Journal of Computer Science

Original Research Paper

Improving Arabic Named Entity Recognition with a Modified Transformer Encoder

[1]Hamid Sadeq Mahdi Alsultani and [2]Ahmed H. Aliwy

[1]*Department of Computer Science, College of Basic Education, University of Diyala, Iraq*
[2]*Department of Computer Science, Faculty of Computer Science and Mathematics, University of Kufa, Iraq*

Article history
Received: 04-12-2022
Revised: 17-01-2023
Accepted: 28-02-2023

Corresponding Author:
Hamid Sadeq Mahdi Alsultani
Department of Computer Science, College of Basic Education, University of Diyala, Iraq
Email: hamedsultani@uodiyala.edu.iq

Abstract: This article investigates the use of a transformer encoder for Arabic Named Entity Recognition (NER). The classic transformer that was originally proposed for machine translation adopts the absolute sinusoidal position embedding which is aware of distance but unfortunately is not aware of the directionality. However, in the NER task, both distance and orientation are crucial. Therefore, in this study, instead of using absolute sinusoidal position encoding, we employ relative positional encoding and incorporate the directionality information in our NER model. More specifically, our proposed model uses Bidirectional Long Short-Term Memory (BiLSTM) for encoding every input token. Then, the output of the encoder is fed to the multi-head attention where both the distance and directionality information are incorporated. The decoder layer with a simple fully connected layer takes as input, the result of the attention layer, and the prediction layer with Conditional Random Fields (CRF) predicts the tag of each token. We validate our proposed approach on two merged public datasets, namely, ANER corp and AQMAR. Our experiment results demonstrate significant improvements when compare to the vanilla Transformer with absolute sinusoidal position encoding while achieving a state-of-the-art result on a merged two Arabic public datasets.

Keywords: Named Entity Recognition, Transformer, Natural Language Processing

Introduction

Named Entity Recognition (NER) is one of the most studied Natural Language Processing (NLP) tasks. With a given text, the NER model's goal is to assign to each word, a label using a pre-defined set of labels (Li *et al*., 2020). NER is an important text pre-processing phase in several NLP tasks such as question answering, relation extraction, and so forth. More specifically, in NLP tasks such as machine translation, question answering, text clustering, and relation extraction, NER produces useful information for chunks of raw text to be used in such tasks (Roy, 2021). In the literature, numerous natural languages, including English (Sun *et al*., 2018), Chinese (Cao *et al*., 2018), Arabic (Ali *et al*., 2020), have been the focus of NER research, and so on.

Arabic is widely spoken. Over 25 nations have 360 million Arabian speakers. Arabic is rich in morphology and semantics which is one of its important features. It's one of the five recognized international languages (Ali *et al*., 2019). When it comes to NER study, Arabic has recently emerged as one of the most significant challenges because of: (1) Particular features and peculiarities (Farghaly and Shaalan, 2009) of the Arabic language and (2) The scarcity of labeled corpora currently available. Several Arabic NER models and a small number of publicly available datasets were suggested in the literature (Shaalan, 2014). Especially, the ANER corp dataset was created by Benajiba and Rosso (2007) while Mohit *et al*. (2012) proposed the AQMAR dataset. These datasets have a limited number of classes. Therefore, the $WikiFANE_{Gold}$ dataset which contains 50 classes was proposed by Alotaibi and Lee (2014). Despite significant efforts that have been made recently in the Arabic NER, challenges still exist because of some characteristics of the Arabic language. Named entity recognition consists of:

- Detection of entities in the raw text: If we can use the capital letters as indicators to determine where named entities start and end as in the English language, it will be quite an easy task. However, the Arabic language does not support capitalization which is one obstacle to obtaining good performance in NER (Benajiba and Rosso, 2007). As illustrated in Fig. 1,

Science Publications

© 2023 Hamid Sadeq Mahdi Alsultani and Ahmed H. Aliwy. This open-access article is distributed under a Creative Commons Attribution (CC-BY) 4.0 license.

we present an example of two words where only one of them is a Named Entities (NE) while both of them start with the same character

- Classification of NEs: To decide the class of a NE, the classification system looks at both the word and the sentence where it occurs. However, Arabic is a highly inflectional language meaning that a word is formed by the following components (Eq. 1):

$$word = prefix(es) + lemma + suffix \tag{1}$$

where the prefixes can be either articles, conjunctions, or prepositions. In the Arabic language, both prefixes and suffixes can be combinations meaning that a word can have one or more affixes. In terms of a statistical viewpoint, this inflectional feature of Arabic makes it a more complex morphology and more sparse language compared to other languages. Therefore, when dealing with Arabic NLP tasks, a large amount of training data must be used in order for the model to work well.

Additionally, the lack of uniformity in writing styles is another challenge. Moreover, compared to other languages, the Arabic language has more speech sounds, that is, when names in another language are transliterated to Arabic, many variants of named entities from the same name indicating the same meaning can be obtained. Also, in Arabic, most vowels are represented by diacritics, which influences the representation of the phonetic and therefore resulting to have words with different meanings. Thus, having the same word with different meanings indicates that a word may be represented by two different named entities.

Considering the above-mentioned challenges, in recent years, researchers have tackled the Arabic NER using either handcrafted rules (Shaalan and Raza, 2008) or statistical learning (Benajiba and Rosso, 2007). In the former case, where handcrafted grammatical rules of linguists are employed, in-depth knowledge of the Arabic language is required.

