Received June 9, 2021, accepted June 19, 2021, date of publication June 24, 2021, date of current version July 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3092261

Classical Arabic Named Entity Recognition Using Variant Deep Neural Network Architectures and BERT

NORAH ALSAARAN AND MAHA ALRABIAH
Computer Science Department, Imam Muhammad Ibn Saud Islamic University, Riyadh 13318, Saudi Arabia
Corresponding author: Norah Alsaaran (NSAlsaaran@imamu.edu.sa)

This work was supported by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University through the Graduate Students Research Support Program.

ABSTRACT Recurrent Neural Networks (RNNs) and transformers are deep learning models that have achieved remarkable success in several Natural Language Processing (NLP) tasks since they do not rely on handcrafted features nor enormous knowledge resources. Named Entity Recognition (NER) is an essential NLP task that is used in many applications such as information retrieval, question answering, and machine translation. NER aims to locate, extract, and classify named entities into predefined categories such as person, organization and location. Arabic NER is considered a challenging task because of the complexity and the unique characteristics of Arabic. Most of the previous research on deep learning based-Arabic NER focused on Modern Standard Arabic and Dialectal Arabic, which are different variations from Classical Arabic. In this paper, we investigate deep learning-based Classical Arabic NER using different deep neural network architectures and a BERT based contextual language model that is trained on general domain Arabic text. We propose two RNN-based models by fine-tunning the pretrained BERT language model to recognize and classify named entities from Classical Arabic. The pre-trained BERT contextual language model representations were used as input features to a BGRU/BLSTM model and were fine-tuned using a Classical Arabic NER dataset. In addition, we explore variant architectures of the proposed BERT-BGRU/BLSTM-CRF models. Experimentations showed that the BERT-BGRU-CRF model outperformed the other models by achieving an F-measure of 94.76% on the CANERCorpus. To the best of our knowledge, this is the first work that aims to recognize named entities in Classical Arabic using deep learning.

INDEX TERMS NER, named entity recognition, classical arabic, BGRU, BLSTM, BERT, CRF, deep learning, natural language processing.

I. INTRODUCTION

NER is an NLP task that involves identifying and classifying named entities in a given text where named entities are pre-defined semantic categories such as person name, location name, and organization name. NER was first introduced in the 1990s as an information extraction task in the Message Understanding Conferences (MUC) [1]. It plays an essential role in several NLP tasks such as information retrieval, question answering, machine translation and text summarization. Therefore, investigating an accurate NER system can serve as a source of information for different NLP applications [1].

There is a fair amount of literature concerning NER research in English, Chinese, and other widely spread languages. Earlier NER models were generally based on three approaches, which are rule-based, machine learning-based and hybrid-based NER. Rule-based NER depends on a set of handcrafted rules extracted by experts in linguistics [2]. Machine learning-based NER relies on feature-engineering and statistical models. While hybrid-based NER combines both rule-based and machine learning-based approaches. Recently, with the breakthrough of multi-layered neural networks, deep learning-based approaches achieved noticeably high performance in many NLP tasks including NER [3]. Deep learning is considered a subfield of machine learning, which uses neural networks of multiple layers. It is a multi-stage approach that can discover and learn the structure and

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa M. Fouda.

representation of unlabeled and unstructured data such as images and documents [4], [5].

The massive growth of Arabic content on the internet caused a heightened need for constructing accurate and robust NLP tools for Arabic. Arabic is considered the official language of the Arab World and is used over a region of 22 countries [6]. It is a Semitic language that is rich in vocabulary, morphology and complex syntactical structures. Arabic exists in different forms including Classical Arabic (CA), Modern Standard Arabic (MSA) and Dialectal Arabic (DA). CA is the original Arabic language spanning from the seventh until the early eleventh century CE [7]. While MSA is a variation of CA that is commonly used in formal contexts such as formal communications, books and news articles. On the other hand, DA is a variation of MSA that is used in everyday life communications. Although CA is not commonly used nowadays, it continues to receive great attention from scholars and common Muslims because it is the language of the Quran (the holy book of Islam), the Hadith (the sayings of Prophet Muhammad) and the Islamic heritage.

Arabic NER is considered a challenging task compared to other languages since it is a morphologically rich language that enables suffixes and prefixes to be conjunctions, prepositions and pronouns such as the word ''ولتنعموا'' *waletana'amo*, which means ''and so that you can rejoice''. Moreover, capitalization of proper nouns, which is considered a considerable NER feature in some languages is not allowed in Arabic. Furthermore, Arabic relies on short vowels (diacritics) to determine the correct word sense. However, it is usually written without vowels, which causes the problem of word sense ambiguity. For example, the word ''درس'' could mean ''دَرْس'' *lesson* or ''دَرَسَ'' *has studied* [1]. In addition, the shortage of datasets and other reliable resources for Arabic introduces another major challenge facing Arabic NER [1].

CA differs from MSA with regard to NER in the sense that it comprises a rich set of named entities that are now abandoned in MSA, especially in the Islamic domain. In addition, many named entities in CA are now changed in MSA. For example, the word ''يلملم'' *ylmlm,* which is the name of a well-known religious place, is now changed to ''السعدية'' *AlsEdyh* [8]. On the other hand, many named entities have evolved in MSA due to cultural and social changes in the Arab world. Generally speaking, Arabic has gone through remarkable lexical, morphological and syntactical changes between its classical and modern periods [7].

Machine learning-based NLP tasks suffer from poor training data, especially for low recourse languages such as Arabic, which causes inadequate model generalization. Transfer learning overcomes this problem by transferring knowledge across domains or tasks. It aims to extract the knowledge from one or more source tasks in order to apply it to other target tasks. The concept of transfer learning is based on the fact that people can intelligently apply knowledge learned previously to solve new problems faster or with better solutions [9]. In transfer learning, a model is trained on a large dataset and then this pretrained model is used to conduct learning for another downstream task [10]. There are two strategies to use pretrained language models for down-stream tasks, which are feature-based and fine-tuning. In feature-based transfer learning, the language model is used as an off-the-shelf feature extractor where the parameters of the model are frozen. This approach requires more complex task-specific model architectures to achieve good accuracy. While in fine-tuned transfer learning, the parameters of the language model are fine-tuned using the down-stream task dataset, which reduces the time to train the down-stream task model. This approach is more suitable for downstream NLP tasks with relatively small dataset sizes, resulting in notable performance improvement for these tasks [11].

In most NLP tasks, it is important to extract the semantic and syntactic knowledge from the used training data. Non-contextual pretrained language models such as Word2vec [12], Glove [13], and Fasttext [14] represent each word in the input sequence in a single vector that combines all contextual meanings of that word. Whereas contextual pretrained language models take the context into consideration by representing each word using more than one vector based on the position of that word in the context. For example, non-contextual word embeddings would produce one vector for the word ''*present*'' despite that it may have different meanings such as ''gift'', ''exist'', and ''show''. This is due to the fact that they cannot understand the context in which the word appears in. On the other hand, contextual word embeddings successfully address this issue by taking the entire input sentence into consideration while calculating the equation for the embeddings. This results in representing each word with different vectors based on its context. Contextual language models have shown that such representations are able to achieve noticeably high performance in many NLP tasks. Bidirectional Encoder Representation from Transformer (BERT) is one of the contextual language models that can be fine-tuned. Its basic component is a multilayer bidirectional transformer encoder that takes into consideration both left and right contexts of a given word in the input sentence [15].

