Word sense disambiguation using optimisation techniques

S. Rajini* and A. Vasuki

Kumaraguru College of Technology,
Coimbatore, Tamil Nadu, 641049, India
Email: rajini.s.cse@kct.ac.in
Email: vasuki.a.ece@kct.ac.in
*Corresponding author

Abstract: In the field of computational linguistics, word sense disambiguation (WSD) is a problem of high significance which helps us to find the correct sense of a word or a sequence of words based on the given context. It is treated as a combinatorial optimisation algorithm wherein the aim is to discover the set of senses which help to improve the semantic relatedness among the target words. Nature inspired algorithms are helpful to find optimal solutions in reduced time. They make use of collection of agents that interact with the surrounding environment in a coordinated manner. In this article, two such algorithms, namely, cuckoo search and firefly algorithms, have been used for solving this problem and their performance have been compared with the D-bees algorithm based on bee colony optimisation algorithm. They have been evaluated using the standard SemEval 2016 task 11 dataset for complex word identification. Experimental results show that firefly algorithm is performing better than the other algorithms.

Keywords: word sense disambiguation; WSD; cuckoo search; optimisation; firefly; bees algorithm; unsupervised.


Biographical notes: S. Rajini has completed her MS (by res.) from the Anna University, Chennai and is working as an Associate Professor in Computer Science and Engineering Department of Kumaraguru College of Technology, Coimbatore, India. She has over two decades of teaching experience. She has published many research articles in journals and conferences. Her areas of interest include distributed computing and artificial intelligence.

A. Vasuki has more than 26 years of experience in teaching, research and administration. She is currently a Professor of Mechatronics Engineering at the Kumaraguru College of Technology, Coimbatore. She has published three book chapters, 28 papers in international journals and over 60 papers in national and international conferences. She has completed three funded research projects under RPS, AICTE and RESPOND, ISRO. She is currently guiding several research scholars for the Doctoral program. Her areas of interest are signal/image processing and communications.

This paper is a revised and expanded version of a paper entitled ‘Word sense disambiguation using optimization techniques’ presented at SEEE International Conference on Innovation in Electrical, Electronics and Intelligent Computing, Coimbatore, 19–20 July 2019.
1 Introduction

Lexical ambiguity is a regular feature of most of the languages used in the world. A word may possess different senses or meanings and so it is considered to be an ambiguous word. The process of resolving ambiguity of a word based on the context in which it occurs is known as word sense disambiguation (WSD) (Agirre and Edmonds, 2006). WSD replaces the ambiguous word by the proper one depending on the surrounding context after identifying the correct meaning of the word (Navigli, 2009). For example, consider the following two sentences:

1. The board meeting of the company took place yesterday.
2. The teacher wrote on the white board.

The meaning of ‘board’ is different in the two sentences. In the first sentence, ‘board’ means “an organized body of administrators” and in the second sentence, ‘board’ refers to “a large vertically positioned flat surface used for writing.”

WSD task can be performed on one or more texts. An extract of text can be considered as a set of words and WSD task is helpful for finding the correct sense(s) for all the textual words or for only a few of them. WSD can also be considered as a classification task in which word senses are the classes and an automatic classification method can be used to assign each word occurrence to one or more classes. This is performed by making use of context information and external knowledge sources.

WSD can be easily achieved by using knowledge-based trained data and feature selection. Knowledge-based trained data can be labelled with word senses or unannotated. For domains (Khapra et al., 2010) like medicine and engineering, separate knowledge bases are created.

The Lesk (1986) algorithm calculates the similarity value by counting the number of overlapping words between two definitions of the senses. Banerjee and Pedersen (2002) modified the Lesk algorithm by taking into account semantically related senses, like hypernyms, hyponyms, etc. to improve the accuracy of similarity value. Sense inventories like WordNet are useful for implementing Lesk algorithm.

WSD task was used for performing machine translation in the early days (Ide and Véronis, 1998). Later, many methods were proposed to resolve the ambiguity. They are classified as supervised learning algorithms, semi-supervised learning algorithms and unsupervised learning algorithms (Mante et al., 2014).

In supervised approach, the algorithms given by Banerjee and Pedersen (2002), Florian et al. (2002), Lee et al. (2004) and Nameh et al. (2011) work on already trained or classified (sense-tagged) data which can be in the form of Wordnets or knowledge bases (Tyar and Win, 2015) to differentiate the new data. In case of semi-supervised approach used by Li and Li (2004) and Yu et al. (2011), the training data was either labelled, unlabelled or partially trained. In unsupervised learning approach, raw data which is not sense tagged was used by Diab and Resnik (2002) and Navigli and Lapata (2010). A lot of recent research is based on knowledge-based unsupervised approaches and it is taking place in order to achieve state of art performance (Navigli, 2009).