This approach consumes time when linguists' knowledge plus background are weak. Contrary, statistical learning uses a training set of instances to pick up the patterns of the model relevant to any NER task. The advantage of such an approach is that no knowledge of the Arabic language is required. Only sufficient corpus is needed to fit Machine Learning (ML) models. The challenge with traditional ML-based methods is that they are not able to deal with the great amount of Arabic data available on the web. Recently, researchers' interest in neural networks has led to the development of several deep learning architecture proposals. More specifically, advanced solutions for NLP tasks in different languages have been made by combining supervised methods with deep neural networks (Melis *et al*., 2017). These neural networks achieved extremely impressive results on different NLP tasks such as NER (Lample *et al*., 2016), relation extraction (Konstantinova, 2014), machine translation (Chu and Wang, 2018), and so forth. These neural networks are mostly built using word representation information or embedding. However, these models struggle when handling Out of Vocabulary (OoV) words. In the past years, several NLP models have considered both word and character embedding to solve the problem of OOV.

For the purpose of machine translation, (Vaswani *et al*., 2017) suggested a fully connected self-attention design called a "transformer" for representing lengthy contexts. The attention mechanism is widely used to improve deep learning models for numerous fields including NLP tasks, computer vision, and soon, even though it was first proposed for machine translation models (Vaswani *et al*., 2017; Luong *et al*., 2015). Especially, in the context of NER, the attention mechanism enables a deep learning model to focus its attention on the relevant part parts of the sequence during the prediction of the label of a word. However, (Yan *et al*., 2019; Guo *et al*., 2019) researchers demonstrated that the vanilla Transformer doesn't perform well on the NER tasks. They found that the sinusoidal position encoding used by Vaswani *et al*. (2017) is aware of distance but not of directionality, which is important for the NER models. According to our best knowledge, the Transformer network is not yet used in Arabic NER. Even though the transformer is used for NER in other languages like English (Liu *et al*., 2022), all the models that employed the transformer used the vanilla one which is not aware of the directionality. In this study, we employ the Transformer for Arabic NER by incorporating it into our model, the directionality aware. We measure the performance of our architecture by using the F1-score metric.

This section provides recent previous works that have been published in the literature in the context of Arabic NER using deep learning. Nevertheless, before deep learning shows its efficiency in NLP tasks, feature engineering methods were used. Especially, (Goyal *et al*., 2018) classified NER approaches into ones that depend on either rules or learning or a mix of both.

Moreover, Etaiwi *et al*. (2017) categorized Arabic NER approaches into the following classes: Markov-based model or HMM (hidden Markov Model), Conditional Random Field (CRF), Naïve Bayes, neural networks, maximum entropy, and support vector machine.

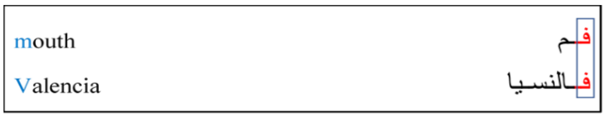

Fig. 1: Illustration of the lack of capital letters in the Arabic language

An in-depth survey on Arabic NLP approaches was reviewed by Al-Ayyoub *et al.* (2018). By comparing the Arabic NLP approaches to works available in the English language, the authors concluded that there exists a considerable gap that needed to be filled.

Dahan *et al.* (2015) developed an ANER model based on HMM where stemming was used to solve some ambiguity problems related to the Arabic language. A decision trees approach was proposed by Al-Shoukry and Omar (2015) for Arabic NER. The authors considered named entities such as a person, time, and date and achieved an F1-measure of 81.35% using a dataset gathered from online resources.

Benajiba *et al.* (2008) replaced maximum entropy with a conditional random field to improve the performance of their NER model. They obtained 72.77, 86.90 and 79.21% respectively for recall, precision, and F-measure.

AbdelRahman *et al.* (2010) used word-level features, POS tagging, BPC, gazetteers, and morphological features to handle Arabic NER. They used CRF to identify named entities such as a person, location, device, cell phone, organization, car, date, and time.

Mohammed and Omar (2012) used Neural Networks (NNs) to classify four entities in the Arabic language: A person, organization, location, and miscellaneous. When comparing to decision trees, the authors have demonstrated that NNs reach 92% of precision while the decision tree achieves only a precision of 87%.

Bazi and Laachfoubi (2018) conducted an investigation on the impact of word representations on the Arabic NER systems. They compared different neural word embedding algorithms on the AQMAR dataset and concluded that combining different approaches together improves the network performance.

Because handcrafted features and machine learning approaches cannot deal with a large volume of data available on the web, current studies shifted towards deep learning to address Arabic NLP problems, including NER. In what follows, we review NER methods proposed for both Arabic and other languages, which rely on deep learning. When it comes to Arabic NER, several deep-learning approaches were recently proposed. More specifically, (Shahina *et al*., 2019) applied Bi-directional recurrent networks on the ANER corp dataset and the results revealed that bidirectional implementations provide excellent performance compared to only one-direction recurrent networks. More specifically, the authors experimented with RNN variants such as LSTM and GRU.

A deep co-learning based on a semi-supervised learning algorithm is proposed by (Helwe and Elbassuoni, 2019) for Arabic NER. The authors first designed and implemented a classifier for Wikipedia articles by using LSTM-DNN and thus obtained a semi-labeled corpus for the Arabic NER application. In the testing phase, the authors used three public datasets and concluded that their proposal outperforms the state-of-the-art approaches.

Using a BiLSTM design, (Ali *et al*., 2018) suggested an approach to Arabic NER. The authors used both word and character embedding. When comparing Bi-LSTM to Bi-GRU on the ANER corp dataset, the authors achieved with Bi-LSTM an F1-score equal to 88.01% while Bi-GRU reached 87.12% of the F1-measure. The same researchers suggested a BiLSTM and multi-attention layer-based NER technique (Ali *et al*., 2019). The first attention layer is the embedding attention layer which concatenates the word and character embedding. The second attention layer is on top of the encoder which is a Bi-LSTM. By using two attention layers, the authors significantly improved the performance of their model and reached an F1-measure of 91% on the ANER corp dataset.

