جامعة بيرزيت

BIRZEIT UNIVERSITY

Faculty of Graduate Studies
Master Program in Applied Statistics and Data Science

Performance Analysis of Machine Learning Algorithms for Predicting Arabic Morphological Features Without Context

مقارنة بين خوارزميات تعلم الآلة في التنبؤ بالخصائص الصرفية للكلمات العربية بدون سياق

Prepared by:
Nour A. Shayeb
1195052

Supervisor:
Dr Mustafa Jarrar

Submitted in fulfillment of the requirements for the "Master's Degree in Applied Statistics and Data Science" from the faculty of Graduate Studies at Birzeit University-Palestine

17-06-2023

جامعة بيرزيت

BIRZEIT UNIVERSITY

Performance Analysis of Machine Learning Algorithms for Predicting Arabic Morphological Features Without Context

مقارنة بين خوارزميات تعلم الآلة في التنبؤ بالخصائص الصرفية للكلمات العربية بدون سياق

Prepared by:

Nour A. Shayeb - 1195052

Committee:

Name	**Signature**
Dr. Mustafa Jarrar (Supervisor)	
Dr. Mohammad Khalilia (Examiner)	MK
Dr. Hassan Abu Hassan (Examiner)	

Submitted in partial fulfillment of the requirements for the "Master's Degree in Applied Statistics and Data Science" from the faculty of Graduate Studies at Birzeit University-Palestine

Jun-23

Abstract

Birzeit University has built the largest Arabic lexicographic database that is composed of 150 lexicons. Some morphological features of the words in the database are not included or are incomplete. In this study, we implemented four Machine Learning algorithms in order to predict the incomplete words' features. Although there are different methods for predicting morphological features of words in context, our goal is to predict them without context. The algorithms we used in this thesis are Logistic Regression, Support Vector Machine, Naive Bayes, and Random Forest. Our target is to predict the features of words (mainly, Part of Speech, Gender, and Number) without context sentences. We adapted a character-based approach without using context or supplementary dictionaries. These algorithms examine both words without diacritics, as well as words with diacritics. Afterwards, we compared the performance of the resulting models for each characteristic, considering different metrics, such as accuracy, recall, precision, the F1-score, the confusion matrix, and the Area Under the Curve of the Receiver Characteristic Operator (AUC-ROC Curve). Based on the results, the random forest algorithm performed the best among the other algorithms. Diacritics were also helpful in increasing the performance of Part of Speech and Number prediction from 89.88% and 90.08% to 95.33% and 97.07% AUC-ROC, respectively. Comparatively, the gender of Arabic words could be predicted using both diacritics as well as without diacritics at an average AUC-ROC of 92.66%.

Keywords: Arabic Word morphological features, Gender Prediction, Number Prediction, POS prediction, Machine Learning, Classification Algorithms.

الملخص

تمتلك جامعة بيرزيت قاعدة بيانات معجمية تضم كلمات اللغة العربية التي تم تجميعها من ما يقارب 150 قاموس لغوي. وتشمل قاعدة البيانات كلمات من اللغة العربية بالإضافة الى عدد من الصفات الصرفية لهذه الكلمات. لكن بعض هذه الكلمات ينقصها أحد أو كل هذه الصفات الصرفية، خاصة قسم الكلام والعدد والجنس. تسعى هذه الدراسة لتطبيق خوارزميات التعلم الآلي المختلفة على الكلمات العربية؛ بهدف استخدامها للتنبؤ بهذه الصفات المفقودة. وعلى الرغم من وجود طرق مختلفة للتنبؤ بصفات الكلمة الصرفية التي تكون ضمن سياق ، فإن هدفنا هو توقعها بدون سياق. وهذه الخوارزميات هي: الانحدار اللوجستي، ومتجهات الدعم، وبايز، والغابة العشوائية. واستُخدِمَت حروف الكلمة العربية نفسها لبناء النماذج، دون الحاجة إلى نصوص تضم مختلف الكلمات العربية، وأيضا دون الاستعانة بقاموس يخص البناء الصرفي للكلمة العربية. ولمقارنة أداء النماذج المختلفة، تم اعتماد مؤشرات تقييم الأداء من درجة دقة القياس، ومعدل الاسترجاع، ودقة النموذج، ودرجة F ، ومنحنى خصائص تشغيل المستقبل. قمنا بتطبيق هذه الخوارزميات بالاستعانة بكل من الكلمات العربية مضافاً لها الحركات (التشكيل) والكلمات العربية بدون أي حركات. تبعاً لذلك، أشارت النتائج الى أن التنبؤ بكل من قسم الكلام والعدد يتأثر بوجود الحركات (التشكيل) على حروف الكلمة بشكل إيجابي، حيث إنّ منحنى خصائص تشغيل المستقبل وصلَ من 89.88% إلى 95% لقسم الكلام، ومن 90.08% إلى 97.07% للعدد. في المقابل، بالإمكان التنبؤ بصفة الجنس للكلمة العربية باستخدام الحروف من الحركات بدرجة دقة مماثلة لتلك الناتجة من استخدام الحروف بدون حركات، والتي وصلت إلى معدل منحنى خصائص تشغيل المستقبل 92.66%.

الكلمات المفتاحية: الصفات الصرفية للكلمة العربية، التنبؤ بجنس الكلمة، التنبؤ بقسم الكلام، التنبؤ بعدد الكلمة، التعلم الآلي، خوارزميات التصنيف.

Contents

List of Tables

List of Figures

Chapter 1: Introduction

Birzeit University has the largest lexicographic database about the Arabic language, which is built using about 150 of the top Arabic lexicons (Jarrar & Amayreh, 2019; Jarrar, 2020). The database consists of tens of millions of Arabic words that are important to the Arabic language research field. That subsequently led different Natural Languages Processing (NLP) and Knowledge Engineering studies to become more applicable in applications like search engines, translations, grammatical error detection, and spell-checkers. This data is available through Birzeit Lexicographic Search Engine (Alhafi et al., 2019; Jarrar, 2018) which helps Arabic language learners - whether they are natives or not- to use a lexicographic that collected and summarized all other sources.

However, this lexicographic dataset has incomplete data, where some word features are missed, such as gender, part of speech, and number. Thus, this study intends to impute these values to enhance data set importance. After that, we can use the generated prediction models to predict such word features.

Arabic language processing is a challenging task. Arabic has unstable rules and complex morphology. The Arabic word consists of various parts, such as prefixes, suffixes, and lemma. And an enormous number of words can be resulted from combining these parts. Furthermore, Arabic language processing faces a lack of resources, where limited adequate resources are available for similar implementations. (Salah & Binti Zakaria, 2017)

Several related accomplishments relate to tagging Arabic words with their morphological features. The first type depends on a sentence-based method. In other words, the model can tag word features based on their position in the sentence (i.e., context). Thus, a huge, labeled corpus should be used to train the model and achieve acceptable results. In contrast, other kinds of research aim to predict different morphological features using a word-based method (i.e., without context). Words' informative parts would be extracted, and it would list all proposed words' labels using Arabic dictionaries.

All available techniques depend on big resources availability, which is a thing that the Arabic language lacks. Our Lexicon with its rich data should be employed in morphological feature prediction studies.

To go through these drawbacks, Artificial Intelligence can also help in such complex issues, especially Machine Learning because different algorithms are used to detect patterns and information from data.

1.1Purpose and Research Questions

Considering the above discussion, the point is whether machine learning can help in predicting Arabic word morphological features without context or background knowledge. For example, given the word (جامعات), is it possible to predict the POS, Gender, and Number of this word without any context? It might be straightforward for some words like (جامعات) but more challenging for words like (ذهب). Different ML algorithms will be applied to predict each feature of the Arabic word. Besides, to support the analysis, different performance evaluation metrics will be adopted. Thus, this study seeks to answer the following questions:

1. How can machine learning algorithms predict Arabic word morphological features?
2. What are the features that mainly contribute to the prediction of morphological characteristics, e.g., gender (male, female); part of speech POS (noun, verb); number (singular, dual, and plural) for each?

3. What is/are the ML algorithm/s that perform better in predicting morphological features for Arabic words (gender, number, and parts of speech)?

The purpose of this study is to create an adaptable model that predicts Arabic word features (gender, part of speech, and number) using an ML algorithm. In addition to input - missing values in Birzeit University Lexicographic database. This study aims to investigate the role of Arabic word characters, specifically the letters at the beginning and the end of a word, to determine the morphological characteristics of words and to explore the ML algorithm's ability to develop an adequate tagging model, without using Arabic texts or dictionaries.

The study is organized as follows: Chapter 2 reviews the theoretical background and related works. Chapter 3 discusses the adopted methodology to achieve the study goals. Chapter 4 analyzes the study results and finally, Chapter 5, concludes the results and recommendations of this study.

Chapter 2: Related Work

Introduction

This chapter presents and reviews related works about Arabic morphological features, specifically that involve the adaptation of classical Machine Learning algorithms, the discuss encompasses POS taggers and morphological analyzers. There have been limited contributions of the Arabic language, with some approaches depending only on features of the word to predict its morphological properties.

2.1 POS tagging

There have been attempts that started with automatically assigning the morphosyntactic categories, such as part-of-speech (i.e., Noun, Verb, or Particle) of a word in a sentence based on contextual information, assuming that those words in the same syntactic contexts have the same part of speech (Kashefi, 2018). This approach is called “part of speech tagging,” also known as “grammatical tagging.”

Building such a POS tagger requires a corpus. Words are extracted as input for the tagger which determines the POS of the word as an outcome. Here, the corpus should be pre-labeled, which is called supervised POS tagging. In addition, the corpus used in POS tagging may not be pre-tagged, so the issue is treated as a clustering model. This is referred to as unsupervised POS tagging.

As described in the literature, POS taggers can be divided into three main groups: rule-based taggers, stochastic taggers, and hybrid taggers. For the Arabic language, different works have been implemented. The following paragraphs provide a background of these approaches emphasizing the models related to Arabic.

With **the rule-based taggers**, every word gets all possible corresponding tags from a lexicon that consists of a tagged bag of words, after that in case the word is not found in that lexicon, the word category will be predicted by a set of linguistic rules that are written manually by linguists. Evidently, this approach requires a linguistics background and extensive labor capacity. There have been different Arabic POS taggers developed, such as Arabic Morphosyntactic Tagger AMT, which uses pattern-based techniques, lexical, as well as contextual techniques. AMT achieved 91% accuracy on the 20,000-word testing corpus. (Alqrainy,2008)

Researchers are recently using other techniques that require lower human efforts known as the statistical/stochastic approach. The approach is based on extracted lexical and contextual probabilities from the corpus under the Markov Assumption, where predicted categories are identified based on previous word categories (Jurafsky & Martin, 2020). Multiple models adopt this technique, such as N-Grams which considers the probability of the current word tag given by the previous *n* words tag, that term unigram (n=1), bigram (n=2), and trigram (n=3). The Hidden Markov Model (HMM) considers the future tags as well as the previous tags (Kumawat & Jain, 2015). There are different machine learning techniques, other than the ones listed above, such as Support Vector Machine, decision trees, and Conditional Random Fields (CRF).

According to the statistical taggers' methodology, some words will be tagged with more than one category. Sometimes other words may not receive any tag. Therefore, **Hybrid approaches** were developed to eliminate this ambiguity, where rule-based and statistical approaches are combined. An example of hybrid Arabic POS taggers is the one developed by Khoja (2001). This tagger - APT tagger -combines statistical and rule-based techniques.

Also, Tlili-Guiassa (2006) tagger relies on rule-based and memory-based learning methods. Another example is Hadni (Hadni et al., 2013) who used rules and HMM methods.

2.2 Morphological Analyzers

Arabic words are constructed by combining morphemes, namely prefixes, suffixes, and stems which convey grammatical information. Analyzers segment a word into its morphemes and use them to derive the word features (e.g., POS, gender, number, person, voice, etc.) (Boudchiche et al., 2017).

Researchers have developed multiple Arabic morphological analyzers. It starts with extracting word morphemes in a process called *Morphological Segmentation.* Several analyzers worked on extracting word morphemes, which are classified based on the initial unit of analysis:

→ **Root-pattern morphology**, which depends on the word root and patterns for analysis, such as Gridach and Chenfou (2011),

→ **Lexeme-based morphology**, where analysis is based on extracting word stem, adopted in ElixirFM, ALMOR and AraComLex analyzers.

→ **Stem-based morphology**, which relies on grammatical specifications including stems, prefixes, suffixes, and patterns. BAMA, SAMA, Al Khalil, and SALMA analyzers were developed using this stem-based morphology approach. (Alothman & Alsalman, 2020)

Analyzers' performance relies on the methodology they adopt. The following paragraphs will discuss open-source analyzers.

The Al **Khalil** morphological analyzer can process non-vocalized and vocalized texts, developed in 2010 and updated in 2016. This analyzer uses root-based morphology, including root-pattern and syntactic features, which lists all possible tags of the stem and features, such as word prefix, suffix, and pattern stem with an addition to POS, gender, and number. The analyzer achieved 99.31% accuracy when tested on more than 72 million words with diacritics (Boudchiche et al., 2017), and 96% accuracy when tested over about 18 million words of KALIMAT dataset (El-haj and Koulali,2013).

The next open-source analyzer is **AraComLex**, developed in 2005 and updated in 2011. This analyzer supports MSA, where a lexicon is created to implement it. Lemma-based methods were used to implement the morphology. Regarding its efficiency, the analyzer achieved an 87.13% coverage rate on words from general news and an 85.73% coverage rate on semi-literary words. Using this analyzer, as well as other ones, a list of proposed tags is presented including word number, gender, case, and clitics (Attia et al.,2011). But this is not the same for the **MADAMIRA** analyzer, where the result features are fed to an SVM model to select the highly probable one. This tool combines MADA/SAMA morphological analysis and stem database, and the AMIRA POS tagger. Pasha et al. (2014) reported that this system was able to achieve 95.9% accuracy in tagging POS for 25K MSA words.

More recently, researchers worked on developing analyzers with a faster performance and a similar accuracy, such as **Farasa** (Abdelali et al., 2016; Darwish and Mubarak, 2016) and **YAMAMA** (Khalifa et al., 2016). According to Darwish et al. (2019) and Khalifa et al. (2016), both tools achieved a level of accuracy similar to MADAMIRA. The comparison was tested on MSA tweets. When tested in MSA tweets Farasa obtained an accuracy of 89.3% while MADAMIRA achieved an accuracy of 88% (Darwish et al., 2019). In a comparative study conducted by Khalifa et al. (2016) using the Penn Arabic Treebank (PATB), YAMAMA and MADAMIRA were evaluated for their performance on various tasks. Regarding part-of-speech (POS) tagging, YAMAMA attained an accuracy of 96.1%, while MADAMIRA achieved 96.8%. In terms of gender and number tagging, YAMAMA achieved an accuracy of 78.8%, whereas MADAMIRA achieved a higher accuracy of 86%.

Word segmentation, which is considered a core task for morphological analysis, aims at splitting words into prefixes, suffixes, and stems. It's an ambiguous task in the Arabic language since the same word can be segmented in multiple ways. Several morphological analysis tools use different datasets that have been segmented, such as FARASA and MADAMIRA, while others segment the data manually (Freihat et al., 2018).

Following the step of segmentation, analyzers list all possible word features based on a lexicon or a dictionary. These dictionaries comprise all words' stems and patterns in addition to their morphological information/features (Alothman & Alsalman, 2020). Some analyzers build their models using specific corpuses while others build their own corpus. Khalifa et al. (2020) claimed that analysis tools that are built using existing linguistic dictionaries have higher quality, and the quality of the analysis tools that build their own dictionaries are affected by the quantity and quality of data in addition to the method used.

