Arabic Named Entity Recognition Using Transformer-based-CRF Model

Muhammad Al-Qurishi
Research Department / Elm Company
Riyadh, 12382-4182, Saudi Arabia

Riad Souissi
Research Department / Elm Company
Riyadh, 12382-4182, Saudi Arabia

Abstract

Named Entity Recognition is an essential component of Natural Language Progressing with countless practical applications. Several different directions of research are currently being pursued, exploring the possibilities in various ways. One particularly fruitful approach is the localization of multilingual deep learning tools based on the BERT architecture, with AraBERT and ARBERT/MARBERT serving as good examples. In this paper, we propose a simple but effective model for Arabic named entity recognition. The architecture of this model consists of three layers, as follows: a transformer-based language model layer, a fully connected layer, and the last layer is a conditional random field(CRF). Our proposed model shows promising results compared to the current state-of-art Arabic-NER models. For example, the F1-macro of the test data scores approximately 89.6% on the ANERCorp and 88.5% on the AQMAR datasets.

1 Introduction

Named Entity Recognition (NER) is an essential task that has numerous practical applications and enables successful performance of other NLP tasks. In the Arabic language, NER is additionally complicated by unique morphology and grammar, requiring specific methods to accurately label parts of text as entities from one of several possible classes.

Since Arabic is a widely spoken language with numerous dialects, there is an apparent need for automated linguistic tools that would serve the huge population of its speakers. At present time, the available Arabic NER tools are very scarce, only partially successful, and insufficiently tested, so new research in this field is urgently needed. Some of the notable NER tools for Arabic language based on machine learning include MADAMIRA by (Pasha et al., 2014), FARASA by (Abdelali et al., 2016), as well as the CAMeL tools suggested by (Obeid et al., 2020).

Discovering the most promising methodology that meets those requirements and allows for consistently accurate entity recognition would enable easier processing of the huge amount of information in Arabic that is already accumulated in various databases. This would in turn provide a boost in numerous research areas and practical applications of new technology, from semantic interpretation of multimedia content to searching for specific data in large data silos or the internet.

The idea of using automated algorithms for Natural Language Processing is several decades old, and has been successfully applied to many different problems. In particular, NER task has been the subject of many studies (Goyal et al., 2018; Li et al., 2020). Some of the earliest works of this kind are based on simple machine learning strategies, for example (Benajiba and Rosso, 2008; Oudah and Shaalan, 2012; Benajiba et al., 2010), while other proposed methods were based on grammatical rules and collections of named entities, i.e. (Khalil et al., 2020). More recently, models based on deep learning are emerging and achieving unprecedented success with a range of linguistic tasks including NER. In particular, BERT – Bidirectional Encoders Representations from Transformers model developed by (Devlin et al., 2018) has been extremely successful and inspired a number of regional variations that deal with specific languages. Some of the notable NER tools for Arabic language based on the BERT architecture include AraBERT and ARBERT/MARBERT serving as good examples.

In this paper we propose a simple but effective model for tagging Arabic named entities using transformer-based language model. Our solution architecture consists of three layers as follows. A transformer-based language model layer. In this layer we fine-tuned a pretrained language model Arabert which is the Arabic version of BERT. We also have fine-tuned and evaluated other models include AraElactra and XLM-Roberta. The second

layer is a fully connected linear layer which helps adjusting the output dimensions and initializing the inputs to the last conditional random field(CRF) layer. With CRF algorithm, tags are assigned based on the tag associated with the previous word in a linear fashion. Starting from the features extracted from the words, CRF estimates the probabilities that the word in question is a named entity or not. The problem is thus reduced to a simple probabilistic decision, while the inclusion of the previous word provides possibilities for capturing contextual clues. Due to its unique combination of simplicity and effectiveness, CRF is one of the most commonly used mathematical procedures used for NER tagging. Finally, we trained our model on two publicly available datasets- ANERCorp and AQMAR. The results of the conducted experiments were very promising and our proposed solution outperforms the current state-of-art in both datasets.

Our research paper organized as follows. We discuss the related studies in Section 2. In Section 3, we describe our solution in details. The experiments description is detailed in Section 4 and the results discussion is presented in Section 5. Finally, we conclude this work in Section 6.

2 Related Works

There are several different approaches to constructing NER tools, all of which are well represented in the modern literature and can be viewed as potentially viable in practice. The first group of solutions is based on simple machine learning strategies such as Support Vector Machines, Conditional Random Fields, or Random Forest, where the algorithm is trained on a labeled dataset and tasked with classifying new tokens based on statistical trends (Muhammad et al., 2020; Hudhud et al., 2021; Alshammari and Alanazi, 2020).