El Bazi and Laachfoubi (2019) came up with a new design for Arabic NER based on BiLSTM and "Conditional Random Fields" (CRF). On the ANER corp dataset, they obtained an F1-measure of 90.6%.

Kuru *et al.* (2016) proposed a NER model for the Arabic language by employing character-level information and word embedding. A deep BiLSTM is used to encode the embedding information. By comparing their approach to hand-engineered features methods, they achieved good results.

Al-Smadi *et al.* (2020) suggested an Arabic NER model based on transfer learning. They used a BiGRU to get the encoded output from a "Universal Sentence Encoder" (USE) that had been pre-trained. Next, the researchers applied "Global Max Pooling" (GMP) and "Global Average Pooling" (GAP). The output of such processes is merged and passed into a feed-forward neural network, which makes predictive inferences based on this data. For the $WikiFANE_{Gold}$ corpora, they got an F1-score of 91.20%.

Ali and Tan (2019) suggested an encoder-decoder NER model for Arabic. The authors added a mirror change to their paper proposed by (Ali *et al*., 2019). Using an attention layer, the word embedding and character embedding in the initial layer were joined together. A BiLSTM is used to encode the result of this layer. Following the construction of the encoder, a second attention layer is added on top, the output of which is fed into the decoder, itself a second BiLSTM. The model made by the researchers did much better on the AQMAR and ANER corp corpora than the model presented by Ali *et al.* (2019).

Alsaaran and Alrabiah (2021) fine-tuned the BERT model for Arabic NER. Especially, the pre-trained BERT model was used for embeddings and this layer's result is utilized as incoming characteristics to a Bidirectional Gated Recurrent Unit (BiGRU). Ultimately, a fully connected layer is utilized for classification in the deepest layer. On the ANER corp corpora, they got an F1-measure of 92.28% and when they added the AQ-MAR corpora to the ANER-Corp corpora, they got an F1-measure of 90.68%.

Gridach (2016) utilized a BiLSTM-CRF system that embeds characters and utilized a pre-trained word2vec to embed words for extracting named entities from Arabian social mediums.

Khalifa and Shaalan (2019) used Convolutional Neural Networks (CNNs) for character-level embedding and the Skip Gram Word2vec model for word embedding. To further train the model, a BiLSTM is given the embedded data and a CRF performs the classification at the last layer. To keep away from using task-related characteristics in the tasks of sequence labeling like NER, Part of Speech (PoS) and Chunking, (Collobert *et al.*, 2011) combined Multi-Layer Perceptron (MLP) a-n-d C-N-N. The BiLSTM was first combined with CNN by Huang *et al.* (2015) to address sequence tagging problems and after that BiLSTM is widely used in NER tasks for various languages such as (Chiu and Nichols, 2016; Dong *et al.*, 2016) which used a combination of BiLSTM and CRF to recognize Chinese named entities.

Materials and Methods

In this part, we describe in depth the methodology behind our suggested system. Firstly, we presented the transformer encoder and its components. We then explained the absolute encoding that is inherent to the classic transformer, before turning to our modified solution. Our suggested system is shown in Fig. 2.

Transformer Encoder

Given a sequence with length *l* and d be the dimension of the input, the transformer encoder of Vaswani *et al.* (2017) takes as input a matrix $H \in \mathbb{R}^{l\times d}$. Then, to project *H* into different spaces, the query matrix W_q, the key matrix W_k and the value matrix W_v with the usual size of $\mathbb{R}^{l\times dk}$ are learned. Here, *dk* is a hyperparameter. After *H* has been projected, the scaled dot product attention is computed by using Eqs. 2-4 respectively:

$$Q, K, V = HW_q, HW_k, HW_v \tag{2}$$

$$A_{t,j} = Q_t K_j^T \tag{3}$$

$$Attn(K, Q, V) = SoftMax\left(\frac{A}{\sqrt{d_k}}\right)V \tag{4}$$

where, Q_t, *j*, and K_j are the query vector of the *t*-th token, the token the *t*-th token attends, and the key vector representation of the *j*-th token respectively. The SoftMax is applied along the last dimension. To improve the capability of self-attention, numerous sets of W_q, W_k, and W_v can be used instead of one set. The multi-head self-attention is formed by using several groups. It can be calculated in Eqs. 5-7 respectively:

$$Q^{(h)}, K^{(h)}, V^{(h)} = HW_q^{(h)}, HW_k^{(h)}, HW_v^{(h)} \tag{5}$$

$$head^{(h)} = Attn\left(Q^{(h)}, K^{(h)}, V^{(h)}\right) \tag{6}$$

$$MultiHead^{(H)} = \left[head^{(1)}, head^{(2)}, \ldots, head^{(n)}\right]W_o \tag{7}$$

Here, *n* represent the number of heads, the superscript *h* represents the head index and [$head^{(1)}$, $head^{(2)}$, ..., $head^{(n)}$] means concatenation in the last dimension of size $\mathbb{R}^{l\times d}$. W_o is a learnable parameter of size $\mathbb{R}^{d\times d}$. $d_k * n = d$.