Based on the previous discussion, we can conclude that these analysis tools are based on the availability of good lexicons, which can be considered a challenge to develop. In addition, as Alothman and Alsalman (2020) mentioned, the morphological analyzers did not adopt new techniques regarding classification. Besides, their process ended with presenting almost all the possible word features and left the user confused with a large list of outputs.

2.3 Machine learning and Arabic Word Tagging

Machine learning algorithms are used for different data science problems to reduce human efforts and capacities. Machine learning algorithms can be classified into two main groups. The first is **supervised algorithms** which deal with predetermined labels. Support Vector Machines, Neural Networks, Bayesian Networks, Decision Trees, and Naïve Bayes belong to this type of algorithm. The second type**, unsupervised algorithms**, is used to find patterns in data with unknown labels. Unsupervised learning can be used for clustering problems (Alzubi et al., 2018).

Machine learning algorithms have been employed in Arabic Natural Language Processing. Among these applications is Arabic Named Entity Recognition (NER) which classifies words in a text into different name classes (e.g., person, organization, sports). For instance, Salah & Binti Zakaria (2017) used multiple machine learning algorithms. Conditional Random Fields (CRF) were employed by Benajiba and Rosso (2008a), Zirikly and Diab (2015), and Abdul-Hamid and Darwish (2010). The Support Vector Machine (SVM) algorithm was utilized by Benajiba and Rosso (2008b). In a more recent study, Jarrar et al. (2022a) employed deep learning techniques to recognize Arabic named entities nested within each other.

Sentiment Analysis also applies machine learning algorithms. It worked on different Arabic texts and classified them into two or more categories (e.g., positive, negative, and neutral). The analysis used both a supervised learning approach (includes classification algorithms) and an unsupervised learning approach (includes sentiment lexicons) (Boudad et al., 2017; Abuaiadah et al. 2017). As Boudad summarizes, the best classification accuracy they reported reached 96.6%.

Machine learning is also becoming a critical tool for understanding the semantics. Recent research in (Al-Hajj and Jarrar, 2021a; Al-Hajj and Jarrar, 2021b) illustrated 84% accuracy in word-sense disambiguation.

In the field of tagging words with some morphological features, a variety of machine learning algorithms are used to predict these features in the form of a classification exercise. The classifier uses extracted features to build its model and predict the word's features. For example, HMM was applied by El Hadj et al. (2009) to Classical Arabic (Hijri texts) and the obtained accuracy was 96%. Random Forest was used for predicting the POS of MSA tweets, and the accuracy was 89%. (Darwish et al., 2018 a)

Other approaches combine both features extracted from the word itself – as well as morphological analyzers- and the features related to the word contextual information as well as the POS taggers. Abdulkareem and Tiun (2017) used features related to sentence form (N-gram words, next word, and word length) and features related to word form (first character, first two characters, first three characters, last character, last two characters, and last three characters). Darwish et al. (2014) in their study used similar features in addition to listing match, word template, and the position of the word in the sentence. The models were applied over different training datasets. Darwish et al. (2014) trained their study on sentences extracted from Wikipedia Alaljazaera.net articles, while Abdulkareem and Tiun (2017) relied on Arabic Tweets and Modern Arabic Text. The two studies adopted machine learning algorithms in developing morphological feature tagging models. The first study (Abdulkareem and Tiun, 2017) compared the performance of different algorithms; K-Nearest

Neighbor, Naïve Bayes and Decision tree models, and the best performance achieved by Naïve Bayes for MSA while ID3 for Arabic tweets, however metrics reached F1-score of 87.97%. In contrast, Darwish et al. (2014) trained conditional random forest (CRF) and they reported an accuracy of 98.1%.

Two other models proposed by (Mahafdah et al, 2014) and (Tnaji et al., 2021) combined two machine learning techniques. Both models used HMM features (tag of the previous and following words in the sentence) in addition to words' prefixes, suffixes, and length. Mahafdah et al (2014) used Quranic Arabic Corpus to employ K-Nearest Neighbors (KNN) and Naïve Bayes (NB). NB's best accuracy reached 91.77%, whereas KNN achieved 95.5%. The researchers enhanced the performance by combining the two algorithms (KNN and NB) using majority voting strategy and accuracy was increased to 98.32%. Tnaji et al. (2021) used NEMLAR dataset with about 500K words, the model based on HMM had an accuracy of 99%, while the one with Decision Tree had 97% accuracy. For combining the two techniques, the model uses the HMM if it were able to find the word tag using the calculated probabilities that can be found in the vocabulary. Otherwise, the Decision Tree model will be used to predict the tag based on word suffixes, prefixes, and word's length. The combined model was evaluated on WikiNews corpus and achieved about 96.06% accuracy.

Nowadays, deep learning is used in the same vein, and multiple studies were applied to the Arabic language. They investigated the neural network effectiveness in tagging Arabic words' morphological features. The Advantage of using deep learning is solving morphological segmentation issues, as it does not require feature engineering. For example, Plank et al. (2016) study evaluated the performance of Bidirectional long short-term memory (biLSTM) over twenty-two languages, including Arabic, the model input was Arabic words embedding vectors and character embedding vectors. The best-achieved model for Arabic was with an accuracy of 98.91%. Also, Alrajhi et al. (2019) investigated the neural network's performance for Quranic Arabic words. They compared the performance of Word2V and LSTM using the input of words and morphemes. According to the study, word decomposition leads to higher precision, while LSTM morphemes achieve the highest accuracy level of 99.72%.

Different POS taggers were developed; however, they were mostly developed for specific projects or objects. Thus, we cannot conduct a valid comparison or determine a standard POS tagger for Arabic words. Studies were conducted to compare different POS taggers. One of these was developed by Alashqar (2012). He used the Quranic Arabic Corpus to compare the performance of N-Gram, Brill, HMM, and TnT taggers. He performed this comparison on both words with diacritics and without diacritics. In his study, Brill tagger achieved the highest accuracy for words without diacritics, and N-gram models were the best for words with diacritics. A conclusion that we must mention is that POS taggers achieved a higher accuracy with non-diacritic Arabic words than words with diacritics, under the justification that diacritics increase the ambiguity and complexity of the model. A similar study was performed by Jacobsen et al. (2021) over multiple languages, including Arabic. The study compared Brill, TnT, SVM Tool, Stanford Tagger, HMM, BiLSTM/Plank, BiLSTM/Yasunaga, Flair, Meta-BiLSTM, and BERT- BPEmb. Flair performed the best within these models with an accuracy of 96.68%

With respect to the achievements of researchers in predicting a specific morphological feature, it is noteworthy that most efforts are focused on the POS of words, and others - such as AL Khalil - work on a bundle of features, but with aggregated evaluation. They focused on the overall performance of all features and did not specifically address one aspect. For Gender and Number features, Alkhairy et al, (2020) show separate performance per each feature, where gender and number can be predicted with an accuracy of 99.4% and 90.3%, respectively. The study of Darwish et al. (2014) that used the Random Forest algorithm attempted to improve these features prediction by including stem template, length of stem template, POS tag, suffixes as well as other features, which were extracted from 8,400 words from Penn Arabic Treebank (PATB). The model was able to predict gender and number with an accuracy of 95.6% and 94.9% respectively.

One of the fundamental issues in POS tagging is the availability of data used in training the model. Limited resources of annotated Arabic texts could be considered a concern to build POS tagger. Based on the studies reviewed earlier, we can see that the data used are extracted from Arabic tweets, traditional Arabic texts, or the Quran. The use of different resources may enhance POS tagger development, but these resources need to be reliable. Although the Quranic text achieved a competitive result, many of its words are no longer found in daily or modern Arabic. Further, in recent decades, new words have appeared in the language.

Lastly, the issue is the size of training data. Albared et al. (2011) show a positive effect of increasing the data size on model accuracy, where they trained an HMM POS tagger on different sizes of Quranic Arabic words and Modern Standard Arabic words. Moreover, plank et al. (2016) showed that bi-LSTM needs more data size than Markovian models for non-Indo-European languages.

Chapter 3: Methodology

Introduction

Given a word without context, our aim is to predict the morphological features of this word. This section describes the dataset used for training and testing and the variables extracted from each word. It also discusses the methodology adopted to predict Arabic words' morphological features without context. Following the steps of a classification model, it started with data preprocessing and preparation. After that, different classification algorithms were applied to predict various word characteristics. The aim of the study was also to evaluate different classification algorithms. To achieve this goal, the data is divided into training and testing sets and uses different evaluation criteria to conduct the comparison.

Data Preparing
Training Classification Model
Testing and Evaluation

Figure 1: Classification Process Steps

3.1 Data Set

To achieve the study goals, the linguistic database was developed by Jarrar & Amayreh (2019), which includes Arabic words from 150 Arabic dictionaries. A subset of the database was used for predicting Arabic word morphological features. It consists of about **7.9 million Arabic words** (one without diacritics and one with diacritics) which are labeled with POS, gender, and number categories. Table 1 shows the data.

Table 1: Original Dataset Snapshot

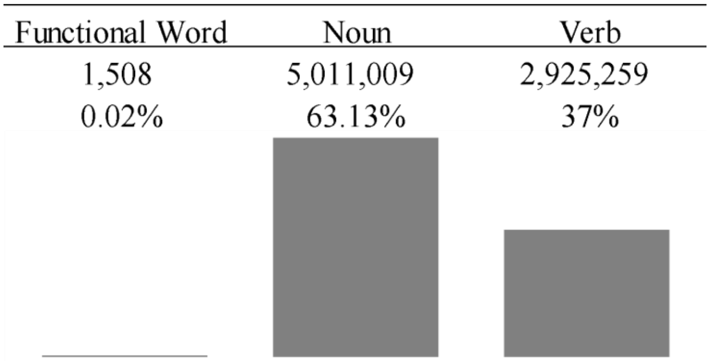
Word_with_Diacritics	Word_without_Diacritics	POS	Number	Gender
نَبِيلَاتِ	نبيلات	Noun	Plural	Female
فَسَادِ	فساد	Noun	Singular	Male
مِكْيَفَ	مكيف	Noun	Singular	Male
فُنُونَ	فنون	Noun	Plural	Male
احْتِفَاءَاتٍ	احتفاءات	Noun	Plural	Female
فَتَّانُ	فتان	Noun	Singular	Male

3.2 Variables

3.2.1 Dependent Variables:

1. ***Part of Speech (*****POS)*****:*** In the dataset, which consists of 7.9 words, each word is labeled with POS, which can be Verb, Noun, or Functional Word. Table 2 illustrates the distribution of the POS categories. Due to the low frequency of words that belong to the functional word category (which is normal in every language), the model can predict only Noun and Verb tags using 7.9 million words.

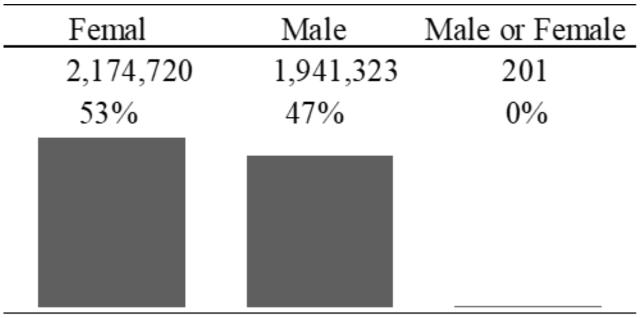
Table 2: Part of Speech Variable Distribution



Functional Word	Noun	Verb
1,508	5,011,009	2,925,259
0.02%	63.13%	37%

2. ***Gender***: In table 3, words in the dataset are labeled with Gender (masculine, feminine, and masculine or feminine). However, the model has been built using words with masculine and feminine tags; to overcome the technical issues resulting from the low frequency of "masculine or feminine" words. Hence, the dataset size to predict Arabic word gender is roughly 4.1 million words.

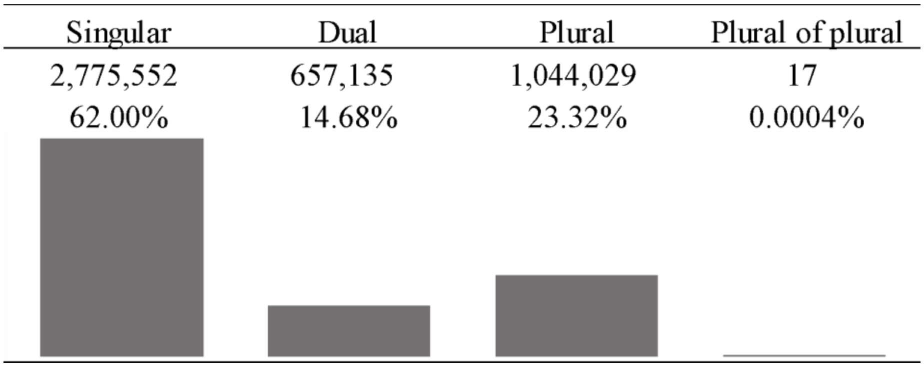
Table 3: Gender Variable Distribution



Femal	Male	Male or Female
2,174,720	1,941,323	201
53%	47%	0%

1. ***Number***: Each word in the dataset is labeled with a number, which can be: Singular, Dual, Plural, and Plural of Plural. In order to avoid the technical issues mentioned for word POS and Gender, the model has been built using Singular, Dual, and Plural words. Hence, the dataset size used is about 4.47 million words. Table 4 shows the number variable distribution.

Table 4: Number Variable Distribution



Singular	Dual	Plural	Plural of plural
2,775,552	657,135	1,044,029	17
62.00%	14.68%	23.32%	0.0004%

3.2.2 Independent Variables:

This study seeks to predict three morphological features (POS, Gender, and Number). The independent variables that were used for the predictions were extracted from the first and last letters of words; based on the length of the word, three characters were examined. For example, only the first and last two characters of a 5-character word were used to develop the prediction model, whereas, for an 8-character word, three characters at the beginning and end of the word were used. Word length is also considered an independent variable. Diacritic words and words without diacritics were both extracted in the same way. In the diacritic words, we selected each letter and its corresponding diacritics, if any. To give an example, the word “سيكتبونها” consists of 9 letters, so we will be able to use the three letters from the beginning and the end of the word. The independent variables are the word length (9), the first -three-letter; “س” as the first letter, “ي” as the second letter, and “ك “as the third letter. The last three- letters, “ن“ ,”هـ” and “ا”. Table 5 shows the characters' extraction for different lengths of words, and Table 6 provides further examples of word's feature extraction.

Table 5: Characters Extraction Model for different word's length

Length of word (Characters)	First Character	Second Character	Third Character	Third at Last Character	Second at Last Character	Last Character
1						*
2 - 3	*					*
4 - 5	*	*			*	*
6 and more	*	*	*	*	*	*

Table 6: Characters Extraction Examples for different word's length

Word	Length of word (Characters)	First Character	Second Character	Third Character	Third to Last Character	Second to Last Character	Last Character
ر	1						ر
حظ	2	ح					ظ
حَظ	2	حَ					ظ
شكر	3	ش					ر
شَكَرَ	3	شَ					رَ
قرية	4	ق	ر			ي	ة
قَرْيَة	4	قَ	رْ			يَ	ة
متجبر	5	م	تَ			ب	ر
مُتَجَبِّرُ	5	مُ	تَ			بِّ	رُ
متجاوز	6	م	ت	ج	ا	و	ر
مُتَجَاوَزٌ	6	مُ	تَ	جَ	ا	وَ	رٌ

There is a four-part distribution of features according to word length: the first set includes word length and the last letter, the second set includes word length and both the first and last letters, the third set includes the first and last two letters as well as word length, and the fourth set includes the first and last three letters and the word length. Thus, we worked on developing a different model for each set, each of which predicts one of our dependent variables (POS, gender, and number). Table 7 illustrates the distribution of each morphological feature for each model, where we can notice the very low number of attributes within the first model (Model 1). As a result, we focused only on the other three models, i.e., Model 2, Model 3, and Model 4.

