The Impact of Arabic Diacritization on Word Embeddings

Mohamed ABBACHE[[1]](#footnote-1)

School of Computer Science and Technology, Tianjin University of Technology, Tianjin, China, m.abbache@yahoo.fr

Ahmed ABBACHE

Mathematics and its Applications Laboratory, Faculty of Exact Sciences and Computing, Hassiba Ben Bouali University of Chlef, Ouled Fares, Chlef Province, Algeria. mr\_abbache@yahoo.co.uk

JingWen Xu

Computer Science, Faculty of Information Engineering, Computer Science and Statistics, Sapienza University of Rome, Rome, Italy, xu.1972349@studenti.uniroma1.it

Farid MEZIANE

Data Science Research Centre, University of Derby, The United. Kingdom, F.Meziane@derby.ac.uk

XianBin WEN

School of Computer Science and Technology, Tianjin University of Technology, Tianjin, China. xbwen@tjut.edu.cn

Word embedding is used to represent words for text analysis. It plays an essential role in many Natural Language Processing (NLP) studies and has hugely contributed to the extraordinary developments in the field in the last few years. In Arabic, diacritic marks are a vital feature for the readability and understandability of the language. Current Arabic word embeddings are non-diacritized. In this paper, we aim to develop and compare word embedding models based on diacritized and non-diacritized corpora to study the impact of Arabic diacritization on word embeddings. We propose evaluating the models in four different ways: clustering of the nearest words; morphological semantic analysis; part-of-speech tagging; and semantic analysis. For a better evaluation, we took the challenge to create three new datasets from scratch for the three downstream tasks. We conducted the downstream tasks with eight machine learning algorithms and two deep learning algorithms. Experimental results show that the diacritized model exhibits a better ability to capture syntactic and semantic relations and in clustering words of similar categories. Overall, the diacritized model outperforms the non-diacritized model. Interestingly, we obtained some more interesting findings. For example, from the morphological semantics analysis, we found that with the increase in the number of target words, the advantages of the diacritized model are also more obvious, and the diacritic marks have more significance in POS tagging than in other tasks.

CCS CONCEPTS: **•Computing methodologies ~ Artificial intelligence ~ Natural language processing ~ Phonology / morphology**

**Key Words and Phrases:** Arabic NLP, Word Embeddings, Diacritization, Morphological Semantics, Semantic Analysis.

Introduction

Word embeddings have entirely changed the approach to research and development in Natural Language Processing (NLP) and Machine Learning (ML), opening new doors for many applications [Stohler 2018]. In word embeddings, words are represented as vectors in a continuous space, capturing many syntactic and semantic relations among them [Soliman et al. 2017]. These dense and distributed latent vectors allow working with text better and in a more meaningful manner than older text vectorization processes such as Bag-of-Words (BOW) [Stohler 2018; Manning et al. 2008].

Modern Standard Arabic (MSA) contains optional diacritical marks, which have become less used in Arabic books and newspapers. Diacritics are very important for readability and understandability, and their absence causes a critical problem of lexical, morphological, and semantic ambiguity [Hadjir et al. 2019]. In the Arabic language, many words appear the same in writing. Yet, they are given different unrelated meanings by simply changing one diacritic mark in a word. To show the importance of diacritic marks in the context of the Arabic language, we introduce a very simple example. From the Arabic word “جد”, we can obtain at least three different writings with different meanings by changing only one diacritic mark: “جَدُّ” (grandpa), “جِدُّ” (seriousness), “جُدُّ” (ruined well). This given example word contains only two characters, as word length increases, so does the number of new words formed from letters and diacritics.Arabic Morphology deals with the meanings that come from Scheme (pattern) and extra letters within the template of a word. These meanings include tense, voice, and added connotations [Esther Fleming 2020]. The Arabic Scheme is a natural representation of Arabic and shows the strength of the Arabic language in its meanings [Alexis Neme 2011]. Mingling with the letters and diacritics of an Arabic language Scheme, will lead to new meanings of the word. This phenomenon in the Arabic language is known as morphological semantics (الدلالة الصرفية) and is based on the semantics performed by the Arabic morphological Scheme and their structures [Qawaqzeh 2019]. Arabic diacritic marks are the main common factor between a word and its morphological Scheme. The words “كَتَبَ” and “كُتُبٌ” have the same writing “كتب”, but different Schemes: “فَعَلَ” and “فُعُلُ” leading to two different morphological semantics; the first word means “He wrote” and the second means “books”. As it can be seen, these two meaning are different and unrelated and are obtained by simply changing diacritic marks, then the Scheme.

All existing Arabic word representation projects support only non-diacritized texts; this is somehow expected because the Arabic language is considered as a low-resource language [Abid et al. 2018], especially when dealing with Arabic diacritized corpora to fit the word embedding models. However, diacritic marks are an essential characteristic of the Arabic language, especially as they are the first source of introducing the exact morphological, and semantic structures, which are needed in word embeddings. The main research question we address in this study is “**What is the impact of diacritization on Arabic non-contextual word embedding?**”

In this paper, our aim is to create and compare word embedding models based on diacritized and non-diacritized corpus, the models are created with FastText. The significant contributions of our work are as follow:

* **Better semantic relations capturing:** When reducing the ambiguity caused by the absence of diacritic marks, using a diacritized word embedding model makes the text more meaningful and allows better semantic relations capturing.
* **Higher accuracy:** Among all the experimental tasks, the word embedding model based on diacritized corpora outperformed the non-diacritized model.
* **Novel datasets[[2]](#footnote-2):** Developed novel evaluation datasets to help researchers to compare word embedding models on future works based on diacritized and non-diacritized corpora.
* **Baseline:** As far as we know, there are no existing Arabic word embedding based on diacritized corpora. Our study could be considered a baseline for future studies in the same field, especially to attract more attention to the importance of diacritization for Arabic Natural Language Processing (ANLP).

The rest of the paper is organized as follows: Section 2 presents some related efforts on the role of diacritization in Arabic NLP applications. Section 3 describes the methodology of our study. Section 4 briefly describes the datasets' challenges. Section 5 summarizes the evaluation results and discussion concerning the impact of diacritization on Arabic word embedding through a series of tests on some downstream tasks. Section 6 gathered a comparison and some limitations of our study. Finally, this paper ends with a conclusion and some ideas for future works.

Related work

[Diab et al. 2007] stated that no systematic study of the impact of diacritization on other NLP applications has been reported. To the best of our knowledge, there are few studies on Arabic NLP applications about the impact of Arabic diacritization [Masmoudi et al. 2019], especially in diacritics Arabic word embeddings. The most comparable research to ours were the two papers: [Younes et al. 2020] in Arabic language, and Adewumi et al. in the Yorùbá language [Adewumi et al. 2020]. We took the challenge of studying the impact of diacritization, especially on Arabic word embeddings.

[Adewumi et al. 2020], studied the effect of diacritic marks on word embeddings in the Yorùbá language. The experiment was based on three datasets: the cleaned 2020 Yorùbá Wikipedia dump containing diacritics to different levels across articles, a non-diacritized version of it, and the most extensive diacritized data used by [Alabi et al. 2020] . After the cleaning process, especially removing the HTML tags using a python script proposed by [Grave et al. 2018] their experimental results showed that the non-diacritized embeddings, based on normalized text, achieved better intrinsic performance than other diacritized models.

[Younes and Weeds 2020], investigated the capability of neural techniques and embeddings to represent language specific characteristics in two sequence labeling tasks: named entity recognition (NER) and part of speech (POS) tagging, conducted on seven (7) non-diacritized datasets (NER: ANERCorp, BinAjeeba, NewsWire, Wikipedia) and (POS: Al Mushaf, PADT, WikiNews). In both tasks, a preprocessing is designed to use enriched Arabic representation by adding diacritics to non-diacritized text by an automatic diacritization model called Shakkala by incorporating an embedding layer for diacritics alongside embedding layers for words and characters, show that embedding the information that is encoded in automatically acquired Arabic diacritics improves the performance across all datasets on both tasks. Embedding the information in automatically assigned POS tags further improves the performance on the NER task.

The main difference between this work and ours are as follows:

* our study focuses on the performance of the non-contextual word embedding models (diacritized and non-diacritized) on different downstream tasks, especially semantic relations, not only NER and POS. In addition, we output a new word embedding model, which is diacritized.
* We avoided using a diacritization system, because the current existed systems cannot be 100% accurate, which will result in a data quality problem, and then affect our experimental results, because we are having a comparative study that requires well-diacritized content.
* Their used dataset is automatically diacritized and was not released. We released three (3) semantic datasets used to evaluate the performance of the future diacritized word embeddings models. These can be used to reproduce the results of this research and to allow its comparison with future developments.

[Diab et al. 2007] explored the impact of Arabic diacritization in the context of Statistical Machine Translation (SMT) by defining several diacritization schemes ranging from complete to partial diacritization and studied two different modes which tease apart the effect of diacritization on the alignment and its consequences on decoding. Their reported results show that none of the partial diacritization schemes significantly varies in performance from the non-diacritization baseline despite increasing the number of types in the data. Finally, the diacritics scheme performs considerably worse than the non-diacritized one. Word ambiguity is a significant limitation for machine translation (MT), where the absence of diacritic marks naturally leads to significant word ambiguity. However, [Alqahtani et al. 2016] got the exact opposite experimental result; in the same SMT field and continuance of the previous work, various partial diacritization schemes preserved some of the semantics that complemented the implicit contextual information present in the sentences were investigated. Diacritized schemes capture and model necessary information at a more appropriate level of granularity. In addition, diacritic schemes perform well in some evaluation metrics and datasets. Moreover, improving the underlying diacritization technology will probably have a significant impact on performance. Finally, the study concludes that diacritic marks help reduce the number of possible lexical word choices assigned to a source word, leading to better-translated sentences.

[Masmoudi et al. 2019] summarized the impact of diacritization in NLP Applications on Arabic speech recognition (ASR) application suggested adding diacritic marks of data to improve Levantine Arabic speech corpus [Alotaibi et al. 2013]. In the same field, [Masmoudi et al. 2014] proposed to use full diacritization to generate a corpus and phonetic dictionary for ASR in the Tunisian Railway Transport Network domain. According to the study, it was reported that the phonetic system has a 10% Word Error Rate (WER) for diacritized words, which is 90% efficient for data diacritization compared to non-diacritization data.

Methodology

The main objective of our work is to create and compare the performance of two kinds of word embedding models: diacritized and non-diacritized models. To ensure the validity of the experimental results, we need to use data that is as highly diacritized as possible. Both of our models are mainly created based on three different freely available sources; the first is the Tashkeela corpus by [Zerrouki and Balla 2017], the second is the holy Quran, and the last are some diacritized Islamic articles collected by using semi-automatic web crawling process from Raslan[[3]](#footnote-3) and others. Tashkeela's corpus is a popular Arabic diacritized corpus with a size of 1.14 GB and contains 75 million vocalized words, mainly collected from 117 books, including classical Arabic (CA) and modern standard Arabic (MSA). Of these, twenty books were written in MSA. Although the Quran is small in size and limited only to Classical Arabic, its content is completely and precisely diacritized, which strengthen our corpus and facilitates our research. Because Tashkeela corpus has covered most of our used words, we will hereinafter refer to all the used sources as Tashkeela+. Our Tashkeela+ corpus contains mostly classical Arabic text and covers mostly Islam related contents. We summarized our study in Figure 1.

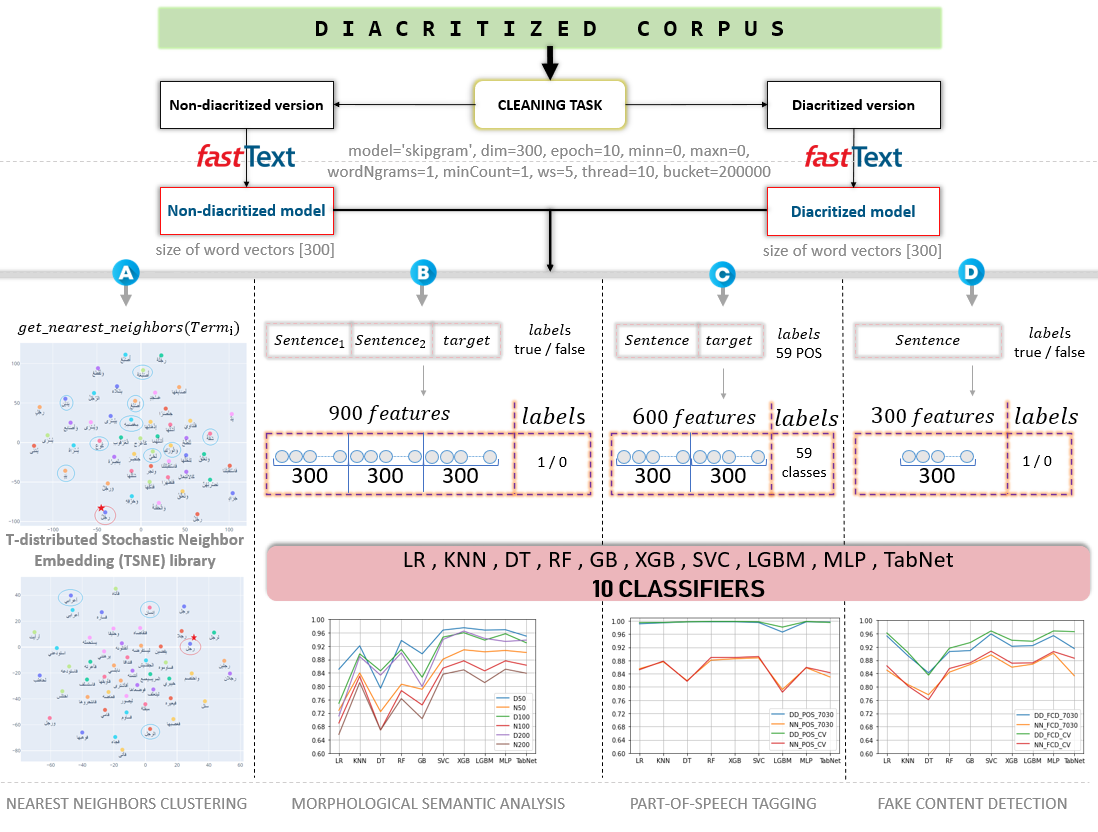


Figure 1 : General schema summarizes the methodology of our study from the diacritics data collection and word representation models to the final results, "A" is the clustering task, and the other three tasks are downstream tasks tagged by alphabetical "B", "C", "D".

