Ant Colony Algorithm for Arabic Word Sense Disambiguation through English lexical information
Bakhoucha Abdelaali, Yamina Tlili-Guiassa, Didier Schwab, Andon Tchechmedjiev

To cite this version:

HAL Id: hal-01684576
https://hal.archives-ouvertes.fr/hal-01684576
Submitted on 15 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Ant Colony Algorithm for Arabic Word Sense Disambiguation through English lexical information

Bakhouche Abdelaali*
Laboratory LRI / Team SRF
Univ. Badji Mokhtar, Annaba / Algeria
P.O. Box 12, 2300 Annaba, Algeria
E-mail: Bakhouche2006@yahoo.fr
*Corresponding author

Yamina Tlili-Guiassa
Laboratory LRI / Team SRF
Univ. Badji Mokhtar, Annaba / Algeria
P.O. Box 12, 2300 Annaba, Algeria
E-mail: guiyam@yahoo.fr

Didier Schwab
LIG (Laboratory of Informatics of Grenoble)
GETALP (Study Group for Machine Translation and Automated Processing of Languages and Speech)
Grenoble University, France
E-mail: Didier.Schwab@imag.fr

Andon Tchechmedjiev
LIG (Laboratory of Informatics of Grenoble)
GETALP (Study Group for Machine Translation and Automated Processing of Languages and Speech)
Grenoble University, France
E-mail: Andon.Tchechmedjiev@imag.fr

Abstract: The ability to identify the intended meanings of words in context is a central research topic in natural language. Many solutions exist for word sense disambiguation (WSD) in different languages, such as English or French, but research on Arabic WSD remains limited. The main bottleneck is the lack of resources. In this article, we show that it is possible to build a WSD system for the Arabic language thanks to the Arabic WordNet and its connections to the English Princeton WordNet. Given that the Arabic WordNet does not contain definitions for synsets, we construct a dictionary that maps the Princeton WordNet definitions to the Arabic WordNet. We also create an Arabic evaluation corpus and gold standard. We then exploit this dictionary and evaluation corpus to run and evaluate an adapted Ant Colony algorithm on Arabic text that can use the Lesk similarity measure thanks to definition mapping. The algorithm shows a performance of approximately 80% compared to the random baseline of 78.9%.

Keywords: Arabic language processing, Word Sense Disambiguation, Ant colony algorithm, Lesk algorithm, Local/Global algorithm, WordNet.

Biographical notes: Abdelaali Bakhouche received his engineering degree in 2001 from the Batna University, Algeria, and his MSc degree 2009 from University of Khemchela, Algeria. He is actually a lecturer at the same university, and prepares a PhD-thesis in natural language processing. He is a member of the Laboratory LRI, Annaba University. His research interest includes Word Sense Disambiguation, semantic Web, and ontology.

Tlili Yamina holds a doctorate degree in Computer Sciences in artificial intelligence and language processing. She is Lecturer in the Department of Computer Sciences at Badji Mokhtar University Annaba, Algeria, since 1985. She is head of project entitled ‘documents indexing and security’. She is an active researcher in texts, images and
opinions mining, with a focus on applications in language processing and artificial intelligence, she has a number of articles in internationals journals and conferences in these subjects.

Didier Schwab received his PhD in Informatics in 2005 from the University of Montpellier, France. He was scientific fellow at University Sains Malaysia, Penang in 2006-2007. He has been assistant professor then associate professor at University of Grenoble Alpes France since 2007. He is researcher at the Study Group for Languages and Speech of the Laboratory of Informatics of Grenoble. His research focuses on the representation, acquisition and exploitation of meaning including Word Sense Disambiguation.

Andon Tchechmedjiev is a PhD candidate in natural language processing at Univ. Grenoble-Alpes in the GETALP research group. His research interests include multilingual lexical resources, word sense disambiguation, computational lexical semantics and machine learning. He has a Master’s degree in Computer Science and A.I. from Univ. Grenoble Alpes and a BSc (Hons) First Class in Computer Science from Oxford Brookes University.

1. Introduction

Word sense disambiguation (WSD) is the process of disambiguating the sense of an ambiguous word when the word has more than one sense by using the neighbours of the word (context) in a sentence or text (Navigli, 2009; Özdemir, 2009). Word sense disambiguation is of interest for several linguistic applications such as automatic translation, multilingual information extraction, semantic indexing, and automatic summarization.

