Disambiguating Arabic Words According to Their Historical Appearance in the Document Based on Recurrent Neural Networks

RIM LAATAR, CHAFIK ALOULOU, and LAMIA HADRICH BELGUITH, MIRACL Laboratory-University of Sfax, Tunisia

How can we determine the semantic meaning of a word in relation to its context of appearance? We eventually have to grabble with this difficult question, as one of the paramount problems of Natural Language Processing (NLP). In other words, this issue is commonly defined as Word Sense Disambiguation (WSD). The latter is one of the crucial difficulties within the NLP field. In this respect, word vectors extracted from a neural network model have been successfully applied for resolving the WSD problem. Accordingly, this article presents an unprecedented method to disambiguate Arabic words according to both their contextual appearance in a source text and the era in which they emerged. In fact, in the few previous decades, many researchers have been grabbling with Arabic Word Sense Disambiguation.

It should be noted that the Arabic language can be divided into three major historical periods: old Arabic, middle-age Arabic, and contemporary Arabic. Actually, contemporary Arabic has proved to be the greatest concern of many researchers. The main gist of our work is to disambiguate Arabic words according to the historical period in which they appeared. To perform such a task, we suggest a method that deploys contextualized word embeddings to better gather valid syntactic and semantic information of the same word by taking into account its contextual uses. The preponderant thing is to convert both the senses and the contextual uses of an ambiguous item to vectors, then determine which of the possible conceptual meanings of the target word is closer to the given context.

CCS Concepts: • **Computing methodologies** → **Lexical semantics**;

Additional Key Words and Phrases: Natural language processing, historical dictionary, contemporary arabic, old arabic, middle-age arabic, word sense disambiguation, contextualized word embeddings, recurrent neural networks

ACM Reference format:
Rim Laatar, Chafik Aloulou, and Lamia Hadrich Belguith. 2020. Disambiguating Arabic Words According to Their Historical Appearance in the Document Based on Recurrent Neural Networks. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 19, 6, Article 86 (October 2020), 16 pages.
https://doi.org/10.1145/3410569

Authors' addresses: R. Laatar, C. Aloulou, and L. H. Belguith, MIRACL Laboratory-University of Sfax, Tunisia; emails: rimlaatar@yahoo.fr, Chafik.aloulou@fsegs.rnu.tn, lamia.belguith@gmail.com.

2375-4699/2020/10-ART86 $15.00
https://doi.org/10.1145/3410569

1 INTRODUCTION

The human language is subjected to several factors and influences that lead to its development and to the evolution of its vocabulary and grammar. It may also provoke its erosion and fragmentation, and at times, its extinction.

In fact, the evolution of the Arabic language from antiquity to the present day has given birth to several linguistic registers during the major historical periods of the Arabic language. Indeed, Arabic went through different experiences. It has also been marked by a lot of foreign influences that generated the evolution of its words. The historical events and political circumstances that humanity has experienced have also had a profound impact on the development of its lexis.

According to Reference [5], the Arabic language can be classified into three periods: old Arabic, middle-age Arabic, and contemporary Arabic. In fact, the old Arabic language extends from 480 BC to 200 AD. It includes the Arabic of the three ages (Jahili, Islamic, and Umayyad) and a part of the Abbasid period. Middle-age Arabic extends from the beginning of the blogging era to the pre-Renaissance era (in Hovef 1804) and includes the majority of Arabs of the Abbasid period and the subsequent era of States and Small States. The contemporary Arabic language stretches from the beginning of the Renaissance under the reign of Muhammad Ali Pasha to the present day and comprises modern Arabic and basically contemporary Arabic language.

This classification is supported by the majority of Arab linguists, but this does not prevent some linguists from adopting a more detailed classification giving birth to five periods. We adopt the classification in terms of three periods in our research and in the rest of this article.

Therefore, like any other Semitic language, Arabic seems to evolve and change in terms of its phonology, syntax, and especially its semantics.

In light of this, it is obvious that the senses of words are not fixed. In fact, they are constantly changing and evolving due to time and events. So, to safeguard their languages, nations have resorted to construct historical dictionaries. According to Reference [5], a historical dictionary is a general dictionary of language that draws its importance from the human heritage gathered from sciences, arts and letters from different ages and places. It analyses the evolution of lexical meanings and forms throughout the chronological stages a language may undergo. The historical dictionaries of a language are thus considered "the language body," which helps to understand the entire human heritage.

Generally speaking, creating such a dictionary must be very demanding and may necessarily go through several stages. One of these stages is the extraction of the appropriate sense of a given word by taking into consideration its appearance in the source text. The problem of determining the appropriate meaning of a given word in its context of appearance is also called Word Sense Disambiguation (WSD). Multiple works have focused on disambiguating terms in modern Arabic, but none seems to be concerned with disambiguating Arabic words according to the historical period in which they occurred.

The main objective of this work is to disambiguate words appearing in old and middle-age Arabic to help linguists built a historical dictionary for Arabic.

Recently, the use of word vectors extracted from a neural network language model has contributed to a series of significant advances in NLP, especially in WSD. In fact, word embeddings are of major importance as they display certain algebraic relations and can, therefore, capture both syntactic and semantic data.

Although, classic pre-trained word embeddings, such as Word2Vec [26] and GloVe [30] have been proved to be a breakthrough of solving many NLP tasks, they obviously produced the same embedding for the same word in different contexts. To overcome this difficulty, a new embedding method has been put forward. It is known as contextualized word embeddings. The latter aims at generating different embeddings for the same word by taking into account its contextual usage.

Such word representation is proved to be highly efficient to tackle several major NLP tasks, like Named Entity Recognition [3], Sentiment Analysis [24], or Word Sense Induction [33].