In our previous work [16], we used BGRU over fine-tuned BERT model for MSA NER and the results were promising. In thiswork, we fine-tune a pre-trained BERT language model for the task of Classical Arabic NER using deep learning. First, we fine-tune the vectors produced by the pre-trained BERT using a CA NER task-specific dataset rather than using them directly. Fine-tuning reduces the time for training the down-stream task model since a considerable part of the training was performed by the pre-trained language model. After that, the BERT representations are forwarded to a deep learning model for further sequence modelling. Two RNN models are investigated in this work; Bidirectional Long Short-Term Memory (BLSTM) and Bidirectional Gated Recurrent Unit (BGRU). The output of the BLSTM/BGRU is then passed to a CRF layer to get the probability distribution over the tags.

The proposed models are then trained and evaluated using a CA NER dataset (CANERCorpus) [8]. Finally, the resulting models are evaluated against different baselines to assess their performance. To the best of our knowledge, this is the first work that tackles the extraction of CA named entities using deep learning.

The rest of this paper is organized as follows. Section II presents a background of the scientific concepts related to this research. Section III describes the proposed BERT-BGRU/BLSTM-CRF model architecture for Classical Arabic NER. Section IV describes the used dataset, the baseline models, the experimentation setup, and the evaluation measures used. Section V presents and discusses the results of the evaluation. Finally, Section VI summarizes the conclusions and future work.

II. RELATED WORK

Recent NER research studies have utilized deep learning, which demonstrate its powerful capacity for feature abstraction. RNN-based models such LSTM and GRU are the most dominating used models in NLP. In addition, Bidirectional RNNs (BRNNs) proved their effectiveness in solving many NLP tasks since they process the input sequence from both sides, obtaining potentially richer representations and capturing patterns that may have been missed by unidirectional RNNs. Li *et al.* [17] studied the performance of several deep learning models in the field of biomedical NER. Their study showed that bidirectional models outperformed unidirectional models for different model architectures including word embedding only, character embedding only, and a concatenation of word and character embedding.

BLSTM with CRF (BLSTM-CRF) is the commonly used architecture in NER task [18]–[23]. However, several research studies proved that BGRU has comparable, and sometimes better, performance compared to BLSTM [17], [24]–[26]. BGRU has been used to tackle NER in several languages including Indonesian, Bengali, and Czech [27]–[29]. On the other hand, Convolutional Neural Networks (CNNs) have been also used in NER, Wang *et al.* [30] utilized a hierarchical CNN to extract context information by applying a gating mechanism into the convolutional layer. In addition, [31] used bidirectional gated CNN for Chinese NER. Gao *et al.* [32] proposed attention-based Iterated Dilated CNNs that benefits from a concatenated input obtained from different embeddings including pre-trained Fasttext embedding, BLSTM based character embedding, POS embedding, and position embedding.

Different word embedding models have been investigated in NER research. Many NER studies [18], [33]–[36] have utilized word2vec, including both of its models: Skip-gram and CBOW successfully. Fei *et al.* [37] and Xiaofeng *et al.* [38] used pre-trained Glove model with BLSTM-CRF. Fasttext has been used by [39] and [40] for Japanese NER and Biomedical NER, respectively. In addition, word2vec, GloVe, and Fasttext vector representations have been combined by several research studies. Kağan Akkaya and Can [41] used different levels of word embeddings, which include word-level word embeddings obtained by word2vec and character n-gram level word embeddings obtained by Fasttext. Zhang *et al.* [42] merged pre-trained Glove and word2vec vectors. Although most of the research studies in NER used the pre-trained word embeddings as features, fixing the values of the vectors during model training, there exist some studies that fine-tuned these pre-trained embeddings during the training process [43]–[44]. To reduce the impact of Out of Vocabulary (OOV) words, word level representations are concatenated with character level representations. Commonly used character embedding models such as CNN, BLSTM and BGRU were investigated by [28], [45]–[48]. Ever since BERT [15] contextual language model was introduced and obtained high performance in many NLP tasks [49], many deep learning-based model architectures were introduced to fine-tune it for NER. Li *et al.* [50] demonstrated that adding a BLSTM-CRF layer on top of the fine-tuned pre-trained BERT model performed better than only fine-tuned BERT. In addition, combining dictionary features and radical features improved the model performance. A BGRU-CRF layer on top of fine-tuned BERT model also achieved good results [51], [52]. Yan *et al.* [52] added a multi-head attention layer on top of BGRU layer to increase the context-dependent semantic vector. On other hand, Straka *et al.* [29] results showed that combining the representation of FastText, BERT and Flair for Czech NER outperformed the BERT only version of their models.

Regarding Classical Arabic, and to the best of our knowledge, there exist no studies related to Classical Arabic NER using deep learning. Sajadi and Minaei [53] used a machine learning approach to classify named entities into person, location, and organization tags. Part of Speech (POS), Base Phrase Chunking (BPC), Gazetteer, and keywords were used as features to improve the model's performance. Their model achieved 96.04% and 67.01% F-measure values on NoorCorp and ANERcorp, respectively. In addition, Harrag *et al.* [54] used a rule-based approach for extracting named entities from prophetic narration texts (Hadith). The authors used finite state transducer to assign a conceptual label among a set of labels that include: Num-Kitab, Title-Kitab, Num-Bab, Title-Bab, Num-Hadith, Saned, Matn, Taalik, and Atraf. Their model achieved a 52.00% F-measure value on a set of prophetic narrations texts from Sahîh Al-Bukhari corpus. On the other hand, there exist several research studies concerning deep learning-based MSA and DA NER in the literature. BLSTM-CRF was the most used model [55]–[60]. In addition, adding character embeddings showed an increase in the performance of many Arabic NER models. This is due to the fact that Arabic has a complex morphological system, making it prone to the OOV problem. Awad *et al.* [57] enhanced their model prediction by 1.05 F-measure points when concatenating a CNN character embedding layer with their word embeddings to solve the problem of OOV. Gridach and Haddad [61] improved their model's performance by