Arabic is the fourth most-widely spoken language in the world. It is a highly inflectional language, with a rich morphology and relatively free word order. Ambiguity in Arabic is present at several levels, such as internal word structure ambiguity, syntactic ambiguity, semantic ambiguity, constituent boundary ambiguity and anaphoric ambiguity. This prevalence of ambiguity has led to difficulty with the automatic processing of Arabic (Attia, 2008).

Ant colony optimization (ACO) was first introduced by Marco Dorigo (Dorigo et al., 1996; Dorigo et al., 1999; Dorigo and Blum, 2005). This algorithm is a fairly novel technique for solving computational problems by mimicking the natural behaviour of ants as they generate and select paths to a food source from a colony. The first application of ant colony optimization was to solve the Travelling Salesman Problem and other similar np-hard combinatorial optimization problems (asymmetric traveling salesman problem, the quadratic assignment problem, and the job–shop scheduling problem). In addition, ACO has been successfully applied to transportation planning problems (Yang et al., 2007; Vitins and Axhausen, 2009).

To disambiguate words in context, Schwab et al. have proposed a distinction between local and global algorithms: the similarity measure between word senses is the local measure, and the combinatorial optimization algorithm that finds the best sense combination is the global algorithm. They proposed a global ACO algorithm and compared it to Simulated Annealing and Genetic Algorithms. The ACO algorithm obtained better precision, recall and execution time (Schwab et al., 2011; Schwab et al., 2012; Schwab et al., 2013).

In this paper, we adapt the approach of (Schwab et al., 2013) to solve the WSD problem in the Arabic language (AWSD henceforth). Firstly, our approach extracts two sets of features, namely the set of words from the Arabic WordNet and the set of meanings from the English WordNet, from which we build the dictionary. Secondly, we present the ACO algorithm used as well as the variant of the Lesk similarity measure applied.

The rest of this paper is organized as follows: in section 2, we present the specificity of the Arabic language; in section 3, we briefly discuss some of the related work in the area of Arabic Word Sense Disambiguation; in section 4, we describe the resources and algorithms used; in section 5, we present the proposed method for AWSD; and finally, we present the results of tests of our model.

2. Specificity of the Arabic language

“Arabic is the official language of hundreds of millions of people in twenty Middle East and northern African countries, and is the religious language of all Muslims of various ethnicities around the world” (Elkateb et al., 2006). Every year, Arabic written content on the web is multiplied by 250; this is the fastest growth of written content for any language on the Internet (Russian is the second fastest at 182x). There are approximately 65 million Arabic speakers on the Internet, which is 18.8% of the global Internet population (Menai et al., 2012).

The literary Arabic language is called Modern Standard Arabic MSA (Robert, 2009). It is currently the only official form of Arabic and is used in most written documents as well as in formal spoken occasions, such as lectures and news broadcasts. It is a difficult language that could hinder the development of tools for the application of natural language processing. The Arabic language has many particularities such as short vowels, the absence of capital letters and complex morphology. The Arabic language is composed of nouns, verbs and particles, wherein these
are morphemes and derived from a closed set of approximately 10,000 roots. Arabic is also highly inflectional and derivational, which makes morphological, syntactic and semantic analysis a very complex task (Farghaly and Shaalan, 2009). There is no capitalization in Arabic, which makes it hard to identify proper names, acronyms, and abbreviations. Furthermore, the Arabic Language is highly ambiguous for several reasons.

- Vowelization is a very interesting characteristic of the Arabic language, which causes ambiguity when it is absent, and this is usually the case.
- Polysemy (when a word or phrase has several meanings) is the second problem that causes semantic ambiguity in Arabic.
- A rarity of resources such as an Arabic Corpora and pre-treatment text tools leads to a lot of time being wasted on the process of collecting and/or modelling and developing these resources.

All of these factors collectively will affect the availability of WSD applications for Arabic.

