Effects of Multiple Annotation Schemes on Arabic Named Entity Recognition

Ikram Belhajem
Faculty of Sciences, Mohammed V University in Rabat, Morocco
i.belhajem@um5r.ac.ma (corresponding author)

Received: 26 July 2024 | Revised: 12 August 2024 | Accepted: 18 August 2024

 DOI: https://doi.org/10.48084/etasr.8528

ABSTRACT

Named Entity Recognition (NER) is considered an important subtask in information extraction that aims to identify Named Entities (NM) within a given text and classify them into predefined categories (e.g., person, location, organization, and miscellaneous). The use of an appropriate annotation scheme is crucial to label multi-word NEs and enhance recognition performance. This study investigates the effects of using different annotation schemes on NER systems for the Arabic language. The impact of seven annotation schemes, namely IO, IOB, IOE, IOBE, IOBS, IOES, and IOBES, on Arabic NER is examined by applying conditional random fields, multinomial Naive Bayes, and support vector machine classifiers. The experimental results reveal the importance of selecting an optimal annotation scheme and show that annotating NEs based on the simple IO scheme yields a higher performance in terms of precision, recall, and F-measure compared to the other schemes.

Keywords-information extraction; named entity recognition; machine learning; conditional random fields; support vector machines

I. INTRODUCTION

The concept of Named Entity (NE) involves the recognition of person names, organization names, geographic location names, time, currency, and percentage expressions within structured and unstructured text using SGML markup [1]. NE Recognition (NER) has emerged as one of the most important subtasks in information extraction, aiming to identify and classify every NE in a document into predefined categories, e.g., person, location, organization, and miscellaneous [2]. Various Natural Language Processing (NLP) applications rely on NER as a crucial preprocessing phase to enhance their overall performance, such as Information Retrieval (IR) [3], Machine Translation (MT) [4], Question Answering (QA) [5], and Search Results Clustering (SRC) [6].

Many studies have investigated NER for many different languages, including Arabic. Arabic is a Semitic language spoken by more than 360 million people in more than 30 countries [7]. It is a highly inflected language with rich morphology and complex syntax. Moreover, Arabic has some peculiarities that make it a highly challenging language to deal with in the context of NER [8]:

- Lack of capitalization
- Agglutination process in which an Arabic word may combine one or more prefixes, a stem or root, and one or more suffixes in different ways.
- Optional short vowels or diacritics included in Arabic may change the phonetic representation and give different meanings to the same lexical form.
- The ambiguity between NE types results in tagging the same word as one or more NE types.
- The absence of uniformity in writing styles leads to many variants of the same word that are spelled differently.
- Spelling mistakes include typographic errors made by Arabic writers about certain characters.
- Lack of free and publicly available Arabic linguistic resources to evaluate proposed Arabic NER (ANER) systems.

Three major approaches are used to perform ANER, including a handcrafted rule-based approach, a statistical Machine Learning (ML) based approach, and a hybrid approach. The rule-based approach [9, 10] is based on manual-crafted local grammatical rules written by expert linguists, so any adjustment required for rule-based NER systems is labor-intensive and time-consuming [8]. The ML-based approach [11, 12] depends on different learning algorithms, e.g., Hidden Markov Models (HMM), Decision Trees (DT), Maximum Entropy Models (ME), Support Vector Machines (SVM), and Conditional Random Fields (CRF) [13], that use feature sets obtained from annotated texts with NEs to build statistical models for NER systems. Therefore, solid linguistic knowledge is not required to develop ML-based NER systems that are adaptable and easily maintained with insignificant effort and minimum time given sufficient large tagged datasets. The hybrid approach [14] combines the previous two by providing the rule-based output as a feature used by the ML classifier. This integration aims to improve the overall performance of

NER systems and overcome the limitations of each approach when processed individually.

Therefore, the need for a large amount of annotated data is a prerequisite for training and testing NER models. Furthermore, since many NEs consist of multiple words, it is not feasible to annotate subsequent entities with the same type [15]. To this end, several annotation schemes have been proposed for labeling multi-word NEs in an attempt to increase recognition performance. However, few studies have focused on analyzing the effects of using different annotation schemes on NER systems for a variety of languages, especially Arabic. To address this shortcoming, this study aimed to investigate the impact of using various annotation schemes on the performance of ANER, as choosing the optimal annotation scheme is not straightforward [15]. Thus, the following annotation schemes were examined:

- IO: This simple scheme uses only two tags, namely the same Inside (I) tag for all words in the NE and the Outside (O) tag for words that are not part of the NE. The weakness of this scheme is its inability to recognize subsequent entities of the same type.
- IOB: In this scheme [16], each word in the text is assigned to a certain tag, whether it is the Beginning (B), the I, or the O of the NE. This scheme is capable of differentiating between consecutive entities and has good support in the literature.
- IOE: This scheme is similar to IOB but it replaces the B tag with the E tag to indicate the end of the NE.
- IOBE: This is a variation of the IOB scheme that additionally distinguishes the last word of multi-word NEs with the E tag to have more information concerning the entity boundaries.
- IOBS: This scheme labels the entities as IOB, with the addition of the Single (S) tag to identify NEs including only a single word.
- IOES: An extension to the IOE scheme that adds the S tag for single-word NEs.
- IOBES: A further extension to the IOB scheme that consists of five tags, namely B, I, and E for multiword NEs, S for one-word NEs, and O for non-entity words.

II. RELATED WORKS

The primitive simplest annotation scheme applied to the NER task is the IO [17]. However, this scheme is unable to represent correctly multi-word NEs, as it cannot recognize subsequent entities of the same type. In [2], the IOB tagging scheme was adopted to annotate the corpus, and since then it has become the most widely used format in NER systems. However, some studies compared and analyzed the impact of different annotation schemes on the performance of NER systems for multiple languages, in particular Arabic.

In [18], the design challenges encountered when developing an efficient and robust NER system were studied, such as issues related to text chunk representations and how to use external knowledge resources in NER. Some interesting findings were reported, particularly the considerable impact of choosing an appropriate tagging scheme on the system performance and the fact that the BILOU format (same as IOBES but using the L tag instead of the E tag and the U tag rather than the S tag) significantly outperforms the most commonly used BIO scheme on CoNLL-2003 and MUC7 datasets. The NER task has also been adopted in the biomedical domain to identify entities such as genes, chemical compounds, viruses, diseases, and drugs mentioned in biomedical text. In [19], the performance of biomedical NER was explored using CRF and SVM classifiers with different annotation schemes, notably IO, IOB1, IOB2, IOE1, IOE2, IOBE, and IOBES. The adopted approach was evaluated on the i2b2/VA 2010 challenge dataset and the JNLPBA 2004 shared task dataset. Simulation results showed that CRF performed better than SVM and that the IO format with only two tags gives the best performance, whereas the IOBE and IOBES schemes resulted in low performance, leading to the conclusion that the number of tags used in the annotation scheme affects the performance of BioNER.

Although many studies analyzed the effects of annotation schemes on NER in English, other languages also received attention on this topic. In [15], the impact of multiple segment representations on NER was examined, in particular IO, IOU (same as IOBES but adds the U tag for unit-word entities), BIO-1 (or IOB1), BIO-2 (or IOB2), BIOU (similar to IOBS), IEO-1 (or IOE1), IEO-2 (or IOE2), IEOU (same format as IOES), BIEO (similar to IOBE) and BILOU. Experimental tests were applied in the corpora of four different languages, including English, Spanish, Dutch, and Czech, using CRF and ME. The corpora for English, Spanish, and Dutch were collected from the CoNLL-2002 and CoNLL-2003 shared tasks, while the CoNLL format version of Czech Named Entity Corpus 1.1 was used for Czech. Based on the findings, BILOU performed the worst in English when using CRF, and IOE-1 and IOE-2 were the most promising representations, as they performed the best in almost all languages and methods. This study concluded that choosing the optimal tagging scheme is not straightforward, since it depends on the language, the approach adopted, and the feature set used. For NER in Russian, the impact of IO and BIO labeling formats on two open Russian text collections was explored in [20], using the CRF method. Experimental results showed that the BIO scheme had a more significant contribution in recognizing NEs and outperformed the IO scheme based on F-measure values. This highlights the weaknesses of the IO format in recognizing subsequent entities of the same type and the importance of the BIO format for representing entities in languages such as Russian, where NEs are usually located beside each other. In [21], the effects of BILOU and IO representation schemes on increasing the learning performance of a Portuguese NER system were explored. Different experiments were carried out on the HAREM corpus using CRF, and the IO scheme achieved better F-measure values compared to BILOU for all categories. Accordingly, it is worth noting that adopting a simple or a complex annotation scheme for NER can increase or decrease its performance depending on the structure of each language.