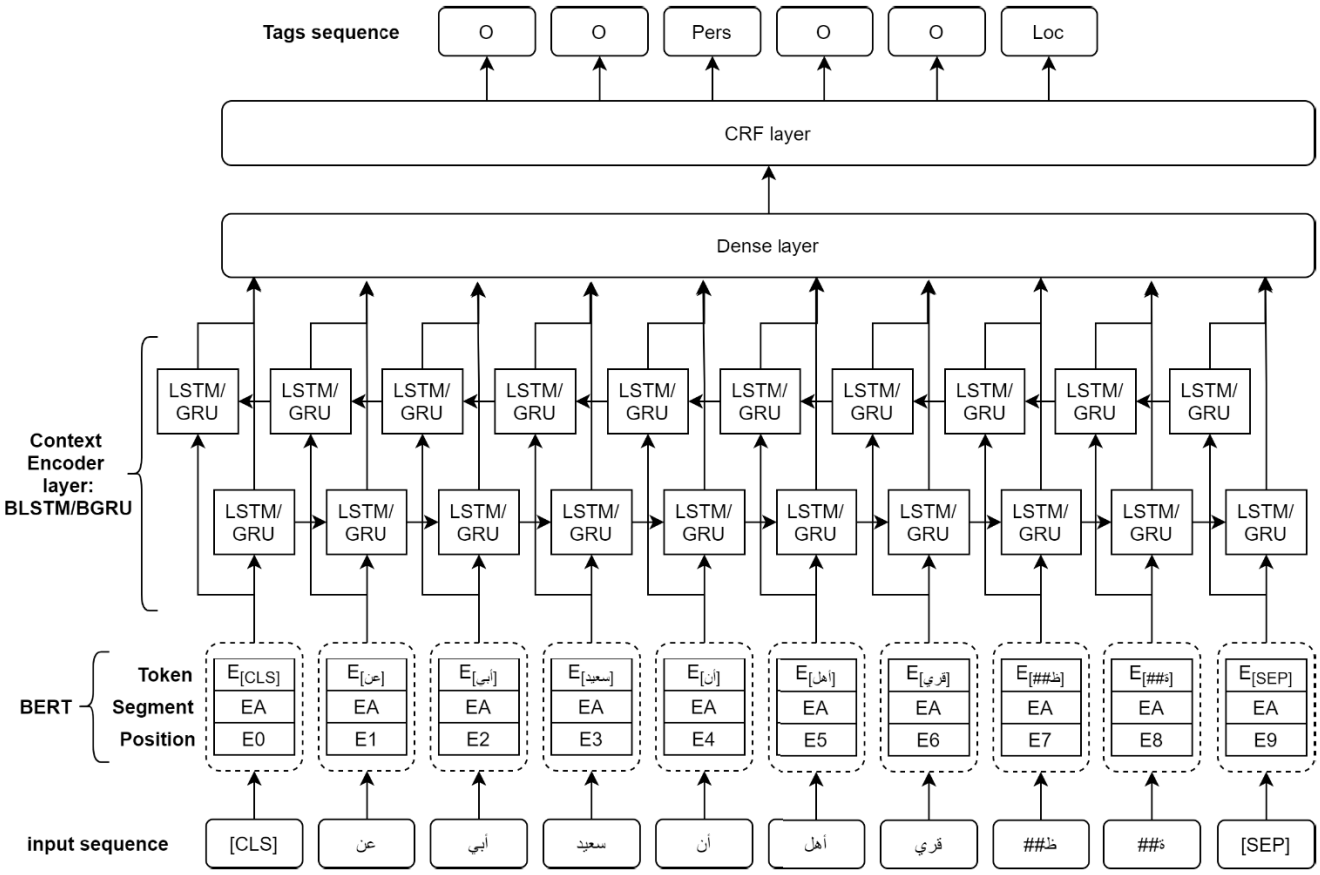

FIGURE 1. Proposed model architecture.

1.61 F-measure points when using BGRU model for character representations.

Other research studies attempted to overcome the problem of lacking sufficient training data for Arabic. Helwe and Elbassuoni [59] used deep co-learning, which is a semi-supervised learning approach that can be trained using both labeled and unlabeled data. Their model used two classifiers that learn from each other using two different views of the data. Furthermore, the attention mechanism demonstrated its effectiveness in many NLP applications such as machine translation. It helps with long input sequences by giving a relative importance for each input word. Ali *et al.* [62] improved their MSA NER model by adding a self-attention layer on top of the encoder in order to provide high or low consideration to words based on their involvement in the creation of the sentence meaning [63]. Ali and Tan [64] used seq2seq model with BLSTM as an encoder and decoder model for MSA NER; their model outperformed the BGRU-CRF model by [61] and the BLSTM-CRF model by [57]. A recent study [65] applied transfer learning with deep neural networks to build a Pooled-GRU model for MSA NER. Their model outperformed the BLSTM-CRF model proposed by [66]. AraBERT [67] is a pretrained BERT model for Arabic that outperformed the BLSTM-CRF model proposed by [66] and BERT multilingual for MSA NER.

III. PROPOSED MODELS ARCHITECTURES

The architecture of deep learning based NER models commonly consists of three main layers: the input representation layer, the encoder layer and the prediction layer. In this work, we fine-tune a pre-trained Arabic BERT model for our input representation. We use BGRU as a context encoder layer and CRF as a prediction layer. Fig. 1 illustrates the architecture of our model.

A. INPUT REPRESENTATION

BERT by Devlin *et al.* [15] is based on a multilayer bidirectional transformer encoder that takes into account both left and right contexts. The used transformer network is developed by [68] and depends on parallel attention layers. BERT is based on two tasks: masked language models and next sentence prediction task. Around 15% of the tokens in the input sequence are randomly masked in the masked language model task. Model uses the multi-layered context to predict the target token and the final hidden vectors corresponding to the masked tokens are fed into an output Softmax over the vocabulary. On the other hand, the next sentence prediction task is used to encode the relationship between two consecutive sentences.

Following the Devlin *et al.* [15], [CLS] and [SEP] tokens should be added at the beginning and end of each input

sentence. BERT relies on WordPiece tokenizer to deal with OOV words since WordPiece segments every unknown word into sub-words. BERT uses token embedding to encode the current word embedding, segment embedding to encode the index embedding of the sentence and position embedding to encode the index embedding of the current word position. All three embeddings are then summed to produce the final BERT representations: the word-level representation and the [CLS] sentence representation.

Arabic pre-trained BERT model, AraBERTv0.1 [67], is trained on 70 million sentences constructed from the manually scraped Arabic news websites for articles and two publicly available large Arabic corpora: the 1.5 billion words Arabic Corpus and the Open Source International Arabic News Corpus (OSIAN). The used dataset was of size 24GB covering news from different media in different Arab regions, and therefore can be representative of a wide range of topics discussed in the Arab world. The total size of the vocabulary was 64k tokens.

On the other hand, character-level representations can also be used to handle OOV words by representing it as the sum of its character n-gram vectors. When dealing with the OOV words, the relative importance of character representation will increase because word representation will be some random values. Character embeddings capture the important morphological and shape information that will help especially with rich morphology languages such as Arabic [61], [63]. In addition, it has the ability for exploiting sub-word-level information such as: prefixes and suffixes. There are two basic architectures that are widely used to extract character-level representations, which are 1D-CNN and RNN.

In this work we investigate the performance of CNN as a character embedding model. CNN is used to produce a word representation by looking at its character-level in order to learn word morphology.

Each word in the input sequence is segmented into its characters. The maximum word length is set to 13 characters, which is the average length of the words in the corpus. As shown in Fig.2, each character is passed into a randomly initialized embedding layer to obtain its representations using a vector with dimension d.

Word w with length l is transformed into a matrix C of size d*l. Convolutional filter H of size d*n is created where n is the filter width and the values within H are randomly initialized. The convolutional operation between C and H is performed to obtain a feature map f with a size of $l-n+1$ as shown in (1), where $\langle C, H \rangle$ is the Frobenius inner product.

$$f_i = \langle C\,[*, i : i+n-1], H \rangle \tag{1}$$

Then, Max-pooling operation (2) is performed to extract a single feature for all feature maps. y_H^w is the character representation of a word w given by the filter H.

$$y_H^w = max\,(f_i) \tag{2}$$

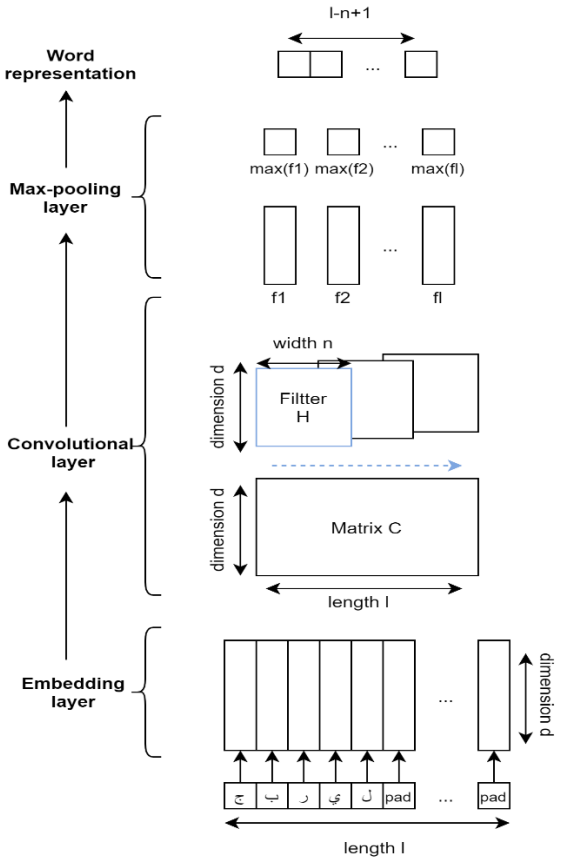

FIGURE 2. CNN based characters embedding model.