Given the success of contextual word representations in generating multiple embeddings for each word based on its context and therefore addressing the problem of polysemous terms, we have investigated in this study the efficacy of contextual word embeddings for Arabic Word Sense Disambiguation. We also trained different neural embedding models by using Flair architecture [2] to be able to identify the meaning of an ambiguous word with reference to the context of its occurrence in a particular text. The rationale behind this methodology is to embed the contextual uses and meanings of the ambiguous word, taken out from Arabic dictionaries, then calculate the similarity between their vector representations.

The major purpose of our method is not only to identify what a word means in a given context but also to disambiguate it according to the era in which it occurred. Therefore, the main contributions of this article are as follows:

- Train a neural language model for Arabic language on the Historical Arabic Dictionary Corpus (HADC) [6] based on the Flair embeddings technique [2],
- Discuss the role of different training parameters of the neural network in WSD performance,
- Put forward a method, which helps to automatically extract the meaning of a given term that occurred at a particular time to disambiguate terms that appear not only in modern standard Arabic but also in old and middle-age Arabic,
- Help linguists create the historical dictionary of Arabic by proposing a method that enables us to disambiguate terms according to the epoch at which they existed.

The rest of the article is structured as follows: in Section 2, we scratchily present a survey on word representation methods that have been widely proposed. We also give an overview of the related works, which focused on Arabic Word Sense Disambiguation. In Section 3, we deal with the dataset used in our experiments, and we present our proposed method for Arabic Word Sense Disambiguation. Section 4 presents our experiments and results. We finally draw a conclusion and future work directions in Section 5.

2 RELATED WORK

The progress that has been achieved in using neural networks to learn word embeddings has shown outstanding performances in a wide variety of NLP applications, such as named entity recognition [14, 36, 39], part-of-speech (PoS) tagging [32], information retrieval [17], and even Word Sense Disambiguation [23, 43]. The fundamental concept of such deep learning techniques is to compute the distributed representations of words [10]. Currently, two major types of methods for word representations have been widely proposed: Classical word embeddings and Contextualized word embeddings [37].

2.1 Classical Word Embeddings

Most current word embeddings algorithms are based on the distributional hypothesis, which claims that terms with similar contexts tend to have similar meanings [18]. There are several existing techniques for constructing vector word representations.

The authors of Reference [13] have suggested a unified architecture for NLP with a deep neural network. They were the first that demonstrated the utility of pre-trained word embeddings. Their deep neural network architecture is jointly trained by making use of many NLP tasks, such as chunking, parts of speech tagging, and semantic role labeling. The architecture, introduced by the authors, forms the cornerstone of many contemporary approaches. The work also establishes word embeddings as a definitely effective tool for many NLP tasks.

Another form of word representation that has been introduced by Reference [28] is called hierarchical log-bilinear model embeddings. It is a probabilistic model that concatenates the embedding of the $n - 1$ words to predict the last word.

The authors of Reference [26] have put forward two approaches to build word representation in vector space: the skip gram and the Continuous Bag of Word (CBOW). They have succeeded in gaining great popularity in natural language processing. These models are based on a neural network architecture. The CBOW [26] architecture predicts a word given in its context while the Skip gram architecture uses a pivot word to predict its context [27].

The authors of Reference [30] have offered an unsupervised learning algorithm for obtaining vector representations for words, named Glove. To build a word-representation model, Glove considered the global statistics of word occurrences in a corpus. Although Word2Vec and GloVe are two successful word embedding algorithms, these methods have some limits. In fact, they failed to provide any vector representation for words that are not in the vocabulary.

Another word embedding method, dubbed FastText, has been put forward by Reference [11]. FastText takes into account the subword information. It represents each word as an n-gram of characters. So, after training the neural network, a vector representation is associated with each character n-gram and the average of these vectors gives the final representation of the word [11]. FastText works well with rare words. So, even if a word is not seen during the training, it can be properly represented.

Although the distributed representations of words have led to great performance improvements in many NLP applications, namely, Word Sense Disambiguation [16, 21, 29] or plagiarism detection [38, 41], they seem to generate one representation of the same word in different contexts. To resolve this problem, new embedding techniques have been recently proposed. They aim at representing words according to their context of use and therefore the same word will have different embeddings depending on its contexts [4].

2.2 Contextualized Word Embeddings

Contextualized word embeddings have recently improved performance in major NLP tasks [42]. This new embedding aims to assign different vectors to the same word based on a surrounding context [34]. In this concern, there are three popular methods: ELMo [31], Bert [15], and Flair [4].

Embeddings from Language Models (ELMo) is a new word representation method proposed by Reference [31]. It uses a bi-directional LSTM model to create word embeddings. Unlike classical embeddings, ELMo takes into consideration the entire sentence to assign the embeddings to each word. Thus, it helps generate more than one vector per word depending on the context of use. ELMo has been trained on One-Billion Word Benchmark (The corpus comprises approximately 0.8 billion words).

Reference [15] proposed deep contextualized word representations endeavor to generate a better word representation for NLP tasks. It is called Bidirectional Encoder Representations from Transformers (BERT). BERT is a method of pre-training language representations that makes use of the transformer and takes into consideration the previous and the next tokens to generate word representations. Two variants of pre-trained embeddings are known as base embeddings and large embeddings. BERT base model uses 12 layers of the transformer encoders and 768-dimensional hidden layers as opposed to BERT large, which is larger and utilizes 24 layers of the transformer encoders and 1,024 hidden size. BERT language model is pre-trained on both the Books Corpus (800M words) and English Wikipedia (2,500M words).

The authors of Reference [4] set forth a novel method for embedding words into real vectors called Flair embeddings. Flair produces embedding for any string of characters in a sentential context [4], that is why it is referred to contextual string embeddings. The latter are produced by using a neural character-level language modeling.

As a language modeling architecture, the authors use the LSTM variant of recurrent neural networks [2]. Then, to create contextualized word embeddings forward and backward, neural networks have been combined. In fact, contextual word embeddings are generated by concatenating the forward LM's hidden state for the word's last character and the backward LM's hidden state for the word's first character [42].