Deep learning methods employ a similar principle, but introduce vastly more complex architecture with far more sensitive capacity for capturing latent trends. Models based on bidirectional transformer stack architecture (such as BERT and its derivatives (Antoun et al., 2020; Abdul-Mageed et al., 2020)) have proven to be very promising, but a number of other designs including Pooled-GRU and CNN deserve to be examined. While deep learning systems are more powerful and accurate, they tend to have higher computational requirements and very long training times, which limits the extent of their practical usefulness (Helwe and Elbassuoni, 2019; Alkhatib and Shaalan, 2020; Al-Smadi et al., 2020).

Another group of solutions leverages certain grammatical rules to recognize named entities, for example using genitive rules to discover multi-word entities. It's also possible to construct knowledge bases that include gazetteers of named entities and semantic interpretation frameworks (ontologies) and use those resources to improve the recognition process (Elgamal et al., 2020; Khalil et al., 2020; Hudhud et al., 2021).

Since Arabic language presents additional challenges for NER algorithms due to its often ambiguous morphology and complex syntax rules, many works focused on this language introduce pre-processing operations and other modifications in order to optimize their tools to the nature of the task at hand, for example by standardizing the writing form for some words and removing diacritical marks (Pasha et al., 2014; Balla and Delany, 2020).

While most tools are trained with samples in Modern Standard Arabic, some of the works attempt to account for the diversity of local variations of spoken and written Arabic and improve performance with content that deviates from the formal language used in mainstream media and academic research. Incorporating data from social media (i.e. Twitter) is an ongoing research trend, contributing to diversification of training material and ultimately an increased capacity for recognizing named entities regardless of the form of writing. Typically, language-specific NER tools are expected to differentiate between several classes of entities, such as persons, organizations, and locations, although it is conceivable to have a much finer granularity and include a large number of classes. They should also be able to operate effectively with large quantities of unstructured data, which is characteristic for the modern online environment (Benali et al., 2021; Gridach, 2016; Zirikly and Diab, 2015).

All of the NER tools suggested in the reviewed works were empirically evaluated, and they typically displayed good overall performance during those tests. In general, the reported results are in the 85-95% range for accuracy and 60-83 for F1-macro score. They feature a limited amount of false positives and false negatives, with some fluctuation from one entity class to another. There is a trend that Arabic language tests yielded lower performance, mostly due to aforementioned complexities inherent in this language, but we have tried in this

paper to present an effective solution that can narrow this gap considerably. Since virtually no NER tools are completely error-free at this time, the differences between methods must be seen as the key to raising the expected level of performance. That's why even marginal improvements over state-of-the-art methods achieved through innovations are seen as very encouraging, providing some indications that with additional optimizations such innovations could lead to more tangible gains in the future.

3 Transformer-based-CRF Model

BERT and all of the derivative models feature an identical architecture which was directly inspired by Transformer (Vaswani et al., 2017). In all cases, there are two-layer stacks – decoder stack and encoder stack, each of which must contain one attention layer at the bottom. Self-attention mechanism in the decoder stack encodes the semantic relationships from the input sequence as attention scores and passes their normalized values to a series of forward-propagating layers. Conversely, in the encoder stack the representations are gradually refined with each new layer, until the correct output sequence can be produced. The number of layers and self-attention heads in the model can be variable, while the model is capable of processing semantic information in both directions, thus treating the entire sequence as a single connected unit.

This simple setup is now broadly accepted as the basis for advanced linguistic tasks, but it can be greatly improved through pre-training on certain supervised benchmark tasks, as well as fine-tuning of relevant model hyperparameters.

Our proposed architecture consists three layers, as shown in Figure 1. In the first layer, we fine-tuned three pretrained models, including AraBERT (Antoun et al., 2020), AraELECTRA (Antoun et al., 2021) and XLM-Roberta (Conneau et al., 2019) for Arabic Named Entity Recognition task.The second layer is a fully connected linear layer that receives the output of the pretrained model and reshapes it to be an input for the third layer. The final layer is the conditional random field (CRF), which is the tagging algorithm. The CRF makes sure that the objective of the model training is to return the most accurate combination of outgoing tags. We applied a dropout procedure in order to keep the training procedure well-balanced between entity classes.