This process is repeated several times with different convolutional filters and their outputs are appended to obtain the final representation of the word.

B. CONTEXT ENCODER LAYER

In NER task, it is commonly to use RNN as context encoder model. It processes the input sequence in order where shuffling or reversing the timesteps affect the extracted representations from the sequence. The longer the input sequence gets, the less accurate RNN becomes because it is difficult for the network to memorize far away from previous time steps outputs [5]. This problem is called the vanishing gradient problem. LSTM and GRU are variations of RNN that help in handling the vanishing gradient problem and can learn long dependency input.

In NER task, both past and future information is useful for prediction. For that, we evaluate the impact of BLSTM and BGRU models as context encoder models after obtaining word embeddings from the BERT model.

1) LSTM

LSTM is composed of a cell state (memory) and three gates that control the flow of information into and out of the network. The input gate controls what information from a new input should be kept in the memory. The forget gate

controls what information from the previous timestep cell state should be forgotten. The output gate decides what information should be passed as output at each timestep [5]. The gates are updated according to the following equations.

$$\mathrm{i_t} = \sigma(\mathrm{W_i}[\mathrm{h_{t-1}}, \mathrm{x_t}]\mathrm{b_i}) \quad (3)$$
$$c'_t = tanh(\mathrm{W_c}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_c}) \quad (4)$$
$$\mathrm{f_t} = \sigma(\mathrm{W_f}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_f}) \quad (5)$$
$$\mathrm{o_t} = \sigma(\mathrm{W_o}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_o}) \quad (6)$$
$$c_t = f_t^* c_{t-1} + i_t * c'_t \quad (7)$$
$$h_t = \mathrm{o_t^*} tanh(c_t) \quad (8)$$

where x_t denotes the input at timestep t, h_{t-1} is the hidden state of the previous timestep t − 1. W_i, W_c, W_f, and W_o denote the weights for the input gate, cell state, forget gate, and output gate, respectively. $\mathrm{c'_t}$ refers to candidate values to be added to the output of the input gate at timestep t. $\mathrm{c_t}$ and $\mathrm{h_t}$ denote the cell state and the hidden state after time step t. i_t, f_t, and o_t denote the input gate, the forget gate and the output gate, respectively. b_i, b_c, b_f, and b_o denote the bias parameters for the input gate, cell state, forget gate, and output gate, respectively.

2) GRU

GRU uses two gates: update and reset gates. The update gate decides what information should be used in the next time step input. While the reset gate uses the previous time step output to decide what information should be deleted and what information is of use to the current time step input [4], [5].

The update gate, reset gate, and hidden state h_t at time step t are updated using the following equations.

$$\mathrm{u_t} = \sigma(\mathrm{W_u}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_u}) \quad (9)$$
$$\mathrm{r_t} = \sigma(\mathrm{W_r}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_r}) \quad (10)$$
$$h'_t = tanh(W * [h_{t-1} * r_t, x_t] + b) \quad (11)$$
$$h_t = (1 - u_t)^* h_{t-1} + u_t . h'_t \quad (12)$$

where $\mathrm{W_u}$, $\mathrm{W_r}$ and W denote the weights for the update gate, the reset gate and the cell state, respectively. $\mathrm{b_u}$, $\mathrm{b_r}$ and b denote the bias parameters for the update gate, the reset gate and the cell state, respectively. $\mathrm{u_t}$ denotes the update gate while $\mathrm{r_t}$ denotes the reset gate.

3) BLSTM AND BGRU

In both BLSTM and BGRU, the input sequence is read one word at time by the forward layer from left-to-right and the words from the other side are read by the backward layer. Then, the forward layer h_t^f hidden state and the backward layer h_t^b hidden state are combined to represent the final hidden state h_t such as $h_t = h_t^f \oplus h_t^b$ [5].

C. TAG PREDICTION LAYER: CRF

The tag sequence that corresponds to the input sequence is produced by the prediction layer. A fully connected layer (dense layer) is fed with hidden state h_t to get the score of each tag. The output of this layer is a matrix (A) with size $k*n$ where n indicates the number of words in the input sequence and k is the number of possible named entity tags each word can have. The nth entry of each row in this matrix represents the score of the kth tag for the nth word.

Although the encoder can consider the context information of sentences, it cannot consider the dependencies between tags. CRF was proposed [69] to ensure the validity of the predicted tags by learning the relationship between adjacent tags. It jointly makes tagging predictions where the probability of assigning a tag to a word depends on the features of that word and the previously assigned tag. The transition matrix T that contains the score from one tag to another is a parameter in the CRF layer where $T_{i,j}$ represents the probability of moving from tag i to tag j. This matrix has two extra tags, which are the start and end tags. The size of this matrix is $(k+2)*(k+2)$. For a given input sequence X, to calculate the score of the labels sequence, $Y = [y_1, y_2, \ldots, y_k]$ (13) is used. A_i is the output matrix of the encoder. The probability of label y_i given the input x_i is calculated by (14), which is a Softmax function used to assign a probability for each tag sequence. Yx represents all possible tag sequences for a sentence x. During training, the log-probability (15) of the correct tag sequence should be maximized. Then the output sequence with the maximum score is predicted by (16).

$$score(x, y) = \sum_{i=1}^{k} A_i y_i + \sum_{i=1}^{k-1} T_{yi,yi-1} \quad (13)$$
$$P(x, y) = \frac{e^{score(x,y)}}{\sum_{\acute{y} \in Yx} e^{s(x,y)}} \quad (14)$$
$$log\,(P\,(x, y)) = score\,(x, y) - log(\sum_{\acute{y} \in Yx} e^{s(x,\acute{y})}) \quad (15)$$
$$y^* = argmax_{\acute{y} \in Yx} score(x, \acute{y}) \quad (16)$$

IV. EXPERIMENTS

We have conducted a series of experimental studies to evaluate our proposed models against two baselines, which include BERT and BERT-CRF. In addition, we investigate other different architectures include stacking encoder layers and utilizing a CNN based-character embedding model.

A. DATASET

We used the CANERCorpus as our dataset, which is a Classical Arabic NER corpus [8] that is manually annotated by human experts. It contains more than 7,000 Hadiths (Prophet Muhammad's sayings) from Sahih Al-Bukhari book that are annotated using 20 named entity classes. These classes include person (Pers), location (Loc), organization (Org), measurement (Meas), money (Mon), book (Book), date (Date), time (Time), clan (Clan), natural object (NatOb), crime (Crime), day (Day), number (Num), God (Allah), prophet (Prophet), religion (Rlig), sect (Sect), paradise (Para), hell (Hell), month (Month) and other (O). The corpus contains around 72,108 named entities and 258,264 words. Fig. 3 shows the number of named entities in each tag. In this work, we only consider person (Pers), God (Allah), prophet

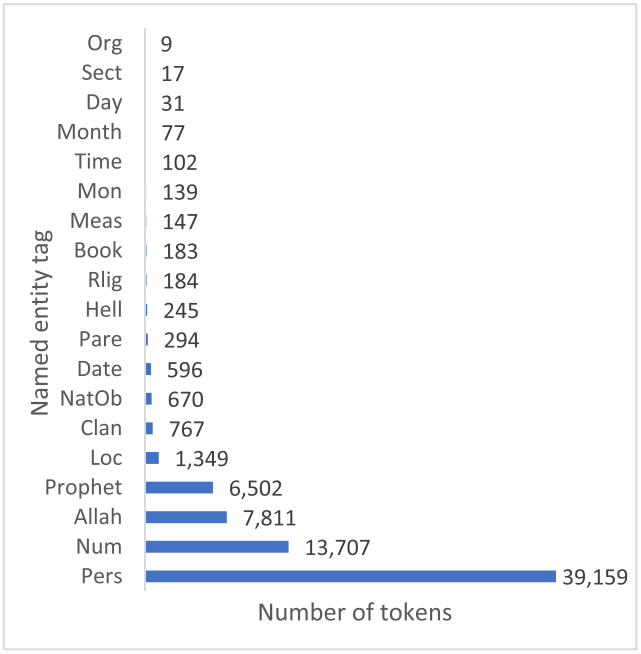

FIGURE 3. CANERCorpus tags distribution based on the number of occurrences in the dataset.