3. Previous Work

Several approaches that address the problem of Arabic word sense disambiguation have been presented in the past few years. Among the most cited is the work of Mona Diab, who used a supervised learning approach. This approach exploits translational correspondence between words in a parallel Arabic English corpus to annotate Arabic text. It exhibits an absolute precision of 56.9% on the AWSD task (Diab, 2004). Menai and Wodjan proposed an approach to solve the Arabic word sense disambiguation problem based on a genetic algorithm. In this algorithm, a text is tokenized, and stop-words are filtered and stemmed. In the pre-processing phase, tokenization splits the text into words. After tokenization, the authors removed stop words to filter out words that were not important in the text, such as prepositions and articles. Subsequently, the authors performed stemming (Khoja’s Stemmer) on the remaining tokens. In stemming, the prefixes and suffixes of the word are removed. The senses of each word were then retrieved from the Arabic WordNet (AWN).

AWSD (GA) is used to find the most appropriate mapping from words to senses retrieved from the AWN in the context T. The authors have shown that GA performs better than the Naïve Bayes algorithm. They obtained an accuracy of 78.9% (Menai et al., 2012; Menai, 2014a; Menai, 2014b).

The system presented by Elmougy, Hamza and Noman (Elmougy et al., 2008; Elmougy et al., 2012) to solve the ambiguity of non-diaccritic words in the Arabic language uses the routing algorithm with a naïve Bayes classifier. It achieved a rate of precision on the order of 76.6%.

To disambiguate Arabic words, Zouaghi, Merhben and Zerigui (Zouaghi et al., 2011) tested the Naïve Bayes algorithm, the decision lists and the exemplar-based algorithm on tagged samples and a large amount of data in the used corpus. For these tests, the exemplar-based algorithms achieve rates of precision of 48.23%, 43.86% and 52.02%, respectively.

Another method for word sense disambiguation was proposed by Zouaghi, Merhben and Zerigui (Merhben et al., 2009). Their method evaluated the variants of the Lesk algorithm and used the Arabic WordNet to perform word sense disambiguation in Arabic. The modified version of the Lesk algorithm achieved a precision of 67%.

The method presented by Nameh et al. (2011) is a supervised learning method for word sense disambiguation based on the Inner Product of vectors. The system extracts two sets of features: the set of words that have occurred frequently in the text and the set of words surrounding the ambiguous word. This approach achieves a precision of 77.1%.

Shah et al. (2010) proposed an approach for accurate prediction of Arabic text consisting of both lemmas and morphological analyses. The accuracy of their system was reported to be 90%.

In spite of the great attention the problem has received in the last years and the important developments achieved, the precision of state-of-the-art algorithms is far from being satisfactory.

4. Resources and algorithms used in our approach

In this section, we present the characteristics of the resources and algorithms used in our approach.

a. WordNet Database

A WordNet is a machine-readable lexical database that groups words into clusters of synonyms called synsets. Every synset can be thought of as representing a unique word sense (meaning or concept). A WordNet provides general definitions and examples for the synsets and includes semantic relations between them. The semantic relations allow for a WordNet to be interpreted hierarchically as a lexical ontology/taxonomy.

WordNet is like a dictionary in that it stores words and meanings. However, it differs from traditional dictionaries in many ways. For instance, words in WordNet are arranged semantically instead of alphabetically. Synonymous words are grouped together to form synonym sets, or synsets. Each such synset therefore represents a single distinct sense or concept (Miller et al., 1990; Miller 1995).

The Arabic WordNet (AWN) is a lexical database of the Arabic language following the development process of the Princeton English WordNet and the Euro WordNet. It utilizes the Suggested Upper Merged Ontology as an Interlingua to link the Arabic WordNet to previously developed WordNet1 (Elkateb et al., 2006). The Arabic WordNet consists of 9228 synsets (6252 nominal, 2260 verbal, 606 adjectival, and 106 adverbial), containing 18,957 Arabic expressions. This number includes 1155 synsets that correspond to Named Entities, which have been extracted automatically and are being checked by lexicographers (Rodriguez et al., 2008; Elberrichi et al., 2012). These numbers are constantly changing. At the time of writing, the English WordNet consisted of 155287 synsets (117798 nominal, 11529 verbal, 21479 adjectival, and 4481 adverbial).