3.1 Proposed Model Architecture

Figure 1 illustrates the proposed model architecture. If we have an input sentence x, it has to be tokenized before it is entered into the BERT model. In this process, the sentence x is padded to reach the maximum length of the sequence. Tokenized x with the corresponding attention mask is passed to the model, which outputs contextual embeddings of x. Transformer models including BERT use numerous separate attention mechanisms in each layer, in case of BERT base model, a total of 12 x 12 attention heads. In practice, this means that each token can be connected with 12 distinct features of any other token in the sequence.

There are two major aspects of BERT output crucial for accurate classification – prediction scores and hidden states. The prediction scores are obtained as the output of the last layer of the model, and thus represent the result of all attention heads in all layers and are relative to parameters such as batch size, hidden states size, and sequence length. Meanwhile, hidden states are the outputs of the individual layers of the model, and their total number is equal to the number of layers + 1 (12+1 for BERT). The output of one layer is immediately used as input for the next layer, which contextualizes its content further using its own attention heads. Thus, the prediction score basically represents a hidden state created by the final layer in the BERT model.

This output of the model can be understood in different ways. Intuitively, it seems logical that the last layer should contain all the information gained through different stages of learning, so its output should be seen as relevant regardless of the way the input vectors changed as they passed through the layers. On the other hand, it is quite possible that some of the vector modifications accidentally eliminated bits of useful information that could have contributed to and accurate prediction. To compensate for it, the input vectors can be partially or completely concatenated, or their sum could be used. We noticed that this procedure provides significant gains in terms of accuracy improvement, which is why we used this technique in our experiments.The model also includes a linear layer that helps reshape the output of BERT into 3 dimensions (batch size, number of tags, sequence length) and then pass it to a CRF layer tasked with making a prediction regarding the probabilities for each of the tags.

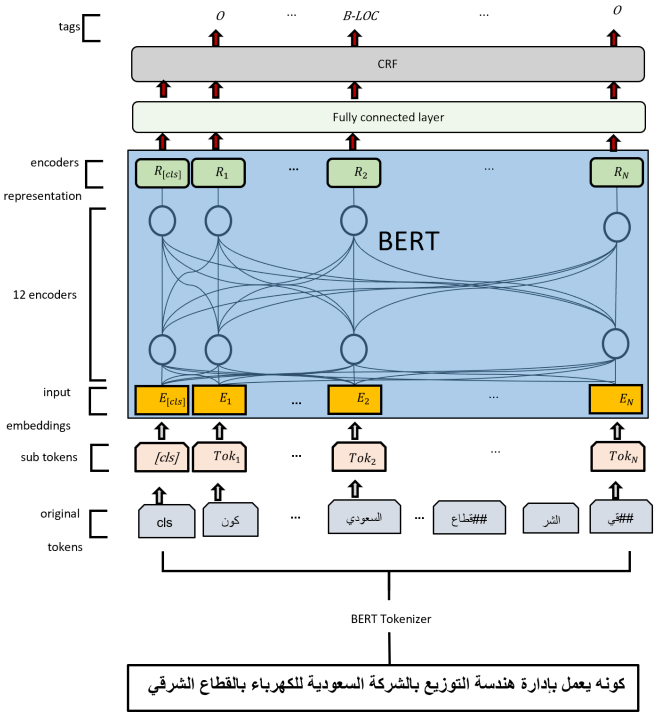

Figure 1: Basic layout of the model. Words are sent into a BERT-CRF model. Tokens are used to contextualize the word, to build the final representation

3.2 CRF Tagging Algorithm

Using transformer-based models, we get full distribution over the potential labels of the actual input possibilities. From this distribution, the classifier can predict the most likely class that the input could belong to. Named Entity Recognition meets this description, as the need to apply rules for semantic interpretation stretches it to a breaking point. That's because, I-PER can't match B-LOC under such conditions, effectively preventing the algorithm from maintaining its independence. Because of this, the selection of tags is performed using the conditional random field (CRF) method.