Whatever method is adopted, whether BERT, Elmo, or Flair embeddings, the main objective of these methods is to address the problem of polysemous words by capturing word semantics in different contexts. Thus, contextualized word representations permit to generate multiple embeddings for one word depending on its context. However, according to Reference [4] Flair outperforms the previous best methods on a range of NLP tasks.

2.3 Arabic Word Wense Disambiguation: State of the Art

To the best of our knowledge, there seems to be no work concerned with disambiguating words that appeared in both old and middle-age Arabic. In fact, all the works that focused on Arabic Word Sense Disambiguation are concerned with identifying the meanings of words in modern Arabic. In this section, we are going to review the existing works related to Word Sense Disambiguation in modern Arabic.

The work, suggested by Reference [12], considers the local and the global contexts as defined by the full text during the disambiguation process. They have represented local context, global context and each sense of the ambiguous item with the help of vectors. Then, the appropriate sense for the target term is defined by the closest semantic proximity to its local and global contexts.

The authors of Reference [7] relied on Arabic Wikipedia to extract the different meanings of the ambiguous word. They have applied Vector Space Model as a mathematical representation of the documents. The Vector Space Model serves to represent each texts, retrieved from Wikipedia, as a vector. After that, each text, represented with the help of vectors, is compared to the context of the word using cosine distance. The appropriate sense of an ambiguous word is measured based on the highest cosine similarity.

Another method was proposed by the author of Reference [25]. He has used genetic algorithms to solve Word Sense Disambiguation problems. They tested their approach by using a sample Arabic text, then they compared it with the Naive Bayes classifier [8].

The authors of Reference [44] have introduced an approach based on information retrieval measures. They have generated the contexts of use of each sense of an ambiguous word with its glosses. Then, the most probable sense is chosen by measuring the similarity between the different generated contexts and the current context of the ambiguous word.

The hybrid method recommended by Reference [45] combines unsupervised and knowledge-based methods. They have used a context matching algorithm that measures the similarity between the contexts of use corresponding to the glosses of the target word and the original sentence.

Through their study, the authors of Reference [20] tried to look into the possibility of using word embeddings to solve Word Sense Disambiguation problems. Their proposed method consists in measuring the semantic relation between the contextual uses of an ambiguous word and its meanings.

The most recent work, performed by the authors of Reference [9], sought to disambiguate Arabic words based on Arabic Wordnet and word embeddings. The objective was to represent each sense of the ambiguous word by a vector by using both word2vec [26] and Glove [30]. The system adopted by the authors of Reference [9] lists all the synsets, which represent the ambiguous term along with its similarity to the context. It also chooses the synset that has the

Table 1. Comparative Study of Some AWSD Approaches

Author	WSD method	Used resources	Testing data	Precision
[9]	Knowledge-based approach	—Arabic WordNet —Watan and Khaleej corpora	—10 ambiguous words —A collected corpus of 240 training samples	79%
[20]	Knowledge-based approach	—HADC corpus —Arabic Dictionary Allwassit	—100 ambiguous words —100 contexts of use for each ambiguous words	56.45%
[7]	Knowledge-based AWSD	—Arabic Wikipedia —Arabic Wordnet	—7 ambiguous words	–
[44]	Knowledge-based approach	—Arabic dictionary Alwassit —A collected corpus of 1500 Arabic texts	—50 ambiguous words —130 contexts of use for every word	73%
[45]	Hybrid AWSD	—Arabic dictionary Alwassit —A collected corpus of 1500 Arabic texts	—10 ambiguous words —130 contexts of use for every word	79%

maximum similarity among all synsets. Table 1 presents a comparative study in the field of Arabic Word Sense Disambiguation. This comparison is performed with respect to these criteria:

- The used method for WSD,
- The resources used to WSD,
- The testing data (the number of ambiguous words that are used in the corpus),
- The rate of precision.

Thanks to this study, we can clearly perceive that most of the works employed a knowledge-based approach for AWSD, since this particular technique is characterized by a higher precision than the unsupervised approach. Besides, all the previous methods that focused on Arabic Word Sense Disambiguation were just concerned with identifying the meaning of terms in modern Arabic. However, there seems to be no work concerned with disambiguating Arabic terms according to the distinct historical period in which they occurred. Hence, the idea of disambiguating old and middle-age Arabic items to build a historical dictionary is by no means original.

3 ARABIC WORD SENSE DISAMBIGUATION APPROACH

We propose here a method that seeks to determine the meaning of an ambiguous word according to its historical epoch.

This method makes use of contextualized embeddings, more specifically, Flair embeddings [4]. Flair is a model that uses contextual string embeddings to estimate a good distribution $P(X_{0:T})$ over sequences of characters $(X_0, X_1, \dots, X_T) = X_{0:T}$. Hence, by training a language model, $P(X_t|X_0, \dots, X_{t-1})$, an estimate of the predictive distribution over the next character given past characters, is learned. The joint distribution over entire sentences can then be decomposed as a product of the predictive distribution over characters conditioned on the preceding characters:

$$P(X_{0:T}) = \prod_{t=0}^{T} P(X_t|X_{0:t-1}).$$

Two major reasons lay behind the choice of Flair that aims at Embedding words into vectors. First, Flair outperforms the previous best methods on a range of NLP tasks [4]. It is characterized by its ability to model words and context fundamentally as sequences of characters, to both

Table 2. Main Characteristics of the HADC Corpus

Historical period	Number of text
Pre-Islamic era	100
Islamic era	101
Abbasid era	383
Middle era	147
Modern era	138
Total	869

better handle rare and misspelled words as well as model subword structures such as prefixes and endings [4]. Second, FLAIR is an easy-to-use framework [2]. It facilitates training the model and producing vector representations of documents. It also offers multiple interesting methods to produce sentences and document embeddings. Moreover, Flair supports a growing list of embeddings such as Elmo [31] and Bert [15]. Furthermore, it can mix several different types of embeddings to build a powerful word representation. In particular, it combines both traditional embeddings such as Glove [30] together with contextual string embeddings.