Table 7: Sub models classes Distribution for Number, Gender, and POS morphological

Dataset	POS			Gender			Number			
	Noun	**Verb**	**N/A**	**Female**	**Male**	**N/A**	**Singular**	**Dual**	**Plural**	**N/A**
Dataset 1	2	13	0	0	0	46	2	0	0	44
Dataset 2	125,432	49,161	0	16,054	96,591	62,227	113,682	2,196	10,965	48,141
Dataset 3	1,672,096	989,571	0	561,848	900,392	1,200,014	1,330,473	142,279	194,287	995,293
Dataset 4	3,213,479	1,886,514	0	1,596,818	944,340	2,559,245	1,331,395	512,660	838,777	2,417,565
Total	5,011,009	2,925,259		2,174,720	1,941,323		2,775,552	657,135	1,044,029	
Total (Excluding Dataset 1)	5,011,007	2,925,246		2,174,720	1,941,323		2,775,550	657,135	1,044,029	

3.4 Characters Encoding

There are machine learning algorithms that are incapable of dealing with categorical data without transforming them into numerical variables. As a result, it needs to convert to categorical variables; First Character, Second Character, Third Character, Third at Last Character, Second at Last Character, and Last Character. Here, a one hot encoding method is adopted., which converts the categorized value to binary values (0 and 1). The procedure starts with preparing a glossary for each feature's characters, and a separate column is prepared for each value of these features. The resulting columns combine both the character (with or without diacritics) and its position in the word. After that, each record is labeled with 1 for the column containing its feature, and 0 for other columns. For example, the word “جَاءَكُم” will be labeled with 1 for columns First_Character_جَ, Second_Character_ا, Second_at_Last_Character_كُ and Last_Character_م. And other columns that express other characters at different positions will be labeled with 0. The result is a binary vector. Similar schemes are presented for words with non-diacritical letters, but with a lower number of columns (variables) due to the absence of diacritical marks.

The dataset sizes were updated with character variables after data encoding, and Table 8 summarizes the data shape that was used to predict each of the three morphological features. The number of words refers to the number of rows or records in each dataset for each of the three morphological features. The number of features reflects the number of columns in each data set (for words with and without diacritics).

Table 8: Datasets Shape after Characters Encoding

	Number Data set	**Gender Data set**	**POS Data set**
No. of words	4,476,714	4,116,043	7,937,730
No. of features- diacritical words	1,553	1,525	1,213
No. of features-non diacritical words	204	204	208

3.5 Splitting

For machine learning to develop an efficient classification model, the data should be split into training and testing sets. **Training set** is the input data from which the model is learned, and data is labeled with the target categories. **Testing data** is the part of data that is used to evaluate the generated model, where the model is applied to the unlabeled testing set and the output is compared with the original labels. Different techniques are available to split data into these two sections, such as simple random sampling and stratified sampling. In simple random sampling, records are selected randomly from the dataset. However, this method does not take into consideration unbalanced distributions. Stratified sampling ensures that the data sets reflect the diversity of the population. This is done by clustering the data according to characteristics and drawing samples from each cluster.

It is also important to determine the size of the testing and training sets. There is no ideal size for training or testing sets, as it depends on multiple factors related to data size and the number of variables.

For this study, a stratified sampling method is implemented for selecting training and testing sets due to imbalanced classes distribution in the different models. In this method, the same percentage of each class is drawn, to ensure the presence of each class in both the training and testing stages.

In addition, we implemented different splitting ratios; (70% for training and 30% for testing), (80% for training and 20% for testing), and (90% for training and 10% for testing). The generated results were then compared.

3.6 Incremental Learning

Incremental learning (also called online learning) helps in applying machine learning algorithms without storing the whole data. Instead of training the algorithm on all data at once, it is applied to a stream of data, where knowledge from the old data is stored and updated with new data. As a result, if new data is added, there is no need to reuse the previous and new data. This method helps to emerge problems related to data size and limited processing power and resources.

For this study, models need to be built using dataset sizes ranging from 100 thousand records to 5 million records. This is due to limitations in computer storage and processing power. Consequently, incremental learning is used, where data is split into batches and algorithms are applied incrementally to each batch.

One of the key aspects of batch incremental learning is batch size. A big-batch size needs more resources, while a small-batch size causes noise in the modeling process. Thus, a trade-off between resources and efficiency. For this study model, the batch size is selected for RAM to handle. Model 2 is applied in one batch that includes all datasets at once, as the data size is not too big, and RAM can handle. In contrast, data sets for Models 3 and 4 split into mini batches. The batch size has an average number of 202,000 records. Table 9 shows further details of batch size and number.

Table 9: Mini-Batches Number and Sizes for Number, Gender, and POS features

	Number		**Gender**		**POS**	
	Batches Number	Batch Size	Batches Number	Batch Size	Batches Number	Batch Size
Model_2	1	126,843	1	112,645	1	174,593
Model_3	8	208,379	7	208,891	15	177,444
Model_4	13	206,372	12	211,763	25	203,999

3.6.1 Tools for Incremental learning

Various machine learning algorithms adapt to incremental learning, so we used some of these algorithms. The following section discusses each of these algorithms and the way each algorithm was applied incrementally.

3.7 Machine Learning Algorithms

Machine learning algorithms can be classified into two main categories: (1) supervised algorithms, which classify data with labeled classes; and (2) unsupervised algorithms, which work with unlabeled data trying to find out hidden structures. As our data set has already been labeled, prediction models will be built using the supervised algorithms, including Support Vector Machine (SVM), Logistic regression (LR), Naïve Bayesian (NB) and Random Forest (DT). These algorithms are as follows:

3.7.1 Support Vector Machine (SVM)

Support vector machine is a linear classifier for binary classes. This classifier firstly plots input data in a n-1 dimensional space, then it works to find the optimal linear surface that separates the two categories - the hyper line. Although it is a linear classifier, this algorithm can also handle non-linear data as well as using a kernel. The kernel can be linear, Polynomial, Radial bias function (RBF) and sigmoid. Further, SVM uses a cascade manner to handle multiclass classification. In other words, we built the N (N-1) classifier model for N categories, or it uses one-many classifications, where models are classified for one class against those not in that class.

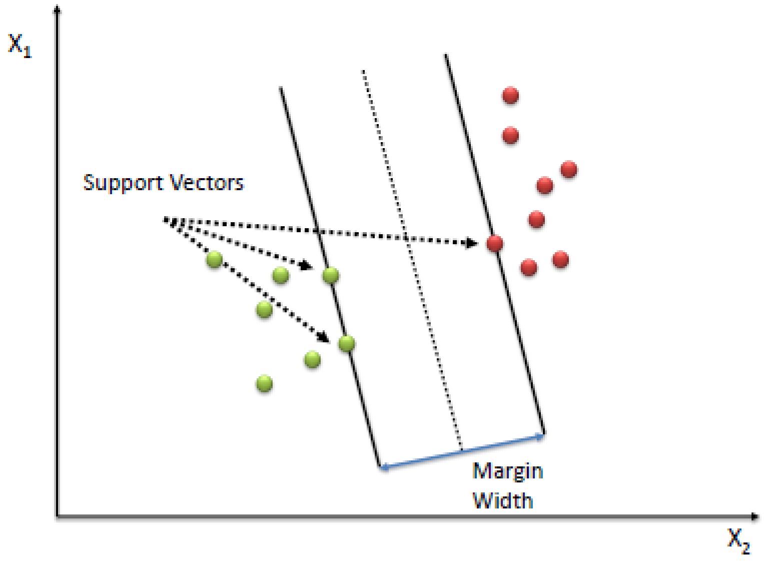


Figure 2: Linear SVM for simple two-class classification with separating hyperplane (Sayad,n.d.)

In Figure 2, we can see a scatter plot for an SVM model. The line that separates points into two categories is the hyperplane, while the points that are close to the hyperplane are support vectors and we can notice that these are the ones that define the orientation and position of the hyperplane. To find the optimal hyperplane, the linear surface should separate the two groups efficiently by maximizing the distance between this line and the nearest input in the space - called the *margin*. However, this hyperplane has to classify points correctly with non-overlapping classes and minimize misclassification. These parameters are incorporated into (1) Cost function: which refers to Hinge loss for SVM. In order to make the prediction accurate, the SVM model works on minimizing this hinge loss. (2) Regularization: is an SVM modification to reduce overfitting in the model and ensure avoiding any misclassification. It is added to the loss function as shown in Equation 1(Liu, 2020).

$$L(w) = \sum_{i=1} \underbrace{max(0, 1 - y_i[w^T x_i + b}_{\text{Loss function}}]) + \underbrace{\lambda||w||_2^2}_{\text{regularization}} \qquad (\textit{Equation } 1)$$

λ=1/C (C, regularization coefficient)

Different regularization functions can be used:

1- **L1 Regularization:** calculated by adding L1 penalty which equals the sum of the absolute value of coefficients. This regulation shrinks coefficients to zero and eliminates the variables that are not important. It plays a feature selection role.

2- **L2 Regularization:** calculated by adding the L2 penalty which equals the sum of the square value of coefficients. This penalty reduces the size of variable coefficients but unlike L1 regulation, it does not remove any coefficient.

3- **Elastic Net:** this regulation combines L1 and L2 regulations.

To apply this regularization, a parameter called alpha (α) controls the weight for each penalty using Equation 2 and it has values between 0 and 1. Here, alpha with zero value gives all weights to L2 Regularization, while alpha with value of 1 gives all weights to L1 Regularization.

$$Elastic\ Net\ Penalty = (alpha \times L1\ Penalty) + \big((1 - alpha) \times L2\ Penalty\big) \quad (Equation\ 2)$$

Stochastic Gradient Descent (SGD) is one of the methods that solve SVM algorithms. SGD works to find the minimum function value. For SVM, SGD aims to find the minimum hinge loss using its derivative function, where the minimum value is the one that has a zero slope. It starts with choosing an arbitrary point on this function and calculating its slope, known as gradient. Then it moves a step towards minimizing the function value and calculating the gradient. So, it keeps traveling down until it finds the minimum value.

A key parameter in SGD is the step size, calculated by multiplying the gradient with a learning rate. A big step size may lead us to skip the minimum value, while a small step size might increase the computation process time. So, the challenge is to select a proper learning rate. The learning rate has a small positive value, and it often ranges between 0 and 1. One approach to find a good value for the learning rate is a grid search for the optimal value using a logarithmic scale for values between 0.1 and 0.1^6.

Incremental learning is adapted by the SVM algorithm, where it does not require access to the original data, instead, it preserves knowledge from previously training data. For incremental SVM, this is the support vector as decisions are dependent on it.

In order to deal with the dataset in this study, large-scale imbalanced data, a linear incremental support vector machine algorithm has been applied using a stochastic gradient descent algorithm with hinge loss type. A grid search was also implemented for the function hyperparameters:

- **Regularization:** Net Elastic regularization with alpha values [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- **Learning Rate:** logarithmic scale for values [1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0].

3.7.2 Logistic regression (LR)

Logistic regression is a binary classifier, it uses the sigmoid function (Equation 3) to convert any probability to a number between 0 and 1. After that, a threshold is used to classify the inputs, where the observation would be classified with class"1" if the probability function returned a value exceeding the threshold, and class "0" if the function output is lower than the threshold.

$$f(x) = \frac{1}{1 + e^{-x}} \qquad (Equation\ 3)$$

Even logistic regression classifies binary classes as it can be extended to classify multiclass variables using the aforementioned techniques for SVM classifier; one-to-many, and one-to-one classification.

For logistic regression, we need to select a model that fits the data the best, in other words, the model with the least error. For logistics regression, this error is called logistic loss (Equation 4). The main goal is to minimize this metric. As discussed in section 3.7.1, SGD was implemented for this target.

In order to avoid overfitting problems resulting from features weights, a “Regularization term” is added to the function; the ways to compute this term are the same as referred to in the previous section (Support Vector Machine).

$$Log\ loss\ function = -(y\ log\ log\ (p)\ + (1-y)(1-p)) \qquad \textit{(Equation 4)}$$

Where:

- Y: Actual output
- P: Probability predicted y Log regression

3.7.2.1 Incremental Logistic Regression

Incremental learning adapted by logistic regression algorithm, where it does not require access to the original data, instead, it keeps updating the gradient per each batch.

In order to deal with the dataset in this study, large-scale imbalanced data, an incremental logistic regression algorithm has been applied using a stochastic gradient descent algorithm with log loss type. A grid search was implemented for the function hyperparameters:

- **Regularization:** Net Elastic regularization with alpha values [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
- **Learning Rate:** logarithmic scale for values [1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0].

3.7.3 Naïve Bayesian (NB)

The Naïve Bayesian classifier is one of the simplest classification algorithms. This algorithm is considered a "probabilistic classifier" as it relies on the Bayes theorem. It assumes independence between the features for a given class. In other words, there is no relationship between the occurrence of specified features and other features for the same class.

Naïve Bayesian determines the class of any observation by calculating "posterior probability" for each class. These probabilities are calculated using the following Equations 5: (Rish, 2001)

$$p(c/x) \ = \frac{p(x/c)\ P(c)}{p(x)} \qquad (Equation\ 5)$$

Where:

- p(c/x) is the posterior probability of specific class c given feature/s x.
- p(x/c) is the probability of a feature x given class c.
- p (c) is the probability of class c; it is also called the posterior probability
- p (x) is the probability of feature x.

Then, label the observation with the class that has the highest probability, which is expressed mathematically using Equation 6.

$$y \ = argmax_y p(y) \prod_{i=1}^{n} p(y) \qquad (Equation\ 6\ )$$

The disadvantage to this equation is it is biased to data distribution. So, a class with low frequency will get a lower accuracy, and probabilities will be biased to the majority class. Therefore, for this study, The Complement Naive Bayes is implemented to overcome the non-uniform distribution in data classes, where probability is computed as shown in equation (7). It can be noticed that this probability is the inverse of equation (6), so it calculated the probability of occurrence of all classes other than the target one.

$$y \ = argmax_y p(y) \prod_{i=1}^{n} \frac{1}{p(y)} \qquad (Equation\ 7\ )$$

3.7.3.1 Incremental Naïve Bayes

Naïve Bayes algorithm adapts incremental learning; the model does not revisit the old data, but it relies on the stored knowledge and parameters kept updated with new data to find the final posterior probabilities. (Ren & Lian ,2014) incremental learning using naïve bayes explained through the Diagram 3. For this study dataset- large imbalance data- we implemented the incremental naïve bayes.