During the cleaning process, we kept only Arabic texts; all non-Arabic texts were removed, including special characters and numbers. After the cleaning task, we obtained a 1.02GB corpus with 77,020,199 terms. By removing the diacritic marks from the diacritized corpus, we obtained the non-diacritized corpus, [Zerrouki and Balla 2017] have indicated the proportion based on the number of terms, where the Classic Arabic represent 98.85% and the MAS 1.15%. However, our reported results of the percentage of MAS and CA were roughly the same, and obtained by using a Binary classifier as described in section A of the Appendix. The analysis of the content of the final Tashkeela+ corpus is shown in Table 1. We used the FastText[[4]](#footnote-4) python library created by Facebook's AI Research lab for developing the word embedding models. It is an open-source, free, lightweight library that allows us to learn text representations based on a given corpus. However, a stable learning algorithm is one whose prediction does not change much when the training data is slightly modified, where FastText is the most stable word embedding model (WEM), between both GloVe[[5]](#footnote-5) and Word2Vec[[6]](#footnote-6) [Borah et al. 2021; Dharma et al. 2022]. We set the parameters as: *model='skipgram', dim=300, epoch=5, minn=0, maxn=0, wordNgrams=1, minCount=1, ws=5, thread=10, bucket=200000.* With FastText, we converted the diacritized and non-diacritized corpus into word embeddings models, and all the Arabic words are represented with 300-dimensional vectors. From Table 1, we observe that the non-diacritized corpus contains only 498,849 unique words, while the diacritized one contains 1,036,989 unique words. **Is this situation leading to ambiguity?** Four experimental tasks were proposed to answer this question and conclude the impact of diacritization on word embeddings: A) clustering of the nearest words, B) morphological semantic analysis, C) part-of-speech tagging, D) semantic analysis for fake content detection, shown on Figure 1.

Table 1 : Statistics of the diacritized and non-diacritized corpora after a cleaning process represent the number of words, unique words, and the size of both the corpora and their word representation models.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Terms** | **source** | **Diacritized corpus** | | | | **Non-diacritized corpus** | | **Binary classification** | |
| Tashkeela+ | 77,020,199 | **Unique Words** | **Diacritized words** | **Non-diacritized** | **Size** | **Unique Words** | **Size** | **MSA%** | **CA%** |
| Tashkeela | 987,418 | 67,210,618 | 1,448,588 | 1,01 GB | 480,843 | 588 MB | 0.88 | 99.12 |
| Quran | 17,576 | 78,238 | 30 | 1,27 MB | 14,872 | 728 KB | 8.41 | 91.59 |
| Others | 452,603 | 7,932,840 | 349,885 | 122 MB | 452,603 | 70,2 MB | 3.50 | 96.50 |
| **Final Tashkeela+** | | | **1,036,989** | **75,221,696** | **1,798,503** | **1,02 GB** | **498,849** | **682 MB** | **0.74** | **99.26** |
| **FastText Model’s size** | | | **2,34 GB** | | | | **1,12 GB** | |

As far as we know, this study is novel for the Arabic language, and no evaluation materials are available for testing the performance of the diacritized word embeddings models. We have created three evaluation datasets for experiments B, C, and D that appeared in Figure 1 from scratch, which will be introduced sequentially in the following sections. All the words and sentences that emerged in the three datasets were converted to 300-dimensional word representation through the two word embedding models using the FastText library to feed ten different Machine learning classifiers: LogisticRegression (LR), KNeighbors Classifier (KNN), DecisionTree Classifier (DT), RandomForest Classifier (RF), GradientBoosting (GB), XGB Classifier (XGB), Support Vector Machine (SVC), LGBM Classifier (LGBM), MLP Classifier (MLP), TabNet Classifier (TabNet). We would like to mention that our study is to compare the diacritized and non-diacritized models. Thus, almost all the classifier’s parameters are set to default. Except that the LGBMClassifier requires defining the type of the classification, whether it is binary or multiclass, and the deep learning classier TabNet requires increasing the number of epochs to 200 to outstrip the random baseline.

Datasets challenge

It is known that any study needs an evaluation based on a dataset to test the model's performance. However, when it comes to an original work with a low-resource language such as the Arabic language, it is a challenge for researchers to create a dataset from scratch or translate an existing dataset from other languages such as English. Making a dataset by translation is even harder for the Arabic language, especially for finding the correct translation between a set of possibilities. WordSim353[[7]](#footnote-7) and SimLex999[[8]](#footnote-8) are non-diacritized datasets for testing semantic performance. It works better on Latin script-based languages. However, in our study, we need an Arabic diacritized semantic dataset for testing the semantic performance; we want to take the challenge of making datasets from scratch to fit our problem and provide a baseline for other researchers on future works on Arabic NLP. We have created three datasets for the three downstream tasks. For ease of reading, we will cover the details of each dataset in their respective sections.

Downstream tasks Experiments

Clustering of the nearest words

It is known that an Arabic word accepts multiple inflections to refer to different categories (sports, body parts, names, etc.) depending on its meaning. However, even for native speakers of Arabic, categorizing non-diacritical words is ambiguous. We evaluated the performance of the word embedding models by observing their ability of capturing word’s similarity. It is worth mentioning that Tashkeela+ is a huge corpus, and it is unrealistic to test and observe by every word. So, in this experiment, we selected 8 different word categories from the Tashkeela+ corpus: body parts, names, measure, build, quantum, family, personality, place, and then selected one word from each category. The words are shown in Table 2. We used the diacritized and non-diacritized models to capture the nearest words in the Tashkeela+ corpus respectively, so as to clarify the power of diacritization in removing ambiguity.

Table 2: A tiny list of words for observing the performance of embedding models on grouping words into the same category.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Non-diacritized** | ***رجل*** | ***ملك*** | ***ضعف*** | ***عمر*** | ***مركب*** | ***خال*** | ***خلق*** | ***منزل*** |
| **Diacritized** | ***رِجْلٌ*** | ***مَلَكٌ*** | ***ضِعْفَ*** | ***عَمَّرَ*** | ***مُرَكَّبٌ*** | ***خَالٌ*** | ***خُلُقٌ*** | ***مَنْزِلٌ*** |
| leg | angel | double | build | complex | uncle | character | home |
| **Categories** | **BODY PARTS** | **NAMES** | **MEASURE** | **BUILD** | **QUANTUM** | **FAMILY** | **PERSONALITY** | **PLACE** |

We used a python script to obtain the vectors representing each of those words from our generated models (FastText diacritized and non-diacritized models); get\_nearest\_neighbors is a predefined method in the Fasttext library to get the nearest words for a given word. In our study, we are trying to get the nearest 50 words. The T-distributed Stochastic Neighbor Embedding (TSNE) library allows 2D graph representation.

In this section, we explain our work with four examples “رجل”, “ملك”, “ضعف”, “عمر”, the rest of the words are given and illustrated in the section B of the Appendix. It is worth mentioning again that a non-diacritized word may have different meanings in different contexts, it is hard to guess the exact meaning because each word can have many possible meanings, depending on its diacritic. We can only translate the diacritized words as they have one clear meaning. First, we compare the result of the non-diacritized word “رجل” in the context of a body part and its diacritized word “رِجْلٌ” meaning a leg. The two graphs in Figure 2 “A” and “B” show the nearest words of the diacritized and non-diacritized models created from the Tashkeela+ corpus. Based on the understanding of the Arabic Language, we observe that the model using the diacritized corpus in Figure 2 “A” has a closer semantic relation than the non-diacritized model of Figure 2 “B”. Moreover, the non-diacritized model has caused an ambiguity, no matter in which context the word “رجل” appears, the model on the non-diacritized corpus always treats it as human or a male person, but it is not the case with the diacritized model. The diacritized word “رِجْلٌ” makes the model results very close to body parts like “شَفَةٌ”, “يَدٍ”, “أُصْبُعٌ”, which mean “finger”, “hand”, “lip”.

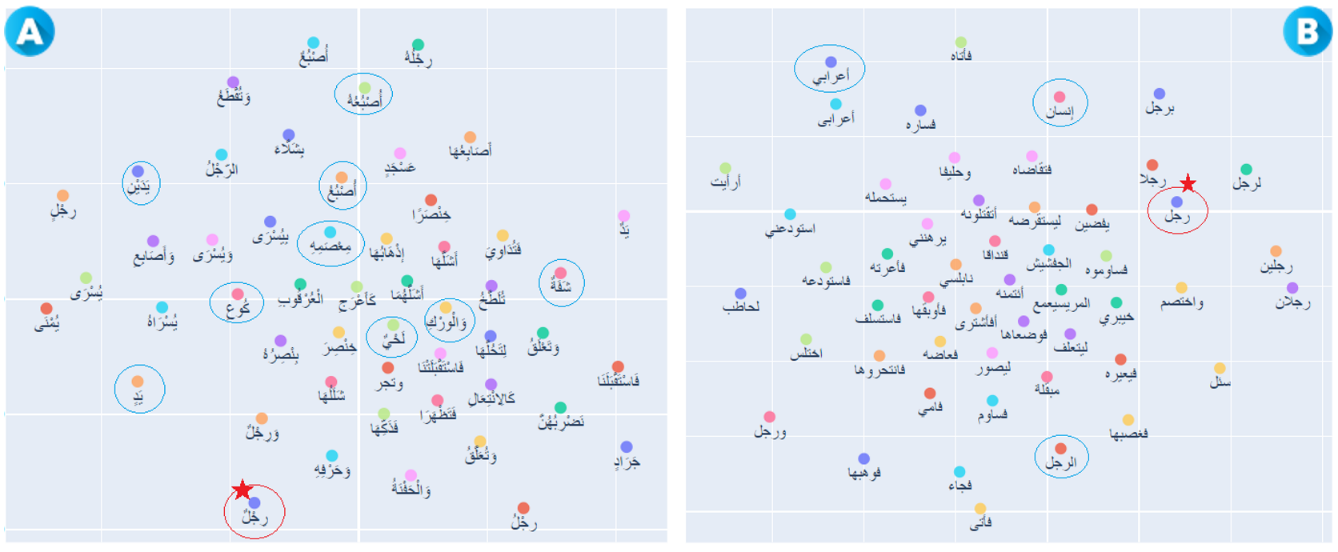


Figure 2: Nearest neighbors for the words “رِجْلٌ” and “رجل”

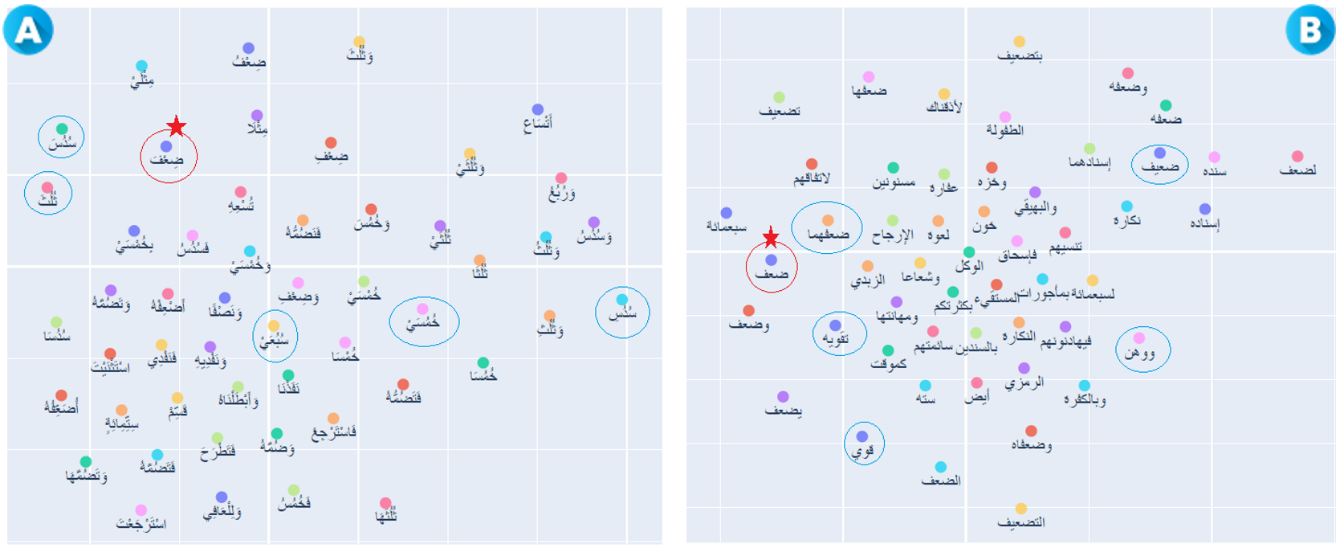


Figure 3: Nearest neighbors for the words “ضِعْفَ” and “ضعف”

The same result appeared in example 2, as shown in Figure 3 “B”, the non-diacritized word “ضعف” is always under the category of “power”, “weakness”, but when it comes to the diacritized word “ضِعْفَ”, in Figure 3 “A”, which means double (x2), we obtain results under the measuring unit category like “ثُلُثَ”, “خُمُسَيْ”, “سُدُسِ”, which means “1/3”, “1/5”, “1/6”.

The third example is with the word “عمر”, the non-diacritized model Figure 4 “B” treats it as a person's name, because most of the nearest words are names like “عبيدالله”, “نافع”, “عباس”, “عثمان”. However, in the diacritized model in Figure 4 “A”, the word “عَمَّرَ” (build) is surrounded by words like “أَجَّرَ” (building), “يُعَمِّرَ” (destroy), “سَكَنَ” (rent), “هَدَمَ” (live).