Gunavathi and Premalatha (2014), Menai and Alsaeedan (2012) and Deb and Yang (2009) used nature inspired algorithms such as genetic algorithm, bee colony optimisation (BCO), cuckoo search (CS), particle swarm optimisation, ant colony optimisation and firefly algorithm (FA) for solving combinatorial optimisation problems.
This paper introduces firefly and CS algorithms which are swarm intelligence methods that are bio-inspired for solving the WSD problem. The results of implementation are compared with those produced by testing D-bees algorithm on a standard dataset. These algorithms are discussed in the next section.

2 Optimisation algorithms

While solving optimisation problems, cost functions called objective functions are optimised where a set of feasible solutions are given. These solutions satisfy the problem’s constraints. For a WSD problem, the objective function is the relatedness measure between two senses and the goal is to attain the senses which maximise the overall relatedness value.

Pedersen et al. (2015) defined WSD as an optimisation problem where the objective function helped to maximise the overall relatedness value among the words within a certain context window of chosen length. The overall relatedness is calculated for each sequence and finally the sequence that results in the best relatedness is considered.

The following subsection relates about D-bees algorithm proposed by Abualhaija and Zimmermann (2016).

2.1 D-bees algorithm

There are several proposed computational methods inspired by honey bees in nature each of which is used in a certain application. In this work, the D-bees algorithm has been implemented and tested on a standard dataset taken from Semeval-2007 (Navigli, 2009). D-bees algorithm is based on the BCO meta-heuristic proposed by Abualhaija and Zimmermann (2016).

Social insects in general are self-organised and adapt well to the environmental changes. This is usually facilitated by exchanging information among the individual insects in order to achieve a collective intelligence (emergence) for the sake of the colony. Bees communicate directly with other bees by performing a sort of dance on a dancing floor in the hive.

First, bee scouts explore the unknown environment looking for a food resource from which they can collect nectar for the hive. Once a food source has been found, these scouts head back to the hive and perform a certain dance based on the goodness of the food resource and the distance to it which amounts to an advertisement to recruit other bee fellows to further exploit this food resource. There are two types of dances, a round dance if the food source is close to the hive, and a waggle dance if the food is farther away, through which the bees also give information about the direction to the food source.

Having watched the dance floor, the uncommitted bees may decide to follow one of the advertised paths. The committed bees can stick to their own path or abandon it and follow one of the other advertised paths. These decisions usually depend on the hive needs and the characteristics of the food resources.

Initially, all bees are in the hive. Figure 1 contains the pseudo code of the BCO algorithm (Teodorović, 2009).
While solving the WSD problem, Abualhaija and Zimmermann (2016) renamed the BCO algorithm as D-bees. The D-bees algorithm for WSD assumes that each bee agent explores part of the search space of the combinatorial problem and generates a particular solution to the problem. For this, the number of bee agents is pre-defined. The process is simulated by two alternating phases, a forward pass and a backward pass.

The bee agents have a local memory in which they store the actual path and its quality. Initially, the path contains a sense of the target word from which the bee is created and the quality is set to zero. In the forward pass, the bees move along the next possible path or randomly based on the usage of frequencies. A bee agent does not evaluate all the next possible moves during a forward pass; rather it chooses a random sense of the next word. After each step, the bee agents update their local memory. They append the chosen sense to the path and update its quality by adding incrementally the similarity value. The bee agent then moves a step further until the number of constructive moves (NC) is reached.

Then, the bee agents return to the hive each of which holds a partial solution. In the hive, they exchange information on a virtual dancing floor and initiate the backward pass. During this pass, each bee calculates the loyalty probability. Based on this, the bee decides whether to stay loyal to its path or to become uncommitted and follow one of the advertised solutions.

The forward and backward passes are alternated until there are no more target words to be disambiguated. The bee agent with the best found solution in terms of path quality is stored. Finally, the best solution is returned as the output of the disambiguating process.

3 Proposed algorithms

In this section, description of the proposed firefly and CS methods is given.
3.1 Firefly algorithm

Yang (2010) has explained about the working of FA algorithm based on the characteristics of firefly insects. These insects produce brief flashes of light in a rhythmic manner to attract partners or potential prey and as a protective warning toward the predator. The light intensity of flashes keeps varying for different observers based on their distance. The light appears to be dim as the distance increases. The light intensity is also influenced by the air absorption capacity of the surroundings (Ali et al., 2014). Therefore, the intensity becomes less appealing as the distance increases.