Regarding the Arabic language, to our knowledge, the only research effort that analyzed the impact of various annotation schemes on NER was proposed in [22]. This study investigated the effects of using IO, IOB, IOE, IOBES, BI, IE, and BIES on the performance of biomedical NER for Arabic. Several experiments were carried out on a dataset extracted from 27 Arabic medical articles applying five different classifiers, namely AdaBoost, DT, K-nearest neighbors, Random Forest (RF), and Gradient Boost (GB), to recognize only a single class of entities (disease names). The results showed that the IO scheme achieved the highest F-score and outperformed the other annotation schemes in terms of cost and running time, as it requires few tags. However, since the IO scheme cannot recognize consecutive entities, the performance of more complex schemes (IOB, IOE, IOBES, BI, IE, and BIES) was explored in the case of adjacent entities. The results showed that the BI scheme with the RF classifier and the IOE scheme with the GB classifier correctly predicted 62.5% of the consecutive entities, while the results of other schemes ranged between 12.5% and 50%, concluding that these annotation schemes have promising potential in recognizing consecutive entities. Moreover, the importance of choosing first the appropriate classifier for NER and then selecting the suitable annotation scheme was highlighted, as it affects the classifier performance.

This study investigates the impact of applying multiple annotation schemes in Arabic NER. The proposed approach differs from [22] in many aspects. First, this study introduces other annotation schemes, i.e., IOBE, IOBS, and IOES, to examine their effects on a standard NER corpus extracted from newspaper articles rather than biomedical datasets, which have different properties. This method also extracts several additional features for each word in the corpus to train the CRF, Multinomial Naive Bayes (MNB), and SVM classifiers, which were not explored in [22]. Finally, the proposed ANER system aims to recognize various classes of NEs, namely Person, Location, Organization, and Miscellaneous.

III. METHOD

The proposed system consists of three main phases, as shown in Figure 1. In the first phase, data annotation is carried out to convert the IOB format applied on the original corpus to the rest of the tagging formats, notably IO, IOE, IOBE, IOBS, IOES, and IOBES. The second phase involves the extraction of NER features related to each word in the text including context words, word length, POS information, and morphological features. Finally, in the third phase, the extracted features are fed into the CRF, MNB, and SVM classifiers to identify Arabic NEs.

A. Data Annotation

The ANERcorp dataset [23] was used following the CONLL-2002 task formulation. ANERcorp classifies NEs into the same four classes defined in the CoNLL-2002, namely Person, Location, Organization, and Miscellaneous. The corpus follows the IOB annotation scheme to assign every word in the text to a specific tag (B, I, or O). Consequently, ANERcorp contains the words of the text along with their corresponding label that indicates both boundary tags along with the NE class that can be one of the following: B-PERS, I-PERS, B-LOC, I-LOC, B-ORG, I-ORG, B-MISC, I-MISC, or O.

Concerning the different annotation schemes adopted, Python scripts were developed to convert the IOB format to other new formats, namely IO, IOE, IOBE, IOBS, IOES, and IOBES. This results in several datasets generated from the original ANERcorp, each corresponding to a certain annotation scheme. Table I provides an example of tagging a text fragment with each annotation scheme. Table II presents the number of annotation labels in each dataset, calculated given the number of tags per annotation scheme and the number of NE categories.

B. NER Features

NER features are characteristic word attributes designed for algorithmic consumption. The feature vectors are fed to the NER classifier as input data, representing each word to be categorized by one or more Boolean or binary, numerical, and nominal values [20]. In this study, the following features were applied for NER:

- Context words: These are preceding and succeeding words related to the current NE within a context window, i.e., the window size is chosen to be ±1. This feature is used under the observation that the surrounding words carry effective information to identify NEs.
- Word prefix and suffix: The extraction of word prefixes and suffixes, if existing, can be a good sign to capture NE presence, as most ANEs have no prefix or suffix [39]. Here, it is generated by Tashaphyne [24], which is an Arabic light stemmer and segmentator.
- Word length: This is a binary feature used to check if the length of the current word is greater than a predefined threshold, which is set to 3 characters. This is based on the fact that very short words are rarely NEs.
- Part Of Speech (POS) information: This feature identifies the POS category, e.g., noun, verb, adjective, preposition, etc., of the current word and its surrounding words, i.e., one previous and one after. The MADAMIRA tool [25] is employed to generate POS information, which is used for morphological analysis and disambiguation in Arabic.
- Morphological features: This is a set of morphological information generated by MADAMIRA, based on exploiting the rich morphological features in Arabic.
- Corresponding English Capitalization: This is a binary feature that checks the capitalization of the English translation corresponding to the current Arabic word. It is used to compensate for the missing capitalization feature in Arabic, based on the English gloss generated by MADAMIRA. If the translated word begins with a capital letter, it is most likely a NE [8].
- Gazetteers: This is a set of binary features that indicate whether the word exists within each of the various gazetteers (predefined lists of typed NEs). This approach uses ANERGaz [23], which consists of three gazetteers: complete names of people, locations (names of continents,

countries, cities, etc.), and organizations (names of companies, football teams, and other organizations).