After that, the result of "multi-head" self-attention is fed into a "position-wise" feed-forward network, Eq. 8:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{8}$$

Here, W_1, W_2, b_1 and b_2 are trainable parameters and $W_1 \in \mathbb{R}^{l\times dff}$, $W_2 \in \mathbb{R}^{dff\times d}$, $b_1 \in \mathbb{R}^{dff}$ and $b_2 \in \mathbb{R}^{d}$. d_{ff} is a hyperparameter. In addition, layer normalization and Residual connection of Vaswani *et al.* (2017) are also included in our architecture.

Sinusoidal Position Encoding and Relative Position Encoding

In this part, we provide the difference between the sinusoidal position encoding utilized in the classic Transformer and the relative position encoding that we employ in our proposed model.

Sinusoidal Position Encoding

In the classic transformer, Vaswani *et al.* (2017) proposed using the embeddings of position produced by sinusoids of different frequencies in order to enable self-attention to be able to catch the languages' sequential features. Especially, the sinusoidal encoding models the position of each token and measures the distance between two tokens. Given a sentence with length *l*, the position encoding of the *t*-th token is obtained by Eqs. 9-10 respectively:

$$PE_{t,2i} = \sin\left(\frac{t}{10000^{\frac{2i}{d}}}\right) \tag{9}$$

$$PE_{t,2i+1} = \cos\left(\frac{t}{10000^{\frac{2i}{d}}}\right) \tag{10}$$

where, d is the dimension of the input sequence while i is in the range of $\left[0, \frac{d}{2}\right]$. With any fixed distance k, PE_{t+k} is represented using linearly transforming PE_t (Vaswani *et al*., 2017).

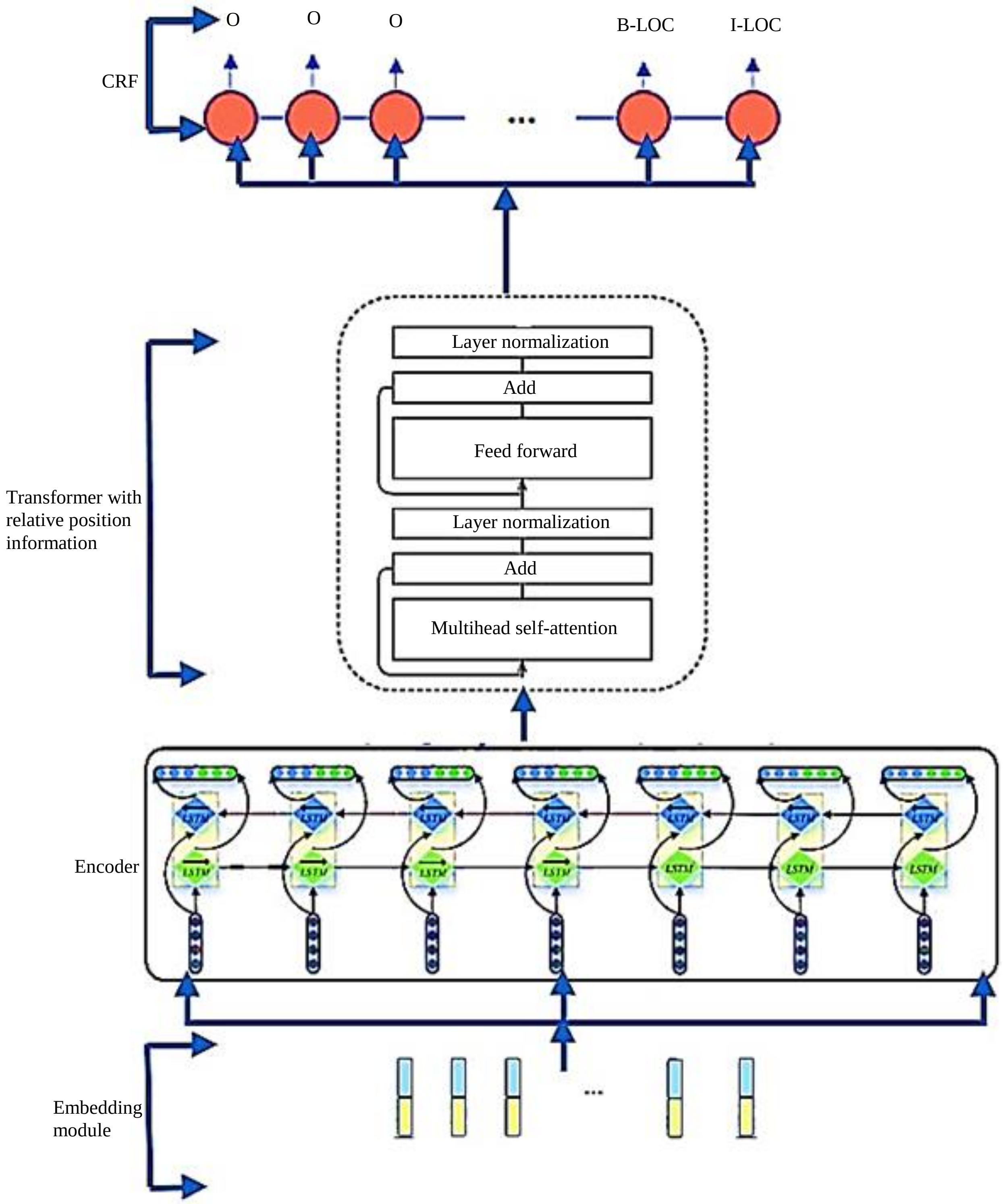

Fig. 2: Architecture of the proposed model

However, as mentioned earlier, this encoding type is not aware of the directionality which is important in NER models.