(Prophet), location (Loc), clan (Clan), date (Date), natural object (NatOb) and other (O) named entities since the rest are insufficient to train the model. We have excluded the number (Num) named entities since 93.33% of them were page numbers, which will trivially cause improvement in the model performance.

1) PREPROCESSING

Sentences boundaries in the CANERCorpus dataset are not marked with a dot. In order to overcome this, we have calculated the average sentence length in *Sahih Al-Bukhari* book, which is 38 words, and use it to denote the boundaries of the sentences by adding a dot after every 38 words. Before marking the end of the sentence, we make sure that the last word is labelled with other (O) named entity to avoid cutting related entities. In addition, and in order to achieve high accuracy results, the dataset was cleaned by removing punctuations, other special characters, and diacritics signs (ً,ٌ,ٍ,َ,ُ,ِ,ّ,ْ). We also enclose each input sentence after tokenization between [CLS] and [SEP] tokens. As for tokens tagging, we assign the tag for the first sub-token only and the rest of the sub-tokens are considered as padding. For example, the word 'صلاته' *SlAth* is tokenized as ['صلات' ,'##ة'] and its given tags would be ['O', 'PAD']. Finally, we split the dataset into three datasets; we used ∼80% as training dataset, ∼10% as validation dataset and ∼10% as testing dataset. Table. 1 shows the named entities distribution in each dataset.

B. BASELINE MODELS

To the best of our knowledge, Classical Arabic NER have not yet been explored using deep learning. Therefore, we have investigated the performance of our proposed models, BERT-BLSTM/BGRU-CRF, over two baseline models; the first is a fine-tuned pre-trained BERT model that is combined to a fully connected layer with Softmax function. The second baseline is a fine-tuned pre-trained BERT model followed by a CRF layer. Moreover, we have evaluated the performance of the proposed models when adding CNN-based character embeddings and when stacking two BLSTM/BGRU layers.

TABLE 1. Named entities distribution over training, testing and validation datasets.

Named Entity	Training	Testing	Validation	Total
Pers	31,875	3,758	3,526	39,159
Allah	6309	778	724	7,811
Prophet	5,259	712	531	6,502
Loc	1,086	116	147	1,349
Clan	589	103	75	767
NatOb	551	63	56	670
Date	476	75	45	596
Other	151643	18931	16882	187459
Total	197788	24536	21986	244313

TABLE 2. Value of the models hyperparameters.

Parameter	Value
Number of epochs	10
Maximum length size	54
LSTM/GRU hidden units	200
Batch size	32 for training and 8 for testing and validation
Optimizer	Adam
Learning rate	1e-4
Dropout	0.5, 0.2
Character embedding dimension	20
CNN filter	50

C. EXPERIMENTATION SETUP

We used Pytorch API for the model implementation and all the experiments were run on the Google Colab platform (https://colab.research.google.com/) with a Tesla T4 GPU. We used the training dataset for training our models and utilized the validation dataset to choose the hyper-parameters. The used values for the models' hyperparameters are shown in Table. 2.

D. EVALUATION MEASURES

To evaluate the performance of our proposed models, we used different measures; precision, recall, and F-measure. Equation (17) presents the precision, which measures the number

TABLE 3. BERT-BLSTM/BGRU-CRF models performance.

Model	Named Entity	Precision	Recall	F-measure
BERT-BLSTM-CRF	Pers	98.96%	98.85%	98.90%
	Allah	99.07%	97.89%	98.48%
	Prophet	98.63%	98.16%	98.39%
	Loc	92.24%	89.94%	91.07%
	Clan	87.77%	98.28%	92.70%
	NatOb	87.30%	91.73%	89.43%
	Date	86.40%	87.26%	86.78%
	Other	99.62%	99.65%	99.63%
	Overall	93.75%	95.22%	94.42%
BERT-BGRU-CRF	Pers	99.01%	98.80%	98.90%
	Allah	99.07%	98.09%	98.58%
	Prophet	98.46%	97.85%	98.15%
	Loc	92.41%	89.64%	91.00%
	Clan	89.13%	98.71%	93.67%
	NatOb	87.62%	93.00%	90.20%
	Date	87.47%	88.52%	87.95%
	Other	99.62%	99.66%	99.64%
	Overall	**94.10%**	**95.54%**	**94.76%**

TABLE 4. Results obtained by BERT-BGRU-CRF model against the baseline models.

Model	Precision	Recall	F-measure
BERT (baseline 1)	94.09%	94.93%	94.45%
BERT-CRF (baseline 2)	94.04%	95.44%	94.68%
BERT-BGRU-CRF	**94.10%**	**95.54%**	**94.76%**
BERT-BLSTM-CRF	93.75%	95.22%	94.42%
BERT-Stacked BLSTM-CRF	93.35%	95.37%	94.29%
BERT-Stacked BGRU-CRF	93.99%	95.10%	94.48%
BERT-CNN-BGRU-CRF	94.03%	95.31%	94.62%

TABLE 5. Results obtained by BERT-BGRU-CRF model with BERT fine-tuning and using BERT as a feature extractor.

Model	Precision	Recall	F-measure
BERT (fine-tuning)	**94.10%**	**95.54%**	**94.76%**
BERT (feature extractor)	88.61%	93.96%	90.96%

of correct named entities recognized by the model out of the overall recognized entities. Equation (18) presents the recall, which measures the number of correct named entities recognized by the model out of the overall entities in the corpus. Equation (19) presents the F-measure, which is used to balance the antagonistic relation between precision and recall [3], [70].

$$Precision = \frac{TP}{TP + FP} \tag{17}$$

$$Recall = \frac{TP}{TP + FN} \tag{18}$$

$$\text{F-measure} = 2^{*}\frac{(Precision * Recall)}{(Precision + Recall)} \tag{19}$$

where TP, FP and FN refer to true positives, false positives and false negatives, respectively.

V. RESULTS AND DISCUSSION

To get a more accurate evaluation of our model, we have trained it for 5 times and then calculated the average precision, recall, and F-measure. This is to overcome the effect of random initialization of weights that is associated with neural networks.

To explore the performance of our proposed BERT-BLSTM/BGRU-CRF models and show the effect of different architectures. We have conducted a series of comparative experiments; first the BERT-BLSTM-CRF model has been run and then the BERT-BGRU-CRF model. Table. 3 presents the details of the BERT-BLSTM-CRF and BERT-BGRU-CRF models results. As shown in Table. 3 both models have the same performance in person (Pers) named entity tag. BERT-BLSTM-CRF outperformed in prophet (Prophet) and location (Loc) named entities while BERT-BGRU-CRF outperformed in the remaining named entities. For the overall model performance, BERT-BGRU-CRF model outperformed by 0.34 F-measure points.