In our study, we employed a method that focuses on both corpus and knowledge to disambiguate Arabic words. This method makes use of Arabic dictionaries and contextualized embeddings. Three major reasons stand for the choice of this method. First, Arabic lacks sense annotated corpora for training, because they are very expensive. So, supervised methods are rarely used to disambiguate Arabic words. Second, contextualized embeddings and more precisely Flair embeddings [2] are proved to be very helpful for WSD due to their ability to capture semantic relatedness that occurs between words, and their suitability to provide different embeddings for polysemous words depending on their context of use. Moreover, creating contextualized representation of English words encourages us to perform a similar work in Arabic for further validation.

Our proposed method consists in building a recurrent neural network model to calculate a distributed representation of both the context of use of the ambiguous term and its corresponding definitions. Alongside, we applied the cosine similarity distance to determine which of the possible word senses for the target item is closer to the context representation. Our experiments depend on two Arabic resources, namely, a large text corpus to train the model, and a number of Arabic dictionaries that contain words and senses to get the possible meanings of the target word.

3.1 Arabic Resources

- **Arabic corpus:** To build a Flair Embedding model for Arabic, we have used the Historical Arabic Dictionary Corpus (HADC) [6], which is originally designed to build a historical dictionary. It includes texts in old, middle-age, and contemporary Arabic with more than 116 million words. We have also added to this corpus about 200 texts extracted from Arabic Wiki Source. The list includes many text types, such as poetry, the Quran, Hadiths, literary prose, history and genealogy, religions and doctrine texts, encyclopedias and dictionaries, newspaper articles, geography, and travel literature. The main characteristics of the HADC corpus are shown in Table 2.
- **Arabic dictionaries:** To infer the meanings of ambiguous words, we have used three Arabic dictionaries that describe the different historical periods of the Arabic language:
  - —For Old Arabic Dictionaries, we utilized Tahdhib Allougha Dictionary[1] by Abou Mansour Azhari;

[1] AlAzhari, Abu Mansour, Refining the Language. Dar Alamaarif, Cairo, 1976.

- Concerning Middle-age Arabic Dictionaries, we adopted Tej-Alarous Dictionary[2] by Murtadha Zbidi;
- For contemporary Arabic Dictionaries, we made use of Contemporary Arabic Language Dictionary.[3]

So, one of the most important parts of our method consists in building the intended lexical dictionary. In fact, for old Arabic, we have used Tahdhib Alougha[1] Dictionary. Moreover, we have semi-automatically developed a structured electronic dictionary with an XML format containing the glosses of 100 ambiguous old Arabic words. Likewise, we have developed a dictionary that contains the glosses of 100 ambiguous words extracted from Tej-Alarouss.[2] Still, the last two dictionaries, Tahdhib Alougha and Tej-Alarous, are manually structured, because they have complex structures that vary from one entry to another and they are characterized by a quasi-absence of markers. For words in modern Arabic, we have relied on Contemporary Arabic Language[3] dictionary. Indeed, we have an HTML version of this dictionary. The latter is distinguished by a set of markers facilitating the transformation of its raw content to a structured version in XML. Then, we automatically converted it to a structured electronic XML format.

3.2 Proposed Method

We preliminarily apply some preprocessing steps (remove punctuation and non-Arabic words) for the texts of our corpus. Then, we calculate the embeddings of both the different senses of the target word and the context in which it appears. Subsequently, a cosine distance is used to determine the closest definition to the context representation. These aforementioned steps are illustrated in Figure 1. Afterwards, we describe piecemeal each step cited above.

- **Step 1: Pretrained Language Models.** In this step, our objective is to train our own contextualized word embeddings based on the Flair framework [2]. Thereafter, we are going to use an LSTM with 1,024 hidden states and one layer.
- **Step 2: Extract the definitions of the ambiguous word.** To extract the senses of an ambiguous word, we relied on our selected resources by studying the existing resources of the Arabic language that allowed us to use three Arabic dictionaries. The latter describe the different historical periods of the Arabic language, namely, the Tahdhib Allougha[1] dictionary, the Tej-Alarous[2] dictionary, and the Dictionary of contemporary Arabic.[3]

  Thus, our primary objective is to extract the various meanings of an ambiguous word using an appropriate dictionary by taking into account the historical period in which the word appeared in the document. Indeed, the title of each document in the HADC corpus [6] is recorded as follows: Author's name—Date of death. Hence, we can extract the historical period in which the ambiguous word appeared from the title of the document. In fact, the date of the author's death is crucial as it highlights the period when that meaning was used. It represents the closest date to all person's works. We chose the date of death of the authors, because the date of birth is not available for all the authors and especially the older ones. Therefore, from the title of the written text, we can extract the date of the author's death, and then we are able to infer the historical period in which the ambiguous word emerged. The choice of an appropriate dictionary for extracting the senses of such terms is therefore based on this historical information.

[2] Zabidi, Sayed Mortadha, Tej-Alarous, Kuwait Government Press and the National Council for Culture and Arts, Kuwait from 1965 to 2002.

[3] Mokhtar, Omar Ahmed, Modern Arabic Language, The Universe of Books, Cairo, 2008.