Relative Position Encoding

Based on Shaw *et al*. (2018), we adopt the relative position encoding instead of using the sinusoidal one used in the classic transformer. In the classic transformer, for a position *t* and a distance *k*, the calculation between PE_t and PE_{t+k}, is $PE_t^T.W_q^T.W_k.PE_{t+k}$, where W_q^T Wk is the learnable parameters of the query and key matrices. To incorporate the directionality in our model, the attention score is computed using Eqs. 11-14 respectively (Shaw *et al*., 2018):

$$Q,K,V = HW_q, H_{dk}, HW_v \tag{11}$$

$$R_{t-j} = \left[\cdots \sin\left(\frac{t-j}{10000^{\frac{2i}{dk}}}\right)\cos\left(\frac{t-j}{10000^{\frac{2i}{dk}}}\right)\cdots\right]^T \tag{12}$$

$$A_{t,j}^{rel} = Q_t K_j^T + Q_t R_{t-j}^T + uK_j^T + vR_{t-j}^T \tag{13}$$

$$Attn(Q,K,V) = SoftMax\left(A_{t,j}^{rel}\right)V \tag{14}$$

where, *t* denotes the target token and *j* denotes the context token. The key vector and query vector of tokens *j* and *t* are denoted by K_j and Q_t respectively. H_{dk} is obtained by splitting *H* into $\frac{d}{dk}$ partitions and one partition is used for each head. $u \in \mathbb{R}^{dk}$ and $v \in \mathbb{R}^{dk}$ are learnable parameters, while $R_{t-j} \in \mathbb{R}^{dk}$ is the relative-positional-encoding. *i* is in the range of $\left[0, \frac{dk}{2}\right]$. The attention value between a couple of tokens can be represented by $Q_t K_j^T$, while $Q_t R_{t-j}^T$ denotes *t*th token's bias on specific relative distance. The *j*th token's bias is denoted uK_j^T while vR_{t-j}^T representing the bias term for a specific orientation and distance. By using Eq. 12 and applying rules sin(*x*) = -sin(*x*) and cos(*x*) = cos(-*x*), the relative position can be expressed as given in Eq. 15:

$$R_t, R_{-t} = \left(\sin(c_o t)\cos(c_o t)\cdots - \sin\left(C_{\frac{d}{2}-1} t\right)\cos\left(c_{\frac{d}{2}-1} t\right)\right), \left(-\sin(c_o t)\cos(c_o t)\cdots\sin\left(C_{\frac{d}{2}-1} t\right)\cos\left(c_{\frac{d}{2}-1} t\right)\right) \tag{15}$$

In the above equation, for a distance t, the forward and backward relative positional encodings are similar according to the cos($c_i t$) terms and are the opposite according to the sin($c_i t$) terms. Consequently, by using R_{t-j}, we enable the attention's value to differentiate between various distances and orientations.

Embedding Layer

To tackle the OOV challenge, we use the embedding of words and characters. In the literature, word-level embedding is tackled in most cases with pre-trained word embedding like a glove (Pennington *et al*., 2014) and fast text (Bojanowski *et al*., 2017). On the other hand, character-level representation is achieved through CNNs or Bi-LSTM (Khalifa and Shaalan, 2019).

Character Level Embedding

Currently, according to our best knowledge, character-level representations are extracted through either CNNs or Bi-LSTM (Gridach, 2016). Initially, vocabulary characters receive arbitrary embedding vectors. After that, a BiLSTM or CNN is applied to each word in the vocabulary to handle its characters one by one. Finally, the Bi-LSTM's result is the word's character representation. We used in this study BiLSTM for character embedding.

Word Level Embedding

Any word's meaning can be described in 40-300-dimensional space by a continuous tensor of real values that shows how the word is embedded. While unrelated words have dissimilar representations in this space, related words are represented by similar vectors. In this study, we use fast text (Bojanowski *et al*., 2017), which is an open-source pre-trained word embedding model maintained by Facebook, to encode the information of each word.

Encoding Layer

We adopt in this study a Bi-LSTM for encoding with an implementation detailed in Eqs. 16-19 respectively:

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \tag{16}$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \tag{17}$$

$$O_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o\right) \tag{18}$$

$$h_t = o_t \, tanh\left(c_t\right) \tag{19}$$

where, *c*, *f*, *i* and *o* represent the cell state, forget, input, and output gates respectively and all *b*s are biased. *σ* is the sigmoid activation function. To consider both the past and the future context of the input sequence, we utilized BiLSTM, which takes the same input and processes it in both directions (right-to-left and left-to-right) before combining the results to form the result. A Bi-LSTM is composed of a forward hidden layer and a backward hidden layer. More specifically, the forward hidden layer h_f^t processes the input sequence from left to right, i.e., *t* = 1, 2, 3, …, *T*, while the backward hidden layer h_b^t

processes the input sequence from right to left, i.e., $t = T, \ldots, 3, 2, 1$. Finally, h_f^t and h_b^t are combined to generate an output y_t. Bi-LSTM is implemented using Eqs. 20-22 respectively:

$$h_f^t = tanh\left(W_{xh}^f x_t + W_{hh}^f h_{t-1}^f + b_h^f\right) \tag{20}$$

$$h_b^t = tanh\left(W_{xh}^b x_t + W_{hh}^b h_{t-1}^b + b_h^b\right) \tag{21}$$

$$y_t = \left(W_{hy}^f h_t^f + W_{hy}^b h_t^b + b_y\right) \tag{22}$$

Multi-Head Attention Layer

In this layer, we implement the transformer encoder with a relative positional encoding network to consider both the directionality and distance features. Both residual and layer normalization of the transformer is also used.

Decoding Layer

We use a simple fully connected layer in the decoding layer and reduce the learnable parameters instead of using a recurrent unit as it is done in other sequence-to-sequence models.