Table. 4 shows the comparison results obtained by comparing the BERT-BGRU-CRF model against the baseline models. The first row presents the performance of fine-tuned BERT with only a linear layer that has a Softmax function to model tags probabilities. The second row denotes the performance of BERT with a CRF layer. The third and fourth rows show the performance of the proposed models BERT-BLSTM/BGRU-CRF. The fifth and sixth rows present the performance of the BERT-BLSTM/BGRU-CRF with stacking two layers of BLSTM/BGRU. The final row denotes the best model performance (BERT-BGRU-CRF) with a CNN character embedding model. As shown in Table. 4, BERT-CRF model outperformed BERT with linear layer by 0.23 F-measure points which reflect the effectiveness of the CRF algorithm in considering the dependencies between tags. Moreover, adding a BGRU layer to learn more context information improved the performance by 0.08 F-measure points. However, concatenating CNN character embedding representations to the pre-trained BERT model representations resulted in declining the model's performance. This confirms that the word-piece tokenizer in BERT obviates the need for a character embedding model to deal with the OOV issue. On the other hand, stacking deep learning models also showed a decline in the performance of our best model, BERT-BGRU-CRF, although it was supposed to increase the representations at different levels of abstraction across layers [4]. However, this was not the case with our proposed model, where stacking more layers did not improve the model's performance. Fig. 4 shows the training loss of a single BGRU based model compared to the two-stacked BGRU-based model.

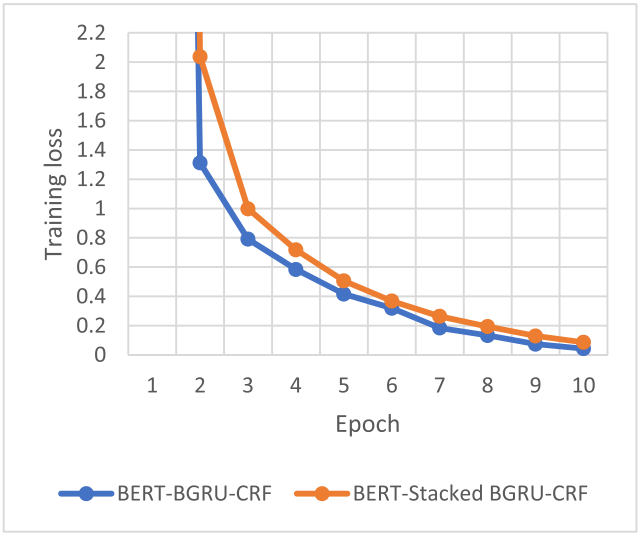

FIGURE 4. BERT-BGRU-CRF and BERT-stacked BGRU-CRF models training loss.

TABLE 6. Example of BERT-BGEU-CRF output using a random sentence from the testing dataset.

Word	Predicted tag	Actual tag	Word	Predicted tag	Actual tag
أبي	Pers	Pers	فاتاه	O	O
زرعة	Pers	Pers	جبريل	Prophet	Prophet
عن	O	O	فقال	O	O
أبي	Pers	Pers	ما	O	O
هريرة	Pers	Pers	الإيمان	O	O
قال	O	O	قال	O	O
كان	O	O	الإيمان	O	O
النبي	Prophet	Prophet	أن	O	O
صلى	O	O	تؤمن	O	O
الله	Allah	Allah	بالله	Allah	Allah
عليه	O	O	وملائكته	O	O
وسلم	O	O	وكتبه	O	O
بارزا	O	O	وبلقائه	O	O
يوما	O	O	ورسله	O	O
للناس	O	O	وتؤمن	O	O

Table. 5 shows the effect of fine-tuning the pre-trained BERT model against using the model directly as a feature extractor. Fine-tuning the pre-trained BERT model increased the performance of our proposed model by 3.8 F-measure points, which proves the effectiveness of fine-tuning BERT parameters with CANERCorpus Classical Arabic NER dataset rather than using them directly.

Table.6 illustrate an example of BERT-BGEU-CRF output for a randomly chosen sentence from the testing dataset. The chosen sentence contains four named entity tags. The columns in Table.6 present the words in the sentence, the predicted tags by the BERT-BGRU-CRF model, and the actual tags of the sentence in the CANERCorpus Classical Arabic NER dataset, respectively.

TABLE 7. Example of BERT-BGEU-CRF output using randomly selected samples from KSUCCA [7].

Word	Predicted tag	Actual tag	Word	Predicted tag	Actual Tag
قالوا	O	O	الله	Prophet	Prophet
يا	O	O	صلى	O	O
موسى	Prophet	Prophet	الله	Prophet	Prophet
إنا	O	O	عليه	O	O
لن	O	O	وسلم	O	O
ندخلها	O	O	وقت	O	O
أبدا	O	O	لأهل	O	O
ما	O	O	المدينة	Loc	Loc
داموا	O	O	.	O	O
فيها	O	O	عبد	Pers	Pers
فاذهب	O	O	الله	Pers	Pers
أنت	O	O	بن	Pers	Pers
وربك	Allah	Allah	صالح	Pers	Pers
فقاتلا	O	O	عن	O	O
إنا	O	O	الليث	Pers	Pers
هاهنا	O	O	بن	Pers	Pers
قاعدون	O	O	سعد	Pers	Pers
.	O	O	قال	O	O
ابن	Pers	Pers	رأيت	O	O
عباس	Pers	Pers	الإبل	NatOb	NatOb
رضي	O	O	التي	O	O
الله	Allah	Allah	تكرى	O	O
عنهما	O	O	للحج	O	O
أن	O	O	تزكى	O	O
رسول	Prophet	Prophet	بالمدينة	Loc	Loc

Moreover, examples of the output of the BERT-BGRU-CRF model with randomly chosen sentences from The Quran, Sahih Muslim, and another Classical Arabic document are illustrated in Table.7. These sentences are selected randomly from KSUCCA [7], which is a raw Classical Arabic corpus. The Actual tags were annotated by two native Arabic speakers.

VI. CONCLUSION AND FUTURE WORK

In this paper, we investigated the performance of fine-tunning pre-trained BERT model on Classical Arabic NER

using variant neural network architectures. Our experiments show the effectiveness of fine-tuning pre-trained BERT language model for languages with rich morphology and low resources, specifically in NER task. We experimented on 6 different model configurations based on BERT. In the first experiment, we found that BERT-BGRU-CRF outperformed BERT-BLSTM-CRF. Moreover, the BERT-BGRU-CRF model performed better than the BERT alone and BERT-CRF models. Furthermore, we investigated the impact of adding CNN-based character embeddings and stacking more than one BGRU layer, both showed declines in the model's performance. The proposed models have been trained and evaluated with the CANERCorpus. Our best model, BERT-BGRU-CRF, performance achieved a 94.76% F-measure value. Our future work will be committed to applying additional features such as dictionary features to improve our results. Also, using the pre-trained BERT model to do other NLP tasks in Classical Arabic.

REFERENCES

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

[2] M. Gridach, "Deep learning approach for Arabic named entity recognition," in *Computational Linguistics and Intelligent Text Processing*, vol. 9623. Cham, Switzerland: Springer, 2018, pp. 439–451.

[3] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Trans. Knowl. Data Eng.*, early access, Mar. 17, 2020, doi: 10.1109/TKDE.2020.2981314.