- Contains Digit: This is a binary feature to examine whether the word contains any digits (0-9). It helps to recognize miscellaneous NEs such as time expressions, measurement expressions, and numerical numbers [26].
- Character n-grams: This is a set of features consisting of the current leading and trailing character unigrams, bigrams, trigrams, and quadrigrams. These character n-gram features implicitly capture valuable morphological and orthographic clues that indicate the presence or absence of NEs [27].
- Stop Words: This binary feature checks whether the word is in the stop words list. Stop words are common words that cannot be part of NEs. The list of stop words was collected based on [28], consisting of 1383 words including prepositions, pronouns, conditional pronouns, verbal pronouns, and adverbs.

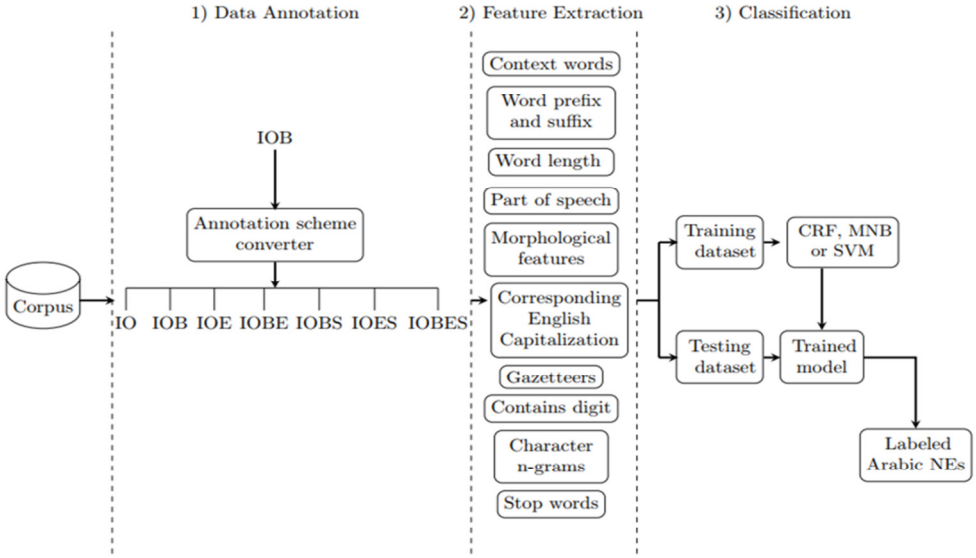

Fig. 1. ANER system architecture.

TABLE I. ANNOTATION OF A FRAGMENT TEXT WITH VARIOUS SCHEMES

Word	IO	IOB	IOE	IOBE	IOBS	IOES	IOBES
…	…	…	…	…	…	…	…
عبر	O	O	O	O	O	O	O
فضائية	O	O	O	O	O	O	O
الجزيرة	I-ORG	B-ORG	E-ORG	B-ORG	S-ORG	S-ORG	S-ORG
أكد	O	O	O	O	O	O	O
السيد	O	O	O	O	O	O	O
حسن	I-PERS	B-PERS	I-PERS	B-PERS	B-PERS	I-PERS	B-PERS
نصر	I-PERS	I-PERS	I-PERS	I-PERS	I-PERS	I-PERS	I-PERS
الله	I-PERS	I-PERS	E-PERS	E-PERS	I-PERS	E-PERS	E-PERS
الأمين	O	O	O	O	O	O	O
العام	O	O	O	O	O	O	O

TABLE II. NUMBER OF ANNOTATION LABELS PER DATASET

Dataset	Number of annotation labels
IO dataset	5
IOB dataset	9
IOE dataset	9
IOBE dataset	13
IOBS dataset	13
IOES dataset	13
IOBES dataset	17

C. Classification

This approach used the supervised ML algorithms CRF, MNB, and SVM to identify Arabic NEs, since the NER task can be regarded as a sequence labeling problem to assign a specific label to each word in a given input sequence.

1) Conditional Random Fields (CRF)

CRF [40] are discriminative probabilistic models that are well suited for segmenting and labeling sequence data. They have been applied successfully for several NLP tasks, particularly NER [29]. They are a type of conditionally trained undirected graphical models whose output nodes represent the label sequence, while the input nodes correspond to the data

sequence. Therefore, CRF aims to find a y that maximizes the conditional probability $P(y|x)$ of a label sequence $y = y_1, \ldots, y_T$ given an input sequence $x = x_1, \ldots, x_T$ as

$$P(x|y) = \frac{1}{Z(x)} exp(\sum_{t=1}^{T} \sum_{k=1}^{N} \lambda_k f_k(y_{t-1}, y_t, x, t)) \quad (1)$$

where T is the sequence length, N is the number of features, $f_k(y_{t-1}, y_t, x, t)$ is a feature function whose value may range from -∞ to ∞ but it is often binary, λ_k represents a learned weight assigned to each feature function f_k, and $Z(x)$ is a normalization factor expressed as

$$Z(x) = \sum_y exp\left(\sum_{t=1}^{T} \sum_{k=1}^{N} \lambda_k \ f_k(y_{t-1}, y_t, x, t)\right) \quad (2)$$