Prediction Layer

When dealing with sequence labeling issues, like NER or POS labeling, it is essential to take advantage of dependencies among tags instead of making decisions about each tag on its own. To give just one example, it is more common for some other "PER" label to succeed a "PER" label as opposed to an "ORG" label to do so. It is possible to make use of such dependencies with the CRF suggested by Lafferty *et al.* (2001).

Given an input sequence $X = \{X_1, X_2, \ldots, X_n\}$, his corresponding labels or tags $y = \{y_1, y_2, \ldots, y_n\}$ has the score expressed in Eq. 23:

$$S(X, y) = \sum_{i=1}^{n} P_{i,yi} + \sum_{i=1}^{n-1} A_{yi,yi+1} \tag{23}$$

where, $P \in \mathbb{R}^{n\times k}$, k, $A_{i,j}$ are respectively the matrix of scores output by the word representation layer, number of possible labels or tags, in addition to the result of moving from label i to label j. For a given set of labels y, the likelihood is given by Eq. 24:

$$P(y \mid X) = \frac{e^{S(X,y)}}{\sum_{y' \in Y_X} e^{S(X,y)}} \tag{24}$$

Here, Y_X represents those potential sequence labels for the provided X. Equation 25 shows that the goal of the learning procedure is to make it more likely that the right labels come in the right order:

$$\log(X) = S(X, y) - \log \sum_{y' \in Y_X} e^{S(X,y)} \tag{25}$$

During the decoding phase, the Viterbi algorithm proposed in 1967 by Viterbi (1967) can be utilized to identify the tags' sequence with the highest value, Eq. 26:

$$y^* = argmax_{y' \in Y_X} S(X, y') \tag{26}$$

Experiments

In this section, we detail our experiment settings, the dataset that we used, the state-of-the-art baseline references that we use for comparison, and briefly define our evaluation metric.

Dataset

In this study, we used ANER corp (Benajiba *et al.*, 2008) and AQMAR dataset (Mohit *et al.*, 2012) which were proposed for Arabic NER tasks for training and testing. On one hand, the ANERCorp dataset is divided into four categories: Pers (39%), Loc (30.4%), Org (20.6%), and Misc (10%). IOB or (inside, outside, beginning) is the standard format of the dataset. In the dataset, if a token comes at the start of any named entity, then it must be labeled as B-label while I-label means that the token comes inside the named entity but not the first token within the named entity. If a token is neither at the start of a named entity nor inside a named entity, it is labeled as *O*. The dataset is composed of 316 articles where articles from different newspapers are chosen. These articles contain many topics enabling thus to create a generic dataset. There are over 150,000 tokens in the dataset and the three named entity classes available for labeling are Persons, Locations, and Organizations. On the other hand, (Mohit *et al.*, 2012) proposed the AQMAR dataset as an Arabic dataset for named entities, which is collected from Wikipedia. AQMAR consists of 74,000 tokens that are hand-labeled for named entities to ensure uniformity of labeling and that it is openly accessible for study needs. By merging these two datasets, 224, 286 tokens were obtained.

Experiment Settings

The following settings are used in our experiments. The maximum sentence length in the used dataset is equal to 212. Each word's features (word embedding dimension) were set to 300 while each character was represented by a vector of length equal to 48 (character embedding dimension). For output tuning, we added the L2 regularization component to our loss function. We used a 50% dropout to control the inputs of the encoder and avoid the overfitting problem. To optimize the cost of the network, we used Adam (Kingma and Ba, 2014) to learn the system with a 64-batch size. To train each model, we used a maximum number of epochs equal to 100. We set an initial learning rate to 1*e*-3 and progressively reduce it during the training with a decay factor of 0.95 until it reaches the minimum learning rate set to 5*e*-5. We employed an early stopping, with min delta and patience equal to 0 and 6 respectively.

Evaluation Metrics

F1-measure is the primary criterion for evaluating the effectiveness of different systems. This measure is contingent on two others: Recall (R) and Precision (P). (*R*), (*P*) and F1-measures can all be calculated with the help of Eqs. 27-29:

$$P = \frac{TP}{TP + FP} * 100 \quad (27)$$

$$R = \frac{TP}{TP + FN} * 100 \quad (28)$$

$$F1 = \frac{2 * P * R}{P + R} * 100 \quad (29)$$

The notations "*TP*," "*FP*," and "*FN*" stand for "true positive," "false positive," and "false negative," correspondingly.

Results and Discussion

Table 1 illustrates the experimental findings. We were able to get a result of 93.27 for the F1-measure by combining the data from ANER corp (Benajiba *et al*., 2008) and AQMAR (Mohit *et al*., 2012) with relative positional encoding. With the same settings and the same merged datasets, the absolute sinusoidal encoding achieved an F1-score of 92.40.

Our experiment results demonstrated that both the distance and directionality information are important for NER as we obtained significant improvement when adding directionality information to the vanilla transformer encoder.

We have chosen three baselines that have used the same data and same performance metrics and compared our proposal with them. More specifically, (Ali and Tan, 2019; Ali *et al*., 2019; Al-Smadi *et al*., 2020) were selected as baselines. Table 1 also displays findings from these baselines using the same experimental conditions. Out of these three baseline models, (Ali and Tan, 2019) are the most optimistic which achieves 92% of the F1-score while (Ali *et al*., 2019; Al-Smadi *et al*., 2020) achieved respectively an F1-score of 91 and 90%. Our proposed approach, therefore, outperformed such baselines on the Arabic NER.