[4] F. Chollet, *Deep Learning With Python*. Shelter Island, NY, USA: Manning Publications, 2018. [Online]. Available: https://www.manning.com/books/deep-learning-with-python

[5] P. Goyal, S. Pandey, and K. Jain, *Deep Learning for Natural Language Processing: Creating Neural Networks With Python*. Berkeley, CA, USA: Apress, 2018. [Online]. Available: https://www.springer.com/gp/book/9781484236840

[6] R. Fabri, M. Gasser, N. Habash, G. Kiraz, and S. Wintner, *Natural Language Processing of Semitic Languages*. Berlin, Germany: Springer, 2014. [Online]. Available: https://link.springer.com/book/10.1007%2F978-3-642-45358-8

[7] M. Alrabiah, A. Alsalman, E. Atwell, and N. Alhelewh, "KSUCCA: A key to exploring Arabic historical linguistics," *Int. J. Comput. Linguistics*, vol. 5, no. 2, pp. 27–36, 2014.

[8] R. E. Salah and L. Q. B. Zakaria, "Building the classical Arabic named entity recognition corpus (CANERCorpus)," in *Proc. 4th Int. Conf. Inf. Retr. Knowl. Manage. (CAMP)*, Mar. 2018, pp. 1–8.

[9] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[10] S. Ruder, M. Peters, S. Swayamdipta, and T. Wolf, "Transfer learning in natural language processing," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Tuts.*, Minneapolis, MN, USA, Jun. 2019, pp. 15–18.

[11] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," 2020, *arXiv:2003.08271*. [Online]. Available: http://arxiv.org/abs/2003.08271

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Scottsdale, AZ, USA, 2013, pp. 1301–3781.

[13] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[14] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[16] N. Alsaaran and M. Alrabiah, "Arabic named entity recognition: A BERT-BGRU approach," *Comput., Mater. Continua*, vol. 68, no. 1, pp. 471–485, 2021.

[17] J. Li, S. Zhao, J. Yang, Z. Huang, B. Liu, S. Chen, H. Pan, and Q. Wang, "WCP-RNN: A novel RNN-based approach for bio-NER in Chinese EMRs," *J. Supercomput.*, vol. 76, no. 3, pp. 1450–1467, Mar. 2020.

[18] Y. Jin, J. Xie, W. Guo, C. Luo, D. Wu, and R. Wang, "LSTM-CRF neural network with gated self attention for Chinese NER," *IEEE Access*, vol. 7, pp. 136694–136703, 2019, doi: 10.1109/ACCESS.2019.2942433.

[19] E. Trandafili, E. K. Meçe, and E. Duka, "A named entity recognition approach for Albanian using deep learning," in *Complex Pattern Mining: New Challenges, Methods and Applications*, vol. 880, A. Appice, M. Ceci, C. Loglisci, G. Manco, E. Masciari, and Z. W. Ras, Eds. Cham, Switzerland: Springer, 2020, pp. 85–101.

[20] M. Cho, J. Ha, C. Park, and S. Park, "Combinatorial feature embedding based on CNN and LSTM for biomedical named entity recognition," *J. Biomed. Informat.*, vol. 103, Mar. 2020, Art. no. 103381, doi: 10.1016/j.jbi.2020.103381.

[21] F. Saad, H. Aras, and R. Hackl-Sommer, "Improving named entity recognition for biomedical and patent data using Bi-LSTM deep neural network models," in *Natural Language Processing and Information Systems*. Cham, Switzerland: Springer, 2020, pp. 25–36.

[22] C. Liu, C. Zhu, and W. Zhu, "Chinese named entity recognition based on BERT with whole word masking," in *Proc. 6th Int. Conf. Comput. Artif. Intell.*, New York, NY, USA, Apr. 2020, pp. 311–316.

[23] F. Li, B. Zhang, and D. Gao, "Chinese named entity recognition for hazard and operability analysis text," in *Proc. Chin. Control Decis. Conf. (CCDC)*, Aug. 2020, pp. 374–378.

[24] K. Dhrisya, G. Remya, and A. Mohan, "Fine-grained entity type classification using GRU with self-attention," *Int. J. Inf. Technol.*, vol. 12, no. 3, pp. 869–878, Sep. 2020.

[25] J. Zhu, P. Ni, Y. Li, J. Peng, Z. Dai, G. Li, and X. Bai, "An word2vec based on Chinese medical knowledge," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 6263–6265.

[26] A. Gopalakrishnan, K. P. Soman, and B. Premjith, "A deep learning-based named entity recognition in biomedical domain," in *Emerging Research in Electronics, Computer Science and Technology*. Singapore: Springer, 2019, pp. 517–526.

[27] J. A. Kosasih and M. L. Khodra, "Transfer learning for Indonesian named entity recognition," in *Proc. Int. Symp. Adv. Intell. Informat. (SAIN)*, Yogyakarta, Indonesia, Aug. 2018, pp. 173–178.

[28] M. J. R. Rifat, S. Abujar, S. R. H. Noori, and S. A. Hossain, "Bengali named entity recognition: A survey with deep learning benchmark," in *Proc. 10th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2019, pp. 1–5.

[29] M. Straka, J. Straková, and J. Hajič, "Czech text processing with contextual embeddings: POS tagging, lemmatization, parsing and NER," in *Text, Speech, and Dialogue*. Cham, Switzerland: Springer, 2019, pp. 137–150.

[30] C. Wang, W. Chen, and B. Xu, "Named entity recognition with gated convolutional neural networks," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. Cham, Switzerland: Springer, 2017, pp. 110–121.

[31] T. Zhao, H. Liu, Q. Wu, C. Sun, D. Zhan, and Z. Li, "BiGCNN: Bidirectional gated convolutional neural network for Chinese named entity recognition," in *Database Systems for Advanced Applications*. Cham, Switzerland: Springer, 2020, pp. 502–518.

[32] M. Gao, Q. Xiao, S. Wu, and K. Deng, "An attention-based ID-CNNs-CRF model for named entity recognition on clinical electronic medical records," in *Artificial Neural Networks and Machine Learning—ICANN 2019: Workshop and Special Sessions*. Cham, Switzerland: Springer, 2019, pp. 231–242.

[33] S. Long, R. Yuan, L. Yi, and L. Xue, "A method of Chinese named entity recognition based on CNN-BILSTM-CRF model," in *Data Science* (Communications in Computer and Information Science). Singapore: Springer, 2018, pp. 161–175.

[34] T. Li, Y. Guo, and A. Ju, "A self-attention-based approach for named entity recognition in cybersecurity," in *Proc. 15th Int. Conf. Comput. Intell. Secur. (CIS)*, Macao, China, Dec. 2019, pp. 147–150.

[35] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Learning distributed word representations for bidirectional LSTM recurrent neural network," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, San Diego, CA, USA, 2016, pp. 527–533.