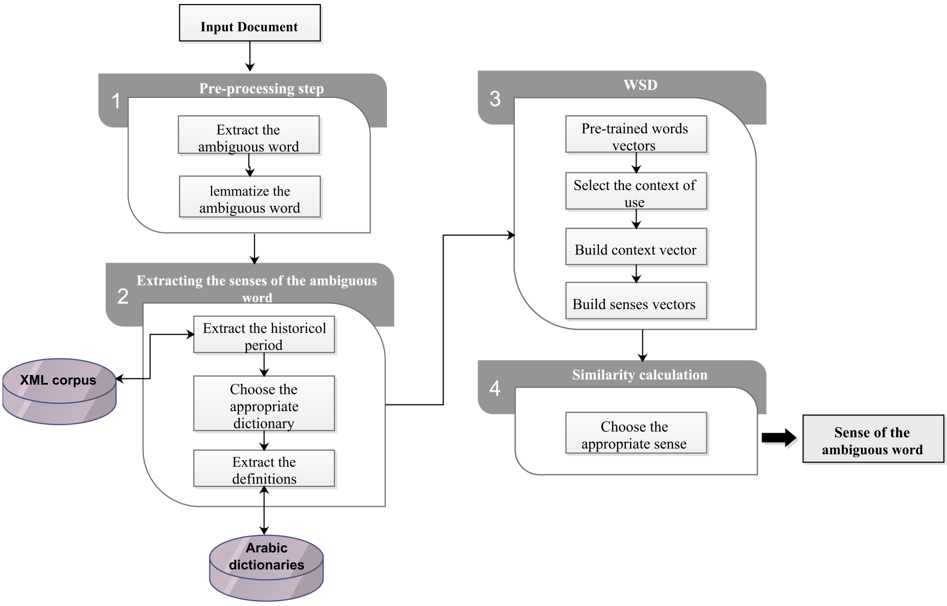
Input Document
1
Pre-processing step
Extract the ambiguous word
lemmatize the ambiguous word
2
Extracting the senses of the ambiguous word
Extract the historicol period
Choose the appropriate dictionary
Extract the definitions
XML corpus
Arabic dictionaries
3
WSD
Pre-trained words vectors
Select the context of use
Build context vector
Build senses vectors
4
Similarity calculation
Choose the appropriate sense
Sense of the ambiguous word

Fig. 1. The different steps of the proposed method for disambiguating Arabic words according to their contextual appearance in a source text and the era in which they emerged.

- **Step 3: Context and sense embeddings.** The goal behind this, is to calculate a distributed representation of both the context of use of the ambiguous word and its different glosses. To attain this purpose, we have chosen an approach by relying on contextualized distributed term representations [4] to represent the context and the different senses of the ambiguous word.

  The context vector of an ambiguous word can be accomplished by computing the mean of all words' vectors surrounding the target word. Then, after obtaining all sense definitions of the word to be disambiguated based on the last step, we will represent each sense as a vector by calculating mean vectors that represent the meanings of the ambiguous word.
- **Step 4: Similarity using cosine distance.** To attribute for each ambiguous word its appropriate sense, we chose the meaning with the closest semantic similarity to the context representation. The context vectors can then be compared to the possible word sense vectors for the abstruse item. To measure the similarity between context vectors and sense vectors, we used a cosine distance metric. In fact, the similarity measure between two vectors, $V = (v_1, v_2, \ldots, v_n)$ and $W = (w_1, w_2, \ldots, w_n)$, can be calculated by the cosine distance metric that is defined as follows [40]:

$$\cos(V, W) = \frac{\sum_{i=1}^{n} v_i \cdot w_i}{\sum_{i=1}^{n} v_i^2 \cdot \sum_{i=1}^{n} v_i^2}.$$

3.3 Arabic Word Sense Disambiguation Algorithm

To put it in a nutshell, we have developed an algorithm[1] that makes us able to disambiguate Arabic words according to their context of appearance in the document. The different steps of this algorithm are:

- Extract a document from the HDAC corpus,
- Select a word,
- Test if the selected word is ambiguous: the algorithm uses the function *IsAmbiguous* to determine if the selected word is ambiguous,
- Extract the historical period: the algorithm uses the function *ExtractHistoricalPeriod* to search the date of appearance of the extracted document and select the appropriate dictionary that contains the glosses of the ambiguous word according to its historical epoch,
- Lemmatize the ambiguous word to be able to extract its definitions. In this concern, Farasa tool [1] is used,
- Extract the senses of the ambiguous word,
- Build context and senses vectors,
- Choose the appropriate correct sense with the highest score.

ALGORITHM 1: Arabic Word Sense Disambiguation Algorithm

```
Input: $d$: document extracted from HADC
Input: $AW$: selected a word from $d$
Output: $sense$: appropriate sense of $AW$ according to its context of appearance in $d$
if IsAmbiguous($AW$) == True then
    Extract the title of the document $d$;
    $DateOfDeath \leftarrow$ date of the author's death;
    ExtractHistoricalPeriod($DateOfDeath$);
    Lemmatize($AW$);
    $glosses \leftarrow$ list of the definitions of $AW$;
    $C \leftarrow$ contextual window of $AW$;
    $V(C) \leftarrow$ context vector;
    $G \leftarrow$ list of glosses vectors;
    foreach $v \in G$ do
        $similarity = cosine(v, V(C))$;
    end
    Choose the sense with the highest score as the correct sense;
Function IsAmbiguous($W$):
    $senses \leftarrow$ number of the glosses of $w$;
    if $senses > 1$ then
        return $True$;
    else
        return $False$;
    end
end
Function ExtractHistoricalPeriod($date$):
    if $date <= 815$ then
        $dic \longleftarrow DicTahdhib$;
    else if $815 < \mathtt{Date} <= 1804$ then
        $dic \longleftarrow DicTejAlarous$;
    else
        $dic \longleftarrow DicAlmouasera$;
    end
    return $dic$;
end
```

3.4 Illustrative Example

For instance, the table below reports some contexts of use of the word W = "القهوة"[4] extracted from documents that appeared in different historical periods.

[4] This word can have four meanings: coffee, milk, coffee shop, or wine. It should be noted that the meanings of the word change over time.