2) MNB (Multinomial Naïve Bayes)

NB is a probabilistic ML algorithm based on the Bayes Theorem with a strong assumption that all the input features are independent of each other. This classifier is employed in a wide range of classification tasks due to its simplicity and effectiveness. MNB [30] is a variant of the NB classifier for multinomially distributed data, suitable for classification with discrete features, e.g., word counts for text classification. The distribution is parametrized by vectors $\theta_y = (\theta_{y1}, .., \theta_{yn})$ for each class y, where n is the number of features and θ_{yi} is the probability $P(x_i|y)$ of feature i appearing in a sample belonging to class y. The parameters θ_{yi} are estimated by a smoothed version of maximum likelihood, i.e., relative frequency counting.

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n} \quad (3)$$

where $N_{yi} = \sum_{x \in T} x_i$ is the number of times that feature i appears in a sample of class y in the training set T, and $N_y = \sum_{i=1}^{n} N_{yi}$ is the total count of all features for class y. The smoothing prior $\alpha \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations.

3) Support Vector Machine (SVM)

SVMs are universal ML algorithms that nonlinearly map the input vector into a high-dimensional feature space using a nonlinear transformation. They are based on the inductive principle of structural risk minimization, which seeks to control model complexity and minimize the upper bound of the generalization error [31]. The main characteristics of SVM include the use of kernels, the absence of local minima, the sparseness of the solution, and the capacity control obtained by optimizing the margin [32]. Thus, they are applied successfully to both classification [33, 34] and regression [35] problems.

This study adopted a multi-class SVM classifier to model the input-output functional relationship. The one-versus-rest approach was used to construct it, involving training a separate binary SVM for each class versus all the other classes and then selecting the one with the highest score. Based on a given set of training data points whose size is l, $\{(x_i, y_i)\}$ $(i = 1, \ldots, l)$, where x_i is the i^{th} sample vector and $y_i \in \{+1, -1\}$ is its associated class, the goal of SVM is to find a linear separating hyperplane with maximal margin (distance of the hyperplane to the nearest training samples) defined as:

$$< w_0, x > + b_0 = 0 \quad (4)$$

where $<.,.>$ represents the dot product, w is the weight vector and b is the bias term. Therefore, SVM constructs the optimal hyperplane (w_0, b_0) by providing a unique solution to the following quadratic programming problem:

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2} \| w^2 \|$$

subject to

$$y_i(< w, x_i > + b) \geq 1 \quad (5)$$

IV. EXPERIMENTS AND RESULTS

Experimental tests were carried out on ANERcorp [23] using CRF, MNB, and SVM to examine the impact of different annotation schemes on ANER. The ANERcorp is a freely available annotated corpus that is manually collected from different article types extracted from various newspapers to obtain a more generalized corpus. It contains more than 150,000 tokens, of which 11% are NEs distributed as 39% for Person, 30.4% for Location, 20.6% for Organization, and 10% for Miscellaneous. It is composed of a training corpus and a test corpus, with 125,000 and 25,000 tokens, respectively.

The metrics used to evaluate the performance of the proposed ANER system are the micro-averaged precision, recall, and F-measure. Precision refers to the percentage of NEs identified by the evaluated system that are correct, recall is the percentage of NEs present in the corpus found by the system [2], and the F-measure is defined as a harmonic mean between precision and recall with equal weights.

During the training phase, different extracted features are used as input to the CRF, MNB, and SVM classifiers to recognize ANEs. The sklearn-crfsuite sequence classification library [36] was employed to implement the CRF model. The scikit-learn Python library [37] was used to implement the MNB and SVM models. The CRF, MNB, and SVM models for the seven annotation schemes, namely IO, IOB, IOE, IOBE, IOBS, IOES, and IOBES, are learned based on the ANERCorp training corpus using 10-fold cross-validation (splits the original data into 10 folds of approximately equal size - at each iteration, one fold is considered as the test set, while the remaining nine folds are used as the training set). The L-BFGS algorithm [38] is used to train the CRF model with the maximum number of iterations set to 100, allowing for all possible transition features, while the values of the L1 and L2 regularization coefficients are set to 0.1. For the MNB model, the value of the smoothing hyperparameter α is selected as 1. For the SVM model, the linear kernel was chosen, since it is the simplest kernel function depending only on the dot products of feature vectors, while the maximum number of iterations is set to 100. Tables III-V summarize the results of training CRF, MNB, and SVM models with 10-fold cross-validation for all various annotation schemes. It should be noted that the O tag is excluded when calculating the evaluation metrics in order not to distort results, since the large majority of ANERcorp tokens are tagged as O. The results show that the IO annotation scheme outperformed the other schemes for the CRF, MNB, and SVM models in terms of precision, recall, and F-measure.