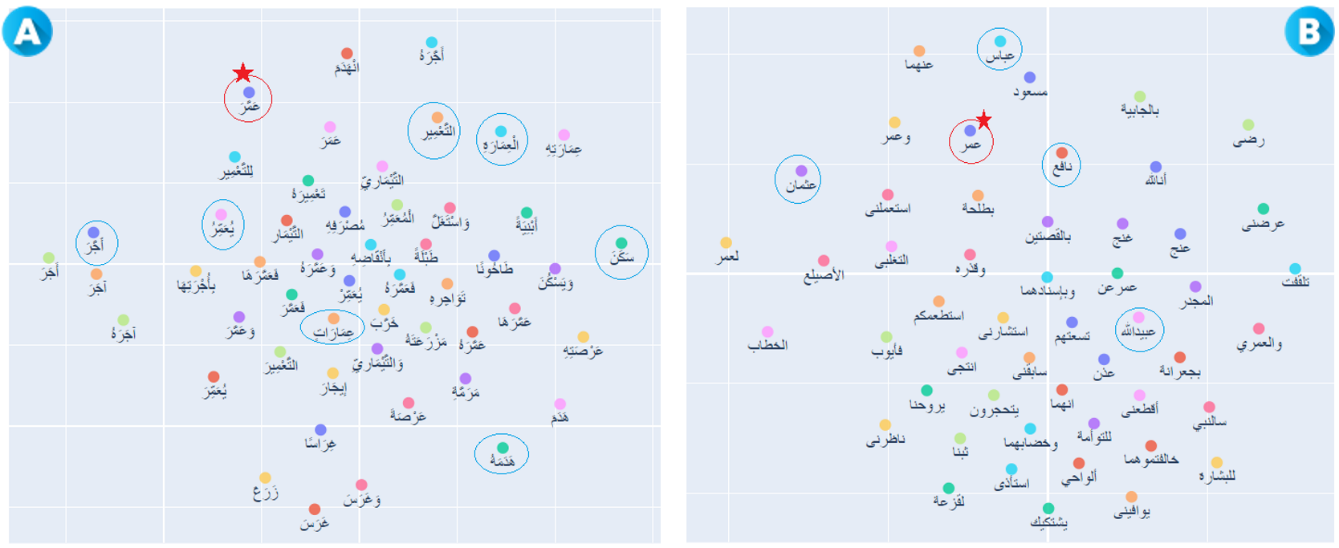


Figure 4: Nearest neighbors for the words “عَمَّرَ” and “عمر”

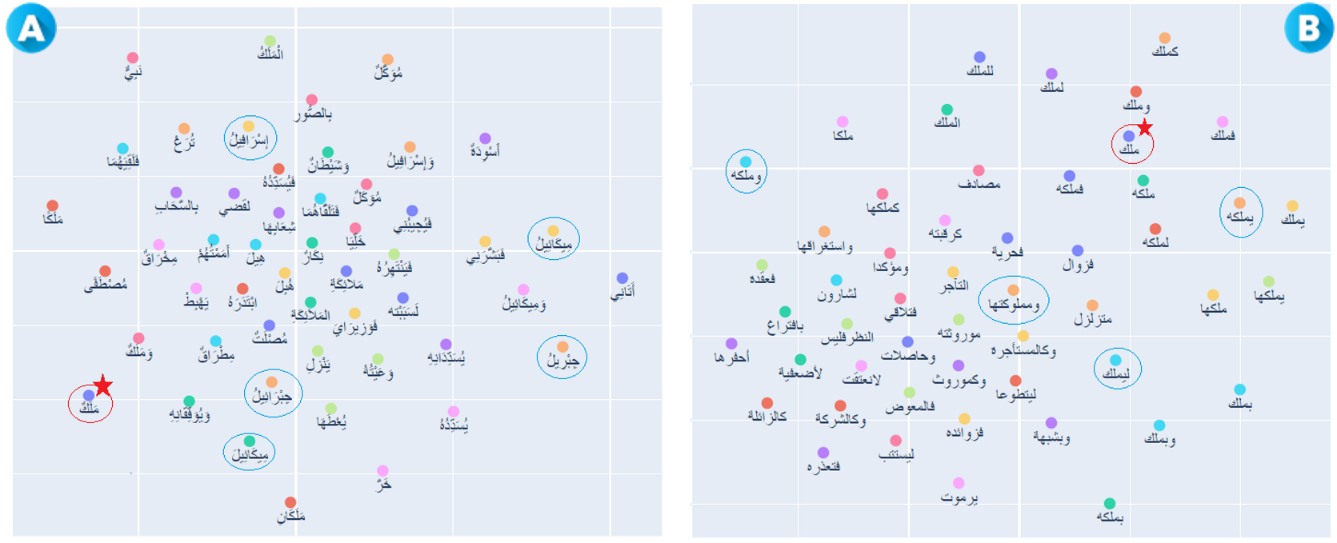


Figure 5: Nearest neighbors for the words “مَلَكٌ” and “ملك”

Finally, for the diacritized word “مَلَكٌ” in Figure 5 “A”, which means Angel, the diacritized model found some angel’s names like: “جِبْرِيلُ” (Gabriel), “وَمِيكَائِيلُ” (Michael), “وَإِسْرَافِيلُ” (Israfil), where the non-diacritized model in Figure 5 “B” poorly clustered it under the category of owning something. This evaluation task shows the power of diacritization on the disambiguity purpose for clarifying the meaning because the diacritic marks are essential characteristics, especially for the word embedding models to capture more syntactic and semantic relations.

Morphological Semantic analysis

In the last section, it was clear that the diacritized model has a better performance in capturing semantics than the non-diacritized one on the word level. In this section, we advance our study of the research question to the sentence level.We propose to analyze and compare the performance of both diacritized and non-diacritized models on morphological semantics.

At present, many popular machine learning algorithms have been developed, and they each have their own advantages and disadvantages. Here, we try to use a variety of algorithms to avoid the disadvantages of using a single one so that the experimental results can be more realistic. This section performed the task with eight well-known machine learning algorithms, including LR, KNN, DT, RF, GB, XGB, SVC, and LGBM. In addition, we also used two deep learning algorithms: MLP and Google’s TabNet 2019. The MLP is commonly used in NLP tasks, while TabNet is one of the newest deep learning algorithms, which is very suitable for tabular datasets.

* + 1. Morphological Semantic Schemes

The Arabic morphological semantic (الدلالة الصرفية) is based on the semantics performed by the Arabic morphological Schemes and their structures. In this section, we target the words that have the same writing but with the same or different diacritic marks. Our study is interested in the ambiguity caused by the words that have multiple possibilities of diacritics. To find them, we have extracted the most frequently used fully diacritized Schemes from Tashkeela+ corpus as shown in Table 3.

Table 3: The 25 most frequent fully diacritized Schemes from the Tashkeela+ corpus and their frequency.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Scheme** | **frequency** | **Scheme** | **frequency** | **Scheme** | **frequency** | **Scheme** | **frequency** | **Scheme** | **frequency** |
| ***فَعَلَ*** | 770830 | ***تَفَعَّلَ*** | 329095 | ***يُفَعَّلُ*** | 102508 | ***فُعِّلَ*** | 33726 | ***تَفَعَّلْ*** | 13425 |
| ***فَعِلَ*** | 721946 | ***فَعْلٌ*** | 245229 | ***فَعَلٌ*** | 97385 | ***أَفْعِلْ*** | 27426 | ***فِعَلٌ*** | 8972 |
| ***يَفْعَلُ*** | 547839 | ***يُفْعَلُ*** | 217885 | ***يُفْعِلُ*** | 79640 | ***فُعْلَلٌ*** | 21403 | ***فُعُلٌ*** | 4221 |
| ***فَعُلَ‏*** | 364936 | ***فُعِلَ*** | 186412 | ***فُوْعِلَ*** | 41944 | ***فِعْلٌ*** | 20613 | ***فَعِلٌ*** | 3932 |
| ***أَفْعَلَ*** | 337114 | ***فَعَّلَ*** | 143586 | ***فُعَلَّلٌ*** | 37982 | ***فُعْلٌ*** | 18529 | ***فَعِّلْ*** | 3378 |

* + 1. Data collection and labeling

We collected and cleaned up 1,708,482 sentences from the Tashkeela+ corpus by a JAVA script. The script helps finding sentence pairs containing target words with the same writing. In addition, the target words must be under one of the 25 most commonly used schemes extracted from the Tashkeela+ corpus shown in Table 3. If the target word appears in both sentences with the same writing and the same Scheme (diacritic marks), then both diacritized words have the same morphological semantic relation; we set “True” as a label. On the other hand, if the target word appears in both sentences with the same writing but not the same Scheme (diacritic marks), then the two words have different morphological semantic relations, the label is “False”. We coded the dataset in a JSON format, labeled and well-shuffled, on two JSON files. The first one, called “documents.json” contains the diacritized and non-diacritized phrases. The other file called “morphological\_signification.json” and has an ID of the phrases, the target word, the position of the word in the two phrases, and the label to describe whether the target words have the same morphological semantic or not. Table 4 shows two different examples. In sample 1, the Scheme of both termA and termB "نَفْسٌ" is "فَعْلٌ", where the target word "نفس" appears in "docA: 4274" and "docB: 124493" with the same writing "نفس", Scheme "فَعْلٌ" and diacritic marks, so termA and termB have the same morphological semantic relation, which is tagged by the label "True". Not the case with sample 2; we note that both terms (termA, termB) from different sentences (docA: 17554, docB: 41322) have the same writing "نفس", but not the same Scheme, where termA's Scheme is "فَعْلٌ", and the Scheme of termB is "فَعَّلَ", we conclude that the terms (termA, termB) do not have the same morphological semantic relation, then the label is "False".

Table 4: Two different examples of the morphological semantic dataset, the first sample: two target words have the same writing and diacritic marks, and the second one: two target words have the same writing but different diacritic marks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Samples** | **Morphological semantic** | **Documents** | **Word translation** |
| **Sample 1** | {  "docA": "4274",  "docB": "124493",  "word": "نفس",  "termA": "**نَفْسٌ**",  "termB": "**نَفْسٌ**",  "indexA": 18,  "indexB": 44,  "label": true  }, | 4274:{  "Diact": "وَإِنْ أَخْطَأَتْ **نَفْسٌ** وَاحِدَةٌ سَهْوًا",  "Ndiact": "وإن أخطأت **نفس** واحدة سهوا",  },  124493:{  "Diact": "لاَ تَأْتِي مِائَةُ سَنَةٍ وَعَلَى الأَرْضِ **نَفْسٌ** مَنْفُوسَةٌ الْيَوْمَ",  "Ndiact": "لا تأتي مائة سنة وعلى الأرض **نفس** منفوسة اليوم"  } | ***نَفْسٌ*** |
| 4274: if one **spirit** sins inadvertently  124493: after one hundred years there will be no **soul** left alive that is living now | Spirit or Soul |
| **Sample 2** | {  "docA": "17554",  "docB": "41322",  "word": "نفس",  "termA": "**نَفْسٌ**",  "termB": "**نَفَّسَ**",  "indexA": 11,  "indexB": 59,  "label": false  }, | 17554:{  "Diact": "أَفْلَحَتْ **نَفْسٌ** زَكَّاهَا اللَّهُ تَعَالَى",  "Ndiact": "أفلحت **نفس** زكاها الله تعالى"  },  41322:{  "Diact": "وَمَثَلُهُ الْمُسَافِرُ إِذَا قَطَعَ مِيلًا أَوْ فَرْسَخًا **نَفَّسَ** ذَلِكَ مِنْهُ",  "Ndiact": "ومثله المسافر إذا قطع ميلا أو فرسخا **نفس** ذلك منه"  } | ***نَفَّسَ*** |
| 17554: a **soul** that God has purified has succeeded  41322: same as a traveler, if he travels a mile, that will **comfort** him. | to comfort , reassure  or  to dispel , banish |

* + 1. Sub datasets

To ensure the validity of the results, we used the same script to create three main morphological semantic datasets, with 50, 100, and 200 target words. The 50 words are included in the 100 words, and the 100 words are included in the 200 words. All the 200 words are shown in Table 5. We could extract hundreds or even thousands of sentence pairs for each target word. We created two sub-data in JSON file format for each main dataset. The first sub dataset was randomly split into 70% training set and 30% test set, which were placed in two folders, respectively. For the second one, we use 10-fold Cross-Validation to split the data into ten (10) parts and place them in ten folders. Each folder contains two JSON files: the training set containing 90% samples and the test set containing 10% samples. So far, we have obtained six datasets, as shown in Table 6.

Table 5 : The extracted 200 non-diacritized words based on the 25 schemes from the diacritized corpus.

|  |
| --- |
| ودع ، قطن ، جلد ، غنم ، عظم ، يصعد ، قلب ، يضحك ، يسر ، تفرغ ، دفن ، تصنع ، أشعر ، أبصر ، بكر ، بقر ، أكرم ، حلم ، ترسل ، أصعد ، أسكن ، تخرج ، غلظ ، برد ، كلب ، تحفظ ، خفض ، ينزع ، تحرق ، شعر ، وضح ، نفس ، يحرق ، طبق ، سخط ، يقرب ، غضب ، أشرب ، نعم ، تشرب ، وسع ، عسر ، وزن ، تهلك ، فرح ، حول ، يهلك ، طرف ، عطش ، خرب ، يفرغ ، أغلق ، أسد ، فزع ، أفلت ، أعجب ، عزر ، يكرم ، تنزل ، تبطل ، رجل ، نفل ، تبدل ، سلب ، وقر ، شرف ، بسط ، محمل ، أنقص ، أضرب ، وجع ، حجب ، تقرب ، يشفع ، يطمع ، تفهم ، كسر ، تنكح ، عذب ، يؤثر ، وتر ، صنف ، شغل ، وسط ، هجر ، كلف ، ترحل ، عمر ، يؤذن ، حرق ، حمد ، أنظر ، يعرض ، نصف ، تفتح ، أسرع ، أقدم ، صبر ، ألزم ، تنكر ، ألصق ، عرج ، يبدل ، تشبه ، تركب ، ورق ، شفع ، زين ، يزرع ، عرق ، قرن ، ثقل ، سفر ، سبح ، وثق ، تحلل ، تمسك ، يعدم ، شرك ، سخر ، فلس ، رقم ، يصلح ، يلصق ، سفه ، رغب ، أصدق ، يرهن ، تثبت ، سبع ، منزل ، يحمد ، صبغ ، تلبس ، تفسخ ، يقلع ، محرز ، يشفق ، غبن ، علف ، يكمل ، يعجز ، تطعم ، يعجل ، أبر ، عقر ، يظلم ، بصر ، خلطة ، يجرح ، بدن ، مبدأ ، تقلع ، حصر ، يتلف ، حزن ، يمرض ، تجبر ، أتبع ، تتبع ، ألبس ، ينفذ ، تتلف ، أعرض ، هدم ، غرق ، بضع ، تبدأ ، أعرف ، طعم ، يغلق ، ستر ، يبغض ، يشغل ، يجلس ، غير ، غمر ، غشي ، تشبع ، خلص ، مركب ، تحدث ، كسل ، طرب ، سكر ، يكذب ، أكذب ، يشبع ، ظفر ، أدب ، يسكر ، خمر ، أقصر ، نظم ، حبل ، معمر ، غلق ، صلب ، صغر ، تسلم ، |

Table 6 : The statistics of the sub datasets.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **50 words** | | | | **100 words** | | | | **200 words** | | | |
| 70% - 30% | | CV (1-fold) | | 70% - 30% | | CV (1-fold) | | 70% - 30% | | CV (1-fold) | |
| Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test |
| Label False | 2735 | 1170 | 3520 | 385 | 5448 | 2332 | 7009 | 771 | 10847 | 4647 | 13954 | 1540 |
| Label True | 1363 | 586 | 1755 | 194 | 2712 | 1166 | 3495 | 383 | 5445 | 2338 | 7012 | 771 |
| Total Pairs | 5854 | | | | 11658 | | | | 23277 | | | |

* + 1. Preprocessing

As we know, the textual data are not ready to be fed to the classifiers; we need to convert them into numerical data. Both the training and testing sets are converted in the way shown in Figure 6. We need to extract the two sentences from the “documents.json” JSON file based on the IDs: “docA” and “docB” found in the “morphological significance” JSON file. Sentences 1, 2, and the target word, are the diacritized and non-diacritized content from both files, respectively. get\_sentence\_vector is a pre-defined FastText method that helps to convert the textual contents to vectors. With 900 numerical features and binary labels, the data is ready to be used in the classifiers.