CRF method was originally publicized in the early 2000's (Lafferty et al., 2001), and has since been much discussed and broadly transferred to a number of different fields. This machine learning approach has proven to be useful in applications as diverse as genetic sequencing and linguistics. In recent years, models of this type are frequently used alongside LSTM-Long Short Term Memory networks to obtain the highest performance possible. This is especially noticeable in the natural language processing field, where this combination is increasingly becoming a standard due to its ability to raise the accuracy level for all tasks where tagging of token sequences is required. Classification of token sequences has the ultimate goal to determine the likelihood that a particular sequence belongs to the class Y from incoming data in vector form X. This section will present a simple type of conditional random field known as the linear chain conditional random field (Larochelle, 2021). We motivate the use of conditional random fields in the context of named entity classification where we want to be able to jointly model the full sequence of labels y associated with a sequence of inputs X as follows: For a given training set $\{(X, y)\}$, where $X = [x_1, ..., x_k]$ is a set of inputs and $y = [y_1, ..., y_k]$ is a target sequences, we calculate the conditional probability $p(y|X)$ as follows:

$$
\begin{aligned}
p(y|X) &= \prod_{i=1}^{k} p(y_i|x_i) \\
&= \frac{\prod_{i=1}^{k} \exp(E(x_i, y_i))}{Z(x_i)} \\
&= \frac{\sum_{i=1}^{k} \exp(E(x_i, y_i))}{\prod_{i=1}^{k} Z(x_i)}
\end{aligned}
\tag{1}
$$

Form equation 1, for a pair (X, y), the normal classification task can be resolved by calculating $P(y|X)$ in the way of multiplication of individual

probabilities for all tokens in the sequence up to the position i, where the value of i is greater than one but lower than the total length of the sequence k. Normalization with exponential function is used in this model, rather than the softmax function that is commonly deployed in the same role in most deep learning models.

Two central variables in this equation of the model are marked by E and Z, with the following definitions:

$E(x, y)$ denotes the emission score and represents the score assigned for the class y based on the vector x after i iterations. In other words, it represents the output of a BERT model after i steps are completed. While the input vector can contain practically any type of information, it is typically populated by a combination of nearby tokens, i.e. words or sentence representations. Each score is assigned its relative weight based on the results of training BERT model.

$Z(x)$ refers to the partition function, and can be viewed as a specific type of normalization function as it serves to discover a distribution of probabilities. All probabilities for different classes must always add up to 1. In this sense, it is a component of the softmax activation and can be calculated as follows.

$$Z(X) = \sum_{y_1^{'}} \sum_{y_2^{'}} ... \sum_{y_i^{'}} ... \sum_{y_k^{'}} \exp(\sum_{i=1}^{k} E(x_i, y_i^{'}) + \sum_{i=1}^{k-1} V(y_i^{'}, y_{i+1}^{'})) \quad (2)$$

Due to numerous loops in the model, calculating the value of $Z(X)$ is not simple. This requires considering all iterations of the input for each step in the model, necessitating k repetitions of all calculations to get the value for the entire set.

While the complexity of this procedure is very high $O(|y|^k)$, it's feasible to use the recurrent properties of the model and decrease the computational requirements. This can be accomplished with the forward/backward algorithm, which is capable of processing the sequence in either direction. Once this parameter is determined, the CRF implementation can be undertaken.

Above is described a standard model that calculates probabilities for each class, but we need to expand it by introducing ponders that can be adjusted through learning. Basically, this means the possibility of following up the label y_i with y_{i+1} can be quantified, linking the neighboring labels to each other, which is why this variation is named Conditional Random Field with a linear chain. All probabilities are thus factored with $P(y_{i+1}|y_i)$, using the exponential function to reformat this value as an emission score $E(x, y)$ expanded by the transition score $V(y_i, y_{i+1})$ as follows.

$$p(y|X) = \frac{\exp(\sum_{i=1}^{k} E(x_i, y_i) + \sum_{i=1}^{k-1} V(y_i, y_{i+1}))}{Z(X)} \quad (3)$$

Parameter $V(y_i, y_{i+1})$ is defined as a matrix consisting of elements obtained through learning while the model transitions from the position i in the sequence to the position $i + 1$ in the same sequence. on another words, it shows the chance that y_{i+1} succeeds after y_i.

3.3 Calculating the NLL Function

With every classification task, a crucial concern is to keep the errors to a minimum while the model is being trained with input data. A common way to achieve this goal is to use a loss function (L), and feed model predictions into it along with the accurate labels. This function has two possible outputs, 0 or a value grater than 0, depending on whether the two input values match or not. When probabilities $P(y|X)$ are calculated, it is obviously important to eliminate erroneous predictions. This problem can be solved by using the negative log value of the probability of error. This quantity is often referred to as the loss based on negative log probabilities, or NLL loss. It can be summarized with the formula $L = -log(P(y|X))$, and modulated with different types of log properties as follows.