Table 3. Some Contexts of Use Extracted from the HADC Corpus

Document	Title	Context of use
1	1031-اَلكشكول (Al-Kachkoul)	يقولون لي قهوة البن هل تباح وتؤمن آفاتها (They ask me about coffee...whether it is permissible and its pests can be averted)
2	1240-عجائب الاثار في اَلتَرَاجم وَالاخبَار (Wondrous Traces in Transla-tions and News)	فيسألون عن القهوجي ويطلبونه ليفتح لهم القهوة ويوقد لهم النار (And they asked for the coffee shop owner to open the café for them and set a fire for them)

Lets S = يقولون لي قهوة البن هل تباح وتؤمن آفاتها (They ask me about coffee...if it is permissible and its pests can be averted). S is taken from the document named 1031- الكشكول.

- **Step 1:** We began by deriving the historical period in which the word W appeared. From the title of this document, we notice that the author died in 1031. Therefore, we can deduce that the current document was written in the middle-age era.
- **Step 2:** We afterwards extracted the glosses of the word W from the appropriate dictionary. In our case, the dictionary Tej-Alarous[2] describes the middle-age Arabic period. So, the senses of W, extracted from Tej-Alarous, are as follows:
  —$Sense_1$: شراب البن المغلى (coffee)
  —$Sense_2$: الخمر (wine)
  —$Sense_3$: اللّبن المحض (milk)
- **Step 3:** We represented the context S in which the word W occurs as a vector by calculating the mean of all word vectors surrounding the target word. $V(S) = \text{Mean}(v(\text{يقولون})) + v(\text{لي}) + v(\text{قهوة}) + v(\text{البن}) + v(\text{هل}) + v(\text{تباح}) + v(\text{وتؤمن}) + v(\text{آفاتها}))$
- **Step 4:** Sense embedding: We mainly calculate embeddings for each definition.
  —$V_1 = Mean(v(\text{شراب}) + v(\text{البن}) + v(\text{المغلى}))$
  —$V_2 = Mean(v(\text{الخمر}))$
  —$V_3 = Mean(v(\text{اللّبن}) + v(\text{المحض}))$
- **Step 5:** in this last stage, we calculate the similarity. Cosine similarity is adopted to calculate the distance between context and definitions.
  —$Similarity(S, Sense_1) = cosine(V(S), V_1) = 0.7114$
  —$Similarity(S, Sense_2) = cosine(V(S), V_2) = 0.2141$
  —$Similarity(S, Sense_3 = cosine(V(S), V_3) = 0.5655$
  Sense 1 is, therefore, the appropriate sense of the ambiguous word "القهوة" according to its context of appearance in the document "1031- الكشكول".

4 EVALUATION AND DISCUSSION

In the following section, we are going to describe the data used in our experiment, present the obtained results and discuss them.

4.1 Code and Data

Our test corpus comprises 183 texts belonging to different historical periods. These texts have been extracted from the Historical Arab Corpus HADC [6] and the Open Source Arabic Corpora (OSAC) corpus [35]. Indeed, the Historical Arab Corpus is divided into two main parts: one for learning and the other for testing. About 149 texts in old and middle-age Arabic were specified and used for the test. As for modern Arabic, along with the texts extracted from the HADC Corpus, we have extracted some texts from the OSAC corpus. The latter involves 22,429 text documents. OSAC Arabic corpus has been collected from various websites. The text documents belong to various categories, such as history, education, entertainment, and other domains.

We have tested about 100 ambiguous words. For each one, we have used AntConc[5] to randomly extract 100 contexts of use for each ambiguous word from each historical period. These contexts were manually annotated.

The algorithm was written in the Python programming language and was trained using Google Colab. The code for the neural network is available at https://github.com/zalandoresearch/flair/blob/master/resources/docs/TUTORIAL_7_TRAINING_A_MODEL.md.

4.2 Results and Discussion

To measure the performance of our method, we have used the precision metric. In our case, for any ambiguous word, the precision measures the number of contexts correctly annotated divided by the total number of the annotated contexts. During the evaluation process, we relatively consider the historical period in which an ambiguous term emerged.

The first part of this assessment concentrated on the impact of different parameters that were used to train the recurrent neural model on WSD performance. In this experiment, we are attempting to test the effect of the batch size, the sequence length and the number of epochs on the evaluation data. Thus, different parameters are tested and the achieved results are presented in Table 4. The results illustrated in this table show that the best architecture that reaches the best result is obtained with $sequence\_length = 10$, $mini\_batch\_size = 16$, and $max\_epochs = 20$.

Our results show that utilizing a small sequence-length will yield better and faster neural network models. In fact, using long input sequences can lead to slow learning and model skill noisy.

In addition, the outcomes indicate that the best performance has been consistently obtained for mini-batch sizes between 16 and 32. Hence, increasing the mini-batch sizes penalized the generalization ability of the obtained model. This engendered a degradation in the quality of the model. Thus, it is preferable to use small mini-batches to achieve the best performance.

Table 4 also displays the precision score for 10, 20, 30, and 50 epochs. The best results have been obtained with 20 epochs. Hence, the best model was obtained with a small number of epochs due to multiple reasons, such as the amount of training data is not so large and the model is thus trained for a small number of epochs.

The second part of this evaluation examines the impact of the variation of the window size around the target ambiguous word. As illustrated in Table 5, increasing the size of the context window has an undesirable impact on the disambiguation of the target words.

The third part diagnoses the effect of removing the stop words from the context of use of the ambiguous word before embedding it. As shown in Table 6, keeping the stop words and then, creating context vectors gives better outcomes. We think that removing stop words will increase the number of content words in the context containing the ambiguous word, and therefore decrease

[5] A freeware corpus analysis toolkit for concordancing and text analysis (https://www.laurenceanthony.net/software/antconc/).