TABLE III. CRF TRAINING RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	85.38	72.86	78.62
IOB	82.79	70.79	76.32
IOE	82.80	70.63	76.23
IOBE	82.33	69.78	75.54
IOBS	82.09	69.68	75.38
IOES	81.75	69.47	75.12
IOBES	81.31	68.89	74.59

TABLE IV. MNB TRAINING RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	70.83	59.99	64.96
IOB	66.40	42.88	52.11
IOE	67.93	44.13	53.50
IOBE	63.56	39.62	48.82
IOBS	61.41	32.94	42.88
IOES	63.46	33.86	44.16
IOBES	58.39	28.86	38.63

TABLE V. SVM TRAINING RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	78.72	69.89	74.04
IOB	75.78	67.43	71.36
IOE	74.75	65.85	70.01
IOBE	73.83	65.70	69.53
IOBS	74.43	65.51	69.69
IOES	74.14	65.06	69.30
IOBES	73.03	64.76	68.65

In the prediction phase, trained CRF, MNB, and SVM models were applied to the ANERcorp test corpus to classify unseen ANEs for each annotation scheme. Tables VI-VIII present precision, recall, and F-measure values for the different experiments. According to the results, learning models based on the IO tagging scheme achieved the best performance compared to the rest schemes, achieving the highest precision, recall, and F-measure scores of 82.05%, 64.43%, and 72.18%, respectively, for the CRF based approach, 80.68%, 49.94% and 61.69%, respectively, for the MNB based approach, and 82.53%, 64.49% and 72.40%, respectively, for the SVM based approach.

The second promising scheme is IOB, since it achieved the closest performance to the IO scheme with a decrease of 1.81% for precision, 1.37% for recall, and 1.56% for F-measure when using the CRF classifier, a decrease of 6.95% for precision, 16.3% for recall and 15.49% for F-measure when using the MNB classifier, and a decrease of 5.5% for precision, 4.55% for recall, and 4.99% for F-measure when using the SVM classifier. However, it should be noted that the MNB model based on the IOE scheme achieved a percentage increase of 1.9%, 1.34%, and 1.63% for precision, recall, and F-measure compared to the IOB scheme. The worst scheme was IOBES, as it resulted in low performance in all evaluation metrics for the CRF, MNB, and SVM models. The IOE scheme yielded significantly better results in precision, recall, and F-measure than IOBE, IOBS, and IOES for CRF, MNB, and SVM classifiers. In addition, the results show that the MNB classifier performed the worst for all annotation schemes compared to the CRF and SVM classifiers. The SVM classifier achieved slightly better performance than the CRF classifier for the IO annotation scheme, with a percentage increase of 0.48%, 0.06%, and 0.22% for precision, recall, and F-measure, respectively. However, the CRF classifier performed better than the SVM classifier for the rest annotation schemes, namely IOB, IOE, IOBE, IOBS, IOES, and IOBES.

TABLE VI. CRF TEST RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	82.05	64.43	72.18
IOB	80.24	63.06	70.62
IOE	78.68	62.46	69.64
IOBE	78.66	61.39	68.96
IOBS	77.26	60.62	67.93
IOES	76.55	60.68	67.70
IOBES	74.52	59.76	66.33

TABLE VII. MNB TEST RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	80.86	49.94	61.69
IOB	73.73	33.64	46.20
IOE	75.63	34.98	47.83
IOBE	72.17	31.62	43.97
IOBS	69.37	26.21	38.04
IOES	68.84	26.03	37.77
IOBES	64.98	22.13	33.02

TABLE VIII. SVM TEST RESULTS

Annotation scheme	Precision (%)	Recall (%)	F-measure (%)
IO	82.53	64.49	72.40
IOB	77.03	59.94	67.41
IOE	76.64	59.34	66.89
IOBE	75.77	58.60	66.09
IOBS	73.64	56.99	64.25
IOES	72.57	56.19	63.34
IOBES	71.74	55.50	62.59

These results show that annotating NEs based on the simplest IO scheme leads to higher performance of the proposed ANER system. This can be explained by the fact that IO requires only I and O tags without any specification of multi-word NE boundaries, allowing the system to reach a maximum number of correctly classified NEs. On the other hand, the more complex IOBES scheme that consists of five tags, including three tags to distinguish boundaries of NEs with multiple words, showed the lowest performance since it generates more annotation labels for classification. The performance of the ANER system depends on the number of tags used in each annotation scheme, as reducing the number of tags significantly improves the overall performance. Furthermore, according to the experimental results, the performance of the ANER system is also affected by the selection of an appropriate classifier, as there are significant differences between the results of the CRF, MNB, and SVM classifiers. As previously shown, some studies came to the same conclusion about the high performance of IO when exploring the effects of different annotation schemes on NER.