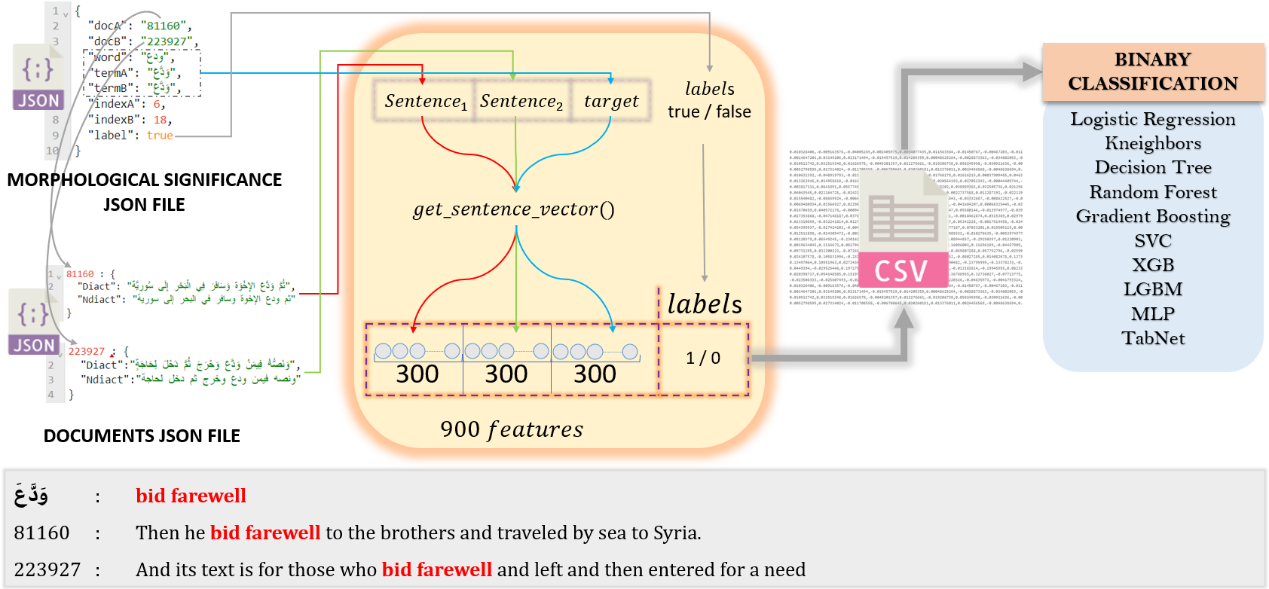


Figure 6: An illustration of morphological semantics pre-processing.

* + 1. Data quality assessment

In our dataset, two factors can affect our results: target words and the collected sentences. First of all, we ensured that all the target words were fully diacritized so that there would be no missed diacritic marks, leading to wrong labeling. Furthermore, because our objective is to investigate the impact of diacritization on word embeddings, to have a reliable accuracy, all the collected sentences are well diacritized. Thus, we ensured the data quality was good enough for testing the morphological semantics.

* + 1. Results and discussion

As we mentioned before, with thousands of sentence pairs that contains the target words, we have created six datasets for the task. Table 7 shows the experimental results on the 70%-30% datasets on 50 words, 100 words, and 200 words. In all the ten algorithms, the diacritized model always yields higher accuracy than the non-diacritized one, whether with 50, 100, or 200 target words. When the target word is 50, the diacritized model is 8.6% more accurate than the non-diacritized model on average. When the number of target words reaches 100 and 200, the diacritized model is higher than the non-diacritized one by 9.356% and 10.533% respectively on average. From this, we can conclude that when there are more target words, the whole text will be more likely to be ambiguous, and at the same time, the diacritized model will have a higher advantage.

Table 7: Morphological semantic experimental results with 70%-30% dataset based on diacritized (DD) and non-diacritized (ND) model

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Train 70% - Test 30%** | **50 words** | | | **100 words** | | | **200 words** | | |
| **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** |
| LogisticRegression (LR) | 85.1 | 72.8 | 12.3 | 74.9 | 69 | 5.9 | 71.1 | 65.6 | 5.5 |
| Kneighbors (KNN) | 92.1 | 84.0 | 8.1 | 89.7 | 83 | 6.7 | 88.9 | 81.1 | 7.8 |
| DecisionTree (DT) | 79.4 | 72.4 | 7 | 84.6 | 66.9 | 17.7 | 83.3 | 66.9 | 16.4 |
| RandomForest (RF) | 93.7 | 80.6 | 13.1 | 91 | 78.7 | 12.3 | 90 | 76.3 | 13.7 |
| GradientBoosting (GB) | 89.7 | 79.1 | 10.6 | 82.7 | 74.4 | 8.3 | 79.9 | 70.3 | 9.6 |
| SVC (SVM) | 96.8 | 88.1 | 8.7 | 94.7 | 85.4 | 9.3 | 94 | 83.6 | 10.4 |
| LGBM | 96.8 | 90.3 | 6.5 | 93.8 | 84.6 | 9.2 | 94.3 | 81.1 | 13.2 |
| DNN (MLP) | 96.9 | 90.7 | 6.2 | 95.7 | 87.6 | 8.1 | 93.4 | 85.1 | 8.3 |
| DNN (TabNet) | 95 | 90.1 | 4.9 | 93 | 86.3 | 6.7 | 93.8 | 83.9 | 9.9 |
| **AVG** | **8.600%** | | | **9,356%** | | | **10,533%** | | |

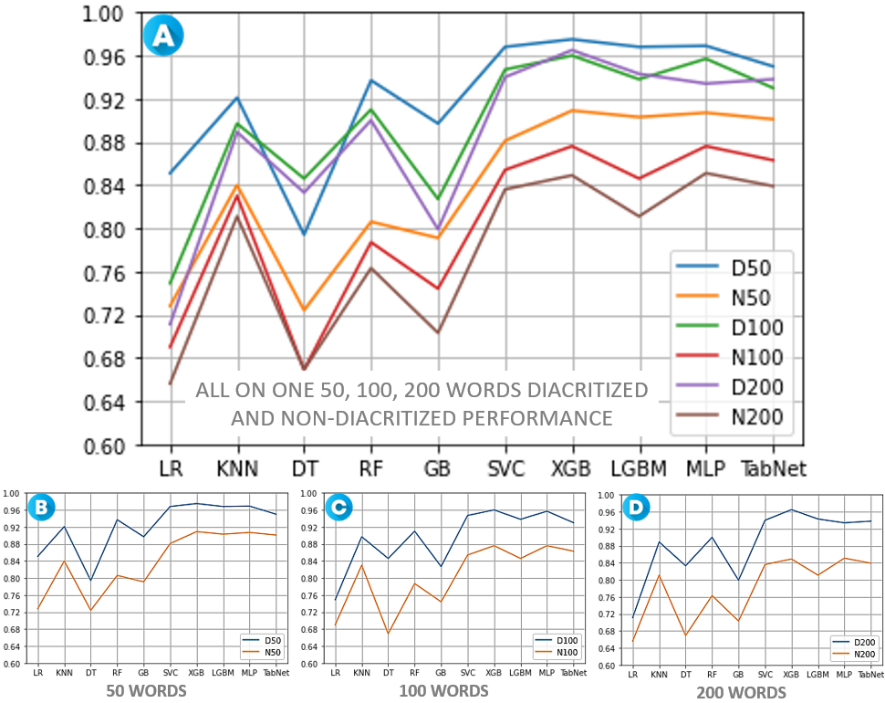


Figure 7: The performance of diacritized (D) and non-diacritized (N) models with 70%-30% dataset in morphological semantic analysis.

In Figure 7, part A generally shows the results of using ten (10) different algorithms to test the diacritized and non-diacritized models on 50, 100, and 200 target words. In order to make the results more apparent, we put the results of 50 words, 100 words, and 200 words in parts B, C, and D, respectively. It is evident that the diacritized model performs better than the non-diacritized model.

Table 8 shows the experimental results on the cross-validation datasets for 50 words, 100 words, and 200 words. In this test, the obtained experimental results were consistent with the previous test. Among the ten (10) algorithms, the diacritized model achieved better results. At the same time, we again see that with the increase in the number of target words, the advantages of the diacritized model are also more apparent.

Table 8: Morphological semantic experimental results with CV dataset based on the diacritized (DD) and non-diacritized (ND) model.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cross Validation**  **10-fold** | **50 words** | | | **100 words** | | | **200 words** | | |
| **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** |
| LogisticRegression (LR) | 87.3 | 78.9 | 8.37 | 82.9 | 75.6 | 7.37 | 78.4 | 71.5 | 6.91 |
| Kneighbors (KNN) | 94.5 | 86 | 8.47 | 94.5 | 85.8 | 8.62 | 94.5 | 85.8 | 8.75 |
| DecisionTree (DT) | 85.8 | 75.7 | 10.13 | 82.8 | 75.9 | 6.83 | 83.4 | 74.5 | 8.96 |
| RandomForest (RF) | 94.5 | 84.4 | 10.13 | 93.9 | 83.3 | 10.58 | 94.5 | 82.4 | 12.11 |
| GradientBoosting (GB) | 92.1 | 82.7 | 9.39 | 86.2 | 78 | 8.24 | 81.3 | 73.1 | 8.23 |
| SVC (SVM) | 97.4 | 89 | 8.38 | 97.2 | 88 | 9.24 | 97.1 | 87.8 | 9.32 |
| LGBM | 97.3 | 91.4 | 5.89 | 96.7 | 89.2 | 7.50 | 95.5 | 85.8 | 9.69 |
| DNN (MLP) | 98 | 91.7 | 6.28 | 97.9 | 90.3 | 7.65 | 97.7 | 90.8 | 6.83 |
| DNN (TabNet) | 97.7 | 90.9 | 6.8 | 97.7 | 89.3 | 8.40 | 97.7 | 88.5 | 9.14 |
| **AVG** | **8.204%** | | | **8.270%** | | | **8.730%** | | |

When we put all the experimental results in Figure 8 part A, the difference between the diacritized and non-diacritized models looks more apparent; the former is entirely above the latter. When we put the test results using different target words in parts B, C, and D, we can also clearly see that the diacritized model outperforms the non-diacritized model.

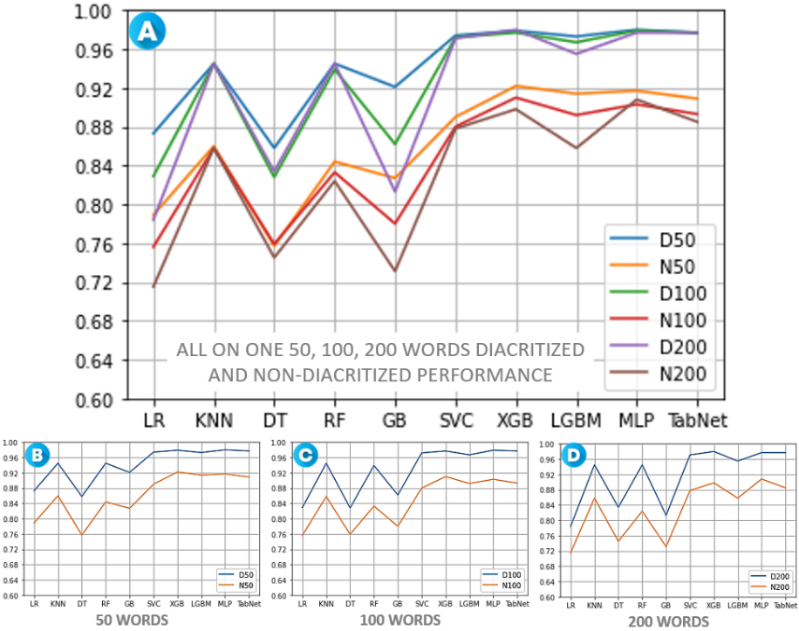


Figure 8: The performance of diacritized (D) and non-diacritized (N) models with the CV dataset in morphological semantic analysis.

At first, we conducted our experiments with the default parameters of the classification algorithms, and the results showed that the diacritized model wins. Then, we investigated on whether the change of parameters will decrease the performance gap between the non-diacritized and the diacritized models. Although hyper-parameter tuning is time-consuming, we still did multiple evaluations on all the six sub-datasets with the TabNet classifier; we found that the gap between the two models did not decrease after tunning. In Table 9, we can find the best parameters for each model with different target words based on the 70%-30% dataset. Once again, we see that the greater the number of target words, the greater the advantage of the diacritized model over the non-diacritized one.

Table 9: The best performance and parameters of discretized and non-discretized models using TabNet based on the 70%-30% datasets.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Best results** | | **DIFF%** | **BEST PARAMETERS** |
| **50 words** | **DD** | 98.3 | **4.9%** | {'mask\_type': 'sparsemax', 'n\_da': 60, 'n\_steps': 2, 'gamma': 1.4, 'n\_shared': 2, 'lambda\_sparse': 3.6373038365307803e-06, 'patienceScheduler': 7, 'patience': 28, 'epochs': 239} |
| **ND** | 93.4 | {'mask\_type': 'sparsemax', 'n\_da': 56, 'n\_steps': 1, 'gamma': 1.4, 'n\_shared': 3, 'lambda\_sparse': 0.00029887415429807467, 'patienceScheduler': 9, 'patience': 26, 'epochs': 242} |
| **100 words** | **DD** | 96.5 | **5.6%** | {'mask\_type': 'sparsemax', 'n\_da': 64, 'n\_steps': 2, 'gamma': 1.2, 'n\_shared': 2, 'lambda\_sparse': 5.253684221521205e-05, 'patienceScheduler': 7, 'patience': 24, 'epochs': 250} |
| **ND** | 90.9 | {'mask\_type': 'entmax', 'n\_da': 64, 'n\_steps': 3, 'gamma': 1.0, 'n\_shared': 2, 'lambda\_sparse': 2.9531668246615665e-05, 'patienceScheduler': 3, 'patience': 15, 'epochs': 227} |
| **200 words** | **DD** | 97.0 | **8.1%** | {'mask\_type': 'sparsemax', 'n\_da': 64, 'n\_steps': 1, 'gamma': 1.4, 'n\_shared': 1, 'lambda\_sparse': 6.1799399573167945e-06, 'patienceScheduler': 6, 'patience': 29, 'epochs': 219} |
| **ND** | 88.9 | {'mask\_type': 'sparsemax', 'n\_da': 56, 'n\_steps': 3, 'gamma': 1.2, 'n\_shared': 2, 'lambda\_sparse': 4.439851102706604e-06, 'patienceScheduler': 4, 'patience': 23, 'epochs': 254} |

Part-Of-Speech tagging

The Arabic Part of Speech (POS) is one of the challenges and a vast domain in Arabic NLP, with a broad community of researchers. To the best of our knowledge, there is not a POS tagged diacritized dataset so far. Hence, we propose creating a diacritized POS dataset that is more specific and limited by using the Arabic rule-based approach to fit our task. To simplify this problem and study the impact of diacritization on word embeddings on Arabic POS, we only focus on Verb and Noun by using a list of specific Schemes. The Scheme matching is the cornerstone of the Arabic morphology; having a list of all schemes referring to the general cases is the best method to extract the exact part of speech from a given diacritized word [Altabba et al. 2010]. Instead of using ten classifiers, we used only nine here. GradientBoosting (GB) suffers from a time-consuming problem in a multiclass classification task; we removed it from the POS tagging task.