Table 4. The Obtained Precision with Proposed Method Using Different Parameters

Parameters	Values	Precision		
		Old Arabic	Middle-age Arabic	Contemporary Arabic
sequence_length[a]	10	48.32	50.15	65.42
	20	47.05	49.68	58.67
	30	46.94	49.52	65.26
	50	46.36	49.61	64.27
	100	47.35	50.06	62.27
	150	46.91	49.36	62.66
	200	47.01	49.20	62.49
mini_batch_size[b]	8	47.52	50.34	65.34
	16	48.32	50.43	66.18
	24	47.05	49.52	65.42
	32	48.32	50.34	65.62
	50	47.68	50.04	64.73
	100	46.36	49.61	64.27
max_epochs[c]	10	47.52	50.15	65.34
	20	49.53	50.34	66.18
	30	48.32	49.52	65.42
	50	47.68	50.04	64.73

[a] Represents the precise number of timed steps that the RNN has unrolled.
[b] The number of training examples in one forward/backward pass.
[c] The number of forward and backward training cycles of a particular training set.

Table 5. The Precision Obtained by Varying the Size of the Context of Use of the Ambiguous Word

Number of words surrounding the target word	Old Arabic	Middle-age Arabic	Contemporary Arabic
3	49.53	50.43	66,18
5	49.39	48.91	65.68
7	45,50	48.05	64.22
9	45.29	46.44	63.07

Table 6. The Average Precision Obtained with Stop Words Removal from the Context of Use of the Ambiguous Word

Method	Old Arabic	Middle-age Arabic	Contemporary Arabic
With stop words removal	49.07	49.74	65.44
Without stop words removal	49.53	50.43	66.18

the chances of matching with the exact sense definition that is often described by the meanings of the words having the same semantic meaning of the ambiguous word.

Finally, we have compared our obtained results to the most recent works of Arabic Word Sense Disambiguation systems that were proposed in the literature. Table 7 delineates this comparison in details.

Table 7. Comparison with Others Methods

Method	Old Arabic	Middle-age Arabic	Contemporary Arabic
Our Work	49.53	50.43	66.18
Reference [22]	48.54	47.48	56.45

Recalling that we have based our comparison on the same test data, our method has given better results compared to that proposed by Reference [22]. The latter used the Skip Gram model[6] to resolve the WSD problem. This result can confirm the good choice and performance of using contextualized word embeddings in the Arabic Word Sense Disambiguation field.

From this comparison, we can assume that the innovation of our work, compared to other works of Arabic Word Sense Disambiguation, can be summarized as follows:

- Putting two and two together, this is a pioneering attempt that is concerned with Arabic Word Sense Disambiguation. We adopted a contextualized word embedding that generates different embeddings for the same term including its contextual usage.
- Another innovative aspect of this piece of research is that we put forward a strategy to disambiguate Arabic terms with reference to the historical period in which they turned out. Consequently, this is the first work with a special focus on disambiguating lexical terms not only from contemporary Arabic but also from old and middle-age Arabic.
- Unlike other studies, where researchers concentrated on modern Arabic that was just dealt with Arabic Wordnet, which affords limited access to items, we have focused in this work on disambiguating Arabic terms based on myriad Arabic dictionaries. We have also set up an electronic version of Contemporary Arabic Language Dictionary in an XML format that can be used to resolve several NLP problems.

5 CONCLUSION

This article presents a strategy for Arabic Word Sense Disambiguation based on recurrent neural networks. The proposed method serves to extract the sense of a given word that emerged in the document. More importantly, this technique focuses equally on disambiguating not only words in modern Arabic but also words that emerged in old and middle-age Arabic periods.

To extract the sense of a given term, we have used contextualized word embeddings by utilizing Flair embeddings [4]. The rationale behind this is to calculate a distributed representation of the different senses and of the context of use of the ambiguous word, then determine which of the possible senses is the closest to the context representation. This method reaches a precision of 66.18% for modern Arabic, 49.53% for old Arabic, and 50.43% for middle-age Arabic.

During our experimentation, we have noticed that some words have meanings that existed in the corpus rather than in the dictionary. As a future work, we will try to overcome this problem by relying on others lexical resources, like Arabic ontology [19]. Furthermore, we will intend to look into ELMO [31] and BERT [15] embeddings to solve the WSD problem.

REFERENCES

[1] Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'16).*

[6] Reference [27].

[2] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR : An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL'19).*

[3] Alan Akbik, Tanja Bergmann, and Rol Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL'19).*

[4] Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the International Conference on Computational Linguistics (COLING'18).*

[5] Almoataz B. Al-Said. 2015. The historical arabic dictionary resources. *J. Arab Lang.* 129 (2015).

[6] Almoataz B. Al-Said and Lucía Medea-García. 2014. The historical arabic dictionary corpus and its suitability for a grammaticalization approach. In *Proceedings of the 5th International Conference in Linguistics.*

[7] Marwah Alian, Arafat Awajan, and Akram Al-Kouz. 2016. Arabic word sense disambiguation using wikipedia. *Int. J. Comput. Info. Sci.* 12 (2016), 857–867.

[8] Marwah Alian, Arafat Awajan, and Akram Al-Kouz. 2017. Arabic word sense disambiguation—Survey. In *Proceedings of the International Conference on New Trends in Computing Sciences.*

[9] Ali Alkhatlan, Jugal Kalita, and Ahmed Alhaddad. 2018. Word sense disambiguation for arabic exploiting arabic wordnet and word embedding and word embedding. In *Proceedings of the 4th International Conference On Arabic Compitational Linguistics (ACLing'18).*

[10] Jiang Bian, Bin Gao, and Tie-Yan Liu. 2014. Knowledge-powered deep learning for word embedding. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECMLPKDD'14).*

[11] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Ling.* 5 (2017), 135–146.

[12] Nadia Bouhriz, Faouzia Benabbou, and El Habib Ben Lahmar. 2016. Word sense disambiguation approach for arabic text. *Int. J. Adv. Comput. Sci. and Appl.* 7, 4 (2016).