However, certain research efforts reported the positive contribution of an annotation scheme other than IO to NER performance, whereas other studies highlighted the inability of IO to recognize subsequent entities of the same type in languages such as Russian [20] and Arabic [22]. This leads us to conclude that the impact of any annotation scheme on NER varies depending on its number of tags, the chosen classifier, and the target language structure.

V. CONCLUSION

This study investigated the impact of using seven different annotation schemes on ANER performance by applying the CRF, MNB, and SVM classifiers. According to the experimental results in the ANERcorp, annotating NEs based on the simplest IO scheme provided the best performance compared to other schemes in terms of precision, recall, and F-measure values. This study shows that choosing an appropriate annotation scheme is not straightforward since its effect on the NER task varies depending on its number of tags, the chosen classifier, and the target language structure. Future work will examine the impact of combining multiple annotation scheme outputs based on voting strategies on the overall performance of the ANER system.

REFERENCES

[1] R. Grishman and B. Sundheim, "Message Understanding Conference-6: a brief history," in *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, Copenhagen, Denmark, May 1996, pp. 466–471, https://doi.org/10.3115/992628.992709.

[2] E. F. Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, Edmonton, Canada, Feb. 2003, pp. 142–147, https://doi.org/10.3115/1119176.1119195.

[3] A. Ababneh, J. Lu, and Q. Xu, "Arabic Information Retrieval: A Relevancy Assessment Survey," *Proceedings of the International Conference on Information Systems Development (ISD)*, Sep. 2016.

[4] A. Alqudsi, N. Omar, and K. Shaker, "Arabic machine translation: a survey," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 549–572, Dec. 2014, https://doi.org/10.1007/s10462-012-9351-1.

[5] L. Abouenour, K. Bouzoubaa, and P. Rosso, "IDRAAQ: New Arabic Question Answering system based on Query Expansion and Passage Retrieval," in *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes*, Rome, Italy, 2012.

[6] Y. Benajiba, M. Diab, and P. Rosso, "Arabic Named Entity Recognition: A Feature-Driven Study," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 5, pp. 926–934, Jul. 2009, https://doi.org/10.1109/TASL.2009.2019927.

[7] "Arabic speaking countries - Worldwide distribution," *Worlddata.info*. https://www.worlddata.info/languages/arabic.php.

[8] K. Shaalan, "A Survey of Arabic Named Entity Recognition and Classification," *Computational Linguistics*, vol. 40, no. 2, pp. 469–510, Jun. 2014, https://doi.org/10.1162/COLI_a_00178.

[9] W. Zaghouani, "RENAR: A Rule-Based Arabic Named Entity Recognition System," *ACM Transactions on Asian Language Information Processing*, vol. 11, no. 1, Nov. 2012, https://doi.org/10.1145/2090176.2090178.

[10] M. Oudah and K. Shaalan, "NERA 2.0: Improving coverage and performance of rule-based named entity recognition for Arabic," *Natural Language Engineering*, vol. 23, no. 3, pp. 441–472, May 2017, https://doi.org/10.1017/S1351324916000097.

[11] Y. Benajiba and P. Rosso, "Arabic Named Entity Recognition using Conditional Random Fields," in *Proceedings of the Workshop on HLT and NLP within the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 2008, pp. 143–153.

[12] J. Li, A. Sun, J. Han, and C. Li, "A Survey on Deep Learning for Named Entity Recognition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, Jan. 2022, https://doi.org/10.1109/TKDE.2020.2981314.

[13] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticæ Investigationes*, vol. 30, no. 1, pp. 3–26, Jan. 2007, https://doi.org/10.1075/li.30.1.03nad.

[14] M. Oudah and K. Shaalan, "A Pipeline Arabic Named Entity Recognition using a Hybrid Approach," in *Proceedings of the Coling Organizing Committee*, Mumbai, India, 2012, pp. 2159–2176.

[15] M. Konkol and M. Konopík, "Segment Representations in Named Entity Recognition," in *Text, Speech, and Dialogue*, Pilsen,Czech Republic, 2015, pp. 61–70, https://doi.org/10.1007/978-3-319-24033-6_7.