* + 1. Part-Of-Speech Schemes

[Altabba et al. 2010] pointed out that Arabic morphological schemes and part-of-speech are closely related, and scheme plays an essential role in determining the POS. As we mentioned earlier, POS tagging itself is a challenging research topic. In order to facilitate our research, we avoid uncertainty and selected only the schemes that accurately correspond to their POS, then create our dataset on this basis. Table 10 shows the POS corresponding to different schemes [Nadeem 2020]. Under certain conditions, words with the same scheme will have the same POS. Let us take the scheme “فَعَلَ” marked in red in Table 10 as an example. “فَعَلَ” means “he did”, and the words based on “فَعَلَ” can be “سَئَلَ”, “جَمَعَ”, “كَتَبَ”, which means, “he wrote”, “he collected” and, “he asked” respectively. These three words have the same POS as the scheme: VERB ACTIVE PERFECT PAST REGULAR I. Thus, we can easily find the POS corresponding to a word through its scheme.

Table 10: List of the most common schemes for Arabic verbs and nouns with a specific Part-of-Speech.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NOUNS** | | | **VERBS** | | | | | | SCHEME | Meaning, use and relationship |
| PECIPIENT passive participle | AGENT active participle | VERBAL NOUN | **PASSIVE** | | IMPERATIVE command | | **ACTIVE** | |
| IMPERFECT present | PERFECT past | IMPERFECT present | PERFECT  past |
| اسم المفعول | اسم الفاعل | المصدر | المضارع | الماضي | الأمر | | المضارع | الماضي | الوزن | المعنى، الاستعمال و العلاقة |
| ***مَفْعُولٌ*** | ***فَاعِلٌ*** | ***فَعْلٌ***  ***فُعُولٌ***  ***فِعَالَةٌ*** | ***يُفْعَلُ*** | ***فُعِلَ*** | ***اُفْعَلْ***  ***اُفْعِلْ***  ***اُفْعُلْ*** | ***اِفْعَلْ***  ***اِفْعِلْ***  ***اِفْعُلْ*** | ***يَفْعَلُ***  ***يَفْعِلُ***  ***يَفْعُلُ*** | ***فَعَلَ***  ***فَعِلَ***  ***فَعُلَ*** | **I** | REGULAR |
| ***مُفَعَّلٌ*** | ***مُفْعَّلٌ*** | ***تَفْعِيلٌ***  ***تَفْعِلَةٌ*** | ***يُفَعَّلُ*** | ***فُعِّلَ*** | ***فَعِّلْ*** | | ***يُفَعِّلَ*** | ***فَعَّلَ*** | **II** | CAUSATIVE INTESIVE OR DENOMINATIVE |
| ***مُفَاعَلٌ*** | ***مُفَاعِلٌ*** | ***مُفَاعَلَةٌ***  ***فِعَالٌ*** | ***يُفَاعَلُ*** | ***فُوعِلَ*** | ***فَاعِلْ*** | | ***يُفَاعِلَ*** | ***فَاعَلَ*** | **III** | RECIPROCAL |
| ***مُفْعَلٌ*** | ***مُفْعِلٌ*** | ***إِفْعَالٌ*** | ***يُفْعَلُ*** | ***أُفْعِلَ*** | ***أَفْعِلْ*** | | ***يُفْعِلُ*** | ***أَفْعَلَ*** | **IV** | CAUSATIVE |
| ***مُتَفَعَّلٌ*** | ***مُتَفَعِّلٌ*** | ***تَفَعُّلٌ*** | ***يُتَفَعَّلُ*** | ***تُفُعِّلَ*** | ***تَفَعَّلْ*** | | ***يَتَفَعَّلُ*** | ***تَفَعَّلَ*** | **V** | REFLEXIVE OF II |
| ***مُتَفَاعَلٌ*** | ***مُتَفَاعِلٌ*** | ***تَفَاعُلٌ*** | ***يُتَفَاعَلُ*** | ***تُفُوعِلَ*** | ***تَفَاعَلْ*** | | ***يَتَفَاعَلُ*** | ***تَفَاعَلَ*** | **VI** | REFLEXIVE OF III |
| ***-*** | ***مُنْفَعِلٌ*** | ***اِنْفِعَالٌ*** | ***-*** | ***-*** | ***اِنْفَعِلْ*** | | ***يَنْفَعِلُ*** | ***اِنْفَعَلَ*** | **VII** | PASSIVE OF I |
| ***مُفْتَعَلٌ*** | ***مُفْتَعِلٌ*** | ***اِفْتِعَالٌ*** | ***يُفْتَعَلُ*** | ***اُفْتُعِلَ*** | ***اِفْتَعِلْ*** | | ***يَفْتَعِلُ*** | ***اِفْتَعَلَ*** | **VIII** | REFLEXIVE OF I |
| ***-*** | ***مُفْعَلٌّ*** | ***اِفْعِلاَلٌ*** | ***-*** | ***-*** | ***-*** | | ***يَفْعَلُّ*** | ***اِفْعَلَّ*** | **IX** | COLORS DEFECTS |
| ***مُسْتَفْعَلٌ*** | ***مُسْتَفْعِلٌ*** | ***اِسْتِفْعَالٌ*** | ***يُسْتَفْعَلُ*** | ***اُسْتُفْعِلَ*** | ***اِسْتَفْعِلْ*** | | ***يَسْتَفْعِلُ*** | ***اِسْتَفْعَلَ*** | **X** | CAUSATIVE REFLEXIVE |

* + 1. Data collection and labeling

In Table 10, there are 86 schemes in total. However, not all of them are suitable for our task (the reasons are explained in section ‎5.3.4). We selected 66 suitable schemes. We have created the dataset through a rule-based approach. Similar to previous tasks, we created a JAVA script to collect 6,668 diacritized words containing a scheme. By removing the diacritic marks of the diacritized words, we got 5,243 non-diacritized words. We extracted 44,831 sentences containing the target word to have 47,068 samples in total. Among the 66 schemes, some contain affixes, such as prefixes, suffixes, and infix, while others do not. For different situations of each scheme, we manually set its rules. When the target word meets these rules, we can determine the accurate POS of the word and set it as the label. The target word must meet two rules to find the exact POS: first, it contains the same diacritic marks as the corresponding scheme; second, it contains the same affix (if any) as the corresponding scheme. In Table 11, we can find three examples. The scheme does not contain any affix in sample A “جَلَسَ”, so the POS will be selected based on the diacritic marks only; we note that the Arabic verbs cannot be ended by “ة” or “ه”, so the tagged POS is “VERB ACTIVE PERFECT PAST REGULAR I”. Sample B is different. The target word “يُسْتَعْمَلُ” has got the same diacritic marks as the scheme “يُسْتَفْعَلُ”, and it has also got the same prefix “يُسْتَ” as the scheme. It can be tagged with POS “VERB PASSIVE IMPERFECT PRESENT CAUSATIVE REFLEXIVE X”. However, in sample C, the target word “بَائِعٌ” and the scheme “فَاعِلٌ” have got the same diacritic marks and the same infix “ا”; thus, we can tag “NOUN AGENT ACTIVE PARTICIPLE DOER REGULAR I” as the POS of the word.

Table 11: Three different samples of the Part-of-Speech dataset.

|  |  |  |
| --- | --- | --- |
| A | {  "DD\_WORD": "جَلَسَ",  "ND\_WORD": "جلس",  "SCHEME": "فَعَلَ",  "DD\_SENTENCE": "فَلَمَّا تَكَلَّمَ جَلَسَ عُمَرُ رَضِيَ اللَّهُ عَنْهُ",  "ND\_SENTENCE": "فلما تكلم جلس عمر رضي الله عنه",  "POS": "VERB\_ACTIVE\_PERFECT\_PAST:REGULAR:I",  "INDEX":  }, | |
| He (sat down) : *verb past* | When someone spoke, 'Umar sat down |
| B | {  "DD\_WORD": "يُسْتَعْمَلُ",  "ND\_WORD": "يستعمل",  "SCHEME": "يُسْتَفْعَلُ",  "DD\_SENTENCE": "وَقَدْ يُسْتَعْمَلُ فِي غَيْرِهِ مَجَازًا",  "ND\_SENTENCE": "وقد يستعمل في غيره مجازا",  "POS": "VERB\_PASSIVE\_IMPERFECT\_PRESENT:CAUSATIVE REFLEXIVE:X",  "INDEX": 1  }, | |
| ~ (be used) : *verb present* | It may be used metaphorically in others. |
| C | {  "DD\_WORD": "بَائِعٌ",  "ND\_WORD": "بائع",  "SCHEME": "فَاعِلٌ",  "DD\_SENTENCE": "لِأَنَّ مُوَاصِلَ الْمُوَاصِلِ بَائِعٌ وَمُشْتَرٍ",  "ND\_SENTENCE": "لأن مواصل المواصل بائع ومشتر",  "POS": "NOUN\_AGENT\_ACTIVE\_PARTICIPLE\_DOER:REGULAR:I",  "INDEX": 3  }, | |
| (seller) : *noun* / A person who sells | Because Muwasil al-Muwasil is a seller and a buyer. |

* + 1. Sub datasets

Similar to the morphological semantics sub-dataset described in section ‎5.2.3, for better evaluation, we created 38521f samples in total and generated two sub-datasets with them: one is a randomly assigned 70%-30% dataset, and the other is a 10-fold Cross Validation dataset. For the 70%-30% dataset, there are 29003 samples in training set and 9518 samples in testing set. For the 10-fold cross validation dataset, each fold has 35521 samples for training and 3000 samples for testing.

* + 1. Data quality assessment

As we mentioned before, our model is built based on the Tashkeela+ corpus. Table 10 shows a set of schemes. If a word's scheme matches with one of them, it is easy to find the POS of the word. However, only being easy to find the POS is not enough. We ensured that in the Tashkeela+ corpus, the target word matched with the scheme is wholly diacritized, and the sentence containing the target word is also well diacritized to compare the different performances of diacritized and non-diacritized models. In addition, we also eliminated the problem of data imbalance and ensured that the training set and the testing set contained the same type of labels to guarantee the validity of the results. After the step-by-step screening, we obtained 66 from 86 schemes and created 59 POS labels on this basis. Finally, we created a high-quality dataset that is completely suitable for our research.

* + 1. Preprocessing

As shown in Figure 9, similar to the morphological semantic preprocessing ‎in section ‎5.2.4, we used get\_sentence\_vertor to convert the training and testing data to numerals. Instead of having two sentences, we have only one here and a total of 600 features. As for the labels, we have obtained 59 numbers corresponding to 59 POS tags.

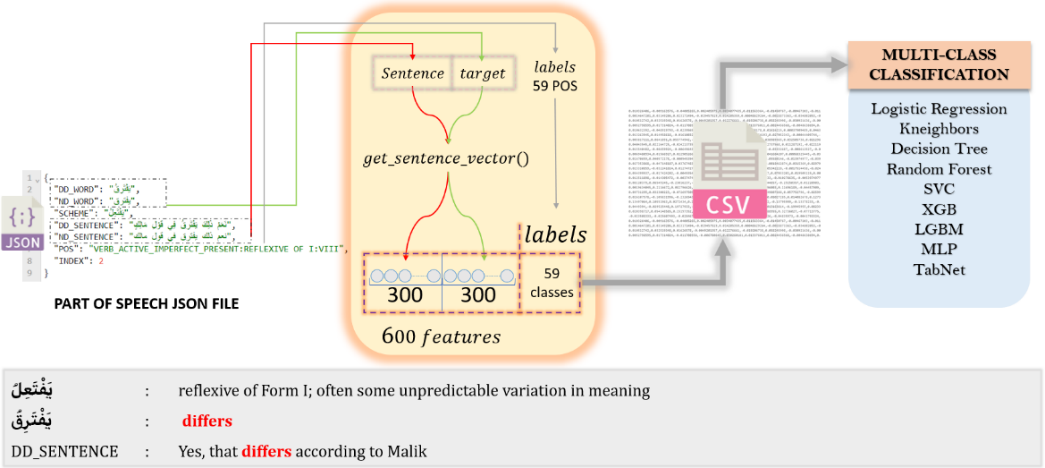


Figure 9: An illustration of Part-of-Speech pre-processing.

* + 1. Results and discussion

From Figure 10, we can easily see that the diacritized model is better than the non-diacritized one. Furthermore, in Table 12, we can find that the former outperforms the latter up to 14.467% on average, which is a more significant gap than the previous morphological semantics analysis. We hence can conclude that diacritization plays a more critical role in POS tagging. Impressively, our results of the diacritized model are very close to 100%. As [Altabba et al. 2010] and many other researchers have declared, Scheme and POS are very closely related. At the same time, the non-diacritized model suffers from massive ambiguity and performs poorly on this task. This again confirms the vital role of Arabic diacritization in disambiguation and helps word embeddings capture semantic relations better.

Table 12: POS tagging experimental results based on the diacritized (DD) and non-diacritized (ND) models.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 10-fold** | | |
| **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** |
| LogisticRegression (LR) | 99.4 | 84.5 | 14.9 | 99.6 | 84.9 | 14.7 |
| Kneighbors (KNN) | 99.4 | 87.2 | 12.2 | 99.5 | 87.4 | 12.13 |
| DecisionTree (DT) | 99.7 | 82.2 | 17.5 | 99.7 | 80.9 | 18.85 |
| RandomForest (RF) | 99.7 | 88.1 | 11.6 | 99.8 | 88.5 | 11.32 |
| XGBoost (XGB) | 99.7 | 88.6 | 11.1 | 99.8 | 88.1 | 11.69 |
| SVC (SVM) | 99.6 | 88.6 | 11 | 99.7 | 89.0 | 10.74 |
| LGBM | 96.3 | 76.7 | 19.6 | 97.9 | 78.5 | 19.4 |
| DNN (MLP) | 99.7 | 84.9 | 14.8 | 99.8 | 85.7 | 14.1 |
| DNN (TabNet) | 99.5 | 82.0 | 17.5 | 99.5 | 82.7 | 16.78 |
| **AVG** | **14.467%** | | | **14.412%** | | |

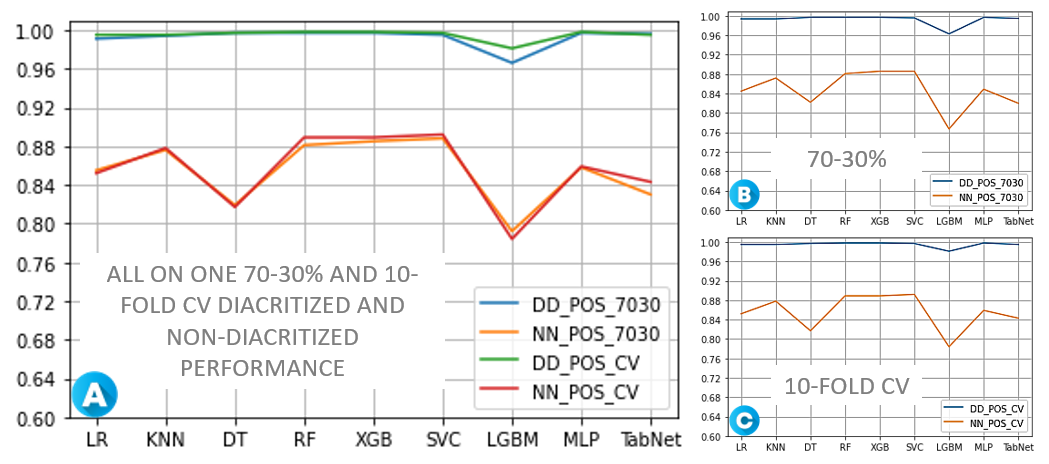


Figure 10: The performance of diacritized (DD) and non-diacritized (ND) models in POS tagging.