[13] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing:Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning.*

[14] Arjun Das, Debasis Ganguly, and Utpal Garain. 2017. Named entity recognition with word embeddings and wikipedia categories for a low-resource language. *ACM Trans. Asian Low-Res. Lang. Info. Process.* 16, 3 (2017).

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL'19).*

[16] O. Dongsuk, Sunjae Kwon, Kyungsun Kim, and Youngjoong Ko. 2018. WordSense disambiguation based on word similarity calculation using word vector representation from a knowledge-based graph. In *Proceedings of the 27th International Conference on Computational Linguistics.*

[17] Jibril Frej, Jean-Pierre Chevallet, and Didier Schwab. 2018. Enhancing translation language models with word embedding for information retrieval. *Comput. Res. Repos.* (2018), 1801.03844.

[18] Zellig S. Harris. 1954. Distributional structure. *Word* 10, 2–3 (1954).

[19] Mustafa Jarrar. 2018. The Arabic Ontology Basics. Retrieved from http://www.jarrar.info/courses/Jarrar.LectureNotes.ArabicOntology.pdf.

[20] Rim Laatar, Chafik Aloulou, and Lamia Hadrich-Belguith. 2018. Word sense disambiguation to create a historical dictionary for arabic language. In *Proceedings of the 8th International Conference on Computer Science and Information Technology (CSIT'18).*

[21] Rim Laatar, Chafik Aloulou, and Lamia Hadrich-Belguith. 2018. Word2vec for arabic word sense disambiguation. In *Proceedings of the International Conference on Natural Language & Information Systems (NLDB'18).*

[22] Rim Laatar, Chafik Aloulou, and Lamia Hadrich-Belguith. 2020. Towards a historical dictionary for arabic language. *Int. J. Speech Technol.* (2020). DOI: https://doi.org/10.1007/s10772-020-09704-z

[23] Minh Le, Marten Postma, Jacopo Urbani, and Piek Vossen. 2018. A deep dive into word sense disambiguation with LSTM. In *Proceedings of the International Conference on Computational Linguistics.*

[24] Yuncong Li, Cunxiang Yin, Ting Wei, Huiqiang Zhong, Jinchang Luo, Siqi Xu, and Xiaohui Wu. 2019. A joint model for aspect-category sentiment analysis with contextualized aspect embedding. *Comput. Res. Repos.* (2019), 1908.11017.

[25] Mohamed El Bachir Menai. 2014. Word sense disambiguation using evolutionary algorithms—Application to Arabic language. *Comput. Hum. Behav.* 41 (2014), 92–103.

[26] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations (ICLR'13).*

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Adgate Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems.*

[28] Andriy Mnih and Geoffrey E. Hinton. 2008. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems*. MIT Press.

[29] Korawit Orkphol and Wu Yang. 2019. Word sense disambiguation using cosine similarity collaborates with word2vec and wordnet. *Big Data Anal. Artific. Intell.* (2019). DOI: https://doi.org/10.3390/fi11050114

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'14).*

[31] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL'18).*

[32] Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the Association for Computational Linguistics (ACL'16).*

[33] Sanjana Ramprasad and James Maddox. 2019. CoKE : Word sense induction using contextualized knowledge embeddings. In *Proceedings of the Spring Symposium on Combining Machine Learning with Knowledge Engineering.*

[34] Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019. Classification and clustering of arguments with contextualized word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.*

[35] Motaz Saad and Wesam Ashour. 2010. OSAC: Open source arabic corpora. In *Proceedings of the International Conference on Electrical and Computer Systems.*

[36] Joaquim Santos, Juliano Terra, Bernardo Consoli, and Renata Vieira. 2019. Multidomain contextual embeddings for namedentity recognition. In *Proceedings of the Iberian Languages Evaluation Forum (IberLEF'19).*

[37] Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. SensEmBERT: Context-enhanced sense embeddings for multilingual word sense disambiguation. In *TProceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20).*

[38] Didier Schwab, Laurent Besacier, Jérémy Ferrero, and Frédéric Agnès. 2017. Using word embedding for cross-language plagiarism detection. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL'17).*

[39] Miran Seok, Hye-Jeong Song, Chan-Young Park, Jong-Dae Kim, and Yu-Seop Kim. 2016. Named entity recognition using word embedding as a feature. *Int. J. Softw. Eng. Appl.* (2016). DOI: 10.14257/IJSEIA.2016.10.2.08

[40] D. Shashavali, V. Vishwjeet, Rahul Kumar, Gaurav Mathur, Nikhil Nihal, Siddhartha Mukherjee, and Suresh Venkanagouda Patil. 2019. Sentence similarity techniques for short vs variable length text using word embeddings. *Comput. Sist.* 23, 3 (2019).

[41] Dima Suleiman, Arafat Awajan, and Nailah Al-Madi. 2017. Deep learning-based technique for plagiarism detection in arabic texts. In *Proceedings of the International Conference on New Trends in Computing Sciences (ICTCS'17).*

[42] Dongfang Xu, Egoitz Laparra, and Steven Bethard. 2019. Pre-trained contextualized character embeddings lead to major improvements in time normalization: A detailed analysis. In *Proceedings of the 8th Joint Conference on Lexical and Computational Semantics (SEM'19).*

[43] Dayu Yuan, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. 2016. Semi supervised word sense disambiguation with neural models. In *Proceedings of the International Conference on Computational Linguistics (COLING'16).*

[44] Anis Zouaghi, Laroussi Merhbene, and Mounir Zrigui. 2012. Combination of information retrieval methods with LESK algorithm for arabic word sense disambiguation. *Artific. Intell. Rev.* (2012). DOI: https://doi.org/10.1007/s10462-011-9249-3

[45] Anis Zouaghi, Laroussi Merhbene, and Mounir Zrigui. 2012. A hybrid approach for arabic word sense disambiguation. *Int. J. Comput. Process. Lang.* (2012). DOI: https://doi.org/10.1142/S1793840612400090

Received February 2020; revised May 2020; accepted July 2020