Arabic Fine-Grained Named Entity Recognition Using Deep Learning

Mohamed Orfy
Computer and Communication
Faculty of Engineering, Alexandria University
mohamedehaborfy@gmail.com

Rahma Abdelhamid
Computer and Communication
Faculty of Engineering, Alexandria University
rahmarizk410@gmail.com

Mazen Gaber
Computer and Communication
Faculty of Engineering, Alexandria University
mazen9gaber@gmail.com

Salma Yousry
Computer and Communication
Faculty of Engineering, Alexandria University
salmaelzohari3@gmail.com

***Abstract*— This paper presents a comprehensive study on fine-grained Arabic named entity recognition (Flat NER) using various deep learning approaches. The researchers explored and evaluated several model architectures, including pre-trained transformer models like AraBERT and XLM-RoBERTa, as well as custom-built Bidirectional LSTM models.**

I. INTRODUCTION

The field of Arabic named entity recognition (NER) has gained significant attention in recent years due to the growing demand for robust and accurate text processing tools in the Arabic-speaking world.

NER is a fundamental task in natural language processing that aims to identify and classify named entities, such as persons, organizations, locations, and other relevant concepts, within unstructured text.

Accurate NER is essential for a wide range of applications, including information extraction, question answering, and knowledge graph construction. However, the unique linguistic characteristics of the Arabic language, such as complex morphology, ambiguity, and the presence of dialects, pose considerable challenges for developing effective NER systems.

The following figure illustrates an example of the Arabic NER:

This study focuses on addressing these challenges by exploring and evaluating various deep learning approaches for fine-grained Arabic NER, leveraging state-of-the-art transformer-based models and custom-built architectures.

II. DATASET

We conducted our experiments on a new version of the Wojood corpus, named Wojood-Fine. The Wojood-Fine corpus consists of two main subsets: Wojood-Fine Flat train dataset and validation dataset. The corpus contains 550K tokens, 75K entity mentions covering the parent types, The train dataset constitutes 70% of the entire corpus, while the validation dataset constitutes 10%.

For final evaluation and testing purposes, we reserved a separate test set comprising 20% of the corpus, which does not include any labels. The data was annotated for 92 entity types with IOB tags in train set and 84 entity types in the validation set, where each token in the text is assigned one of three tags: O (Outside), B-Entity (Beginning of Entity), or I-Entity (Inside Entity).

The Arabic named entity recognition (NER) task requires careful preprocessing of the input text data to improve the performance of the NER model. The dataset utilized in this task was provided in the form of CoNLL files, containing tokenized text along with corresponding labels.To ensure the integrity of the dataset, duplicate token-label pairs were removed. This step involved creating a set to keep track of unique token-label combinations encountered during the extraction process. To prepare the dataset for further analysis, several preprocessing steps were undertaken:

1.Stop words Removal

Common stop words in Arabic, such as conjunctions, prepositions, and pronouns, were removed from the text.That was executed using the NLTK library, supplemented by manual addition of specific stopwords to tailor the preprocessing to the context of the dataset. This helps to focus the NER model on the more informative words in the input.

2. Diacritic and Elongation Removal

The removal of diacritics and elongations from the text was facilitated by the PyArabic library. Using its functionality, the **strip_tashkeel** method was employed to eliminate diacritics, while the **strip_tatweel** method efficiently removed elongation characters. These preprocessing steps contribute to text normalization, aiding in improving the accuracy of the model training.

3. Stemming

Words were stemmed to reduce them to their morphological roots, a process essential for normalizing input text and capturing semantic relationships among related words. The FarasaStemmer, an Arabic-specific stemming algorithm, was employed for this purpose, further enhancing the dataset's consistency and facilitating more effective analysis.Also, The Arabic definite article "ال" and the conjunction "و" were removed during the stemming process from the beginning of words, as these are very common and do not contribute directly to named entity recognition.

4. English words Removal

Any remaining English words in the Arabic text were identified and removed, as the NER model is designed to work on Arabic text.

5. Lemmatization

ISRIStemmer from NLTK was used for lemmatization but it's performance didn't meet the expectations

6. Removing non AlphaNumeric characters

non-alphanumeric characters were removed from tokens for data consistency, then filtered out empty tokens for integrity, ensuring a clean dataset.

III. Model Approaches

To tackle the Fine-Grained ArabicNER task, several different model architectures were explored and evaluated. For each model. The input text underwent preprocessing steps described in the previous section before being fed into the model. The models were trained on the labeled dataset (token, tag) and the performance was evaluated as well.

The approaches are categorically divided into 2: Pretrained transformer models and Implemented models from scratch (Bidirectional LSTM).