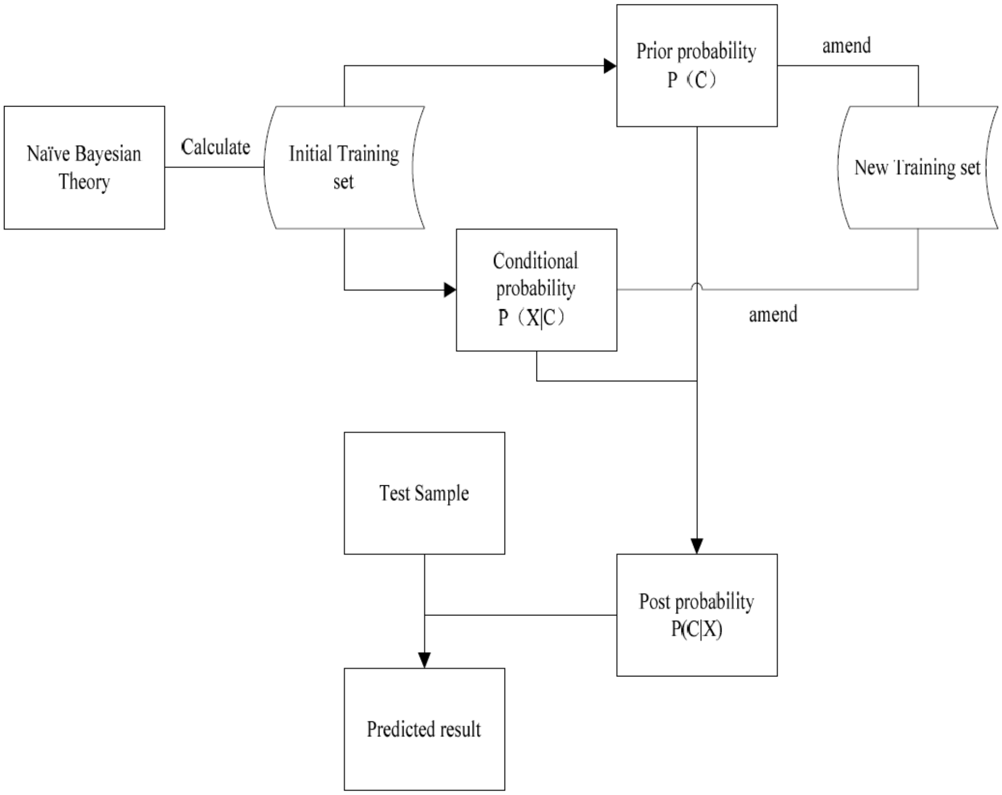


Figure 3: Flow chart of incremental Naïve Bayesian classification, (Ren & Lian ,2014)

3.7.4 Random Forests (RF)

Random forest is one of the machine learning algorithms that helps in regression and classification issues. It works on creating a number of samples of data, where attributes are drawn with replacement. Then decision trees are built using these samples which result in a set of trees “forest”. In the end, the majority of the votes for these trees will be the final prediction. Models were trained on two-thirds of each data sample, while the remaining one-third - known as the out-of-bag (OOB) sample - was used in testing the trees that were produced using the other trees. As a result, this strategy ensures random forest model validation, as well as bootstrapping and cross-validation.

To clarify how these decision trees are made, the algorithm rebuilds the data into a tree form through well-defined true/false queries. This tree consists of three main types of nodes: root node, internal node, and leaf node. These nodes have queries to be answered, where the root node is the starting part of the tree, the internal node has incoming and outgoing edges, and the leaf one has only incoming edges. Instance labeled by putting it firstly in the root, then it gets through the tree nodes until it reaches a final leaf, so it is labeled with this leaf class. In building the tree, nodes split in a way to ensure the impurity of the sub-nodes using different calculations, Entropy and Gini.

- the formula for calculating the Gini:

$$Gini = \sum_{i=1}^{n} {p_i}^2 \qquad (Equation\ 8)$$

$$p_i\text{: } the\ frequentist\ probability\ of\ a\ \ class\ `i'\ in\ data$$

- the formula for calculating entropy:

$$Entropy = -\sum_{i=1}^{n} p_i \, log_2 p_i \qquad (Equation\ 9)$$

p_i: *the frequentist probability of a class 'i' in data*

Random forest is a good predictor for big datasets that have a large number of features.

3.7.4.1 Incremental Random Forest

The classic random forest model is extended to be applied incrementally, where the forest is not built from all the data at once, instead, it is built from batches of data and updated when a new batch arrives. The split calculations are updated each time the model receives a new batch.

For this study case, an incremental random forest has been implemented to handle the size of our data. The implementation applied the different split calculations: Gini and Entropy, to find the most appropriate hyperparameter.

3.8 Performance Evaluation Metrics

After applying different machine learning algorithms to predict our study word characteristics, a comparison is made using several metrics of evaluation, such as Recall, Precision, Accuracy, Confusion Matrix, F1-Score, and Area Under the Curve of Receiver Characteristic Operator (AUC-ROC). These metrics are defined in this section:

3.8.1 Accuracy

Accuracy is considered as the first option to use for such evaluations. It is calculated as the ratio between the number of correct predictions to the total number of predictions as shown in Equation 10.

$$Accuracy = \frac{\#\, correct\ predictions}{\#\, total\ predictions} \qquad (Equation\ 10)$$

Although the accuracy measure is simple to calculate and interpret, it may be deceptive or misleading when used with an imbalanced date. The reason is it does not distinguish between correct predictions for each class. As a result, additional metrics that are more appropriate for the study's imbalanced data were used and discussed in the following sections.

3.8.2 Confusion Matrix

Confusion matrix shows a cross tabulation for predicted and actual values. As shown in Table 10. the matrix presents four terms:

- True positive (T_p): the number of positive instances that were predicted correctly.
- False positive (F_p): the number of positive instances that were predicted incorrectly.
- True negative (T_n): the number of negative instances that were predicted correctly.
- False negative (F_n): the number of negative instances that were predicted incorrectly.

The same terms will be also used to find other metrics, such recall, precision, F1-score and AUC-ROC Curve.

Table 10: Confusion Matrix

		Predicted	
		Positive	Negative
Actual	Positive	T_p	F_n
	Negative	F_p	T_n

3.8.3 Recall

This metric presents the ratio of actual positive records that are correctly classified. In other words, it shows the model's ability to find all relevant instances in a dataset. See Equation 11 for binary classification.

$$Binary\ Classification\ Recall \quad = \frac{True\ positive\ (T_p)}{True\ positive\ \left(T_p\right) + False\ negative\ (F_n)} \qquad (Equation\ 11)$$

$$= \frac{True\ positive\ (T_p)}{Total\ Actual\ Positive}$$

For Multiclass classification, we find a set of binary problems (one class vs other classes) and calculates their metric, then average is calculated. Here there are two specifications: macro recall and micro recall. Macro recall calculates the average of metrics without considering the size of each class as shown in (Equation 12) and micro recall is the sum of true positives for all the classes divided by the actual positives, so it expresses the weight of each class. See Equation 13. As a result, micro recall was selected as an evaluation method for our imbalanced classes.

$$Multi-class\ classification\ Macro\ Recall = \frac{\sum_{k=1}^{K} Recall_k}{k} \qquad (Equation\ 12)$$

$$Multi-class\ classification\ Micro\ Recall = \frac{T_{p1} + T_{p2}}{T_{p1} + F_{n1} + T_{p2} + F_{n2}} \qquad (Equation\ 13)$$

3.8.4 Precision

Precision metric presents the ratio of positive predicted values that are predicted correctly. So, it indicates how precise the model is in predicting the positive instances. It is calculated as presented in Equation 14 for binary classification and Equation 15 for multi-class classification.

$$Binary\ Classification\ Precision = \frac{True\ positive\ (T_p)}{True\ positive\ \left(T_p\right) + False\ positive\ \left(F_p\right)} \qquad (Equation\ 14)$$

$$= \frac{True\ positive\ (T_p)}{Total\ Predicted\ Positive}$$

The binary precision equation was extended to find the average precision for the multiclass model, by finding the arithmetic mean of precision for each binary comparison, which is known as macro precision (Equation 15). The macro precision is the sum of true positives for individual classes divided by the sum of predicted positives for all classes as shown in Equation 16 . For this study, micro-precision was applied in the evaluation process.

$$Multi-class\ classification\ Macro\ Precision = \frac{\sum_{k=1}^{K} Precision_k}{k} \qquad (Equation\ 15)$$

$$Multi-class\ classification\ Micro\ Precision = \frac{T_{p1} + T_{p2}}{T_{p1} + F_{p1} + T_{p2} + F_{p2}} \qquad (Equation\ 16)$$

3.8.5 F1-Score

F1-Score, also named F-score or F-measure. It presents the average of recall and precision metrics, using harmonic means. So, it takes into account both metrics. It is calculated as follows:

$$F1 - score = \frac{2 \times Recall \times Precision}{Recall + Precision} \qquad (Equation\ 17)$$

This equation can be extended for multiclass classification to a Micro F1-Score or Macro F1-Score. To calculate Micro F1-Score, first find micro recall and micro precision. This measure takes into consideration each class size, as small classes have the same weight as large classes. Therefore, this measure is more suitable for datasets of various sizes. On the other hand, Macro F1-Score is the same as the accuracy metric, which ignores class sizes. As a result, the Micro F1 Score is adopted. (Hossin & Sulaiman, 2015)

$$Micro\ F1 - Score = \frac{2\ \times\ Average\ Recall\ \times Average\ Precision}{Average\ Recall + Average\ Precision} \qquad (Equation\ 18)$$

$$Macro\ F1 - Score = \frac{\sum_{k=1}^{K} True\ positive}{Grand\ Total} \qquad (Equation\ 19)$$

3.8.6 Area Under the Curve of Receiver Characteristic Operator (AUC-ROC Curve)

The ROC curve plots the True-positive ratio versus the False-positive ratio, and the AUC-ROC curve represents the area under the curve, which indicates how well the model performs at distinguishing different classes. In this metric, values range from 0 to 1. For example, AUC of 0.5 indicates that the model is unable to discriminate between classes, whereas AUC of 1 indicates that the model discriminates completely between classes. Figure 4 shows AUC-ROC Curve and ROC Curve

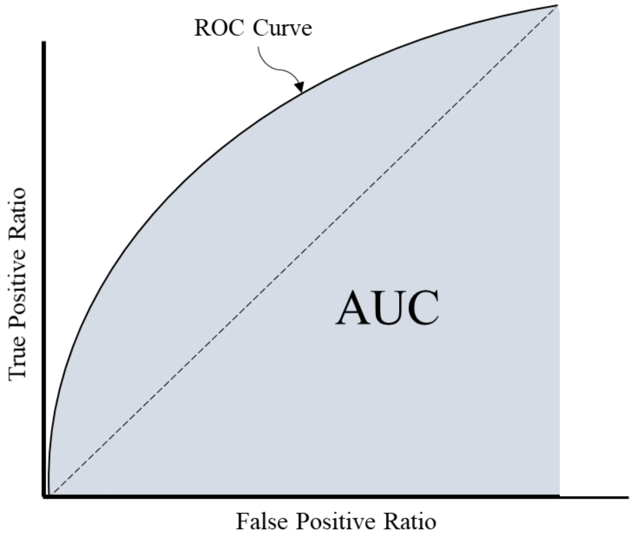


Figure 4: AUC-ROC Curve

Chapter 4: Results and Analysis

4.1 Introduction

Multiple models were developed to conduct comparisons and draw the desired conclusions. These models were implemented to predict different outcomes over multiple data sets using several splitting ratios and hyperparameters.

We walk through the steps that were discussed in Chapter 3. First, we encoded characters to create independent variables. After that, we split the data based on word length into three sub-datasets: a data set with 2–3 characters of word length, a data set with 4-5 characters of word length, and a data set for words with a length of more than 5 characters. After that, we created mini batches to handle our large data set (7.9 million words) and to implement the algorithms incrementally. We then proceeded with splitting these mini batches into training and testing sets, and we used three splitting ratios: 70%, 80%, and 90% for training sets and 30%, 20%, and 10% for the corresponding testing sets.

The four machine learning algorithms were implemented (namely, SVM, LOG, RF, and NB) and hyperparameters tuned separately for each sub-data set. In the end, an error analysis was performed by calculating different evaluation metrics.

Throughout this chapter, we describe all models developed to predict the three morphological features of number, gender, and part of speech. A comparison was conducted between the algorithms using different evaluation metrics. Further, we examined the character's importance in each prediction model.

4.2 Experiments for Words Without Diacritics:

There have been attempts to predict the three morphological features - POS, Number and Gender- of Arabic words without diacritics.

This study used a separate set of experiments to find out the best performance of each algorithm with respect to different hyperparameters. These hyperparameters (learning rate and regularization for SVM and LOG, splitting method for RF) are chosen in a way that maximizes the AUC_ROC metric. We tried to find the combination of these hyperparameters that generates the highest performance after conducting experiments using all parameter combinations. Table 11 shows the hyperparameters details that are used for each algorithm and Table 12 summarizes the best hyperparameters for the final models to predict the study's target features.

Table 11:Hyperparameters Details for Algorithms

Algorithm	**Package and Library-Python**	**Parameters**	**Parameters details**
SVM	Package: sklearn. linear_model Library: SGDClassifier	loss	hinge
		penalty	L1, Elastic net, L2
		alpha	[0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1]
		l1_ratio	[0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
		fit_intercept	Default (True)
		max_iter	Default (1000)
		tol	1e-3,
		shuffle	Default (True)
		verbose	Default (0)
		epsilon	Default (0.1)
		random_state	Fixed number "XXX"
		learning_rate	Optimal
		eta0	Default (0.0)
		power_t	Default (0.2)
		early_stopping	Default (False)
		validation_fraction	Default (0.1)
		n_iter_no_change	Default (5)
		warm_start	Default (False)
		average	Default (False)
		n_jobs	Default (None=1)
		class_weight	Based on trained data subset
RF	**Package: Incremental-trees** **Library: StreamingRFC [1]**	criterion	Gini & Entropy
		max_depth	Default (None)
		min_samples_split	Default (2)
		min_samples_leaf	Default (1)
		min_weight_fraction_leaf	Default (0.0)
		max_features	Default (sqrt)
		max_leaf_nodes	Default (None)
		min_impurity_decrease	Default (0.0)
		bootstrap	Default (True)

[1] https://pypi.org/project/incremental-trees/

Algorithm	Package and Library-Python	Parameters	Parameters details
		oob_score	Default (False)
		n_jobs	1
		random_state	fixed number "XXX"
		verbose	Default (0.0)
		warm_start	Default (bool=True)
		class_weight	Based on trained data subset
		ccp_alpha	Default (0.0)
		max_samples	Optional[int] = None,
		dask_feeding	Default (bool=True)
		n_estimators_per_chunk	Default (1)
		max_n_estimators	Default (10)
		spf_n_fits	Default (100)
		spf_sample_prop	Default (0.1)
LOG	Package: sklearn.linear_model Library: SGDClassifier[2]	loss	log
		penalty	L1, Elastic net, L2
		alpha	[0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1]
		l1_ratio	[0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
		fit_intercept	Default (True)
		max_iter	Default (1000)
		tol	1e-3,
		shuffle	Default (True)
		verbose	Default (0)
		epsilon	Default (0.1)
		random_state	Fixed number "XXX"
		learning_rate	Optimal
		eta0	Default (0.0)
		power_t	Default (0.2)
		early_stopping	Default (False)
		validation_fraction	Default (0.1)
		n_iter_no_change	Default (5)
		warm_start	Default (False)
		average	Default (False)
		n_jobs	Default (None=1)
		class_weight	Based on trained data subset
NB (Complement NB)	Package: sklearn.linear_model Library: ComplementNB	Smoothing parameter (alpha)	1
		Force alpha	Default (False)
		Class weight	Based on trained data subset
		fit_prior	Default (True)
		class_prior	Default (None)
		norm	Default (False)

[2] https://pypi.org/project/incremental-trees/

Table 12: Hyperparameters Tuning for words without diacritics.