$$\begin{aligned} &-log(P(y|X)) = \\ &-log(\frac{\exp(\sum_{i=1}^{k} E(x_i, y_i) + \sum_{i=1}^{k-1} V(y_i, y_{i+1}))}{Z(X)}) \\ &= log(Z(X)) - log(\exp(\sum_{i=1}^{k} E(x_i, y_i) \\ &+ \sum_{i=1}^{k-1} V(y_i, y_{i+1}))) \\ &= log(Z(X)) - (\sum_{i=1}^{k} E(x_i, y_i) + \sum_{i=1}^{k-1} V(y_i, y_{i+1})) \end{aligned} \quad (4)$$

The quantity $log(Z)$ denotes the logarithmic value calculated while the partition was per-

formed (Larochelle, 2021). This value is very useful for the implementation of the forward algorithm. NLL loss represents the forward pass, and can be calculated by reversing the sign before a normal value of log probabilities. Those probabilities can be obtained by calculating all the scores based on the partition and determining the difference between them. This procedure can be made significantly more efficient by the use of a mask matrix, which allows the model to skip any operations that refer to non-essential elements.

4 Experiment

In this section, we will evaluate the training techniques used to improve the algorithm, as well as its performance with different tasks and objectives, and the relationship between architecture of the proposed model and quality of the output.

4.1 Tagging Types

The ultimate objective of NER procedure is to associate a label belonging to a particular class to each included word. It's important to note that some complex named entities could stretch across multiple words, but are always contained in a single sentence. The predominant sentence representation form used in this field is IOB, with words that start a name of an entity marked with B, internally located words marked as I, and other tokens marked with O.

4.2 Data Samples

The model was evaluated using two Arabic public available datasets including ANERcorp and AQMAR. They are formulated specifically for the Arabic NER task.

4.2.1 ANERcorp Dataset

ANERcorp is a high-quality, annotated dataset containing Arabic language content gathered from a variety of publically available media sources. It was created in 2008 and has since been widely accepted as one of the standard datasets used for a variety of linguistic tasks. There are four classes of named entities included in this set, namely Persons, Locations, Organizations, and Miscellaneous. Based on those classes, the dataset includes a number of tags that pertain to named entities, including: *B-PERS*: Beginning of the name of a PERSon. *I-PERS*: Continuation (Inside element) of the name of a PERSon. *B-LOC*: Beginning of the name of a LOCation. *I-LOC*: Inside element present within the name of a LOCation. *B-ORG*: Beginning of the name of an ORGanization. *I-ORG*: Inside element present within the name of an ORGanization. *B-MISC*: Beginning of the name of an entity which doesn't belong to any of the previous classes (Miscellaneous). *I-MISC*: Inside element present within the name of a miscellaneous entity *O*: The word is not a named entity (Other).

Source	Ratio %
http://www.aljazeera.net	34.8
http://www.raya.com	15.5
http://ar.wikipedia.org	6.6
http://www.alalam.ma	5.4
http://www.ahram.eg.org	5.4
http://www.alittihad.ae	3.5
Other newspapers and magazines	17.8

Table 1: Overview of sources of articles container in ANERcorp dataset

There are a total of 316 articles within the ANERcorp dataset, all taken from journalistic sources and online publications. An overview of the content of the dataset regarding the sources, expressed in percentages as in Table 1.

In total, there are more than 150,000 tokens of more than 32,000 types within this dataset, with an average of 4.67 tokens per type. 11% of the content consists of proper names. In collaboration between the original creator of the corpus Yassine Benajiba and researchers from CAMeL Lab and Mind Lab, the dataset was slightly revised in 2020 to correct some imperfections and make it better suited for the type of studies it is needed for. Some of the corrections agreed upon involved spelling errors, blank Unicode characters and diacritical marks. At this time, the dataset was also divided into a training portion (125K words) and testing portion (25K words) to facilitate even better performance. In this study, the latest version of the ANERcorp corpus was used.

4.2.2 AQMAR Dataset

AQMAR (American and Qatari Modeling of Arabic) is a relatively small Arabic language dataset created specifically for the purpose of natural language processing evaluation, including named entity recognition (NER). It was created in collaboration between the US-based Carnegie Mellon University and Qatari governmental institutions, and is widely used in projects of various types.

This dataset consists of more than 3000

sentences taken from around 30 representative Wikipedia articles in Arabic, touching on a wide range of topics from history and science to politics and sports. This dataset was manually annotated with standard four NER classes – persons, locations, organizations, and miscellaneous, in addition to other many tags pertaining to sentiments, relations, etc. Total number of tokens contained in the AQMAR dataset is at 74,000, providing more than enough variability for NLP research purposes.