Diving through the models tested and implemented:

A. AraBERT

AraBERT is a transformer-based language model specifically tailored for the Arabic language, is trained on a large corpus of Arabic text, enabling it to capture complex linguistic patterns and semantic nuances inherent to Arabic.Our model serves as a bridge between the pre-trained AraBERT v02 transformer model and task-specific fine-tuning for fine-grained NER tasks,it's Initialized with a pre-trained AraBERT v02 weights Before feeding the data into the model, we preprocess it to ensure compatibility with AraBERT's input requirements.

This preprocessing step involves tokenization using AraBERT's tokenizer, where each text sequence is split into tokens and encoded into numerical representations.

Our modelis extended by adding two bi -LSTM layers , which operate on top of the AraBERT embeddings to capture sequential dependencies within input sequences. A dropout layer is applied to regularize the model and mitigate overfitting, followed by a fully connected layer responsible for fine-grained NER classification.

During training, we iterate over multiple epochs, optimizing themodel parameters using the Adam optimizer with a learning rate of 2e-3. The training process involves monitoring both training and validation accuracy to assess model performance iteratively. A learning rate scheduler is employed to adjust thelearning rate dynamically based on the epoch loss, facilitating better convergence

B. XLM-RoBERTa

XLM-RoBERTa model is a multilingual language model developed by Hugging Face. It is an extension of the RoBERTa model which was originally trained on English text. However, it was also trained on texts from 100 different languages, which was consequently chosen as an approach for the task.

Since the main task is token classification. The model was allowed to learn to identify and classify different entities and concepts within the Arabic text.

Within training the model on the split dataset (training and validation), a technique was applied which was preferably called 'Dump and Load'.

Dump and Load method heavily relies on *pickle* library from python. It was applied by dividing the dataset into subset samples then batches, training the model on these samples, then dumping the model into a pickle file for later use on the rest of the samples.

This method preserved a lot of effort in case of any environmental circumstances just like the power cut. It was also helpful in case of early detection of overfitting and hence fixing the model parameters.

Primarily, the subset size was set to 5000 samples (nearly 0.1 of the dataset size) and later it was readjusted to be 10,000 samples for further training. Equivalently, the validation set was divided into subsets of 1000 samples and was later resetto 2000 samples.

Moreover, Mixed precision training was leveraged via the PyTorch autocast functionality. It allows the model to utilizeboth 16-bit and 32-bit floating point representations during the training process which significantly improved the training efficiency and speed. This optimization technique was particularly beneficial for the large model used.

C. GPT-2

The Generative Pre-trained Transformer 2 (GPT-2) model stands as a cornerstone in contemporary natural language processing research, developed by OpenAI. Representing a remarkable advancement in language modeling, GPT-2 belongs to the family of transformer-based architectures, renowned for their efficacy in capturing long-range dependencies in sequential data.

Unlike traditional models that rely on task-specific architectures or supervision, GPT-2 adopts a self-supervised learning paradigm, where it learns to predict the next word in a sequence based solely on the preceding context.

Through unsupervised pre-training on vast corpora of text data, GPT-2 acquires a deep understanding of syntactic and semantic structures inherent in natural language. Moreover, GPT-2 exhibits remarkable capabilities in generating coherent and contextually relevant text, a testament to its ability to grasp intricate linguistic nuances. Given its prowess, GPT-2 serves as a foundational tool across a spectrum of natural language processing tasks, including text generation, summarization, and sentiment analysis.

In our study, we propose a novel approach for Arabic NER leveraging a combination of word-level and character-level embeddings, along with a pre-trained GPT-2 model. Word embeddings, learned during training using Keras' Tokenizer, capture semantic similarities between words in a high-dimensional space. Simultaneously, character-level embeddings decompose words into constituent characters, providing morphological information and handling out-of-vocabulary words.

These embeddings are concatenated and passed through a pre-trained GPT-2 model, which generates contextualized representations capturing rich semantic information from the input text. The model architecture involves pooling the GPT-2 output using global average pooling to obtain a fixed-size representation of the input sequence.

Finally, a fully connected layer with softmax activation predicts the probabilities of different named entity classes. Our approach capitalizes on both word-level and character-level information, augmented by contextual embeddings from a pre-trained language model, to enhance the accuracy and robustness of Arabic NER. Through rigorous training and evaluation, we demonstrate the effectiveness of our model in accurately identifying named entities in Arabic text, contributing to the advancement of NER techniques for Arabic language processing.

D. Bidirectional LSTM

LSTMs are a type of Recurrent Neural Network (RNN) address the vanishing gradient problem that hinders traditional RNNs in capturing long-term dependencies within sequences.

BiLSTMs take LSTMs a step further by processing the sequence in both the forward and backward direction, allowing the model to capture contextual information from both sides of a word, leading to a more comprehensive understanding of its meaning. Unlike English, Arabic is written from right to left. This approach accounts for this by processing the text in both directions.

A typical BiLSTM architecture for Arabic NER involves the following steps:

a) Word Embeddings: Words are converted into numerical vectors (embeddings) that capture their semantic meaning and relationships.

b) BiLSTM Layer: The preprocessed text sequence is fed into the BiLSTM model.

c) Output Layer: The final layer predicts the most likely NER tag for each word in the sequence.