1http://www.globalwordnet.org/AWN
b. LESK algorithm

The Lesk algorithm is a disambiguation method based on counting the overlaps between the definitions of sense of a target word (supplied by a machine readable dictionary), and the definitions of words in the context (Lesk, 1986; Banerjee, 2002). The selected sense for this word corresponds to the definition of the sense containing the maximal number of overlaps with the context. The algorithm begins anew for each word and does not utilize the senses it previously assigned. Formally, given two words w₁ and w₂, the following score is computed for each pair of word senses: 

$$\text{Score}(S₁, S₂) = |\text{gloss}(S₁) \cap \text{gloss}(S₂)|,$$

where gloss(S₁) is the bag of words in the textual definition of sense S₁ of w₁. The senses that maximize the above formula are assigned to the respective words. However, this requires the calculation of $$|\text{Senses}(w₁)| \cdot |\text{Senses}(w₂)| \cdot |\text{gloss overlaps}.$$ The original Lesk algorithm relies on glosses found in traditional dictionaries such as the Oxford Advanced Learner’s Dictionary of Current English, Collins English Dictionary and the Oxford English Dictionary (Banerjee, 2002). Another version of the Lesk algorithm takes advantage of the highly inter-connected set of relations among synonyms that WordNet offers.

c. Ant colony algorithm (ACO)

Ants are social insects that live in colonies that may include thousands of individuals. When the ants find food, each individual ant does not know where to go. First, ants randomly chose their path. However, their path is chosen depending on the pheromone trails that have been laid on the ground by previous ants. The trail of pheromones on the shortest path is increased faster than the longest path. In the end, all ants choose the shortest path. Three factors that allow the ants to find the shortest path are as follows:

- Ants follow pheromone trails;
- Pheromone trails degrade over time;
- Short paths take less time to traverse.

The idea of the ant colony algorithm is to mimic this behaviour with "simulated ants" walking around the graph representing the problem to be solved. The first application of ant colony optimization was to solve the Travelling Salesman Problem (Dorigo et al., 1996; Dorigo et al., 1999; Dorigo and Blum, 2005). In this well-known problem, the goal is to find, for finitely many towns whose pairwise distances are known, the shortest path connecting them. Several improvement algorithms have been used with ACO to solve such problems as the asymmetric traveling salesman problem, the quadratic assignment problem, and the job-shop scheduling problem. In addition, ACO has been applied successfully for transportation planning problems (Yang et al., 2007; Vitins and Axhausen, 2009).

d. Local/Global algorithm

Several word sense disambiguation systems are based on the notion of local and global algorithms (Ratinov et al., 2011; Schwab et al., 2013b). A local algorithm allows the calculation of the semantic similarity between two lexical objects (sense, words, constituents, etc.) in the textual segment. Global algorithms propagate local measures at the upper level. In other words, this technique permits a global extension of a local algorithm based on semantic relatedness to an entire text. We use this technique in our approach: the LESK algorithm is the local algorithm, and the ant colony algorithm is the global algorithm.

5. Generation of an Arabic dictionary

External knowledge is a fundamental component of WSD. In Arabic, there are no available dictionaries in these forms. However, there has been much work done on bilingual dictionaries based on WordNet architecture (Black et al., 2006). Therefore, a bilingual dictionary has been implemented successfully for common communication words using WordNet. This source provides data that are essential to associate English senses with Arabic words (see Fig. 1). However, it has difficulty providing the Arabic equivalent word with certainty to that of an English word.

Disambiguation of ambiguous Arabic words

The proposed work consists of developing a method used to resolve semantic ambiguity for the Arabic language using the Ant colony algorithm (see Fig. 2). In this section, we explain the most important points of this approach.

Construction of the text graph

The environment consists of a graph with ants. It is simply organized according to the elements of the text (text, sentence, word) without external language information to better understand the mechanics of the algorithms.

In our graph, we distinguish two types of nodes. Nests produce ants that move on the graph in order to find energy and bring it back to their nest. The more energy brought back by the ants, the more ants the nest can in turn produce.
Types of nodes
Nests and plain nodes are two distinct types of nodes in this graph. Following Schwab et al. (2012), each possible word sense is associated with a nest. From the point of view of an ant, a node can be any of the following:
- The ant’s nest (anthill), where it was born;
- An enemy nest, which corresponds to another sense of the same word;
- A friendly nest: all of those nests that are not enemies;
- A plain node: any node that is not a nest.

Vector of definition
The odour of a nest is the numerical sense vector and corresponds to the definition of the sense associated with the nest (see Fig. 3). All ants born in the same nest have the same odour vector.