		Number			Gender			POS		
Algorithm	**Parameters**	**Model_2**	**Model_3**	**Model_4**	**Model_2**	**Model_3**	**Model_4**	**Model_2**	**Model_3**	**Model_4**
LOG	Learning Rate	0.01	0.01	0.001	0.1	0.1	0.000001	0.01	0.001	0.00001
	Regularization	0.5	0	1	0.1	0.1	1	1	0.2	0.9
	Accuracy (80%)	89.60%	85.20%	82.80%	93.10%	87.40%	90.90%	79.40%	82.80%	84.70%
	AUC ROC (80%)	72.10%	80.50%	86.50%	96.20%	89.20%	89.90%	75.60%	81.80%	84.10%
SVM	Learning Rate	0.001	0.001	0.0001	0.1	0.1	0.00001	0.00001	0.000001	0.00001
	Regularization	0.3	1	0.8	0	0	0	0.6	0.5	0.5
	Accuracy (80%)	88.30%	84.40%	84.00%	93.10%	86.10%	91.30%	78.80%	82.50%	84.90%
	AUC ROC (80%)	67.00%	78.80%	87.20%	96.20%	89.20%	90.20%	75.00%	81.50%	84.20%
RF		Over_all_models			Over_all_models			Over_all_models		
	Splitting Method	Gini	Entropy		Gini	Entropy		Gini	Entropy	
	Accuracy (70%)	90.20%	90.20%		91.80%	91.80%		90.30%	90.40%	
	AUC ROC (70%)	90.00%	90.10%		91.70%	91.70%		89.50%	89.50%	
	Accuracy (80%)	90.20%	90.20%		91.80%	91.80%		90.50%	90.40%	
	AUC ROC (80%)	90.10%	90.00%		91.70%	91.70%		89.60%	89.70%	
	Accuracy (90%)	90.20%	90.20%		91.80%	91.80%		90.60%	90.50%	
	AUC ROC (90%)	90.10%	90.00%		91.70%	91.70%		89.90%	89.90%	

Moreover, other experiments were conducted to find the optimal splitting ratio between the training and testing data sets. Table 13 summarizes the highest ROC-AUC performance achieved at each splitting ratio. The models are compared based on the AUC-ROC metric (See subsection 3.8.6 for the definition). Splitting ratios do not appear to make significant differences in how the algorithm performs for words without diacritics. Yet, SVM performance fluctuates slightly at different splitting ratios.

Table 13: Splitting Ratio Tuning for words without diacritics.

	Algorithm	Split Ratio-Training			Max. AUC_ROC	Best Split Ratio
		70%	80%	90%		
Number	LOG	85.90%	85.92%	85.87%	85.92%	80%
	NB	80.15%	80.10%	80.17%	80.17%	90%
	RF	90.06%	90.07%	90.08%	90.08%	90%
	SVM	87.18%	88.18%	86.79%	88.18%	80%
Gender	LOG	89.15%	89.67%	83.68%	89.67%	80%
	NB	86.59%	86.62%	86.59%	86.62%	80%
	RF	91.71%	91.69%	91.70%	91.71%	70%
	SVM	89.56%	89.94%	85.08%	89.94%	80%
POS	LOG	82.82%	82.70%	82.82%	82.82%	70% / 90%
	NB	81.62%	81.60%	81.62%	81.62%	70% / 90%
	RF	89.55%	89.67%	89.88%	89.88%	90%
	SVM	78.17%	81.84%	82.10%	82.10%	90%

The results are shown in Table 14. The performance metrics of accuracy, recall, precision, and F1-score for each of the different algorithms were equal. This can be justified due to the usage of micro-metrics, i.e., micro-recall, micro-precision, and micro-F1-score. As discussed, the false positive value for a specific class shows the number of words that were predicted incorrectly. The false negative for that class indicates the number of words that were incorrectly predicted in other classes. False positives present the opposite error of false negatives. So, if class A has a false positive, then class B has a false negative, and vice versa. Consequently, the increase in false positives also leads to an increase in false negatives. Micro-metrics, as well as the micro-F1-score, are equal in precision and recall because of this.

It is noticeable that the Random Forest algorithm outperforms other techniques in predicting all word morphological features in terms of accuracy, recall, precision, F1-score, and AUC-ROC. It can predict Arabic words' morphological features without diacritical marks with an AUC-ROC between 89% and 91.71%. For the other algorithms, SVM performed better, followed by logistic regression, and then Naïve Bayes.

Table 14: Evaluation Metrics for models for words without diacritics

Number-Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	76.91%	83.96%	84.51%	90.25%
Precision	76.91%	83.96%	84.51%	90.25%
Recall	76.91%	83.96%	84.51%	90.25%
F1-Score	76.91%	83.96%	84.51%	90.25%
AUC-ROC Curve	80.17%	85.92%	88.18%	90.08%
Gender- Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	86.53%	89.47%	89.75%	91.78%
Precision	86.53%	89.75%	89.47%	91.78%
Recall	86.53%	89.75%	89.47%	91.78%
F1-Score	86.53%	89.75%	89.47%	91.78%
AUC-ROC Curve	86.62%	89.67%	89.94%	91.71%
POS-Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	80.69%	83.75%	83.78%	90.56%
Precision	80.69%	83.75%	83.78%	90.56%
Recall	80.69%	83.75%	83.78%	90.56%
F1-Score	80.69%	83.75%	83.78%	90.56%
AUC-ROC Curve	81.62%	82.10%	82.82%	89.88%

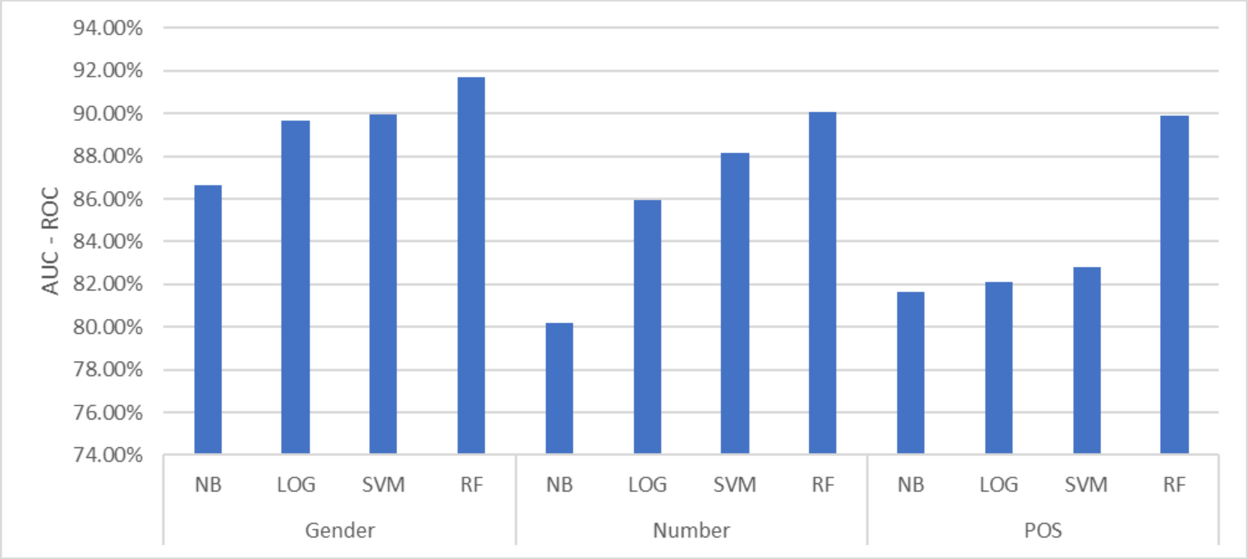


Figure 5: Evaluation Metrics for models for words without diacritics

To further examine the performance of the proposed models, Table 15 shows the confusion matrices for each morphological feature, in addition to the recall, precision, and F1-score obtained for each class. Clearly, in the result of Noun feature, the "Singular" class metrics are higher than those of the "Dual" and "Plural" classes. Furthermore, model metrics for the "Noun" class are higher than those of the other POS class, "Verb." Interestingly, the model can predict "female" words with the same accuracy as "male" words.

Table 15: Evaluation Metrics for models for words without diacritics/ per classes

			Predicted			
			Singular	Dual	Plural	All
Number	Actual	Singular	260,378	8,521	3,662	272,561
		Dual	3,552	58,061	4,101	65,714
		Plural	7,800	11,064	85,543	104,407
		All	271,730	77,646	93,306	447,682

			Predicted		
			Female	Male	All
Gender	Actual	Female	405,346	29,602	434,948
		Male	38,279	349,991	388,270
		All	443,625	379,593	823,218

			Predicted		
			Noun	Verb	All
POS	Actual	Noun	1,392,104	111,208	1,503,312
		Verb	118,646	758,938	877,584
		All	1,510,750	870,146	2,380,896

	Class Label	**Precision**	**Recall**	**F1-Score**
Number	Singular	96%	94%	95%
	Dual	75%	88%	81%
	Plural	87%	82%	84%

	Class Label	**Precision**	**Recall**	**F1-Score**
Gender	Female	92%	93%	92%
	Male	92%	90%	91%

	Class Label	**Precision**	**Recall**	**F1-Score**
POS	Noun	93%	92%	93%
	Verb	87%	87%	87%

4.2.1 Number

Based on our best model to predict the number of non-diacritic Arabic words (random forest), we can conclude which features are the most significant. The top twenty features are illustrated with their importance for each model of the three sub-datasets in Table 16. We can see that defining the word number is primarily dependent on the presence of the letters "ا" and "ي" at the end of words with 2–5 letters, whereas defining the number for words with more than five letters is dependent on the presence of the letters "ة" and "ن" as last letters. Other characters also have a notable role in determining the non-diacritic Arabic word, regardless of the word length. Similarly, the significance of the character "ت" at the end of the word changes as the word length increases. Furthermore, the character "ن" helps at various positions in the word, such as the first, second, and last characters.

The linguistic rules regarding Arabic word number feature, some plural words in Arabic can be formed by adding the suffixes "ين" ,"ون", or "ات" at the end of the word. Also, dual words are formed by adding the suffixes "ان" or "ين". These rules justify the importance of the letters "ن" and "ت" as the last letters and the presence of "ا" and "ي" as the second-last letters. In the same vein, making Arabic words dual or plural involves replacing or deleting the "ة" letter at the end of the word so that we can notice its influence on the importance of the features.

Table 16: Features Importance for Number morphological feature - using non-diacritic words

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	Feature Importance	Feature	Feature Importance	Feature	Feature Importance
1	last_letter_ا	38.56%	last_letter_ا	15.64%	last_letter_ن	12.02%
2	last_letter_ي	23.76%	last_letter_ي	7.40%	last_letter_ة	8.19%
3	**char1_ن**	5.70%	recent_letter_ا	5.70%	recent_letter_و	7.93%
4	last_letter_آ	3.88%	last_letter_ة	5.26%	last_letter_ت	7.56%
5	last_letter_ة	3.37%	**char2_ن**	5.09%	recent_letter_ا	7.01%
6	char1_م	2.22%	**last_letter_ن**	4.71%	length_of_word	4.65%
7	length_of_word	2.02%	**char1_ن**	3.72%	recent_letter_ي	3.44%
8	char1_ي	1.83%	length_of_word	2.82%	last_letter_ا	3.31%
9	last_letter_ن	1.50%	last_letter_ت	2.81%	brecent_letter_ا	2.59%
10	char1_ت	1.39%	recent_letter_ي	2.66%	last_letter_ي	2.56%
11	char1_أ	1.34%	char1_م	2.49%	brecent_letter_ت	2.35%
12	last_letter_ت	1.01%	recent_letter_و	2.49%	char3_ن	1.40%
13	char1_آ	0.85%	char2_ت	2.30%	brecent_letter_ي	1.38%
14	last_letter_ع	0.57%	char1_أ	1.46%	recent_letter_ه	1.33%
15	last_letter_ر	0.50%	last_letter_و	1.26%	recent_letter_ت	1.21%
16	char1_ش	0.45%	char1_ت	1.20%	last_letter_ه	1.10%
17	last_letter_ف	0.40%	recent_letter_ن	1.10%	char2_ت	0.99%
18	char1_غ	0.33%	char1_ي	0.94%	last_letter_ك	0.91%
19	char1_ص	0.32%	char2_ا	0.92%	char3_ت	0.89%
20	char1_و	0.32%	char2_ي	0.82%	last_letter_م	0.79%

4.2.2 Gender

Table 17 illustrates the final model with the highest performance for the gender feature and the importance of each feature across different word lengths. It is obvious that defining the word's gender is related to the presence of the character "ة " at the end of the word, especially with words with 2-3 characters and 51% importance. That is because most feminine Arabic words end with this letter. Moreover, the character "ت" has a considerable effect on predicting whether the word is masculine or feminine.

Table 17: Features Importance for Gender morphological feature - using non-diacritic words.

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	**Feature Importance**	**Feature**	**Feature Importance**	**Feature**	**Feature Importance**
1	last_letter_ة	51.92%	last_letter_ة	39.39%	last_letter_ة	16.26%
2	char1_ت	6.26%	last_letter_ت	4.74%	brecent_letter_ت	9.32%
3	last_letter_ي	3.56%	char1_م	4.50%	last_letter_ن	7.78%
4	length_of_word	3.52%	recent_letter_ت	4.19%	last_letter_ت	6.71%
5	last_letter_ت	3.52%	length_of_word	4.05%	recent_letter_ت	5.93%
6	char1_ي	3.24%	char2_ت	2.46%	recent_letter_و	5.27%
7	last_letter_ل	2.46%	char1_ت	2.18%	length_of_word	2.99%
8	char1_أ	2.45%	last_letter_ن	2.05%	recent_letter_ا	2.77%
9	last_letter_ن	2.18%	char1_ي	1.83%	last_letter_ي	2.38%
10	char1_م	2.15%	recent_letter_ا	1.58%	recent_letter_ي	1.76%
11	last_letter_و	1.91%	char1_أ	1.42%	char1_م	1.53%
12	last_letter_ف	1.42%	char2_ي	1.40%	last_letter_ه	1.43%
13	last_letter_ع	1.32%	last_letter_ي	1.18%	brecent_letter_ي	1.35%
14	last_letter_خ	1.09%	last_letter_ا	1.10%	char3_ي	1.34%
15	last_letter_ق	0.97%	last_letter_م	0.95%	char3_ت	1.32%
16	last_letter_ا	0.56%	recent_letter_و	0.92%	recent_letter_ه	1.24%
17	char1_ش	0.45%	last_letter_ر	0.84%	char2_ت	1.18%
18	last_letter_ء	0.45%	recent_letter_ي	0.65%	last_letter_ا	1.16%
19	char1_آ	0.44%	char1_و	0.64%	last_letter_ك	1.10%
20	last_letter_ط	0.41%	char1_ل	0.64%	char2_ي	1.06%

4.2.3 Part of Speech

Continuing to explore algorithms' performance, Table 18 shows the characters' share in making the decision in the random forest algorithm for the POS feature. The main point to mention is that these models, unlike number and gender models, consider the word beginning as well as the characters at the end of the word. To clarify, the characters "أ" ,"ي "and "ت "are the most common characters that affect predicting the POS feature of non-diacritic Arab words when they are the first character in the word with 2–3 characters. The character "ت" has a role in words with more than three characters and in different positions in the word.

The character "ة" at the end of the word is one of the characters with a similar significance for the three models. That's because Arabic verbs do not end with this letter.

Table 18: Features Importance for POS morphological feature - using non-diacritic words.