Table 1: Experiment results

Model	F1-score
Ali and Tan (2019)	92.00
Ali *et al*. (2019)	91.00
Al-Smadi *et al*. (2020)	90.00
Our model with absolute sinusoidal	92.40
Our model with relative positional	93.27

Conclusion

In our article, we suggested a Transformer-based Arabic NER model. More specifically, a BiLSTM layer was utilized for encoding each input token of the sentence. Next, the multi-head attention layer receives the encoder's output where both the distance and directionality information are incorporated. The attention layer's results are passed into the decoder layer, which is a simple fully connected layer. Then the prediction layer with Conditional Random Fields (CRF) predicts the tag of each token. We validate our proposed approach on two merged public datasets, namely, ANER corp and AQMAR. The use of relative positional encoding enables our network to be aware of both the distance and the directionality. Our model used fast text to embed each word and a BiLSTM to embed each character. Our experiment results have demonstrated that the directionality feature is important when using the transformer for NER. We outperformed the vanilla transformer and some recent works that have been carried out using the same datasets and the same performance metrics.

In the future, we aim to rely on a fully connected self-attention architecture (a.k.a transformer) without an RNN variant for Arabic NER.

Acknowledgment

The authors greatly appreciate the thoughtful comments made by the manuscript's editors and reviewers.

Funding Information

This study paper was not awarded funds from public, commercial, or non-profit funding organizations.

Author’s Contributions

Hamid Sadeq Mahdi Alsultani: The paper background work, conceptualization, methodology, dataset pre-processing, implementation, result analysis and comparison, prepareds, edited and drafted.

Ahmed H. Aliwy: The reviewed of work and project administration.

Ethics

The corresponding author confirms that there are no ethical issues or conflict of interest.

References

AbdelRahman, S., Elarnaoty, M., Magdy, M., & Fahmy, A. (2010). Integrated machine learning techniques for Arabic-named entity recognition. *IJCSI*, *7*(4), 27-36.

Al-Ayyoub, M., Nuseir,A., Alsmearat, K., Jararweh, Y., & Gupta, B. (2018). Deep learning for Arabic NLP: A survey. *Journal of Computational Science*, *26*, 522-531. https://doi.org/10.1016/j.jocs.2017.11.011

Ali, B. A. B., Mihi, S., El Bazi, I., & Laachfoubi, N. (2020). A Recent Survey of Arabic Named Entity Recognition on Social Media. *Rev. d'Intelligence Artif.*, *34*(2), 125-135. https://doi.org/10.18280/ria.340202

Ali, M. N. A., Tan, G., & Hussain, A. (2019). Boosting Arabic named-entity recognition with multi-attention layer. *IEEE Access*, *7*, 46575-46582. https://doi.org/10.1109/ACCESS.2019.2909641

Ali, M. N., & Tan, G. (2019). Bidirectional encoder-decoder model for Arabic named entity recognition. *Arabian Journal for Science and Engineering*, *44*, 9693-9701. https://doi.org/10.1007/s13369-019-04068-2

Ali, M. N., Tan, G., & Hussain, A. (2018). Bidirectional recurrent neural network approach for Arabic named entity recognition. *Future Internet*, *10*(12), 123. https://doi.org/10.3390/fi10120123

Alotaibi, F., & Lee, M. (2014, August). A hybrid approach to features representation for fine-grained Arabic named entity recognition. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers* (pp. 984-995). https://aclanthology.org/C14-1093.pdf

Alsaaran, N., & Alrabiah, M. (2021). Arabic named entity recognition: A BERT-BGRU approach. *Comput. Mater. Contin*, *68*, 471-485. https://doi.org/10.32604/cmc.2021.016054

Al-Shoukry, S., & Omar, N. (2015). Proper nouns recognition in Arabic crime text using machine learning approach. *Journal of Theoretical & Applied Information Technology*, *79*(3).

Al-Smadi, M., Al-Zboon, S., Jararweh, Y., & Juola, P. (2020). Transfer learning for Arabic named entity recognition with deep neural networks. *IEEE Access*, *8*, 37736-37745. https://ieeexplore.ieee.org/abstract/document/8993806/

Bazi, I. E., & Laachfoubi, N. (2018). Arabic named entity recognition using word representations. *arXiv preprint arXiv:1804.05630*. https://doi.org/10.48550/arXiv.1804.05630

Benajiba, Y., & Rosso, P. (2007, December). ANERsys 2.0: Conquering the NER task for the Arabic language by combining the maximum entropy with POS-tag information. In *IICAI* (pp. 1814-1823). http://personales.upv.es/prosso/resources/BenajibaRosso_IICAI07.pdf

Benajiba, Y., Diab, M., & Rosso, P. (2008, October). Arabic named entity recognition using optimized feature sets. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing* (pp. 284-293). https://aclanthology.org/D08-1030.pdf

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135-146. https://doi.org/10.1162/tacl_a_00051

Cao, P., Chen, Y., Liu, K., Zhao, J., & Liu, S. (2018). Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (pp. 182-192). https://doi.org/10.18653/v1/D18-1017

Chiu, J. P., & Nichols, E. (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, *4*, 357-370. https://doi.org/10.1162/tacl_a_00104

Chu, C., & Wang, R. (2018). A survey of domain adaptation for neural machine translation. *arXiv preprint arXiv:1806.00258*. https://doi.org/10.48550/arXiv.1806.00258

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(ARTICLE), 2493-2537. https://www.jmlr.org/papers/volume12/collobert11a/collobert11a.pdf?source

Dahan, F., Touir, A., & Mathkour, H. (2015). First order hidden Markov model for automatic Arabic name entity recognition. *International Journal of Computer Applications*, *123*(7). https://doi.org/10.5120/ijca2015905397