<table>
<thead>
<tr>
<th>Vector of sense’s definition ‘wash’</th>
</tr>
</thead>
<tbody>
<tr>
<td>101 101 101 101 101 101 101 101 101</td>
</tr>
</tbody>
</table>

Movement of ants
An ant is born (produced by its nest) and goes in search of energy. It is drawn by nodes that carry a lot of energy and avoids arcs with too much pheromone to enable exploration of more solutions.

Bridge
When an ant arrives on a node adjacent to a potential friend nest (i.e., that corresponds to a sense of a word), it has to decide between taking any of the possible paths or going to that nest node. The only difference is that if the ant chooses to go on the potential friend nest, a bridge between that nest and the ant’s home nest is built and the ant follows it to go home. Bridges behave like regular edges, except that if the concentration of pheromone on them reaches a certain level, they collapse and are removed.
Interpretative paths
For all intents and purposes, bridges behave exactly like ordinary paths, however, when the concentration of pheromone on a bridge reaches zero, the bridge "evaporates" and ceases to exist. When many bridges have been created, some will be strengthened (when many ants follow them) and other will disappear. This will lead to a monopolization of energy by the anthills linked by the strongest bridges. From the point of view of linguistics, our bridge construction model corresponds to emergent interpretative paths. At the end of the simulation, for each word node, the anthill with the highest energy level is chosen as the solution.

The choice of an answer
The selected answers correspond to words in the nest node with the highest energy value. The reason for this choice over using the pheromone concentration is that empirically, the energy level correlates better with the actual F1 scores. In turn, the global LESK score of a selected sense combination has an even better correlation with the F1 score. This is why we retain the sense combinations resulting from each iteration of the algorithm (highest energy nests at each iteration) and select the one with the highest global LESK score as the final solution.

Illustrated example
The easiest way to understand how the ant colony works is by means of an example; we illustrate the steps of the algorithm through the path taken by an ant on the phrase: "poet said rhyme".

We follow the path of an ant that was born in the anthill بيت/ Rhyme - corresponding to the second meaning for the word بيت.

**Step 1:** The ant begins to search for food to take energy to its nest. It begins to explore the graph starting with the node بيت.

**Step 2:** On the path to this node, it deposits its pheromone, and arriving, it takes energy. It also deposits some of its odour on the node. This odour is marked by the deposition of two components taken at random from its mother nest.

**Step 3:** The ant then chooses to follow another path where it deposits its pheromone. It will also deposit two vector components of the mother nest at random and take energy. At this stage, the ant has a full quantity of energy, and a pseudo-random decision will make it return to the mother nest node.

**Step 4:** As an ant cannot go on a node from which it came, unless that is its only chance, our ant has the choice between going on to قال or الشاعر.

**Step 5:** Ants coming from الشاعر/ Poet part of their vector on الشاعر have more in common with the vector node than our ants, بيت.

**Step 6:** Our ant therefore is more likely to go to الشاعر. It does this while depositing its pheromone along the way. The ant leaves its odour on the destination node but does not take energy, as it is a back node. Now, its only option is to go to Poet.

**Step 7:** For the ant, the Poet-friend is a potential node. It therefore has the opportunity to build a bridge to its mother anthill. Therefore, the ant chooses to borrow and deposit its odour on the Poet node and creates a bridge to it.
pheromone. It reaches its mother nest and deposits all of its energy. The ant then returns to search mode and resumes its travels to the node where it will die and deposit its transported energy and the energy it took to make it.

6. Experimental evaluation

In this section, we present the results of our experiments.

6.a. Statistics of the Arabic dictionary

The main statistics of the Arabic dictionary that we created are summarized in Table I. The first column of this table gives the parts of speech, while the other columns indicate the number of Arabic parts of speech and the number of English sense definitions. Fig. 5 shows the screen of an English-Arabic dictionary based on WordNet capture. We see that Arabic words do not sufficiently correspond to all possible English senses.

<table>
<thead>
<tr>
<th>Arabic POS</th>
<th>English Sense</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nouns</td>
<td>9641</td>
</tr>
<tr>
<td></td>
<td>14680</td>
</tr>
<tr>
<td>Verbs</td>
<td>2777</td>
</tr>
<tr>
<td></td>
<td>6084</td>
</tr>
<tr>
<td>Adjectives</td>
<td>662</td>
</tr>
<tr>
<td></td>
<td>762</td>
</tr>
<tr>
<td>collocations</td>
<td>12905</td>
</tr>
<tr>
<td></td>
<td>263</td>
</tr>
<tr>
<td>Total</td>
<td>25985</td>
</tr>
<tr>
<td></td>
<td>21789</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the Arabic dictionary

6.b. Data pre-processing

The experimental dataset consists of six Arabic texts of various domains (culture, economy, international, local, religion, and sports). These texts are extracted from newspaper articles, which were recorded in the corpus of Elwatan (Abbas et al., 2011). Table 2 describes the characteristics of the corpus.