Semantic Analysis

In order to further explore the impact of Arabic diacritization, we propose to have the semantic analysis through Fake Content Detection (FCD). We have witnessed a rise in fake content, and fake Arabic content appeared significantly on the Internet. The presence of these contents is one of the most critical problems facing automatic processing. Often, Arabic speakers can hardly distinguish between genuine and fake content. However, we sorely need algorithms that mimic a human’s ability to identify fake Arabic content for automatic processing. We used the diacritized and non-diacritized models for the FCD task, and by the accuracy of the prediction results, we can see the performance of the two different models in semantic analysis.

* + 1. Data collection and labeling

When collecting data for semantic analysis, we faced some challenges. For example, we need to find some fully diacritized texts, and these texts should be able to be labeled; more importantly, all the words appearing in the texts must belong to our Tashkeela+ corpus; otherwise, there will be an Out of Vocabulary (OOV) problem. In the end, we managed to find very suitable data: Quotes from the Muslim's Prophet, the so-called Hadith. A hadith refers to sayings, actions, and characteristics of the Prophet Muhammad peace be upon Him (PBUH). It is considered the second reference of legislation for Muslims after the Holy Quran [Binbeshr et al. 2021]. It is fully diacritized and included in the Tashkeela+ corpus. From two of the most famous Muslim Sunni books: "**AL-BUKHARI**" and "**MUSLIM**", we have collected 1600 Hadiths labeled as "True", and from social network platforms, we collected 461 fake contents, which were identified as misrepresented data, labeled as "False". The dataset contains 2061 samples with 12,758 diacritized words and 10,522 non-diacritized words coded in a JSON file. Figure 11 shows the structure of the dataset.

* + 1. Data quality assessment

In this experiment, a good dataset needs to meet three conditions: first, it should be completely diacritized; Second, all words that appeared in the dataset should be included in the Tashkeela+ corpus; Third, it should be well labeled based on different data sources. Our dataset fully meets the above three conditions, which avoids the problem of Out-of-Vocabulary and fulfills the needs of our comparative research.

* + 1. Preprocessing

As shown in Figure 11, like the preprocessing strategies in morphological semantic analysis and POS tagging preprocessing sections (‎5.2.4, ‎‎5.3.5), we converted text to numbers by *get\_sentence\_vector()*. The difference is: In the FCD task, we only want to know whether the target sentence is fake or not. Thus, 300 features plus binary labels are enough. Instead of using 10-fold cross-validation, we used 5-fold so that we could have sufficient data for training and testing. After putting the final data into the CSV files, we can use machine learning algorithms to complete the classification task.

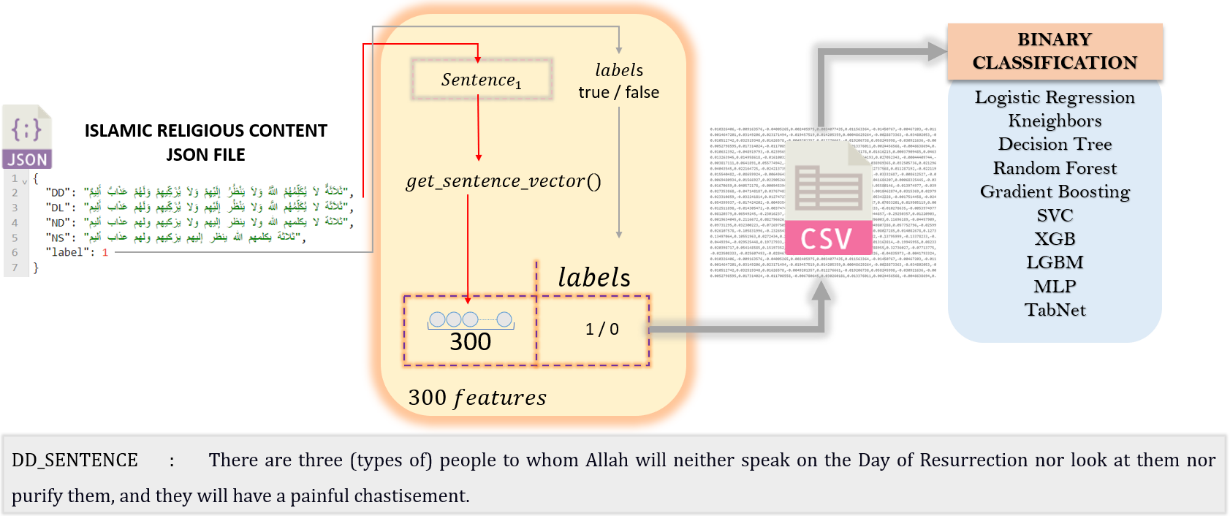


Figure 11: An illustration of Fake Content pre-processing.

* + 1. Results and discussion

Through Table 13 and Figure 12, we can still see that the diacritized model outperforms the non-diacritized one. In the FCD task, the dataset we use is tiny, and the difference between the two models is only 7%. According to the findings in morphological semantics analysis and POS tagging (‎5.2.6, ‎5.3.6) sections, we speculate that if we use a larger dataset, we should be able to see a bigger difference.

Table 13: Fake Content Detection experimental results based on the diacritized (DD) and non-diacritized (ND) models.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 5-fold** | | |
| **DD%** | **ND%** | **DIFF%** | **DD%** | **ND%** | **DIFF%** |
| LogisticRegression (LR) | 95.3 | 85.0 | 10.3 | 96.2 | 86.3 | 9.86 |
| Kneighbors (KNN) | 89.4 | 80.7 | 8.7 | 90.4 | 80.3 | 10.16 |
| DecisionTree (DT) | 84.2 | 77.6 | 6.6 | 83.5 | 76.1 | 7.32 |
| RandomForest (RF) | 90.6 | 84.4 | 6.2 | 91.6 | 85.5 | 6.14 |
| GradientBoosting (GB) | 90.9 | 86.7 | 4.2 | 93.3 | 87.2 | 6.04 |
| SVC (SVM) | 95.9 | 89.6 | 6.3 | 96.8 | 90.7 | 6.16 |
| XGBoost (XGB) | 92.2 | 85.9 | 6.3 | 94.0 | 87.1 | 6.94 |
| LGBM | 92.4 | 86.8 | 5.6 | 93.7 | 87.2 | 6.48 |
| DNN (MLP) | 95.4 | 90.1 | 5.3 | 96.8 | 90.6 | 6.14 |
| DNN (TabNet) | 91.5 | 83.3 | 8.2 | 96.6 | 88.6 | 7.94 |
| **AVG** | **6.770%** | | | **7.318%** | | |

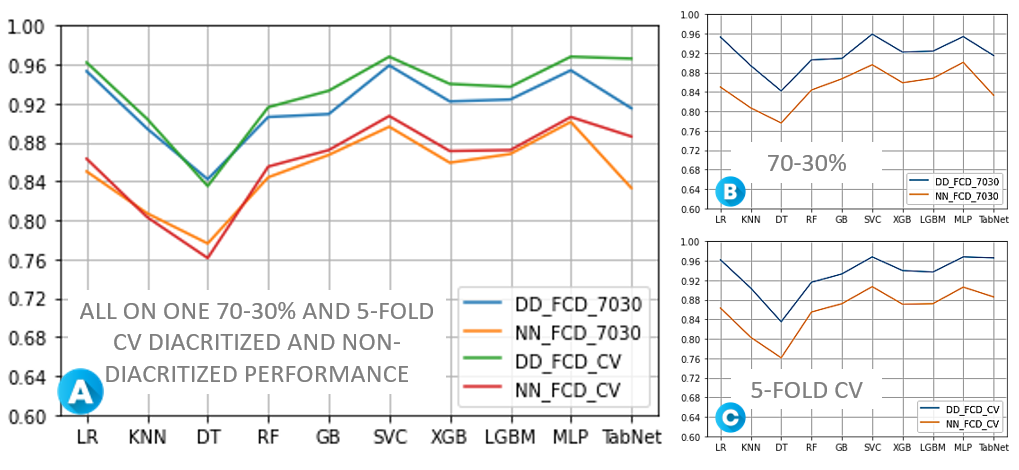


Figure 12: The performance of diacritized (DD) and non-diacritized (ND) models in Fake Content Detection.

Comparison AND LIMITAION

Most current Arabic researchers focus more on collecting non-diacritized data or removing the diacritical marks in the cleaning task. Although cleaning data is an essential process for Latin script-based languages like English, it is not the case for the Arabic language, especially when ignoring one of the critical characteristics of the language, posing a crucial problem, morphological, grammatical, and semantic ambiguity.

Nowadays, Modern Standard Arabic (MSA) texts are written without diacritic marks, making building a non-diacritized corpus and word embedding models easy. This brings much convenience to the research in Arabic NLP applications. On the contrary, adding diacritic marks to a paragraph of MSA text is a complex process. In recent years, a research field has emerged called Arabic diacritization, which specializes in how to restore the diacritic marks of Arabic text. For these reasons, the existing corpora are very few and expensive. However, there is a well-known and freely available corpus called Tashkeela, to which we have added some diacritized texts into it and call it Tashkeela+ and used in this study. It covers a wide range of contents and provides convenience for our research.

A comparative study was the aim of this paper. When your research is faced with a low-resourced language, it is set with difficulties. Creating three new datasets is a time-consuming and challenging process, and we finally customized three datasets for this study. Furthermore, our datasets can be considered a baseline for future studies in the same field, which are time-saving, flexible, convenient, and freely available.

Although we have found the impact of Arabic diacritization on word embeddings through various aspects and demonstrated the power of diacritization, our study still has certain limitations. We used FastText for word representation, but it is a non-contextual tool. So, this study is limited to non-contextual word embeddings only. This study can be further developed in two different ways, non-contextual and then contextual. Our POS dataset is limited to a rule-based approach that uses a list of known schemes (verbs and nouns). A broader diacritized POS resources are required. We are inviting our community for more investigation on the impact of the diacritization on POS in the future. Besides, due to limited resources, our Tashkeela+ corpus mainly contains classical Arabic and Islam-related content. The content of the corpus can be enriched by adding more Modern Standard Arabic content and from different domains such as economics, sports and politics in the future.

Conclusion

This paper developed and compared word embedding models based on two corpora: diacritized and non-diacritized. To address the impact of diacritization on word embeddings and see its power in removing ambiguity, we tested our approach in four ways: clustering of the nearest words, semantic analysis, morphological semantic analysis, and part-of-speech tagging. We found that in clustering the nearest words, the diacritized model has a closer semantic relation than the non-diacritized model. In order to investigate the impact of diacritization on word embeddings, we created three new datasets from scratch to be used in the three downstream tasks. Our results show that the diacritized model outperforms the non-diacritized model. In addition, with the increase in the number of target words, the advantages of the diacritized model are also more apparent. In the POS tagging task, the gap between the diacritized and non-diacritized model is much higher than in the other tasks, revealing that diacritization has more significance in POS tagging. Four different tests all point to the same result. We hence concluded that diacritization in the Arabic language positively impacts word embeddings, and improving the underlying diacritization technology can significantly enhance the performance.

Our article opens up many new research directions:

* First, due to the limited resources of Arabic diacritized data, future work can focus on establishing high-quality diacritized corpora, with adding more Model Standard Arabic and covering multi-topics, to provide the community more benchmarks on our three semantic datasets.
* Second, we propose to enrich the datasets presented in this work or create more diacritized semantic datasets for evaluating the performance of the word embedding models.
* Third, a new study can be addressed on the impact of diacritization on contextual word embedding models.
* Furthermore, a comparison between diacritized non-contextual word embedding models and the contextual word embedding models is also worthy to do in the future.

References

1. Wael Abid, Younes Bensouda Mourri. 2018, Improving English to Arabic Machine Translation. In proceedings of the 32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.
2. Tosin P. Adewumi, Foteini Liwicki, Marcus Liwicki. 2020. The Challenge of Diacritics in Yorùbá Embeddings. In Proceedings of the ML4D Workshop at 34th Conference on Neural Information Processing Systems (NeurIPS) 2020 Workshop on Machine Learning for the Developing World. Vancouver, Canada. arXiv preprint arXiv:2011.07605.
3. Jesujoba Alabi, Kwabena Amponsah-Kaakyire, David Adelani, Cristina España-Bonet. 2020. Massive vs. curated embeddings for low-resourced languages: the case of Yorùbá and Twi. In Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020). Computation and Language. arXi v preprint arXiv:1912.02481. Version 2.
4. Yousef Alotaibi, Ali Meftah, Sid Ahmed Selouani. 2013. Diacritization, automatic segmentation and labeling for Levantine Arabic speech. In Proceedings of IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE). Pages 7–11.
5. Muhammad Altabba, Ammar Al-Zaraee, Mohammad Arif Shukairy. 2010. An Arabic Morphological Analyzer and Part-Of-Speech Tagger. A Thesis Presented to the Faculty of Informatics Engineering, Arab International University, Damascus, Syria.
6. Sawsan Alqahtani, Mahmoud Ghoneim, Mona Diab. 2016. Investigating the Impact of Various Partial Diacritization Schemes on Arabic-English Statistical Machine Translation. In Proceedings of the Association for Machine Translation in the Americas: MT Researchers' Track Conferences. Pages:191–204. Austin, TX, USA.
7. Farid Binbeshr, Amirrudin Kamsin, Manal Mohammed. 2021. A Systematic Review on Hadith Authentication and Classification Methods. ACM Trans. Asian Low-Resour. Lang. Inf. Process., Vol. 20, No. 2, Article 34.
8. Angana Borah, Manash Pratim Barman, Amit Awekar. 2021. Are Word Embedding Methods Stable and Should We Care About It?. Proceedings of the 32nd ACM Conference on Hypertext and social media Pages 45–55 https://doi.org/10.1145/3465336.3475098
9. Eddy Muntina Dharma, Ford Lumban Gaol, Harco Leslie Hendric Spits Warnars , Benfano Soewito. 2022. The Accuracy Comparison Among Word2vec, Glove, And Fasttext Towards Convolution Neural Network Text Classification. Journal of Theoretical and Applied Information Technology.
10. Mona Diab, Mahmoud Ghoneim, Nizar Habash. 2007. Arabic Diacritization in the Context of Statistical Machine Translation. In Proceedings of Machine Translation Summit XI, Copenhagen, Denmark.
11. Esther Fleming. 2020. Is Madinah Arabic free? Retrieved Mai 20, 2022 from : <https://www.sidmartinbio.org/is-madinah-arabic-free/>
12. Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, Tomas Mikolov. 2018. Learning word vectors for 157 languages. arXiv preprint arXiv:1802.06893.
13. Ismail Hadjir, Mohamed Abbache, Fatma Zohra Belkredim. 2019. An Approach for Arabic Diacritization. In International Conference on Applications of Natural Language to Information Systems. NLDB 2019: Natural Language Processing and Information Systems. Pages 337–344. Springer.
14. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze. 2008. An Introductionto Information Retrieval. Cambridge University Press.
15. Abir Masmoudi, Mariem Ellouze Khemakhem, Yannick Estève, Lamia Hadrich Belguith, Nizar Habash. 2014. A Corpus and Phonetic Dictionary for Tunisian Arabic Speech Recognition. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), ID L14-1385, pages 306–310, Reykjavik, Iceland.
16. Abir Masmoudi, Salima Mdhaffar, Rahma Sellami, Lamia Hadrich. 2019. Automatic Diacritics Restoration for Tunisian Dialect. ACM Transactions on Asian and Low-Resource Language Information Processing. 18(3): 1–18.
17. Ayman Nadeem. 2020. Arabic trilateral roots. Retrieved Mai 20, 2022 from : <https://medium.com/@aymannadeem/arabic-trilateral-roots-3186e8319b0>
18. Abu Bakr Soliman, Kareem Eissa, Samhaa R. El-Beltagy. 2017. AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP. Procedia Computer Science. 117: 256-265.
19. Robin Stohler. 2018. Document Embedding Models - A Comparison with Bag-of-Words. Master Thesis, Supervisors: Abraham Bernstein. Merlin - OEC Faculty Information System. University of Zurich. Zurich ZH, Switzerland.
20. Othman Salem Bakheet Qawaqzeh. 2019. Morphological Indication of Al-Khasa'is Book for Ebn Jini: Descriptive Analytical Study. University of Jordan Deanship of Academic Research (DAR).
21. Alexis Neme. 2011. A lexicon of Arabic verbs constructed on the basis of Semitic taxonomy and using finite-state transducers. In Proceedings of the WoLeR 2011 Conference at ESSLLI International Workshop on Lexical Resources At: Ljubliana.
22. Ahmed Younes, Julie Weeds. 2020. Embed More Ignore Less (EMIL): Exploiting Enriched Representations for Arabic NLP. In Proceedings of the Fifth Arabic Natural Language Processing Workshop (pp. 139-154).
23. Taha Zerrouki, Amar Balla. 2017. Tashkeela: novel corpus of Arabic vocalized texts, data for autodiacritization systems. Data Brief, pages147–151.s