[36] D. C. Wintaka, M. A. Bijaksana, and I. Asror, "Named-entity recognition on Indonesian tweets using bidirectional LSTM-CRF," *Procedia Comput. Sci.*, vol. 157, pp. 221–228, Jan. 2019.
[37] H. Fei, Y. Ren, and D. Ji, "Dispatched attention with multi-task learning for nested mention recognition," *Inf. Sci.*, vol. 513, pp. 241–251, Mar. 2020.
[38] M. Xiaofeng, W. Wei, and X. Aiping, "Incorporating token-level dictionary feature into neural model for named entity recognition," *Neurocomputing*, vol. 375, pp. 43–50, Jan. 2020.
[39] S. Misawa, M. Taniguchi, Y. Miura, and T. Ohkuma, "Character-based bidirectional LSTM-CRF with words and characters for Japanese named entity recognition," in *Proc. 1st Workshop Subword Character Level Models NLP*, Copenhagen, Denmark, 2017, pp. 97–102.
[40] H. Huggard, A. Zhang, E. Zhang, and Y. S. Koh, "Feature importance for biomedical named entity recognition," in *Advances in Artificial Intelligence*. Cham, Switzerland: Springer, 2019, pp. 406–417.
[41] E. K. Akkaya and B. Can, "Transfer learning for turkish named entity recognition on noisy text," *Natural Lang. Eng.*, vol. 27, no. 1, pp. 35–64, Jan. 2021.
[42] S. Zhang, Y. Sheng, J. Gao, J. Chen, J. Huang, and S. Lin, "A multi-domain named entity recognition method based on part-of-speech attention mechanism," in *Computer Supported Cooperative Work and Social Computing*. Singapore: Springer, 2019, pp. 631–644.
[43] M. Du, M. Pang, and B. Xu, "Multi-task learning for attribute extraction from unstructured electronic medical records," in *Semantic Technology*. Singapore: Springer, 2020, pp. 117–128.
[44] G. Jin and Z. Yu, "A Korean named entity recognition method using Bi-LSTM-CRF and masked self-attention," *Comput. Speech Lang.*, vol. 65, Jan. 2021, Art. no. 101134.
[45] X. Li, C. Fu, R. Zhong, D. Zhong, T. He, and X. Jiang, "A hybrid deep learning framework for bacterial named entity recognition with domain features," *BMC Bioinf.*, vol. 20, no. S16, p. 583, Dec. 2019, doi: 10.1186/s12859-019-3071-3.
[46] H. Zhou, S. Ning, Z. Liu, C. Lang, Z. Liu, and B. Lei, "Knowledge-enhanced biomedical named entity recognition and normalization: Application to proteins and genes," *BMC Bioinf.*, vol. 21, no. 1, p. 35, Jan. 2020, doi: 10.1186/s12859-020-3375-3.
[47] J. M. Giorgi and G. D. Bader, "Towards reliable named entity recognition in the biomedical domain," *Bioinformatics*, vol. 36, no. 1, pp. 280–286, Jan. 2020, doi: 10.1093/bioinformatics/btz504.
[48] C. Song, Y. Xiong, W. Huang, and L. Ma, "Joint self-attention and multi-embeddings for Chinese named entity recognition," in *Proc. 6th Int. Conf. Big Data Comput. Commun. (BIGCOM)*, Deqing, China, Jul. 2020, pp. 76–80.
[49] W. Zhang, S. Jiang, S. Zhao, K. Hou, Y. Liu, and L. Zhang, "A BERT-BiLSTM-CRF model for Chinese electronic medical records named entity recognition," in *Proc. 12th Int. Conf. Intell. Comput. Technol. Automat. (ICICTA)*, Xiangtan, China, Oct. 2019, pp. 166–169.
[50] X. Li, H. Zhang, and X.-H. Zhou, "Chinese clinical named entity recognition with variant neural structures based on BERT methods," *J. Biomed. Informat.*, vol. 107, Jul. 2020, Art. no. 103422.
[51] Q. Cai, "Research on Chinese naming recognition model based on BERT embedding," in *Proc. IEEE 10th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Beijing, China, Oct. 2019, pp. 1–4.
[52] S. Yan, J. Chai, and L. Wu, "Bidirectional GRU with multi-head attention for Chinese NER," in *Proc. IEEE 5th Inf. Technol. Mechatronics Eng. Conf. (ITOEC)*, Chongqing, China, Jun. 2020, pp. 1160–1164.
[53] M. B. Sajadi and B. Minaei, "Arabic named entity recognition using boosting method," in *Proc. Artif. Intell. Signal Process. Conf. (AISP)*, Shiraz, Iran, Oct. 2017, pp. 281–288.
[54] F. Harrag, E. El-Qawasmeh, and A. M. S. Al-Salman, "Extracting named entities from prophetic narration texts (Hadith)," in *Software Engineering and Computer Systems*. Berlin, Germany: Springer, 2011, pp. 289–297.
[55] M. Khalifa and K. Shaalan, "Character convolutions for Arabic named entity recognition with long short-term memory networks," *Comput. Speech Lang.*, vol. 58, pp. 335–346, Nov. 2019.
[56] I. E. Bazi and N. Laachfoubi, "Arabic named entity recognition using deep learning approach," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 3, p. 2025, Jun. 2019.
[57] D. Awad, C. Sabty, M. Elmahdy, and S. Abdennadher, "Arabic name entity recognition using deep learning," in *Statistical Language and Speech Processing*. Cham, Switzerland: Springer, 2018, pp. 105–116.
[58] M. Gridach, "Character-aware neural networks for Arabic named entity recognition for social media," in *Proc. 6th Workshop South Southeast Asian Natural Lang. Process. (WSSANLP)*, Osaka, Japan, 2016, pp. 23–32.
[59] C. Helwe and S. Elbassuoni, "Arabic named entity recognition via deep co-learning," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 197–215, Jun. 2019.
[60] M. Attia, Y. Samih, and W. Maier, "GHHT at CALCS 2018: Named entity recognition for dialectal Arabic using neural networks," in *Proc. 3rd Workshop Comput. Approaches Linguistic Code-Switching*, Melbourne, VIC, Australia, 2018, pp. 98–102.
[61] M. Gridach and H. Haddad, "Arabic named entity recognition: A bidirectional GRU-CRF approach," in *Computational Linguistics and Intelligent Text Processing*. Cham, Switzerland: Springer, 2018, pp. 264–275.
[62] M. Ali, G. Tan, and A. Hussain, "Bidirectional recurrent neural network approach for Arabic named entity recognition," *Future Internet*, vol. 10, no. 12, p. 123, Dec. 2018.
[63] M. N. A. Ali, G. Tan, and A. Hussain, "Boosting Arabic named-entity recognition with multi-attention layer," *IEEE Access*, vol. 7, pp. 46575–46582, 2019.
[64] M. N. A. Ali and G. Tan, "Bidirectional encoder–decoder model for Arabic named entity recognition," *Arabian J. Sci. Eng.*, vol. 44, no. 11, pp. 9693–9701, Aug. 2019.
[65] M. Al-Smadi, S. Al-Zboon, Y. Jararweh, and P. Juola, "Transfer learning for Arabic named entity recognition with deep neural networks," *IEEE Access*, vol. 8, pp. 37736–37745, 2020, doi: 10.1109/ACCESS.2020.2973319.
[66] S. D. A. Alzboun, S. K. Tawalbeh, M. Al-Smadi, and Y. Jararweh, "Using bidirectional long short-term memory and conditional random fields for labeling Arabic named entities: A comparative study," in *Proc. 5th Int. Conf. Social Netw. Anal., Manage. Secur. (SNAMS)*, Oct. 2018, pp. 135–140.
[67] W. Antoun, F. Baly, and H. Hajj, "AraBERT: Transformer-based model for Arabic language understanding," in *Proc. 4th Workshop Open-Source Arabic Corpora Process. Tools, With Shared Task Offensive Lang. Detection*, Marseille, France, 2020, pp. 9–15.
[68] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*. [Online]. Available: http://arxiv.org/abs/1706.03762
[69] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 8th Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2001, pp. 282–289.
[70] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2019.

NORAH ALSAARAN received the B.S. degree in computer science from Imam Muhammad Ibn Saud Islamic University, Saudi Arabia, in 2016, where she is currently pursuing the M.S. degree. She is also a Teaching Assistant with the Computer Science Department, Imam Muhammad Ibn Saud Islamic University. Her research interests include deep learning and natural language processing.

MAHA ALRABIAH received the Ph.D. degree in computer science from King Saud University, Saudi Arabia, in 2014. She is currently an Assistant Professor with the Computer Science Department, Imam Muhammad Ibn Saud Islamic University, Saudi Arabia. Her research interests include computational linguistics and artificial intelligence.