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	Feature Importance	Feature	Feature Importance	Feature	Feature Importance
1	char1_ي	20.96%	last_letter_ة	10.59%	length_of_word	5.18%
2	char1_أ	17.02%	char1_م	8.68%	last_letter_ة	5.14%
3	char1_ت	11.33%	char1_ي	5.85%	recent_letter_ا	3.73%
4	last_letter_ة	9.06%	recent_letter_ا	3.97%	recent_letter_ي	3.71%
5	char1_م	5.72%	char2_ت	3.48%	char1_ب	3.19%
6	last_letter_ي	5.26%	last_letter_ت	3.38%	char2_ا	3.05%
7	last_letter_ت	5.20%	char1_أ	3.21%	char2_ي	2.97%
8	length_of_word	4.44%	length_of_word	2.63%	char2_م	2.71%
9	char1_ا	4.19%	char2_ا	2.05%	char3_ت	2.59%
10	char1_ن	2.98%	last_letter_ي	1.95%	char1_م	2.52%
11	last_letter_و	1.10%	char2_ي	1.70%	char3_ل	2.46%
12	char1_آ	1.04%	char1_و	1.53%	char1_ي	2.43%
13	last_letter_ء	1.02%	char1_ب	1.51%	char2_ت	2.07%
14	last_letter_ى	1.01%	last_letter_ك	1.41%	char3_ا	2.03%
15	last_letter_ا	0.59%	recent_letter_ي	1.37%	last_letter_ت	1.99%
16	last_letter_أ	0.41%	last_letter_ا	1.36%	char1_و	1.90%
17	last_letter_ن	0.40%	char1_س	1.32%	brecent_letter_ا	1.81%
18	char1_ب	0.30%	char1_ت	1.32%	char1_س	1.61%
19	last_letter_ل	0.30%	char2_أ	1.31%	char2_ل	1.42%
20	char1_إ	0.29%	char2_ن	1.26%	char3_ي	1.41%

4.3 Experiments for Words with Diacritics:

A similar experiment was conducted on Arabic words with diacritics. Following the same steps that were outlined previously, we looked for hyperparameters that would maintain our data and maximize the AUC_ROC metric. Different experiments were conducted using all possible hyperparameter combinations. The results were compared and a hyperparameter combination was selected that yielded the highest performance. Table 19 summarizes the hyperparameters for SMV, LOG, and RF for morphological features (gender, number, and POS).

Table 19: Hyperparameters Tuning for words with diacritics.

		Number			**Gender**			**POS**		
Algorithm	**Parameters**	**Model_2**	**Model_3**	**Model_4**	**Model_2**	**Model_3**	**Model_4**	**Model_2**	**Model_3**	**Model_4**
LOG	Learning Rate	0.001	0.0001	0.00001	0.01	0.00001	0.000001	0.000001	0.0001	0.000001
	Regularization	0.5	0.9	0.5	0.1	1	0.1	1	1	0
	Accuracy (80%)	91.9%	92.2%	93.8%	92.5%	90.3%	93.7%	86.1%	89.0%	
	AUC ROC (80%)	79.6%	87.1%	94.6%	84.6%	91.3%	93.0%	86.5%	88.1%	94.1%
SVM	Learning Rate	0.01	0.0001	0.000001	0.01	0.000001	0.0001	0.00001	0.0001	0.00001
	Regularization	0.1	0.7	0.5	0	0.3	0	0.2	1	0.6
	Accuracy (80%)	90.8%	92.8%	94.4%	92.6%	90.0%	93.9%	86.1%		
	AUC ROC (80%)	79.2%	88.0%	95.3%	84.8%	91.4%	93.2%	86.6%	89.6%	94.2%
RF	Splitting Method	Over_all_models			Over_all_models			Over_all_models		
	Accuracy (70%)	Gini	Entropy		Gini	Entropy		Gini	Entropy	
	AUC ROC (70%)	96.9%	96.9%		93.6%	93.7%		95.4%	95.5%	
	Accuracy (80%)	97.0%	96.9%		93.6%	93.6%		94.9%	95.0%	
	AUC ROC (80%)	96.9%	96.9%		93.5%	93.6%		95.5%	95.6%	
	Accuracy (90%)	97.0%	97.0%		93.5%	93.5%		95.1%	95.1%	
	AUC ROC (90%)	97.0%	96.9%		93.5%	93.5%		95.7%	95.8%	

Using different splitting ratios (70%, 80%, and 90%) across different datasets, we trained and tested the different models and compared the results to find the splitting ratio that generated the highest performance as measured by ROC-AUC. Table 20 shows the highest ROC-AUC performance achieved for each splitting ratio. We can conclude that the splitting ratio has an unseen effect on the results.

Table 20: Splitting Ratio Tuning for words without diacritics.

	Algorithm	Split Ratio-Training			Max. AUC_ROC	Best Split Ratio
		70%	80%	90%		
Number	LOG	94.26%	94.36%	94.35%	94.36%	80%
	NB	86.80%	86.79%	86.83%	86.83%	90%
	RF	96.96%	97.01%	97.07%	97.07%	90%
	SVM	93.48%	94.49%	90.09%	94.49%	80%
Gender	LOG	90.47%	92.63%	92.31%	92.63%	80%
	NB	86.78%	86.82%	86.81%	86.82%	80%
	RF	93.60%	93.53%	93.49%	93.60%	70%
	SVM	92.65%	92.69%	92.67%	92.69%	80%
POS	LOG	92.59%	92.34%	92.79%	92.79%	90%
	NB	90.84%	90.82%	90.84%	90.84%	70% / 90%
	RF	95.00%	95.14%	95.33%	95.33%	90%
	SVM	91.33%	92.74%	91.72%	92.74%	80%

Table 21 lists the highest results reached through different experiments of each algorithm and lists all performance metrics: accuracy, recall, precision, and F1-score. As noted, and discussed in the previous section, the metrics are equal for each model due to using macro-metrics. Additionally, algorithms are tested on words with and without diacritics. Random forest achieved the highest performance over the remaining algorithms, with AUC-Roc reaching 97%, 93%, and 95% for number, gender, and POS, respectively. In addition, logistic regression and SVM perform similarly.

Table 21: Evaluation Metrics for models for words with diacritics

Number-Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	85.75%	93.67%	93.72%	96.96%
Precision	85.75%	93.67%	93.72%	96.96%
Recall	85.75%	93.67%	93.72%	96.96%
F1-Score	85.75%	93.67%	93.72%	96.96%
AUC-ROC Curve	86.83%	94.36%	94.49%	97.07%
Gender- Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	86.69%	92.47%	92.64%	93.65%
Precision	86.69%	92.47%	92.64%	93.65%
Recall	86.69%	92.47%	92.64%	93.65%
F1-Score	86.69%	92.47%	92.64%	93.65%
AUC-ROC Curve	86.82%	92.63%	92.69%	93.60%
POS-Hit Ratios				
Metric	**NB**	**LOG**	**SVM**	**RF**
Accuracy	90.54%	92.89%	92.92%	95.77%
Precision	90.54%	92.89%	92.92%	95.77%
Recall	90.54%	92.89%	92.92%	95.77%
F1-Score	90.54%	92.89%	92.92%	95.77%
AUC-ROC Curve	90.84%	92.74%	92.79%	95.33%

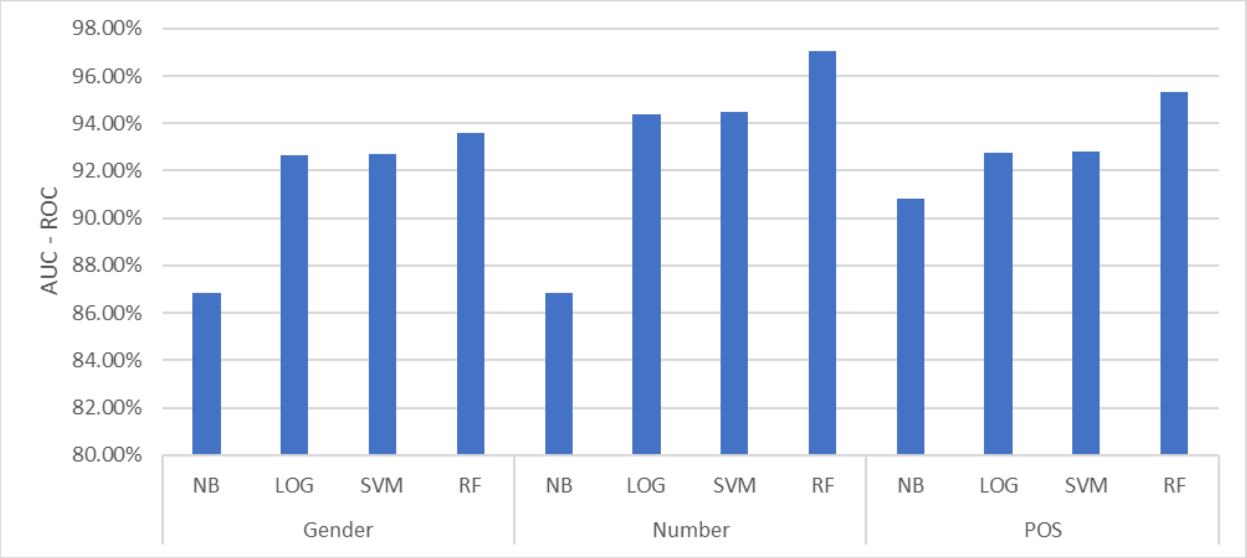


Figure 6: Evaluation Metrics for models for words with diacritics

Following through with discussing other details of the final models, Table 22 summarizes the confusion matrices for the best models of the three features and performance metrics per each class. Comparing the metrics for models that used words with and without diacritics, we can see that the metrics using diacritics increased for all classes. Also, the gap in prediction per class for the same feature decreases for number and POS features. However, it is still higher for the noun POS feature than for the verb POS feature. In contrast, the updated model achieved equal performance in predicting singular and plural words but lower performance for dual words.

Table 22: Evaluation Metrics for models for words with diacritics/ per classes

			Predicted			
			Singular	Dual	Plural	All
Number	Actual	Singular	64,310	524	880	65,714
		Dual	632	97,619	6,156	104,407
		Plural	1,138	4,293	272,130	277,561
		All	66,080	102,436	279,166	447,682

			Predicted		
			Female	Male	All
Gender	Actual	Female	616,815	35,604	652,419
		Male	42,746	539,655	582,401
		All	659,561	575,259	1,234,820

			Predicted		
			Noun	Verb	All
POS	Actual	Noun	486,068	15,037	501,105
		Verb	18,531	274,002	292,533
		All	504,599	289,039	793,638

Table 21(continued): Evaluation Metrics for models for words with diacritics/ per classes.

Number	Class Label	Precision	Recall	F1-Score	*F1-Score (without diacritics)*
	Singular	97%	98%	98%	*95%*
	Dual	95%	93%	94%	*81%*
	Plural	97%	98%	98%	*84%*

Gender	Class Label	Precision	Recall	F1-Score	*F1-Score (without diacritics)*
	Female	94%	95%	94%	*92%*
	Male	94%	93%	93%	*91%*

POS	Class Label	Precision	Recall	F1-Score	*F1-Score (without diacritics)*
	Noun	96%	97%	97%	*93%*
	Verb	95%	94%	94%	*87%*

4.3.1 Number

To investigate more about the characters and diacritics that most affect the model decision, Table 23 illustrates the top twenty most noteworthy features for predicting the number tag. The same characters as the ones that were shown in the previous model for the non-diacritics words are also presented here, i.e., "ن" ,"ي" ,"ا". However, the character "ة" was less significant. In the study of words with more than five characters, the letter "ت" with different diacritics, is ranked most prominently. This is due to the letter's role in defining plural words, which typically consist of more than five letters.

Table 23: Features Importance for Number morphological feature - using diacritic words

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	Feature Importance	Feature	Feature Importance	Feature	Feature Importance
1	last_letter_ا	21.43%	last_letter_ا	11.94%	last_letter_نِ	11.75%
2	last_letter_ي	11.09%	last_letter_يْ	6.80%	recent_letter_يْ	7.30%
3	last_letter_يْ	10.19%	last_letter_نَ	4.32%	last_letter_نَ	5.39%
4	char1_نُ	4.12%	last_letter_نِ	4.03%	recent_letter_ا	3.93%
5	char1_نَ	1.87%	char2_نُ	3.60%	recent_letter_و	3.35%
6	last_letter_نَّ	1.46%	recent_letter_يْ	2.35%	last_letter_يْ	3.16%
7	char1_بُ	1.33%	last_letter_يَّ	2.17%	length_of_word	2.99%
8	last_letter_آ	1.23%	recent_letter_و	2.06%	last_letter_ا	2.58%
9	last_letter_يَّ	1.20%	recent_letter_ا	1.98%	last_letter_تِ	2.15%
10	char1_شُ	1.08%	length_of_word	1.97%	last_letter_يَّ	1.61%
11	char1_أَ	0.94%	char2_نَ	1.96%	last_letter_ةِ	1.60%
12	length_of_word	0.94%	char1_نُ	1.80%	last_letter_ة	1.49%
13	char1_جُ	0.87%	char1_نَ	1.71%	last_letter_تٍ	1.43%
14	char1_خُ	0.85%	last_letter_ي	1.57%	brecent_letter_ا	1.42%
15	char1_مُ	0.83%	char1_مُ	1.37%	brecent_letter_تَ	1.39%
16	char1_عُ	0.82%	last_letter_و	1.06%	brecent_letter_يْ	1.33%
17	last_letter_نَ	0.73%	char2_تَ	1.06%	last_letter_ت	1.27%
18	char1_تَ	0.70%	char2_تُ	1.04%	last_letter_تُ	1.21%
19	char1_عَ	0.65%	last_letter_ةٍ	0.97%	last_letter_تٌ	1.18%
20	char1_حُ	0.65%	last_letter_ةَ	0.90%	char3_نُ	0.93%

4.3.2 Gender

For the characters that influence gender prediction, it is noticed from Table 24 that the highest importance is related to the same characters that affect gender prediction for words without diacritics but with different diacritics. The thing that may justify the low percentage of importance compared to the previous models for characters "ة" and "ت".

Table 24: Features Importance for Gender morphological feature - using non-diacritic words.

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	Feature Importance	Feature	Feature Importance	Feature	Feature Importance
1	last_letter_ةِ	7.46%	last_letter_ةٍ	4.98%	recent_letter_و	4.41%
2	last_letter_ةُ	6.60%	last_letter_ةٌ	4.70%	brecent_letter_تَ	3.92%
3	last_letter_ةَ	5.75%	last_letter_ةُ	4.52%	last_letter_نِ	3.65%
4	last_letter_ةٌ	5.61%	last_letter_ةَ	4.42%	last_letter_نَ	3.52%
5	last_letter_ةٍ	5.57%	last_letter_ةً	4.39%	length_of_word	3.06%
6	last_letter_ةً	5.24%	length_of_word	3.06%	last_letter_ةِ	2.74%
7	last_letter_ت	3.70%	char1_مُ	3.00%	recent_letter_يْ	2.29%
8	char1_تَ	3.27%	last_letter_ة	2.15%	last_letter_ة	2.27%
9	char1_تُ	2.78%	last_letter_تِ	2.06%	brecent_letter_تِ	1.67%
10	last_letter_ة	2.32%	last_letter_ت	1.97%	brecent_letter_تُ	1.65%
11	last_letter_نّ	2.29%	last_letter_ا	1.76%	last_letter_ةٍ	1.54%
12	last_letter_تِّ	1.34%	char2_تَ	1.32%	recent_letter_تَ	1.50%
13	char1_مَ	1.30%	last_letter_نَ	1.31%	last_letter_ت	1.45%
14	length_of_word	1.10%	char2_يُ	1.03%	last_letter_يْ	1.44%
15	last_letter_تِ	1.09%	recent_letter_و	1.02%	recent_letter_تِ	1.43%
16	char1_أَ	1.08%	recent_letter_ا	0.95%	last_letter_ةَ	1.38%
17	char1_يَ	1.02%	char1_يَ	0.93%	last_letter_تِ	1.37%
18	char1_مُ	0.98%	char1_تُ	0.92%	recent_letter_ي	1.34%
19	char1_وَ	0.88%	recent_letter_يّ	0.89%	last_letter_ةُ	1.31%
20	char1_يُ	0.83%	char1_ل	0.64%	char3_يُ	1.30%

4.3.3 Part of speech

For diacritical Arabic words, Table 25 shows the most significant features for predicting the POS. We cannot define a shared pattern for the three types of models. The characters that highly affect the decision for words with 2-3 characters are not the same for words with more than 5 characters. Despite this, we can see that the character "مُ" has an impact on various models. In addition, the character "ت" influences the POS prediction with different diacritics and in different positions in the word, especially at the beginning.