Dong, C., Zhang, J., Zong, C., Hattori, M., & Di, H. (2016). Character-based LSTM-CRF with radical-level features for Chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications: 5th CCF Conference on Natural Language Processing and Chinese Computing, NLPCC 2016 and 24th International Conference on Computer Processing of Oriental Languages, ICCPOL 2016, Kunming, China, December 2-6, 2016, Proceedings 24* (pp. 239-250). Springer International Publishing. https://doi.org/10.1007/978-3-319-50496-4_20

El Bazi, I., & Laachfoubi, N. (2019). Arabic named entity recognition using deep learning approach. *International Journal of Electrical & Computer Engineering (2088-8708)*, *9*(3). https://doi.org/10.11591/ijece.v9i3.pp2025-2032

Etaiwi, W., Awajan, A., & Suleiman, D. (2017). Statistical Arabic name entity recognition approaches: A survey. *Procedia Computer Science*, *113*, 57-64. https://doi.org/10.1016/j.procs.2017.08.288

Farghaly, A., & Shaalan, K. (2009). Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, *8*(4), 1-22. https://doi.org/10.1145/1644879.1644881

Goyal, A., Gupta, V., & Kumar, M. (2018). Recent named entity recognition and classification techniques: A systematic review. *Computer Science Review*, *29*, 21-43. https://doi.org/10.1016/j.cosrev.2018.06.001

Gridach, M. (2016, December). Character-aware neural networks for arabic named entity recognition for social media. In *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP2016)* (pp. 23-32). https://aclanthology.org/W16-3703/

Guo, Q., Qiu, X., Liu, P., Shao, Y., Xue, X., & Zhang, Z. (2019). Star-transformer. *arXiv preprint arXiv:1902.09113*. https://doi.org/10.48550/arXiv.1902.09113

Helwe, C., & Elbassuoni, S. (2019). Arabic named entity recognition via deep co-learning. *Artificial Intelligence Review*, *52*, 197-215. https://doi.org/10.1007/s10462-019-09688-6

Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*. https://doi.org/10.48550/arXiv.1508.01991

Khalifa, M., & Shaalan, K. (2019). Character convolutions for Arabic named entity recognition with long short-term memory networks. *Computer Speech & Language*, *58*, 335-346. https://doi.org/10.1016/j.csl.2019.05.003

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Konstantinova, N. (2014). Review of relation extraction methods: What is new out there? In *Analysis of Images, Social Networks and Texts: 3rd International Conference, AIST 2014, Yekaterinburg, Russia, April 10-12, 2014, Revised Selected Papers 3* (pp. 15-28). Springer International Publishing. https://doi.org/10.1007/978-3-319-12580-0_2

Kuru, O., Can, O. A., & Yuret, D. (2016, December). Charner: Character-level named entity recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (pp. 911-921). https://doi.org/10.1109/ACCESS.2020.2973319

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*. https://doi.org/10.18653/v1/N16-1030

Li, J., Sun, A., Han, J., & Li, C. (2020). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, *34*(1), 50-70. https://doi.org/10.1109/TKDE.2020.2981314

Liu, X., Chen, H., & Xia, W. (2022). Overview of named entity recognition. *Journal of Contemporary Educational Research*, *6*(5), 65-68. https://doi.org/10.26689/jcer.v6i5.3958

Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*. https://doi.org/10.18653/v1/D15-1166

Melis, G., Dyer, C., & Blunsom, P. (2017). On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*. https://doi.org/10.48550/arXiv.1707.05589

Mohammed, N. F., & Omar, N. (2012). Arabic named entity recognition using artificial neural network. *Journal of Computer Science*, *8*(8), 1285. https://doi.org/10.3844/jcssp.2012.1285.1293

Mohit, B., Schneider, N., Bhowmick, R., Oflazer, K., & Smith, N. A. (2012, April). Recall-oriented learning of named entities in Arabic Wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 162-173). https://aclanthology.org/E12-1017.pdf

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532-1543). https://doi.org/10.3115/v1/D14-1162

Roy, A. (2021). Recent trends in named entity recognition (NER). *arXiv preprint arXiv:2101.11420*. https://doi.org/10.48550/arXiv.2101.11420

Shaalan, K. (2014). A survey of arabic named entity recognition and classification. *Computational Linguistics*, *40*(2), 469-510. https://doi.org/10.1162/COLI_a_00178

Shaalan, K., & Raza, H. (2008). Arabic named entity recognition from diverse text types. In *Advances in Natural Language Processing: 6th International Conference, GoTAL 2008 Gothenburg, Sweden, August 25-27, 2008 Proceedings* (pp. 440-451). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-85287-2_42

Shahina, K. K., Jyothsna, P. V., Prabha, G., Premjith, B., & Soman, K. P. (2019, March). A sequential labelling approach for the named entity recognition in Arabic language using deep learning algorithms. In *2019 International Conference on Data Science and Communication (IconDSC)* (pp. 1-6). IEEE. https://doi.org/10.1109/IconDSC.2019.8817039

Shaw, P., Uszkoreit, J., & Vaswani, A. (2018). Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*. https://doi.org/10.18653/v1/N18-2074

Sun, P., Yang, X., Zhao, X., & Wang, Z. (2018, November). An overview of named entity recognition. In *2018 International Conference on Asian Language Processing (IALP)* (pp. 273-278). IEEE. https://doi.org/10.1109/IALP.2018.8629225

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, *13*(2), 260-269. https://doi.org/10.1109/TIT.1967.1054010

Yan, H., Deng, B., Li, X., & Qiu, X. (2019). TENER: Adapting transformer encoder for named entity recognition. *arXiv preprint arXiv:1911.04474*. https://doi.org/10.48550/arXiv.1911.04474