[16] L. A. Ramshaw and M. P. Marcus, "Text Chunking Using Transformation-Based Learning," in *Natural Language Processing Using Very Large Corpora*, S. Armstrong, K. Church, P. Isabelle, S. Manzi, E. Tzoukermann, and D. Yarowsky, Eds. Springer Netherlands, 1999, pp. 157–176.

[17] D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel, "Nymble: a High-Performance Learning Name-finder." arXiv, Mar. 27, 1998, https://doi.org/10.48550/arXiv.cmp-lg/9803003.

[18] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL '09*, Boulder, CO, USA, 2009, pp. 147–155, https://doi.org/10.3115/1596374.1596399.

[19] H. L. Shashirekha and H. A. Nayel, "A comparative study of segment representation for biomedical named entity recognition," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Jaipur, India, Sep. 2016, pp. 1046–1052, https://doi.org/10.1109/ICACCI.2016.7732182.

[20] V. A. Mozharova and N. V. Loukachevitch, "Combining Knowledge and CRF-Based Approach to Named Entity Recognition in Russian," in *Analysis of Images, Social Networks and Texts*, Yekaterinburg, Russia, 2017, pp. 185–195, https://doi.org/10.1007/978-3-319-52920-2_18.

[21] D. O. F. do Amaral, M. Buffet, and R. Vieira, "Comparative analysis between notations to classify named entities using conditional random fields," in *Proceedings of the 10th Brazilian Symposium in Information and Human Language Technology*, Natal, Brazil, Nov. 2015, pp. 27–31.

[22] N. Alshammari and S. Alanazi, "The impact of using different annotation schemes on named entity recognition," *Egyptian Informatics Journal*, vol. 22, no. 3, pp. 295–302, Sep. 2021, https://doi.org/10.1016/j.eij.2020.10.004.

[23] Y. Benajiba, P. Rosso, and J. M. BenedíRuiz, "ANERsys: An Arabic Named Entity Recognition System Based on Maximum Entropy," in *Computational Linguistics and Intelligent Text Processing*, Mexico City, Mexico, 2007, pp. 143–153, https://doi.org/10.1007/978-3-540-70939-8_13.

[24] T. Zerrouki, "Tashaphyne: Tashaphyne Arabic Light Stemmer and segmentor." [Online]. Available: https://pypi.org/project/Tashaphyne/0.2/.

[25] A. Pasha *et al.*, "MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic," *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*, pp. 1094–1101, 2014.

[26] A. Ekbal and S. Bandyopadhyay, "Named Entity Recognition using Support Vector Machine: A Language Independent Approach," *International Journal of Electrical, Computer, and Systems Engineering*, vol. 4, no. 2, pp. 155–170, 2010.

[27] A. A. Hamid and K. Darwish, "Simplified Feature Set for Arabic Named Entity Recognition," in *Proceedings of the 2010 named entities workshop*, 2010, pp. 110–115.

[28] I. A. El-Khair, "Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study." arXiv, Feb. 07, 2017, https://doi.org/10.48550/arXiv.1702.01925.

[29] A. McCallum and W. Li, "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003* -, Edmonton, Canada, 2003, vol. 4, pp. 188–191, https://doi.org/10.3115/1119176.1119206.

[30] "Naive Bayes," *scikit-learn*. https://scikit-learn/stable/modules/naive_bayes.html.

[31] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Science & Business Media, 2013.

[32] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[33] C. W. Hsu, C. C. Chang, and C. J. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science and Information Engineering, National Taiwan University, Technical Report, 2010.

[34] A. Alzahrani, "Explainable AI-based Framework for Efficient Detection of Spam from Text using an Enhanced Ensemble Technique," *Engineering, Technology & Applied Science Research*, vol. 14, no. 4, pp. 15596–15601, Aug. 2024, https://doi.org/10.48084/etasr.7901.

[35] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, Aug. 2004, https://doi.org/10.1023/B:STCO.0000035301.49549.88.

[36] M. Korobov, "sklearn-crfsuite - documentation." https://sklearn-crfsuite.readthedocs.io/en/latest/.

[37] "scikit-learn: machine learning in Python." https://scikit-learn.org/stable/.

[38] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, Aug. 1989, https://doi.org/10.1007/BF01589116.

[39] S. Abdelrahman, M. Arnaoty, M. Marwa, and A. Fahmy, "Integrated Machine Learning Techniques for Arabic Named Entity Recognition," *International Journal of Computer Science Issues*, vol. 7, no. 4, pp. 27–36, Jul. 2010.

[40] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," *Icml*, vol. 1, no. 2, 2001.