Table 25: Features Importance for POS morphological feature - using diacritic words.

No.	Words with 2-3 characters		Words with 4-5 characters		Words with more than 5 characters	
	Feature	Feature Importance	Feature	Feature Importance	Feature	Feature Importance
1	char1_أَ	3.99%	char1_مُ	6.63%	char3_ل	3.72%
2	char1_يَ	3.80%	recent_letter_ا	2.86%	char1_مُ	3.64%
3	char1_يُ	2.89%	char1_يَ	2.67%	length_of_word	2.66%
4	char1_مُ	2.74%	char1_يُ	2.66%	recent_letter_ا	2.51%
5	char1_تُ	2.07%	char1_تُ	2.43%	char2_مُ	2.37%
6	char1_أُ	1.77%	char2_تُ	2.40%	char1_بِ	2.13%
7	char1_ا	1.73%	char1_أُ	1.92%	char2_تُ	1.93%
8	length_of_word	1.72%	char2_تَ	1.76%	char2_يُ	1.91%
9	char1_تَ	1.50%	last_letter_ةٍ	1.76%	char1_سَ	1.73%
10	last_letter_قَ	1.50%	char1_أَ	1.29%	char2_يَ	1.70%
11	last_letter_نَ	1.22%	char2_يُ	1.28%	char2_سَ	1.50%
12	last_letter_ى	1.19%	char1_بِ	1.28%	char1_ا	1.49%
13	last_letter_مَ	1.14%	last_letter_ةَ	1.23%	char1_ل	1.45%
14	last_letter_عَ	1.06%	last_letter_ةٌ	1.23%	char3_تَ	1.44%
15	last_letter_ا	0.94%	char1_سَ	1.19%	char1_تُ	1.28%
16	last_letter_رْ	0.94%	last_letter_تُ	1.13%	char3_تُ	1.27%
17	last_letter_تُ	0.94%	last_letter_ةُ	1.13%	char1_وَ	1.25%
18	last_letter_تِ	0.92%	last_letter_ةً	1.13%	char1_يُ	1.24%
19	last_letter_لْ	0.87%	char1_تَ	1.06%	char3_ا	1.23%
20	last_letter_أَ	0.86%	char2_يَ	1.04%	char1_م	1.18%

4.4 Results and Discussion:

4.4.1 Results summary

Table 26 summarizes the model's result for words with and without diacritics. The table shows that Random Forest resulted in the best performance. The performance of different models was compared using the ROC-AUC score due to the data imbalanced distribution, and this metric increased in the presence of diacritics, especially for Number and POS features.

Table 26: Results Summary

Output	**Algorithm**	**Testing_roc_auc_score** **WITHOUT DIACRITICS**	**Testing_roc_auc_score** **WITH DIACRITICS**
Gender	LOG	89.67%	92.63%
	NB	86.62%	86.82%
	RF	**91.71%**	**93.60%**
	SVM	89.94%	92.69%
Number	LOG	85.92%	94.36%
	NB	80.17%	86.83%
	RF	**90.08%**	**97.07%**
	SVM	88.18%	94.49%
POS	LOG	82.82%	92.79%
	NB	81.62%	90.84%
	RF	**89.88%**	**95.33%**
	SVM	82.10%	92.74%

Underlined numbers refer to maximum value for each algorithm.

4.4.2 Discussion

In what follows, we discuss our results and compare them with others' results. However, this is not a straightforward task, because there are several factors to consider. Among these factors are:

- **Dataset used for training and testing**:

  Studies of predicting Arabic word morphological features have used various sources of datasets, such as the Quranic corpus, Arabic tweets, and Arabic articles. Additionally, some of these studies have prepared datasets specifically for the prediction study, while others have relied on existing corpuses. When more patterns or types of words are added to the training dataset, the learning model's capabilities are enhanced.

  Our study used a dataset comprising 7.9 million words collected from 150 lexicons. This dataset is considered a rich source of Arabic words due to its diversity and comprehensiveness, and thus it contributes to the study of natural language processing in general, and to the prediction of word morphological features in particular.

- **Dataset Size used for training and testing**:

  Studies used varied dataset sizes for training and testing, ranging from tens of thousands to hundreds of thousands of words or morphemes. In this study, we used a large dataset consisting of approximately four million words, and in some models, up to eight million words. The data size is considered large compared to the datasets used in similar studies.

- **Tag Set:**

  Different tag sets were used for predicting Arabic words' morphological features. A large number of tags were added to the POS tag set, and the tag set size reached 70 tags. There is no standard or unified tag set used, and researchers can rely on any tag set for their applications.

In our study, we predicted words with selected tags using a simplified tag set, verb and noun for POS, masculine, feminine for gender, and singular, dual, and plural for number.

- **Words with or without diacritics:**

Researchers also differ in how they predict words, with or without diacritics. Most researchers use words with diacritics when training their models or tools. However, very few studies explored the effect of diacritics on predicting Arabic words' morphological features.

In this study, we have developed models that deal with words with diacritics and words without diacritics. In addition, a comparison was conducted to explore the effects of diacritics on similar rules.

-Models/tools inputs:

The inputs used in morphological features prediction can be divided into two main groups; the first group includes features derived from the words themselves only, such as stems, affixes, word length, and some morphological features. However, these input extraction processes may require a separate detection model, such as the stem detection model used by Darwish et al. (2014), or the use of morphological parsing to extract word morphemes, which entails the presence of a lexicon, morphotactic, and orthographic rules. The other group includes inputs related to neighboring words and word position in relation to context or neighboring words. These positions require the presence of words within a sentence.

This study proposes a classifier that accounts for extraction process obstacles, relying on word characters as model inputs that are simply extracted from input words.

- **Ability to predict features of words without context:**

Multiple studies used words within the context in developing prediction tools, and some of them relied on words without context.

In this study, we extract features from words isolated from context or sentences. This makes it easier to obtain information about individual words and can be useful in predicting morphological features since there are limited annotated Arabic corpora available. Additionally, such data is easier to obtain, as it consists only of separated Arabic words and does not require a large amount of annotated text to be collected.

- **Methodology:**

Studies also differ in the methodology they apply to develop prediction models or tools. The methodology could be rule-based, statistical or hybrid. Also, researchers can use classical machine learning, deep learning or try to merge different approaches.

Table 27 summarizes the research of predicting Arabic words' morphological features and shows the specifications of each study regarding several factors; predicted tag, words with or without diacritics, with or without context approach, tag set, features, training dataset, and methodology.

Table 27: Discussions and Related Studies Summary

Author	Tag set	With\without diacritics	Predicted tag	With/ Without context	Features	Trained Dataset - Dataset Size	Methodology	Results
(Darwish et al., 2014)	masculine or feminine singular, dual, or plural	-	Gender Number	With context	Stem template Length of the stem template POS tag Attached suffix	Wikipedia Alaljazaera.net articles	Random forest Classifier	95.6% (Accuracy) 94.9% (Accuracy)
(Alkuhlani and Habash, 2012)	masculine or feminine singular, dual, or plural	with diacritics with diacritics	Gender & Number	Without context With context	**Orthographic Features** (unnormalized form of the word, unnormalized form of the word plus first letter, second letter, last letter, and last two letters of the word form) **Morphological Features** (POS tags, Lemma, Form-based features) **Syntactic Features**	Penn Arabic Treebank PATB	MLE with Back-of Support Vector Machine Based Sequence Tagger (Yamcha Sequence Tagger)	88.5% (Accuracy) 91-91.4% (Accuracy) 94.1% (Combined Models Accuracy)
(Darwish et al., 2014)	a simplified PATB tag set	with diacritics	POS	With context	List Match Stem template Prefixes The position of the word in the sentence	Penn Arabic Treebank PATB	Random Forest Classifier	98.10%
(Abdulkareem and Tiun, 2017)	-	-	POS	With context	**Context-based features** (N-gram words, Next word, Word length, Is the word containing digit) **Word affixes** (first character, first two characters, first three characters, last character, last two characters and last three characters)	Arabic Tweets and Modern Arabic Text	Naïve Bayes K-Nearest Neighbors Algorithm Decision Tree-ID3	87.97% (F1-score) 87.2 (F1-score) 86.78 (F1-score)
(Mahafdah et al, 2014)	Nominals, Proper Nouns, Pronouns, Adjectives, Verbs, Particles, Prepositions, Uranic Initials (Disconnected Letters)	with diacritics	POS	With context	Word features Pos features	Quranic Arabic Corpus	K-Nearest Neighbors Algorithm & Naïve Bayes	98.32% (Accuracy)
(Tnaji et al., 2021)	the 4 elementary tags [Noun, Verb, Particle, Punctuation]	may contain tweets both with and without diacritics	POS	Combined	Prefixes, suffixes and word length - for unknown words using decision tree algorithm	NEMLAR corpus- 500,000 words, tested on WikiNews corpus	HMM and Decision tree	96.06% (Accuracy)
Alashqar (2012)	33-tag tag set 9-tag tag set	With and without diacritics	POS	Without context With context	words, POS	Quranic Arabic Corpus- 77,430 words	Unigram Bigram Trigram Brill HMM TnT	Highest results 82.50% 82.30% 82.40% 83.20% 77.50% 69.20%
Plank et al. (2016)	17 tags	without diacritics	POS	Without context	Word embeddings	Data from the Universal Dependencies project	Bi-LSTM	98.91%
Alrajhi et al. (2019)	tag set of 37 tags for words and 87 tags for Morphemes	without diacritics	POS	Without context	Words or Morphemes (highest result generated from models that used morphemes)	Quran - 14,901 unique words.	LSTM Word2Vec POS tag	99.72% 99.55%
Darwish et al. (2020)	Farasa POS tags- 18 tags for MSA	without diacritics	POS	With context	Linguistic features (stem templates and clitic meta types) and others Clitic and character-level inputs + the features for CRF	Arabic tweets - 12496 clitics	CRF Deep Neural Network Approaches	94.7% 94.7%
Inoue et al. (2022)	-	with diacritics	POS Gender Number	With context	Stem base (as well as MADAMIRA)	Penn Arabic Treebank PATB (629K)	MADAMIRA with pretrained CAMeLBERT-MSA	68.9% (31.1 error rate) 94.9% (5.1 error rate) 96.5% (3.5 error rate)
Zalmout and Habash (2020)	-	with diacritics	POS Gender & Number			Penn Arabic Treebank PATB (628K)	LSTM	98% 93.5% (for set of features)

To assess our model results in relation to related research results, we would conduct a comparison with experiments under the same conditions. In other words, the comparison should be with studies that have used words without context, words with and without diacritics, classical machine learning algorithms, and a similar tag set. However, based on the details expressed in Table 27 of related studies, we can notice that a direct comparison is not applicable.

Therefore, we focused on studies that have used words without context in the prediction process. Table 28 summarizes these studies, where limited studies tried to predict Arabic words' morphological features without context, and most of them worked on POS feature only.

Table 28: Related Studies Summary- Without context

Author	Tag Set	With\Without diacritics	Predicted tag	With\ Without context	Features	Trained and Tested Dataset - Dataset Size	Methodology	Results
(Alkuhlani and Habash, 2012)	Masculine or feminine singular, dual, or plural	With diacritics	Gender & Number	Without context	**Orthographic Features** (unnormalized form of the word, unnormalized form of the word plus first letter, second letter, last letter, and last two letters of the word form)	Penn Arabic Treebank PATB	MLE with Back-of	88.5%
Alashqar (2012)	33-tag tag set 9-tag tag set	With and without diacritics	POS	Without context	Words, POS	Quranic Arabic Corpus- 77,430 words	Unigram	82.50%
Alrajhi et al. (2019)	Tag set of 37 tags for words and 87 tags for Morphemes	Without diacritics	POS	Without context	Words or Morphemes (highest result generated from models that used morphems)	Quran - 14,901 unique words.	LSTM	99.72%
Plank et al. (2016)	17 tags	Without diacritics	POS	Without context	Word embeddings	Data from the Universal Dependencies project	Bi-LSTM	98.91%
Our Model-POS-without diacritics	2 tags (Noun, Verb)	Without diacritics	POS	Without context	First and last 1-3 characters, and word length	Tested on Data from the Universal Dependencies project V2.0	Random Forest	84.63%
(Tnaji et al., 2021)	The 4 elementary tags [Noun, Verb, Particle, Punctuation]	May contain tweets both with and without diacritics	POS	Combined	Prefixes, suffixes and word length - for unknown words using decision tree algorithm	NEMLAR corpus- 500,000 words, tested on WikiNews corpus	HMM and Decision tree	96.06% (Accuracy)
Our Model-POS-without diacritics	2 tags (Noun, Verb)	Without diacritics	POS	Without context	First and last 1-3 characters, and word length	Tested on WikiNews corpus	Random Forest	83.08%

For the comparison, we have tested our model on the dataset used by these researchers, as shown in Table 28. However, Penn Arabic Treebank PATB is not available, so we excluded the model of Alkuhlani and Habash (2012). For the Universal Dependencies project dataset used by Plank et al. (2016), we have used version 2.0 of this data instead of the version 1.2 used by Plank and his colleagues, as it is no longer available. Also, we have only used words labeled with noun, verb, and adjective POS tags, where the adjective tag was mapped to the noun tag. For the Wikinews corpus used by Tnaji et al. (2021) to test their model on unseen words, we tested our model on this data too, using only words with verb, noun, and adjective POS tags, where the adjective tag was mapped to the noun tag.

Table 28 also shows the results and details of the comparison; we were only able to evaluate the model of POS tag for words without diacritics. The results showed that the Bi-LSTM model used by Plank et al. (2016) and the hybrid model of Tnaji et al. (2021) achieved higher accuracy than our model. However, for the hybrid model of Tnaji et al. (2021), the researchers did not refer to the accuracy of predicting unknown words separately, as these words only present 17.68% of the tested dataset. In addition, this model uses word prefixes and suffixes for predicting word POS, which requires supplementary resources for feature extraction, the thing that our model eliminate through using first and last characters.

For researchers that have used classical machine learning algorithms to predict morphological tags, which is like our approach. For example, in Abdulkareem and Tiun (2017), Naive Bayes outperformed K-Nearest Neighbors (KNN) and Decision Tree models in predicting the part of speech of Arabic words. The models were built using the MSA corpus and Arabic tweets, with context-based features and word affixes as features. For MSA words, the models that were trained with Naive Bayes, KNN, and Decision Tree achieved

F1-scores of 82.59%, 82.91%, and 85.55%, respectively; while for Arabic tweets, the models that were trained with Naive Bayes, KNN, and Decision Tree achieved F1-scores of 87.97%, 87.22%, and 86.79%, respectively. Mahafdah et al. (2014) also compared KNN and Naive Bayes to explore the role of word features and POS features in predicting POS tags; they relied on a Quranic Arabic corpus of 77,430 words, with KNN resulting in higher accuracy. The highest accuracy was achieved by KNN at 95.5%, while NB was at 91.77%. In this study, Random Forest outperformed Naïve Bayes, SVM, and logistic regression. Although we used a different dataset and relied on word features (i.e., without context), we got superior results (ROC-AUC of 95.33% for diacritic words and 89.88% for non-diacritic words).