Our model leverages pre-trained word embeddings(GloVe). They represent words as numerical vectors in a high-dimensional space. These vectors encode semantic similarities between words, where words with similar meanings tend to have closer vectors in the embedding space.The model can exploit the inherent semantic relationships between words, enhancing the model's capability to represent text data and potentially improving its performance on the NLP task.

The standard categorical crossentropy loss functioncan be susceptible to class imbalance issues, where some classes have significantly fewer examples compared to others. To address this, we define a custom loss function (weighted_categorical_crossentropy) , If class imbalance is detected, the function dynamically computes class weights.

To prevent the model from getting stuck in local minima and improve convergence, we employ a learning rate scheduler implemented through (ReduceLROnPlateau) and to prevent overfitting and excessive training time, we use early stopping.

E. Figures and Tables

In this section, The following figures and tables provide visual representations and summaries of key information and results from the analysis performed of the different models.

TABLE I. ARABERT MODEL

Model	Accuracy		Evaluation Metrics		
	Training	*Validation*	*Precision*	*Recall*	*F1-score*
AraBERT	0.7347	0.7230	0.723	0.723	0.723

Fig. 1. Analysis of AraBERT Model

TABLE II. XLM-ROBERTA MODEL

Model	Accuracy		Evaluation Metrics		
	Training	*Validation*	*Precision*	*Recall*	*F1-score*
XLM-RoBERTa	0.7315	0.7493	0.5228	0.7230	0.6068

Fig. 2. Analysis of XLM-RoBERTa Model

TABLE III. GPT-2 MODEL

Model	Accuracy		Evaluation Metrics		
	Training	*Validation*	*Precision*	*Recall*	*F1-score*
GPT-2	0.7468	0.7231	0.5253	0.7231	0.6069

Fig. 3. Analysis of GPT-2 Model

TABLE IV. BIDIRECTIONAL LSTM MODEL

Model	Accuracy		Evaluation Metrics		
	Training	*Validation*	*Precision*	*Recall*	*F1-score*
Bidirectional LSTM	0.7458	0.7431	0.5607	0.7431	0.6390

Fig. 4. Analysis of Bidirectional LSTM Model

IV. MODEL INFERENCE

After evaluating the performance of the tested models, the Bidirectional LSTM model as chosen as the preferred approach for the inference task.

The Bidirectional LSTM architecture demonstrated robust performance in capturing the sequential dependencies within the input text, which proved crucial for the fine-grained NER task.

Additionally, the custom-built nature of the Bidirectional LSTM model allowed for better integration with the dataset-specific preprocessing steps, leading to improved overall performance compared to the off-the-shelf transformer-based models.

V. REFERENCES

[1] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C., 2016. Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360.

[2] Jarrar, M., Abdul-Mageed, M., Khalilia, M., Talafha, B., Elmadany, A., Hamad, N. and Omar, A., 2023. WojoodNER 2023: The First Arabic Named Entity Recognition Shared Task. arXiv preprint arXiv:2310.16153.

[3] Liqreina, H., Jarrar, M., Khalilia, M., El-Shangiti, A.O. and AbdulMageed, M., 2023. Arabic fine-grained entity recognition. arXiv preprint arXiv:2310.17333.

[4] Antoun, W., Baly, F. and Hajj, H., 2020. Arabert: Transformer-based model for arabic language understanding. arXiv preprint arXiv:2003.00104.

[5] Hussein, M., Khaled, S., Torki, M. and El-Makky, N.M., 2023, December. Alex-u 2023 nlp at wojoodner shared task: Arabinder (bi-encoder for arabic named entity recognition). In Proceedings of ArabicNLP 2023 (pp. 797-802).

[6] Elkordi, S., Adly, N. and Torki, M., 2023, December. Alexu-aic at wojoodner shared task: Sequence labeling vs mrc and swa for arabic named entity recognition. In Proceedings of ArabicNLP 2023 (pp. 771-776).

[7] Jarrar, M., Khalilia, M. and Ghanem, S., 2022. Wojood: Nested arabic named entity corpus and recognition using bert. arXiv preprint arXiv:2205.09651

[9] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. 2020). Unsupervised cross-lingual representation learning at scale. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440–8451. Association for Computational Linguistics. https://huggingface.co/xlm-roberta-base

[10] GPT-2: "Language Models are Unsupervised Multitask Learners" by Alec Radford, Karthik Narasimhan, et al.

[11] Anbil Parthipan, S.C., 2020. On challenges in training recurrent neural networks.

[12] Brownlee, J., 2017. Long short-term memory networks with python: develop sequence prediction models with deep learning. Machine Learning Mastery.

[13] Liu, G. and Guo, J., 2019. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Neurocomputing, 337, pp.325-338.