<table>
<thead>
<tr>
<th>Domains</th>
<th>Number of articles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Culture</td>
<td>2782</td>
</tr>
<tr>
<td>Economy</td>
<td>3468</td>
</tr>
<tr>
<td>International</td>
<td>2035</td>
</tr>
<tr>
<td>Local</td>
<td>3596</td>
</tr>
<tr>
<td>Religion</td>
<td>3860</td>
</tr>
<tr>
<td>Sports</td>
<td>4550</td>
</tr>
<tr>
<td>Total</td>
<td>20291</td>
</tr>
</tbody>
</table>

Table 2: Statistics of the corpus

To study the effect of the ACO algorithm, we have conducted six groups of experiments, for each group and for each text category, one-third of the articles were randomly specified and used for testing and the remaining articles were used for training. Table 3 illustrates the domains addressed by these texts and the distribution of words as described in the texts.

<table>
<thead>
<tr>
<th>Text</th>
<th>Domain</th>
<th>Words</th>
<th>Annotated</th>
</tr>
</thead>
<tbody>
<tr>
<td>D001</td>
<td>Culture</td>
<td>800</td>
<td>242</td>
</tr>
<tr>
<td>D002</td>
<td>Economy</td>
<td>2130</td>
<td>1133</td>
</tr>
<tr>
<td>D003</td>
<td>International</td>
<td>600</td>
<td>325</td>
</tr>
<tr>
<td>D004</td>
<td>Local</td>
<td>755</td>
<td>432</td>
</tr>
<tr>
<td>D005</td>
<td>Religion</td>
<td>295</td>
<td>125</td>
</tr>
<tr>
<td>D006</td>
<td>Sport</td>
<td>450</td>
<td>298</td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td>5030</td>
<td>2555</td>
</tr>
</tbody>
</table>

Table 3: The six articles in the dataset

Pre-processing aims to transform the Arabic text documents into a form that is suitable for the algorithm.

- Each document in the Arabic dataset is processed to remove digits, numbers, hyphens, and punctuation marks.
- We have normalized the letters “
- All non-Arabic texts were removed.
- Arabic function words (stop words) were removed. In these experiments, we utilized the stop word list generated by the website Sourceforge, a list containing 26037 words.

All of the results of the algorithm were obtained using a PC with Intel Core Duo CPU 2.66 and 6 GB RAM of main memory on Windows 7. The application was developed in Java language. Figure 8 shows the screen of our applications.

6.c. Evaluation metrics

Evaluation criteria are attempted to indicate how many words the system can disambiguate. The precision is a measurement of how many target words are correctly disambiguated and gives the accuracy of the system.

Precision = \( \frac{\text{correctly disambiguated words}}{\text{disambiguated words}} \)

Recall is defined by the ratio of the number of correctly disambiguated target words to the total number of target words in the dataset.

2http://sourceforge.net/projects/arabicstopwords/
Recall = \frac{\text{correctly disambiguated words}}{\text{tested set words}}

The F-measure is the harmonic mean of the precision and recall values, as described in the following equation:

F - measure = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}

For all of our experiments, we used these three evaluation measures (Recall, Precision, and F-measure) as the bases of our comparisons.