4.3 Fine-tuning Process

Our proposed model was trained and tested on both ANERCorp and AQMAR datasets, with the idea of fine-tuning the hyperparameters in every iteration. Hyperparameters are set up to create the best possible conditions for recognizing named entities in a labeled dataset based on the words found in the input sequence as shown in Table 2. When the dataset is used for model training and testing, individual sentences from unseen articles are selected at random and fed into the model as input. 80% of the dataset is typically used for training, 10% as a validation set, and the remaining 10% to evaluate the performance of the model on unseen examples.

Both AQMAR and ANERCorp datasets are very useful in model evaluation due to their versatility and the relevance of the content, making it a logical choice to include in this study. We have fine-tuned three pretrained models, including AraBERT, AraElectra, and XLM-Roberta. In the first experiment, we fine-tuned AraBert V2, the large model containing 24 layers of encoders stacked on top of each other, 16 self-attention heads, and a hidden size of 1024. In the second experiment, we used only the discriminator base model for AraElactra. This model has 12 attention heads, 12 hidden layers, and 768 hidden states size. In the third experiment, we used XLM-Roberta, a model with 12 attention heads, 12 hidden layers, and 768 hidden states.

During the three experiments we used an AdamW optimizer, which is useful when fine-tuning a pre-trained model as frozen layers. In all experiments, the learning ratio was $5e-5$, and the number of epochs was 5. The model input is a sequence of tokens that are processed to two vectors, input IDs, and attention masks using the BERT tokenizer. The tested sequence lengths were 128, 256, and 512 tokens. Dropout optimization was applied where a dropout step was introduced immediately before the data reaches the linear layer. We used the same dropout ratio of 0.1.

5 Results

In Table 3, the output of the proposed model is compared with state-of-the-art NER models for the Arabic language. Since the proposed solution doesn't use any sources other than the training sample, the performance of the competing methods was presented without access to such sources for the sake of fair evaluation. The AraBertv2 + CRF model achieved the most impressive result; the F1-macro score of almost 89.6% for this model concatenates the last 6 hidden layers. However, this model achieved these results on a sequence length of 256 tokens. Therefore, we applied the same model on a 512 token length, and we found that the best result is attained when we sum the 11 hidden layers; the F1-macro has not significantly change. In general, all the fine-tuned models outperform the state-of-the-art models, with significant improvements by almost 5% more than the best one, as we can see in Table 3. Tables 4- 6 show the experimental results of our proposed model with respect to the AQMAR dataset, using AraBert and AraElectra models, respectively.

We also tested our model on CANERCorpus a Classical Arabic Named Entity Recognition Corpus that was built by (Salah and Zakaria, 2018). This dataset was used by (Alsaaran and Alrabiah, 2021) and we compare our proposed model against theirs as shown in table 5. CANERCorpus was compiled starting from more than 7,000 hadiths - religious texts written by Islamic scholars that include mentions of a large number of named entities, totaling around 250 thousand words. There were approximately 13,000 named entities identified in the reviewed texts, and they were separated in 20 different classes based on the general category they are related to. In the pre-processing stage (Salah and Zakaria, 2018), the texts were segmented into sentences before they were annotated by three human operators, with majority opinion taken as valid in cases when there was disagreement. Words were annotated in the IOB2 format, which determines whether they are included in the beginning, middle, or the end of the entity name. This dataset is notable for very fine granulation with a lot of unique classes related to Islamic tradition (i.e. Allah, Prophet, Clan) in addition to standard NER classes such as person, time, or organization. For this reason, the dataset provides a strong foundation

Parameter	AraBERT	AraElectra	XLM-Roberta
Max sequence length	[128,256,512]	[128,256,512]	[128,256,512]
No. heads	16	12	12
No. hidden layers	24	12	12
Hidden layer size	1024	768	768
Batch size	4	16	4
Vocab size	64000	64000	250002
loss	Crf loss	Crf loss	Crf loss
Dropout prob	0.1	0.1	0.1
Optimizer	AdamW	AdamW	AdamW
Learning rate	5e-5	5e-5	5e-5
Number of epochs	5	5	5