AppendiCES

* 1. Binary classification:

In this section, the main objective was to clarify the percentage of MSA and CA in the Tashkeela+ corpus we used. The Tashkeela+ corpus has three data sources, Tashkeela corpus, Quran, and Others (crawled from the web). It is worth mentioning that in the Tashkeela corpus, the authors have indicated the proportion based on a number of terms, where the Classic Arabic accounts for 98.85% and the MSA is 1.15%. the Quran is only composed of classical Arabic, but the proportion of crawled content is not obvious, and due to a large amount of text data, manual classification is also unrealistic. Thus, an automatic classification is required, and we created a python notebook[[9]](#footnote-9) to conduct this task. FastText text classification was our selected tool. We passed through sequential steps:

* Data collection: we extracted classical Arabic content from the Tashkeela corpus, an amount of 97 classical books. For the MSA content, we selected two sub-corpora from Ibrahim Abu El-Khair corpora[[10]](#footnote-10), named: Almasryalyoum and Techreen.
* Data cleaning: we split the textual content into lines, and remove all non-Arabic content. We set a minimum size of a term in each line, to avoid learning too short lines, especially when it contains only one character. The cleaning task ended up with an amount of 6.006.983 samples, including 2.973.342 CA and 3.033.641 MSA samples. We can see that the data is well-balanced.
* Preprocessing: FastText format is required by the FastText algorithm, where the label should be under the format: ‘\_\_label\_\_MSA’ and ‘\_\_label\_\_CA’ for MSA and CA text, respectively. Each line in the FastText textual file should be under the format:

|  |
| --- |
| \_\_label\_\_MSA بما يضمن توفير مصادر وموارد مالية للإنفاق علي النادي وأنشطته المختلفة |
| \_\_label\_\_MSA In order to ensure the provision of financial resources to spend on the club and its various activities |

* Shuffling: this task is very important for the reliability of the results.
* Data splitting: we have split the FastText formatted data into 70% training and 30% testing. We have checked, and guaranteed that the samples in the training and test files are balanced, where the training file contains 4204889 samples in total, including 2123196 (50.49%) MSA and 2081693 (49.51 %) CA samples. And the test file contains 1802094 samples in total, including 910445 (50.52 %) MSA and 891649 (49.48 %) CA samples.
* Training: we fit the FastText classifier with the training file, with the parameter: dim=25, epoch=15, ws=5, "dim" is the dimension of the vectors; "epochs" are the iterations, and "ws" is the window size.
* Testing: the evaluation of the model output great results : positive : 891649, negative : 910445, true positive : 887480, false positive : 4169, true negative : 903537, false negative : 6908, precision : 0.99, recall : 0.99, accuracy : 0.99.
* Deploy the model and get the statistics: for better results, we combined the training and testing data into one, to train through the same training method and output a final FastText model. We make statistics for each corpus: The detailed results are shown in Table 14.

Table 14 Text classification statistics for our used corpora.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **TOTAL** | **MSA** | **CA** | **MSA%** | **CA%** |
| **Tashkeela** | 4,519,919 | 39,834 | 4,480,085 | 0.88 | 99.12 |
| **Quran** | 6,347 | 534 | 5,813 | 8.41 | 91.59 |
| **Others** | 1,029,569 | 36,039 | 993,530 | 3.5 | 96.50 |
| **Tashkeela+** | **758,736** | **56,191** | **7,531,170** | **0.74** | **99.26** |

* 1. Clustering of the nearest words

In this section we are going to present four more examples to illustrate the impact of diacritization on clarifying the meaning. Similar to what we have done in Section 5.1, here the selected words are from category: quantum, family, personality, place. From these Figures: 13, 14, 15 and 16, we can clearly see that the diacritized model has a better performance in clustering the nearest word than the non-diacritized model.

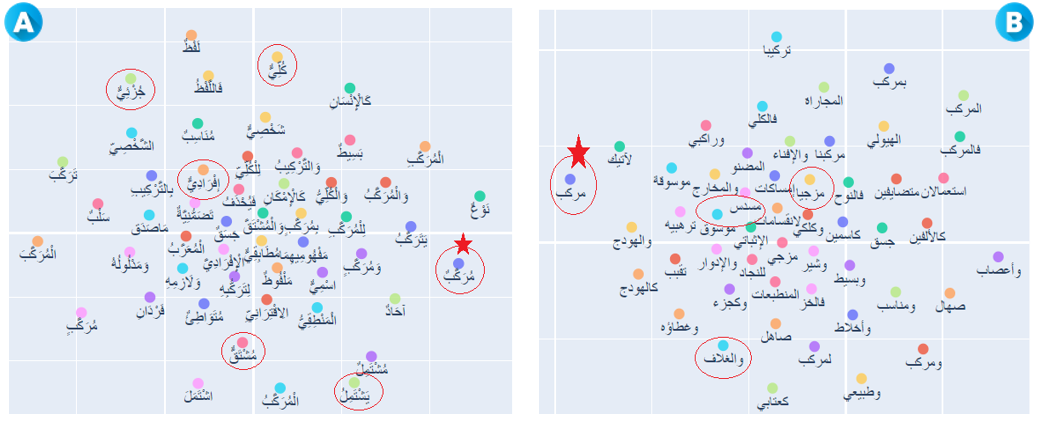


Figure 13 : Nearest neighbors for the words “مُرَكَّبٌ” and “مركب”

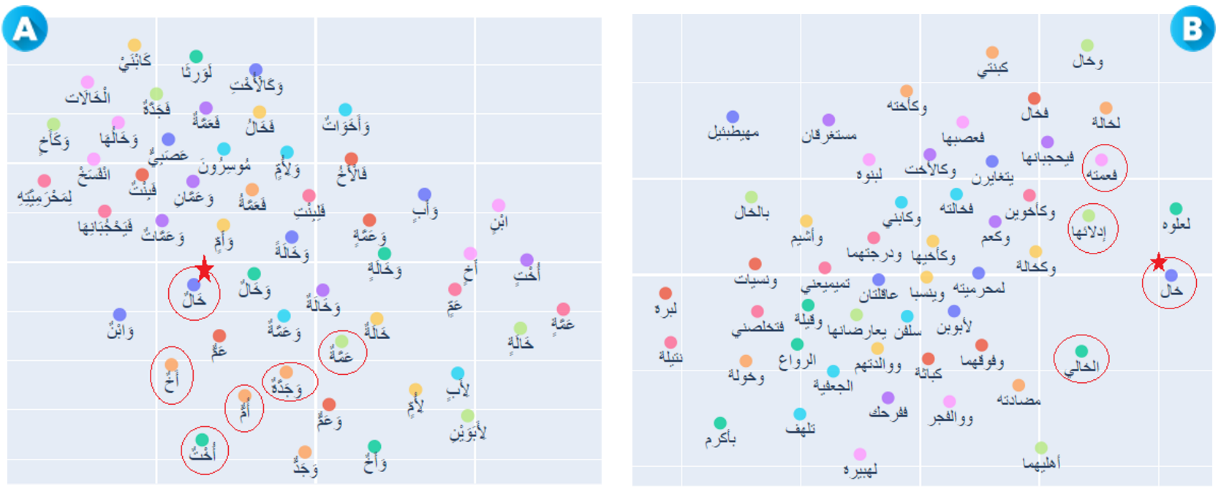


Figure 14: Nearest neighbors for the words “خَالٌ” and “خال”

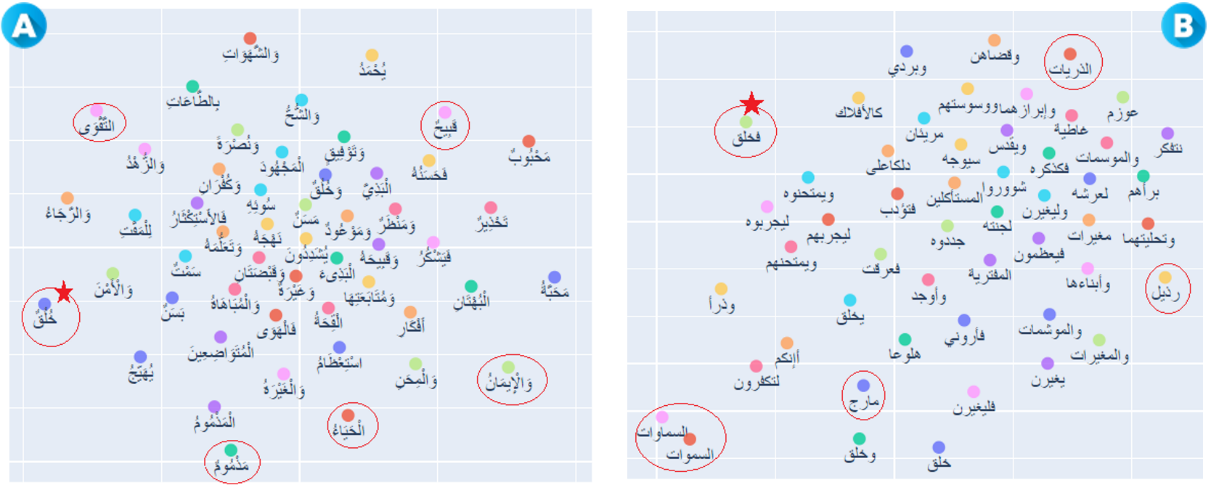


Figure 15: Nearest neighbors for the words “خُلُقٌ” and “خلق”

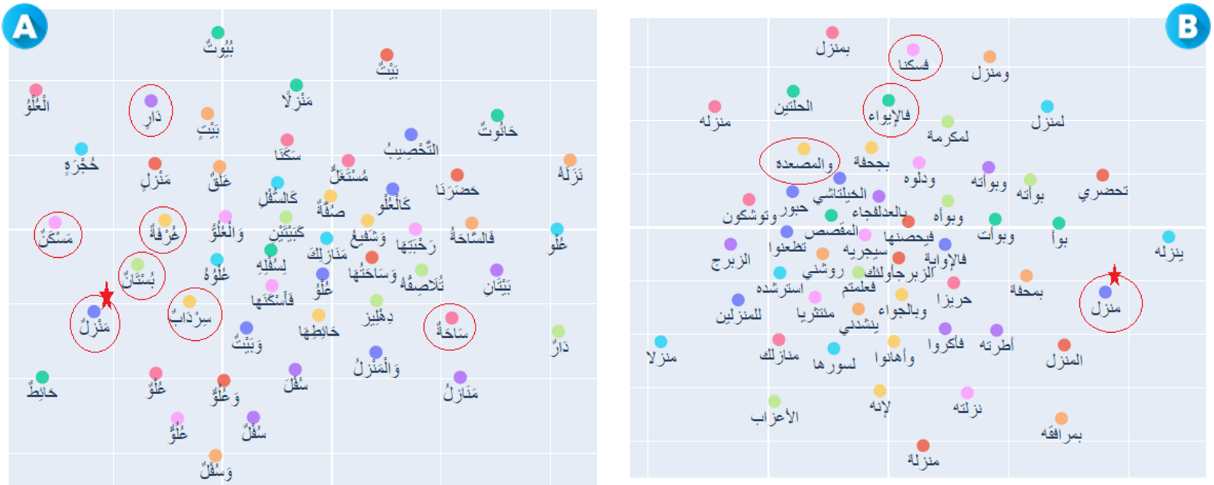


Figure 16: Nearest neighbors for the words “مَنْزِلٌ” and “منزل”

* 1. The impact of the last vowel:

In this section, we are going to check the impact of including the last diacritic mark, we compare two models, the first one is the fully diacritized model, the second one is the diacritized model without the last vowels. We evaluate both models on the same morphological semantics sub-datasets (50, 100, 200) on two deferent splitting formats: 70% training-30% testing data, and 10-folds cross validation, using 10 classification algorithms. From the results, we observe that including the last mark help us to achieve a better result. Among all the sub-datasets. Obviously, the diacritized model, including the last mark, has the best performance, which outperformed the model without the last diacritic mark, of the sub-dataset (50, 100, 200) by (3.35%, 3.3%, 2.72%) on 70-30 datasets, and (2.167%, 2.518, 2.567) for 10-folds cross validation. We have concluded that the last diacritic mark has an impact on the word embedding model, of removing ambiguity and help extracting more semantic and syntactic relations.