d. Selection of the parameters

The choice of parameters is critical to the performance of the algorithm. The first set of experiments involved the numerical investigation of the ACO parameters and their impact on the performance of AWSD. Table 4 shows the parameters of the ACO.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
<th>Value range</th>
<th>Estimation for Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetEnerg</td>
<td>Energy taken by an ant when it arrives on a node</td>
<td>1-30</td>
<td>9.0</td>
</tr>
<tr>
<td>EnergMax</td>
<td>Maximum quantity of energy an ant can carry</td>
<td>1-60</td>
<td>22.142</td>
</tr>
<tr>
<td>pheroEvap</td>
<td>Evaporation rate of the pheromone between two cycles</td>
<td>0.0-1.0</td>
<td>0.3577</td>
</tr>
<tr>
<td>EnergInit</td>
<td>Initial quantity of energy on each node</td>
<td>5-60</td>
<td>32.0</td>
</tr>
<tr>
<td>AntLife</td>
<td>Ant-life-span</td>
<td>1-30</td>
<td>27.0</td>
</tr>
<tr>
<td>TailleVector</td>
<td>Odour vector length</td>
<td>20-200</td>
<td>135.637</td>
</tr>
<tr>
<td>depotVector</td>
<td>Percentage of the odour vector component (words) deposited by an ant when it arrives on a node</td>
<td>0-100</td>
<td>0.9775</td>
</tr>
<tr>
<td>NbrCycle</td>
<td>Number of cycles of the simulation</td>
<td>1-100</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4: Parameters of the algorithm

<table>
<thead>
<tr>
<th>Notation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Nodes</td>
<td>415</td>
</tr>
<tr>
<td>Number of Paths</td>
<td>4096</td>
</tr>
<tr>
<td>Number of Ants</td>
<td>1026</td>
</tr>
<tr>
<td>Cycle Number</td>
<td>99</td>
</tr>
</tbody>
</table>

Table 5: Simulation Environment for Text d001

The algorithm does not give the same meanings on each execution; execution of the repeat is required at least twice to get the exact sense. We have noted that after several cycles (approximately 70), the results obtained are similar, as shown in Figure 4, and we can achieve the same results between two executions.

e. Comparison with other methods

Several approaches that address the problem of Arabic word sense disambiguation have been presented in section 3. The result of these methods cannot be compared directly to our algorithms’ results because those methods are for different tasks and their results were generated using different datasets.

For that reason, in this comparison, we apply the same experimental data that we have used before in the experimental study of the Genetic algorithm (Menai et al., 2012), Naïve Bayes classifier (Elmougy et al., 2008) and Modified version of the Lesk (Merhben et al., 2009) methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant colony algorithm (our approach)</td>
<td>80 %</td>
</tr>
<tr>
<td>Genetic algorithm</td>
<td>78.9 %</td>
</tr>
<tr>
<td>Naïve Bayes classifier</td>
<td>76.6 %</td>
</tr>
<tr>
<td>Modified version of the Lesk</td>
<td>67 %</td>
</tr>
</tbody>
</table>

Table 6: Comparison with other methods

The results obtained with the ant colony algorithm corroborate those obtained in previous studies on ACO for WSD in English (Schwab et al., 2011, 2012, and 2013) even though they are not comparable.

Our experiments show that our approach exhibits superior results when compared with existing techniques. This is not surprising because our approach attempts to globally optimize the text cohesion, while the methods we used as a baseline optimize it locally. Much better precision and recall were obtained by other
methods for more specific WSD tasks in Arabic, such as finding the correct sense of query translation terms, and disambiguation of polysynous and homograph Arabic nouns.

We evaluated the performance of different variants of the ACO algorithm on a set of 5000 words extracted from an Arabic corpus. The results obtained show that ACO is the best performing algorithm.

7. Conclusion and future work

In this paper, we presented an approach of word sense disambiguation for the Arabic language that tackled several difficulties, in particular the rareness of studies in the field of semantics offering integrated solutions and the lack of lexical resources, in addition to technical problems related to programming imposed by the nature of the Arabic language.

To achieve this, we set our objectives at the beginning and we organized our work into three main steps. First, we conducted a theoretical study on the field. We collected and prepared all of the necessary linguistic data: a body of work and a lexical database. Then, we developed our system. Finally, we concluded with a quantitative and qualitative evaluation of our system.

To the best of our knowledge, there is no published research proposing the Ant colony algorithm-based approach to solve the AWSD problem. The ACO algorithm has the advantage of being easily adapted to a dynamic environment, which is important for the WSD problem because natural languages can evolve quickly.

There is still a lot of work to be done to introduce the Arabic language to the world of NLP, with wider opportunities than is the case with other languages. We believe and hope that this study offers solutions that can be exploited in the developing use of the Arabic language in the world of information technology, communications and electronic management.