Table 2: Table of hyperparameters for training the proposed Arabic NER model

Model	Seq	F1-Macro	Accuracy	Precision	Recall	F1-measure
AraBertv1	512	0.82767	0.971329	0.83902	0.816630	0.842
mBERT	512	0.76721	0.96293	0.79244	0.74354	0.784
gigaBERT	512	NA	0.969889	0.82820	0.812253	0.82015
AraBertv2	512	0.8043	0.97004	0.82900	0.81050	0.81965
Mawdoo3	512	NA	0.964331	0.79183	0.755798	0.77339
MARBERT	512	NA	0.966730	0.81267	0.774617	0.79318
ARBERT	512	NA	0.97224	0.84656	0.82582	0.83606
AraBertv2 + CRF (last)	512	0.88888	0.9916173	0.905367	0.912455	0.90889
AraBertv2 + CRF (concat 6)	256	**0.8956**	**0.9919526**	**0.913135**	**0.91378**	**0.91345**
AraBertv2 + CRF (concat 9)	512	0.8933	0.9916918	0.910310	0.913536	0.91192
AraBertv2 + CRF (sum 11)	512	**0.8946**	**0.9917663**	**0.908192**	**0.915954**	**0.91205**
AraElectra +CRF (concat 5)	512	0.872115	0.98755	0.9104595	0.894163	0.9022379
XLM-Roberta +CRF (concat 9)	256	0.875697	0.98968	0.90608	0.90181	0.90181

Table 3: The performance of the proposed Arabic NER model vs state of the art on ANERCorp

for testing Arabic language AI tools with sophisticated NLP capacities.

Thus, it's fair to say that stack architecture of the transformer-based models in combination for conditional random field currently represents one of the most successful NER methodologies that are independent from external sources. Its surprising performance can be explained by the strength of contextualization unique for this approach, which allows it to be accurate even when only limited information is available.

6 Conclusion

In some studies, scalability and versatility of the NER solution were considered alongside accuracy, reflecting the objective to create tools that could be used in practice without too many limitations. In this study, a simple architectural design of transformer-based model was presented, created specifically for tagging of word sequences in the context of Named Entity Recognition. This solution outperforms most of the state-of-the-art methods for this task, including those that rely on knowledge bases. A crucial element of the proposed solutions is their ability to track interdependencies between labels. This can be done with a CRF layer. Also, the process of summing and concatenating vectors has been shown to be effective in generating additional information that helps improve the tagging accuracy. Since vector representations are made on the words level, the model is able to collect contextual clues related to both syntax and morphology. The work in the future will be in two parts as follows. The first is to work on disambiguate the label when there is a word that expresses two different entities, such as the name of a place and a person at the same time. The second is to increase the number of named entity classes as well as working on our own dataset.

Model	#state	Seq	F1-Macro	Accuracy	Precision	Recall	F1-measure
last	1	128	0.862648	0.984333	0.8567	0.879594	0.8680425
concat	12	128	0.88176	0.985708	0.87530	0.895202	0.8851435
sum	8	128	0.88222	0.984944	0.864197	0.88832	0.876095
last	1	256	0.852139	0.983815	0.848448	0.878862	0.8633879
concat	6	256	0.875361	0.985300	0.866348	0.888616	0.8773413
sum	9	256	0.8772	0.98478	0.85322	0.8982412	0.875
last	1	512	0.86588	0.984261	0.8651	0.8809234	0.8729
concat	10	512	0.87350	0.98500	0.87350	0.8766467	0.875
sum	11	512	0.87789	0.985374	0.871121	0.8902439	0.88

Table 4: The performance of the proposed Arabic NER model using bert-large-arabertv2+crf test results on AQMAR dataset

Model	Seq	F1-Macro	Precision	Recall	F1-measure
(Alsaaran and Alrabiah, 2021)	54	NA	94.10	95.54	94.76
AraBertv2 + CRF (concat 6)	54	**91**	**98.11**	**98.42**	**98.26**

Table 5: The performance of the proposed Arabic NER model using bert-large-arabertv2+crf test results on CANERCorpus dataset

Acknowledgements

This work was supported by the Research Department in Elm Company under the Arabic language processing initiative.

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Demonstrations*, pages 11–16.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2020. Arbert & marbert: deep bidirectional transformers for arabic. *arXiv preprint arXiv:2101.01785*.

Mohammad Al-Smadi, Saad Al-Zboon, Yaser Jararweh, and Patrick Juola. 2020. Transfer learning for arabic named entity recognition with deep neural networks. *Ieee Access*, 8:37736–37745.