Table A: Morphological semantic experimental results with 70%-30% dataset based on diacritized (DD) and without the last vowel (DL) model

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Train 70% - Test 30%** | **50 words** | | | **100 words** | | | **200 words** | | |
| **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** |
| LogisticRegression (LR) | 85.1 | 77.9 | 7.2 | 74.9 | 71.2 | 3.7 | 71.1 | 67.6 | 3.5 |
| Kneighbors (KNN) | 92.1 | 90 | 2.1 | 89.7 | 87.8 | 1.9 | 88.9 | 86.6 | 2.3 |
| DecisionTree (DT) | 79.4 | 76.6 | 2.8 | 84.6 | 76.3 | 8.3 | 83.3 | 80.9 | 2.4 |
| RandomForest (RF) | 93.7 | 89.9 | 3.8 | 91 | 86.9 | 4.1 | 90 | 85.9 | 4.1 |
| GradientBoosting (GB) | 89.7 | 89.9 | -0.2 | 82.7 | 79.7 | 3 | 79.9 | 76.2 | 3.7 |
| SVC (SVM) | 96.8 | 87.2 | 9.6 | 94.7 | 90.5 | 4.2 | 94 | 90.9 | 3.1 |
| XGB | 97.5 | 94.4 | 3.1 | 96 | 93 | 3 | 96.5 | 93.8 | 2.7 |
| LGBM | 96.8 | 90 | 0.9 | 93.8 | 91.1 | 2.7 | 94.3 | 90.2 | 4.1 |
| DNN (MLP) | 96.9 | 94.4 | 2.5 | 95.7 | 92.6 | 3 | 93.4 | 93.1 | 0.3 |
| DNN (TabNet) | 95 | 93.3 | 1.7 | 93 | 94 | -1 | 93.8 | 92.8 | 1 |
| **AVG** | **3.350%** | | | **3,300%** | | | **2,720%** | | |

Table B: Morphological semantic experimental results with CV dataset based on the diacritized (DD) and without the last vowel (DL) model

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cross Validation**  **10-fold** | **50 words** | | | **100 words** | | | **200 words** | | |
| **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** |
| LogisticRegression (LR) | 87.3 | 83.6 | 3.66 | 82.9 | 79.6 | 3.31 | 78.4 | 75.8 | 2.69 |
| Kneighbors (KNN) | 94.5 | 92.2 | 2.32 | 94.5 | 91.6 | 2.91 | 94.5 | 92.1 | 2.38 |
| DecisionTree (DT) | 85.8 | 84.2 | 1.62 | 82.8 | 81.2 | 1.56 | 83.4 | 81.3 | 2.15 |
| RandomForest (RF) | 94.5 | 92.1 | 2.47 | 93.9 | 91.2 | 2.71 | 94.5 | 90.6 | 3.91 |
| GradientBoosting (GB) | 92.1 | 89.6 | 2.49 | 86.2 | 84 | 2.23 | 81.3 | 78.5 | 2.76 |
| SVC (SVM) | 97.4 | 95.4 | 2.05 | 97.2 | 93.7 | 3.53 | 97.1 | 94.5 | 2.55 |
| XGB | 97.9 | 96.2 | 1.71 | 97.7 | 95.9 | 1.79 | 98 | 95.7 | 2.23 |
| LGBM | 97.3 | 95.6 | 1.75 | 96.7 | 94.4 | 2.27 | 95.5 | 92.2 | 3.25 |
| DNN (MLP) | 98 | 96.2 | 1.8 | 97.9 | 95.7 | 2.29 | 97.7 | 96.1 | 1.53 |
| DNN (TabNet) | 97.7 | 95.9 | 1.8 | 97.7 | 95.1 | 2.58 | 97.7 | 95.4 | 2.22 |
| **AVG** | **2.167%** | | | **2.518%** | | | **2.567%** | | |

Table C: POS tagging experimental results based on the diacritized (DD) and without the last vowel (DL) models

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 10-fold** | | |
| **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** |
| LogisticRegression (LR) | 99.4 | 99.1 | 0.3 | 99.6 | 99.4 | 0.19 |
| Kneighbors (KNN) | 99.4 | 99.4 | 0 | 99.5 | 99.5 | -0.04 |
| DecisionTree (DT) | 99.7 | 99.5 | 0.2 | 99.7 | 99.5 | 0.18 |
| RandomForest (RF) | 99.7 | 99.5 | 0.2 | 99.8 | 99.7 | 0.15 |
| XGBoost (XGB) | 99.7 | 99.5 | 0.2 | 99.8 | 99.6 | 0.23 |
| SVC (SVM) | 99.6 | 99.6 | 0 | 99.7 | 99.7 | 0.05 |
| LGBM | 96.3 | 93.9 | 2.4 | 97.9 | 96.2 | 1.72 |
| DNN (MLP) | 99.7 | 99.6 | 0.1 | 99.8 | 99.3 | 0.51 |
| DNN (TabNet) | 99.5 | 99 | 0.5 | 99.5 | 99.3 | 0.11 |
| **AVG** | **0.433%** | | | **0.344%** | | |

Table D: Fake Content Detection experimental results based on the diacritized (DD) and without the last vowel (DL) models

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 5-fold** | | |
| **DD%** | **DL%** | **DIFF%** | **DD%** | **DL%** | **DIFF%** |
| LogisticRegression (LR) | 95.3 | 94.1 | 1.2 | 96.2 | 95.2 | 1,00 |
| Kneighbors (KNN) | 89.4 | 85.4 | 4 | 90.4 | 86.5 | 3,90 |
| DecisionTree (DT) | 84.2 | 83.6 | 0.6 | 83.5 | 81.9 | 1,60 |
| RandomForest (RF) | 90.6 | 91.4 | -0.8 | 91.6 | 91.2 | 0,42 |
| GradientBoosting (GB) | 90.9 | 91.4 | -0.5 | 93.3 | 92.2 | 1,08 |
| SVC (SVM) | 95.9 | 95.4 | 0.5 | 96.8 | 96.3 | 0,58 |
| XGBoost (XGB) | 92.2 | 93.8 | -1.6 | 94.0 | 93.5 | 0,58 |
| LGBM | 92.4 | 92.7 | -0.3 | 93.7 | 93.0 | 0,72 |
| DNN (MLP) | 95.4 | 94.8 | 0.3 | 96.8 | 95.8 | 1,02 |
| DNN (TabNet) | 91.5 | 88.5 | 3 | 96.6 | 96.9 | -0,28 |
| **AVG** | **0.670%** | | | **1.062%** | | |

* 1. Compare the our non-diacritized model to Facebook model:

In this section, we propose checking the validity of the proposed word embedding model, by comparing our non-diacritized word embedding model, with Facebook [[11]](#footnote-11)pretrained model. Facebook distribute pre-trained word vectors for 157 languages, trained on Common Crawl and Wikipedia using FastText. both models represent words in dimension 300 and have the same window size of 5. We evaluate the model’s performance in three different datasets: morphological semantic analysis; part-of-speech tagging; and semantic analysis datasets, using two strategies of splitting data: 70%30%, 10 folds and 5 folds cross validation (for analysis dataset). We observe that our non-diacritized model slightly outperform the Facebook model in a range average of [0.33, 1.01] percent in both morphological semantic analysis and part-of-speech tagging datasets. In the other hand, Facebook model also outperform slightly out non-diacritized model in a range average of [0.71, 1.146] percent. Although we noticed a slight fluctuation accuracy depended on the classification algorithm, and the difficulty in determining who is the winning model, we confirm that the two models give me almost the same performance. The results prove the validity of our non-diacritized model, in the other hand our study is realistic and not biased.

Table E: Morphological semantic experimental results with 70%-30% dataset based on our non-diacritized (O\_ND) and non-diacritized Facebook (F\_ND) model

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Train 70% - Test 30%** | **50 words** | | | **100 words** | | | **200 words** | | |
| **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** |
| LogisticRegression (LR) | 72.8 | 70.9 | 1.9 | 69 | 68 | 1 | 65.6 | 66 | -0.4 |
| Kneighbors (KNN) | 84.0 | 85 | -1 | 83 | 82.6 | 0.4 | 81.1 | 80.9 | 0.2 |
| DecisionTree (DT) | 72.4 | 71.6 | 0.8 | 66.9 | 66.1 | 0.8 | 66.9 | 65.9 | 1 |
| RandomForest (RF) | 80.6 | 80.4 | 0.2 | 78.7 | 77.4 | 1.3 | 76.3 | 75.5 | 0.8 |
| GradientBoosting (GB) | 79.1 | 80.7 | -1.6 | 74.4 | 73.1 | 1.3 | 70.3 | 71.1 | -0.8 |
| SVC (SVM) | 88.1 | 87.4 | 0.7 | 85.4 | 83.6 | 1.8 | 83.6 | 81.9 | 1.7 |
| XGB | 90.9 | 90.7 | 0.2 | 88.1 | 88.1 | -05 | 84.9 | 84.4 | 0.5 |
| LGBM | 90.3 | 89.1 | 1.2 | 84.6 | 83.2 | 1.4 | 81.1 | 79.7 | 1.4 |
| DNN (MLP) | 90.7 | 91.1 | -0.4 | 87.6 | 87 | 0.6 | 85.1 | 85.4 | -0.3 |
| DNN (TabNet) | 90.1 | 88.8 | 1.3 | 86.3 | 84.3 | 2 | 83.9 | 80.4 | 3.5 |
| **AVG** | **0.330%** | | | **1,010%** | | | **0,760%** | | |

Table F: Morphological semantic experimental results with CV dataset based on our non-diacritized (O\_ND) and non-diacritized Facebook (F\_ND) model

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cross Validation**  **10-fold** | **50 words** | | | **100 words** | | | **200 words** | | |
| **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** |
| LogisticRegression (LR) | 78.9 | 77.5 | 1.47 | 75.6 | 75.4 | 0.13 | 71.5 | 71.8 | -0.24 |
| Kneighbors (KNN) | 86 | 85.4 | 0.59 | 85.8 | 85.2 | 0.62 | 85.8 | 85.2 | 0.54 |
| DecisionTree (DT) | 75.7 | 76.5 | -0.87 | 75.9 | 73.5 | 2.48 | 74.5 | 73.8 | 0.63 |
| RandomForest (RF) | 84.4 | 83.4 | 0.99 | 83.3 | 82.1 | 1.25 | 82.4 | 81.5 | 0.86 |
| GradientBoosting (GB) | 82.7 | 82.3 | 0.42 | 78 | 77.5 | 0.49 | 73.1 | 73.4 | -0.31 |
| SVC (SVM) | 89 | 88.8 | 0.25 | 88 | 87.3 | 0.68 | 87.8 | 87 | 0.76 |
| XGB | 92.2 | 91.2 | 0.12 | 91 | 90.1 | 0.93 | 89.8 | 89.6 | 0.25 |
| LGBM | 91.4 | 92.1 | 0.23 | 89.2 | 88.2 | 0.91 | 85.8 | 85.4 | 0.35 |
| DNN (MLP) | 91.7 | 91.2 | -0.02 | 90.3 | 90.1 | 0.19 | 90.8 | 90.2 | 0.66 |
| DNN (TabNet) | 90.9 | 89.6 | 1.26 | 89.3 | 88.3 | 0.94 | 88.5 | 87.4 | 1.09 |
| **AVG** | **0.444%** | | | **0.862%** | | | **0.459%** | | |

Table G: POS tagging experimental results based on our non-diacritized (O\_ND) and non-diacritized Facebook (F\_ND) model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 10-fold** | | |
| **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** |
| LogisticRegression (LR) | 84.5 | 84.7 | -0,2 | 84.9 | 84.6 | 0,28 |
| Kneighbors (KNN) | 87.2 | 85.7 | 1,5 | 87.4 | 86.3 | 1,05 |
| DecisionTree (DT) | 82.2 | 81.1 | 1,1 | 80.9 | 82.5 | -1,67 |
| RandomForest (RF) | 88.1 | 87.1 | 1 | 88.5 | 87.8 | 0,66 |
| XGBoost (XGB) | 88.6 | 87.2 | 1,4 | 88.1 | 87.1 | 0,98 |
| SVC (SVM) | 88.6 | 86.9 | 1,7 | 89.0 | 87.9 | 1,07 |
| LGBM | 76.7 | 79.4 | -2,7 | 78.5 | 79.2 | -0,72 |
| DNN (MLP) | 84.9 | 84.8 | 0,1 | 85.7 | 81.5 | 4,24 |
| DNN (TabNet) | 82.0 | 82.0 | 0 | 82.7 | 82.9 | -0,26 |
| **AVG** | **0.433%** | | | **0.626%** | | |

Table D: Fake Content Detection experimental results based on our non-diacritized (O\_ND) and non-diacritized Facebook (F\_ND) model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Train 70% - Test 30%** | | | **Cross Validation 5-fold** | | |
| **O\_ND%** | **F\_ND%** | **DIFF%** | **O\_ND%** | **F\_ND%** | **DIFF%** |
| LogisticRegression (LR) | 85.0 | 86.8 | -1,8 | 86.3 | 88.4 | -2,10 |
| Kneighbors (KNN) | 80.7 | 79.5 | 1,2 | 80.3 | 79.1 | 1,16 |
| DecisionTree (DT) | 77.6 | 77.2 | 0,4 | 76.1 | 79.4 | -3,24 |
| RandomForest (RF) | 84.4 | 87.3 | -2,9 | 85.5 | 87.7 | -2,20 |
| GradientBoosting (GB) | 86.7 | 88.6 | -1,9 | 87.2 | 88.0 | -0,76 |
| SVC (SVM) | 89.6 | 90.2 | -0,6 | 90.7 | 92.0 | -1,36 |
| XGBoost (XGB) | 85.9 | 87.2 | -1,3 | 87.1 | 88.5 | -1,38 |
| LGBM | 86.8 | 87.8 | -1 | 87.2 | 88.2 | -1,00 |
| DNN (MLP) | 90.1 | 88.9 | 1,2 | 90.6 | 90.1 | 0,50 |
| DNN (TabNet) | 83.3 | 83.7 | -0,4 | 88.6 | 89.7 | -1,08 |
| **AVG** | **-0.710%** | | | **-1.146%** | | |

1. Corresponding author [↑](#footnote-ref-1)
2. https://icrodev.com/icrosim/ [↑](#footnote-ref-2)
3. https://www.rslantext.com/ [↑](#footnote-ref-3)
4. https://fasttext.cc/ [↑](#footnote-ref-4)
5. https://nlp.stanford.edu/projects/glove/ [↑](#footnote-ref-5)
6. https://cbail.github.io/textasdata/word2vec/rmarkdown/word2vec.html [↑](#footnote-ref-6)
7. http://alfonseca.org/eng/research/wordsim353.html [↑](#footnote-ref-7)
8. https://fh295.github.io/simlex.html [↑](#footnote-ref-8)
9. https://github.com/icroob/MSAvsCA/ [↑](#footnote-ref-9)
10. http://www.abuelkhair.net/index.php/en/arabic/abu-el-khair-corpus [↑](#footnote-ref-10)
11. https://fasttext.cc/docs/en/crawl-vectors.html [↑](#footnote-ref-11)