Fig. 6 Part of the Arabic Dictionary
References

Notes
2. http://wordnet.princeton.edu/wordnet/map/wntats/7WN.html
Figures, images and tables

Fig. 1 Arabic Dictionary

Fig. 2 Diagram of the proposed approach for AWSD

Fig. 3 Vector of sense’s definition ‘wash’

Fig. 4 Example of phrase”poet said rhyme”

Table 1: Statistics of the Arabic dictionary

<table>
<thead>
<tr>
<th>Arabic POS</th>
<th>English Sense</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nouns</td>
<td>9641</td>
</tr>
<tr>
<td>Verbs</td>
<td>2777</td>
</tr>
<tr>
<td>Adjectives</td>
<td>662</td>
</tr>
<tr>
<td>ncallations</td>
<td>12908</td>
</tr>
<tr>
<td>Total</td>
<td>25985</td>
</tr>
</tbody>
</table>

Table 2: Statistics of the corpus

<table>
<thead>
<tr>
<th>Text</th>
<th>Domain</th>
<th>Words</th>
<th>Annotated</th>
</tr>
</thead>
<tbody>
<tr>
<td>D001</td>
<td>Culture</td>
<td>800</td>
<td>242</td>
</tr>
<tr>
<td>D002</td>
<td>Economy</td>
<td>2130</td>
<td>1133</td>
</tr>
<tr>
<td>D003</td>
<td>International</td>
<td>600</td>
<td>325</td>
</tr>
<tr>
<td>D004</td>
<td>Local</td>
<td>755</td>
<td>432</td>
</tr>
<tr>
<td>D005</td>
<td>Religion</td>
<td>295</td>
<td>125</td>
</tr>
<tr>
<td>D006</td>
<td>Sport</td>
<td>450</td>
<td>298</td>
</tr>
<tr>
<td>TOTAL</td>
<td>-</td>
<td>5030</td>
<td>2555</td>
</tr>
</tbody>
</table>

Table 3: The six articles in the dataset
<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
<th>Value range</th>
<th>Estimation for Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetEnerg</td>
<td>Energy taken by an ant when it arrives on a node</td>
<td>1-30</td>
<td>9.0</td>
</tr>
<tr>
<td>EnergMax</td>
<td>Maximum quantity of energy an ant can carry</td>
<td>1-60</td>
<td>22.142</td>
</tr>
<tr>
<td>pheroEvap</td>
<td>Evaporation rate of the pheromone between two cycles</td>
<td>0.0-1.0</td>
<td>0.3577</td>
</tr>
<tr>
<td>EnergInit</td>
<td>Initial quantity of energy on each node</td>
<td>5-60</td>
<td>32.0</td>
</tr>
<tr>
<td>AntLife</td>
<td>Ant-life-span</td>
<td>1-30</td>
<td>27.0</td>
</tr>
<tr>
<td>TailleVector</td>
<td>Odour vector length</td>
<td>20-200</td>
<td>135.637</td>
</tr>
<tr>
<td>depotVector</td>
<td>Percentage of the odour vector component (words) deposited by an ant when it arrives on a node</td>
<td>0-100</td>
<td>0.9775</td>
</tr>
<tr>
<td>NbrCycle</td>
<td>Number of cycles of the simulation</td>
<td>1-100</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4: Parameters of the algorithm

<table>
<thead>
<tr>
<th>Notation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Nodes</td>
<td>415</td>
</tr>
<tr>
<td>Number of Paths</td>
<td>4096</td>
</tr>
<tr>
<td>Number of Ants</td>
<td>1026</td>
</tr>
<tr>
<td>Cycle Number</td>
<td>99</td>
</tr>
</tbody>
</table>

Table 5: Simulation Environment for Text d001

Fig. 5 The F-measure result generated by the algorithm against six Arabic datasets. In each dataset, we arbitrarily consider 70% of documents for training and 30% for testing.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant colony algorithm (our approach)</td>
<td>80 %</td>
</tr>
<tr>
<td>Genetic algorithm</td>
<td>78.9 %</td>
</tr>
<tr>
<td>Naïve Bayes classifier</td>
<td>76.6 %</td>
</tr>
<tr>
<td>Modified version of the Lesk</td>
<td>67%</td>
</tr>
</tbody>
</table>

Table 6: Comparison with other methods
Fig. 6 Part of the Arabic Dictionary

Fig. 7 Sample of graph text
Fig. 8. The output annotation in the format required for the evaluation script