Manar Alkhatib and Khaled Shaalan. 2020. Boosting arabic named entity recognition transliteration with deep learning. In *The thirty-third international flairs conference*.

Norah Alsaaran and Maha Alrabiah. 2021. Classical arabic named entity recognition using variant deep neural network architectures and bert. *IEEE Access*, 9:91537–91547.

Nasser Alshammari and Saad Alanazi. 2020. The impact of using different annotation schemes on named entity recognition. *Egyptian Informatics Journal*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2021. Araelectra: Pre-training text discriminators for arabic language understanding. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 191–195.

Husamelddin AM Balla and Sarah Jane Delany. 2020. Exploration of approaches to arabic named entity recognition. In *CLEOPATRA@ ESWC*, pages 2–16.

Yassine Benajiba and Paolo Rosso. 2008. Arabic named entity recognition using conditional random fields. In *Proc. of Workshop on HLT & NLP within the Arabic World, LREC*, volume 8, pages 143–153. Citeseer.

Yassine Benajiba, Imed Zitouni, Mona Diab, and Paolo Rosso. 2010. Arabic named entity recognition: using features extracted from noisy data. In *Proceedings of the ACL 2010 conference short papers*, pages 281–285.

Brahim Ait Benali, Soukaina Mihi, Ismail El Bazi, and Nabil Laachfoubi. 2021. New approach for arabic named entity recognition on social media based on feature selection using genetic algorithm. *International Journal of Electrical and Computer Engineering*, 11(2):1485.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised

Model	#states	Seq	F1-Macro	Accuracy	Precision	Recall	F1-measure
last	1	128	0.869587	0.978834	0.875871	0.86620689	0.871012482
concat	10	128	0.865022	0.97758	0.86192	0.8583333	0.860125261
sum	2	128	0.859811	0.978417	0.853556	0.8547486	0.854152128
last	1	256	0.862752	0.978032	0.864902	0.8758815	0.870357393
concat	2	256	0.875771	0.978344	0.87883	0.87638888	0.877607788
sum	3	256	0.861978	0.978552	0.864902	0.8601108	0.8625
Last	1	512	0.872761	0.979593	0.862116	0.8792613	0.870604781
concat	10	512	0.862209	0.977719	0.86908	0.85714285	0.863070539
Sum	2	512	0.8559	0.977719	0.866295	0.859116	0.862690707

Table 6: The performance of the proposed Arabic NER model using araelectra-base-discriminator +crf test results on AQMAR dataset

cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Marwa Elgamal, Mohamed Abou-Kreisha, Reda Abo Elezz, and Salwa Hamada. 2020. An ontology-based name entity recognition ner and nlp systems in arabic storytelling. *Al-Azhar Bulletin of Science*, 31(2-B):31–38.

Archana Goyal, Vishal Gupta, and Manish Kumar. 2018. Recent named entity recognition and classification techniques: a systematic review. *Computer Science Review*, 29:21–43.

Mourad Gridach. 2016. 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)*, pages 23–32.

Chadi Helwe and Shady Elbassuoni. 2019. Arabic named entity recognition via deep co-learning. *Artificial Intelligence Review*, 52(1):197–215.

Mohammad Hudhud, Hamed Abdelhaq, and Fadi Mohsen. 2021. Arabianer: A system to extract named entities from arabic content. In *ICAART (1)*, pages 489–497.

Hussein Khalil, Taha Osman, and Mohammed Miltan. 2020. Extracting arabic composite names using genitive principles of arabic grammar. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(4):1–16.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Hugo Larochelle. 2021. lectures on conditional random fields.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Marwa Muhammad, Muhammad Rohaim, Alaa Hamouda, and Salah Abdel-Mageid. 2020. A comparison between conditional random field and structured support vector machine for arabic named entity recognition.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. Camel tools: An open source python toolkit for arabic natural language processing. In *Proceedings of the 12th language resources and evaluation conference*, pages 7022–7032.

Mai Oudah and Khaled Shaalan. 2012. A pipeline arabic named entity recognition using a hybrid approach. In *Proceedings of COLING 2012*, pages 2159–2176.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *Lrec*, volume 14, pages 1094–1101. Citeseer.

Ramzi Esmail Salah and Lailatul Qadri Binti Zakaria. 2018. Building the classical arabic named entity recognition corpus (canercorpus). In *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*, pages 1–8. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ayah Zirikly and Mona Diab. 2015. Named entity recognition for arabic social media. In *Proceedings of the 1st workshop on vector space modeling for natural language processing*, pages 176–185.