Referring to studies that employed machine learning algorithms and examined features like those in our study (i.e., word affixes and length of words) without relying on context, the test implemented by Tnaji et al. (2021) applied similar features to predict only unknown words (i.e., words that were not part of the training corpus). Even though the share of unknown words in the testing dataset was around 18%, the overall accuracy for the POS was 96.06%, a performance comparable to this study's performance (ROC-AUC of 95.33% for diacritic words and 89.88% for non-diacritical words). In contrast, Alkuhlani and Habash (2012) used Maximum Likelihood Estimation (MLE) methodology to predict the gender and number of words without context, and the accuracy was 88.5%, a performance lower compared to our work.

Based on previous studies that also employed machine learning algorithms and investigated similar features to those in our study, in addition to features extracted from the context, and as shown in Table 26, our model achieved a similar result for gender tags (ROC-AUC of 93.6% for diacritic words and 91.72% for non-diacritic words), while it achieved a slightly higher result for numbers tags (ROC-AUC of 97.07% for diacritic words and 90.08% for non-diacritic words). In addition, the comparison has shown that our study's performance in predicting POS tags is as effective as theirs.

Compared to other studies exploring Deep Learning to predict Gender, Number, and POS for Arabic, little work has been done on predicting Gender and Number tags using deep learning, so the results achieved are acceptable, especially for Number tags (ROC-AUC of 97.07% for diacritic words and 90.08% for non-diacritic words). Additionally, more studies focused on predicting POS tags using deep learning. Most of these studies worked on predicting POS without context and achieving performance better than the results of our study.

Deep learning has been applied in tagging words with morphological features, helping researchers avoid feature engineering. However, they still need to choose between encoding approaches and Arabic text representation techniques that comply with the methodology used (e.g., embeddings and Word2V). For example, Alrajhi et al. (2019) used the reversible integer transformation (RIT) algorithm over one hot encoding to encode the input for the LSTM model.

4.4.3 Contributions

In summary, in this study we have contributed to the Arabic word morphological features field of study as we have:

- Employed the Birzeit lexicon - a rich Arabic language lexicon.
- Explored the performance of different machine learning algorithms.
- Tried to eliminate some of the morphological features' limitations, by:
    1. Using words only without context.
    2. Employing characters at the beginning and end of the word and avoiding complex feature extraction processes.
    3. Using a reliable tagged dataset for training with a size of 7.9 million words.
- Studied the role of diacritics in morphological feature predictions.
- Achieved a satisfactory performance for Number and Gender tags, especially for our diacritic models, and a performance in line with the average performance of other studies in predicting POS tags, with comparable accuracy levels.

Chapter 5: Conclusions and Recommendations

5.1 Conclusions

In this study, we presented a character-based approach for predicting Arabic word morphological features without context, which helps overcome the disadvantages of existing Arabic word morphological feature taggers. We used a combination of the first and last one to three characters of a word, along with its length, to predict morphological features. Each model and algorithm were evaluated using different evaluation metrics using four supervised machine learning algorithms.

The first experiments were conducted on words without any diacritics. And the highest performance is produced by the Random Forest algorithm. The AUC-ROC for the most optimal model for number, gender, and POS was 90.08%, 91.72%, and 89.88%, respectively.

In the second experiment, we used words with diacritics. The Random Forest algorithm also outperformed the other three algorithms. The AUC-ROC for the number and POS performance of the model increased to 97.07% and 95.33%, respectively. Besides, the gender feature showed a similar result to the aforementioned performance, with an AUC-ROC of 93.7%.

Overall, the models that have been developed can simply be applied and updated when needed.

5.2 Future Work

For this section, we present future work to improve prediction models of Arabic words with morphological features without context:

1. To enhance performance comparison and improve generated models, we suggest applying additional machine learning algorithms, such as neural networks and K-nearest-neighbor algorithms.

2. Technical challenges faced in data dimensions. In this study, we implemented one-hot encoding for each character at different positions in the word, which increased the number of variables. So, the suggestion is to use different methods, such as long-short-term memory (LSTM), and to explore the effectiveness of this approach.

Bibliography

Abdelali, A., Darwish, K., Durrani, N., & Mubarak, H. (2016). *Farasa: A Fast and Furious Segmenter for Arabic*.

Abdul-Hamid, A., & Darwish, K. (2010). *Simplified Feature Set for Arabic Named Entity Recognition*. Proceedings of the 2010 Named Entities Workshop, (pp. 110–115). Uppsala, Sweden.

Abdulkareem, M.A., & Tiun, S. (2017). *Comparative analysis of ML POS on Arabic tweets*. Journal of theoretical and applied information technology, 95, 403-411.

Alashqar, A. (2012). A comparative study on Arabic POS tagging using Quran corpus. NLP-29.

Albared, M. & Omar, N. & Ab Aziz, M. (2011). *Developing a Competitive HMM Arabic POS Tagger Using Small Training Corpora*. 288-296. 10.1007/978-3-642-20039-7_29.

Al-Hajj, M., Jarrar, M., (2021). *ArabGlossBERT: Fine-Tuning BERT on Context-Gloss Pairs for WSD*. In Proceedings – the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), PP 40--48.

Al-Hajj, M., Jarrar, M., (2021). LU-BZU at SemEval-2021 Task 2: *Word2Vec and Lemma2Vec performance in Arabic Word-in-Context disambiguation*. In Proceedings – the 11th Workshop on Semantic Evaluation (SemEval2021), PP 748--755, Association for Computational Linguistics.

Alkhairy, M., Jafri, A., & Smith, D.A. (2020). Finite *state machine pattern-root arabic morphological generator, analyzer and diacritizer*. LREC 2020 - 12th International Conference on Language Re-

Alkuhlani, Sarah & Habash, Nizar. (2012). Identifying broken plurals, irregular gender, and rationality in Arabic text. 675-685.

Alluhaibi, Reyadh & Alfraidi, Tareq & Abdeen, Mohammad & Yatimi, Ahmed. (2021). *A Comparative Study of Arabic Part of Speech Taggers Using Literary Text Samples from Saudi Novels Information* (Switzerland). 12. 10.3390/info12120523.

Alothman, A., & Alsalman, A. (2020). *Arabic Morphological Analysis Techniques*. International Journal of Advanced Computer Science and Applications (IJACSA), 2(11), 214-222. doi:10.14569/ijacsa.2020.0110229

Alqrainy, S. (2008). A morphological-syntactical analysis approach for Arabic textual tagging. Leicester, UK: Phd.

Alrajhi et. al., K. (2019). *Automatic Arabic Part-of-Speech Tagging: Deep Learning Neural LSTM Versus Word2Vec*. International Journal of Computing and Digital Systems. 8(3), 307-315. https://doi.org/10.12785/ijcds/080310

Alzubi, J., Nayyar, A., & Kumar, A. (2018). *Machine Learning from Theory to Algorithms: An Overview*. Journal of Physics: Conference Series, 1142, 012012. doi:10.1088/1742-6596/1142/1/012012

Attia, M., Pecina, P., Toral, A., Tounsi, L., & Genabith, J. v. (2011*). An Open-Source Finite State Morphological Transducer for Modern Standard Arabic. Blois*, France:

Proceedings of the 9th International Workshop on Finite State Methods and Natural Language Processing.

Benajiba, Y., Diab, M., & Rosso, P. (2008). *Arabic Named Entity Recognition using Optimized Feature Sets*. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (pp. 284–293). Honolulu: Association for Computational Linguistics.

Benajiba, Y., Diab, M., & Rosso, P. (2008b). Arabic Named Entity Recognition: An SVM-Based Approach.

Boudad, N., Faizi, R., Haj Thami, R. O., & Chiheb, R. (2017). *Sentiment analysis in Arabic: A review of the literature*. Ain Shams Engineering Journal, 9, 2479–2490. doi: dx.doi.org/10.1016/j.asej.2017.04.007

Boudchiche, M., Mazroui, A., Ould Bebah, M. O., Lakhouaja, A., & Boudlal, A. (2017). *AlKhalil Morpho Sys 2: A robust Arabic morpho-syntactic analyzer*. Journal of King Saud University –Computer and Information Sciences, 29, 141–146. doi: 10.1016/j.jksuci.2016.05.002

Darwish, K., & Mubarak, H. (2016). Farasa: A New Fast and Accurate Arabic Word Segmenter. .

Darwish, K., Attia, M., Mubarak, H., Samih, Y., Abdelali, A., Màrquez, L., Eldesouki, M., & Kallmeyer, L. (2019). *Effective Multi Dialectal Arabic POS Tagging*. Natural Language Engineering. 26. 10.1017/S1351324920000078.

Darwish, K., Mubarak, H., Abdelali, A., Eldesouki, M., Samih, Y., Alharbi, R., ... Kallmeyer, L. (2018). *Multi-dialect arabic pos tagging: A CRF approach*. In

Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)

Darwish, Kareem & Mubarak, Hamdy & Abdelali, Ahmed & Eldesouki, Mohamed. (2014). *Using Stem-Templates to Improve Arabic POS and Gender/Number Tagging*. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 2926–2931, Reykjavik, Iceland. European Language Resources Association (ELRA).

El-haj, M. and Koulali, R. (2013). *KALIMAT a multipurpose Arabic Corpus*. In Second Workshop on Arabic Corpus Linguistics (WACL-2), pages 22–25

Freihat, A.A., Bella, G., Mubarak, H., & Giunchiglia, F. (2018). *A single-model approach for Arabic segmentation, POS tagging, and named entity recognition*. 2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP), 1-8.

Gridach, M., & Chenfour, N. (2011). *Developing a New System for Arabic Morphological Analysis and Generation*. Thailand: Proceedings 2nd Workshop on South Southeast Asian Natural Language Processing (WSSANLP).

Hadj, Y.O., Alsughayeir, I.A., & Al-Ansari, A. (2009). Arabic part-of-speech tagging using the sentence structure.

Hadni, M., Ouatik El Alaoui, S., LACHKAR, A., & Meknassi, M. (2013). Hybrid Part-Of-Speech Tagger for Non-Vocalized Arabic Text. International Journal on Natural Language Computing, 2, 1-15. doi:10.5121/ijnlc.2013.2601

Hossin, M., & Sulaiman, M. (2015*). A Review on Evaluation Metrics for Data Classification Evaluations*. International Journal of Data Mining & Knowledge Management Process, 01-11. doi:10.5121/ijdkp.2015.5201

Jacobsen, M. & Sørensen, M. & Derczynski, L. (2021). *Optimal Size-Performance Tradeoffs: Weighing PoS Tagger Models*. 10.31219/osf.io/azfu2.

Jarrar, M. (2018). *Search Engine for Arabic Lexicons.* In Proceedings of the 5th Conference on Translation and the Problematics of Cross-cultural Understanding, The Forum for Arab and International Relations, Qatar.

Jarrar, M. (2020). *Digitization of Arabic Lexicons*. Arabic Language Status Report. UAE Ministry of Culture and Youth. Pages 214-2017. Dec 2020

Jarrar, M., & Amayreh, H. (2019). *An Arabic-Multilingual Database with a Lexicographic Search Engine*. Springer Nature Switzerland AG 2019, 13, 234-246. doi:10.1007/978-3-030-23281-8_19

Jarrar, M., Khalilia, M., Ghanem, S. (2022). *Wojood: Nested Arabic Named Entity Corpus and Recognition using BERT*. In Proceedings of the International Conference on Language Resources and Evaluation (LREC 2022), Marseille, France.

Jurafsky, D., & Martin, J. (2020). *Speech and Language Processing*.

Go Inoue, Salam Khalifa, and Nizar Habash. 2022. Morphosyntactic tagging with pre-trained language models for arabic and its dialects. Findings of the Association for Computational Linguistics: ACL 2022, pages 1708–1719.

Kareem Darwish, Mohammed Attia, Hamdy Mubarak, Younes Samih, Ahmed Abdelali, Lluís Màrquez, Mohamed Eldesouki, and Laura Kallmeyer. (2020). *Effective multi-dialectal Arabic POS tagging. Natural Language Engineering*, 26:677–690

Kashefi, O. (2018). *Unsupervised Part-of-Speech Induction*. Intelligent Systems Program University of Pittsburgh.

Khalifa, S., Zalmout, N., & Habash, N. (2016). *YAMAMA: Yet Another Multi-Dialect Arabic Morphological Analyzer*. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, pages 223–227, Osaka, Japan. The COLING 2016 Organizing Committee.

Khalifa, S., Zalmout, N., & Habash, N. (2020). *Morphological Analysis and Disambiguation for Gulf Arabic: The Interplay between Resources and Methods*. In Proceedings of the Twelfth Language Resources and Evaluation Conference, 3895–3904, Marseille, France. European Language Resources Association.

Khoja, S. (2001). *APT: Arabic part-of-speech tagger*. In Proceedings of NAACL Student Research Workshop.

Kumawat, D., & Jain, V. (2015). POS Tagging Approaches: A Comparison. International Journal of Computer Applications, 118(6), 32-38. doi:10.5120/20752-3148

Lie, C. (2020). *A Top Machine Learning Algorithm Explained: Support Vector Machines (SVMs)*. Velocity Business Solutions Limited. Retrieved from https://www.vebuso.com/2020/02/a-top-machine-learning-algorithm-explained-support-vector-machines-svms/

Mahafdah, R & Omar, N & Al-Omari, O. (2014). *Arabic part of speech tagging using K-Nearest Neighbour and Naive Bayes classifiers combination*. Journal of Computer Science. 10. 1865-1873. 10.3844/jcssp.2014.1865.1873.

Nasser Zalmout and Nizar Habash. (2020). Joint diacritization, lemmatization, normalization, and fine-grained morphological tagging. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8297–8307. Association for Computational Linguistics.

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 Twelfth Language Resources and Evaluation Conferen*ce, pages 7022–7032, Marseille, France. European Language Resources Association.

Pasha, A. & Elbadrashiny, Mohamed & Diab, Mona & Elkholy, A. & Eskandar, Rushdi & Habash, Nizar & Pooleery, M. & Rambow, Owen & Roth, Ryan. (2014). *MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic*. Proceedings of the 9th International Conference on Language Resources and Evaluation. 1094-1101.

Plank, B. & Søgaard, A. & Goldberg, Y. (2016). Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. 412-418. 10.18653/v1/P16-2067.

Ren, S., Lian, Y., & Zou, X.Y. (2014). *Incremental Naïve Bayesian Learning Algorithm based on Classification Contribution Degree*, Journal of Computers vol. 9, no. 8, pp. 1967-1974, 2014.

Salah, R. I., & Binti Zakaria, L. Q. (2017*). A Comparative Review of Machine Learning for Arabic Named Entity Recognition*. International Journal on Advanced Science, Engineering, and Information Technology, 7(2), 511--518. doi: 10.18517/ijaseit.7.2.1810

Sayad, S. (n.d.). *Support Vector Machine*. Retrieved from https://www.saedsayad.com/support_vector_machine.htm

sources and Evaluation, Conference Proceedings, pages 3834–3841

Tlili-Guiassa, Y. (2006). *Hybrid Method for Tagging Arabic Text. Journal of Computer Science*, 2(3), 245-248. doi:10.3844/JCSSP.2006.245.248

Tnaji, K. & Bouzoubaa, K. & Aouragh, Si. (2021). *A Light Arabic POS Tagger Using a Hybrid Approach*. Lecture Notes in Networks and Systems, LNNS:199–208. 10.1007/978-3-030-73882-2_19.

Appendix

- Link to Google Colab

  https://colab.research.google.com/drive/1FtnDQwoLgG2jE5JxhvGjOzWTo5LRTyQH?usp=sharing

- Link to Results and Outputs

  https://docs.google.com/spreadsheets/d/1XLRbATFrj1-9G0Q_zXieZsmGR8dPnQ2O6J_RFD-gT0w/edit?usp=sharing