The FA for WSD (Figure 2) takes sentence as input and finds the sense of a target word which is represented as intensity using similarity value of the senses. Then, it compares the similarity of senses between pairs of words to find the best solution. Maximisation of intensity of senses is helpful for finding actual sense of a word. The new solutions are generated based on distances between two words and loss factor. As the distance decreases, similarity value increases.

**Figure 2** Pseudo code of FA (Yang, 2010) modified for WSD

<table>
<thead>
<tr>
<th>Objective function f(a), a = (a1, a2, ..., ad)T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate initial population of target words ai (i = 1, 2, ..., n)</td>
</tr>
<tr>
<td>Similarity sense Si of ai is determined by f(ai)</td>
</tr>
<tr>
<td>Define loss factor y</td>
</tr>
<tr>
<td>While (t &lt; MaxGeneration)</td>
</tr>
<tr>
<td>For k = 1:n (all n words in a sentence)</td>
</tr>
<tr>
<td>For l = 1: n (all n words in a sentence)</td>
</tr>
<tr>
<td>If (Sk &lt; Sl) move word k towards word l;</td>
</tr>
<tr>
<td>end if</td>
</tr>
<tr>
<td>Change attractiveness value with distance r between k and j using exp(–yr)</td>
</tr>
<tr>
<td>Evaluate new solutions and update similarity</td>
</tr>
<tr>
<td>Sense</td>
</tr>
<tr>
<td>End for j</td>
</tr>
<tr>
<td>End for i</td>
</tr>
<tr>
<td>Perform ranking of similarity senses and determine the global best g</td>
</tr>
<tr>
<td>End while</td>
</tr>
<tr>
<td>Examine the results</td>
</tr>
</tbody>
</table>

3.2 CS algorithm

CS is an optimisation algorithm (Figure 3) and it was proposed by Deb and Yang (2009). Cuckoo birds lay their eggs in the nests of other birds belonging to varying species which act as hosts. Host birds may throw away such eggs or build a new nest elsewhere after abandoning the old one. All eggs are considered as solutions whereas cuckoo eggs are treated as new solutions.
Figure 3  WSD algorithm using CS

Objective function $f(x) = (x_1, x_2, \ldots, x_n)^T$

Generate an initial population of $n$ target words $x_i$ ($i = 1, 2, 3, \ldots, n$);

While stopping criterion or limit for maximum generations is not reached

a  Choose a word sense (say $i$) randomly using Levy distribution

b  Evaluate its similarity score/fitness

c  Choose another word sense (say $j$) randomly

d  If (fitness value of $i >$ fitness value of $j$)

Substitute the new solution for $j$

End while

A part of the worst senses is abandoned and new ones are built;

Keep the best solutions/word senses;

Rank the solutions/word senses and find the current best;

Pass the current best sense to the next generation;

The first round of CS algorithm consists of selecting nests at random, employing of Levy flights for substituting the solutions in a nest, followed by abandoning of relatively unfit solutions (Deb and Yang, 2009). Each egg is tested before discarding. Discarding of egg is carried out based on lack of fitness. The algorithm sends the successful solutions through a second round and so on until an optimal solution emerges. This is carried on until maximum numbers of generation are reached or the stopping criterion is met. Stopping criterion is usually taken to be as convergence of the fitness function.

CS algorithm takes a sentence as an input such that one word is chosen to represent a nest which produces cuckoo agents (word senses) and sends them to the words in the context window. The algorithm considers a bag of words as they appear in the sentence, excluding the nest word. The aim is to use the new and potentially better solutions/word senses to replace a worse solution in the nests until there are no more target words to be disambiguated. The definitions of the senses as well as the semantic relations are retrieved from WordNet.

4  Implementation

The optimisation algorithms are implemented in MATLAB with the use of Java functions along with WordNet browser to get the sense of a word. They are evaluated on a standard SemEval 2016 task 11 complex word identification dataset (Paetzold and Specia, 2016). The flow diagram of the proposed model is as shown in Figure 4. It is based on three algorithms, namely D-bees, CS and FAs.

For all the three algorithms which have been tested, the total number of sentences examined = 200.

From each of the sentences in the dataset that is derived from SemEval 2016 task 11, four target words per sentence are extracted and for these disambiguation is carried out. The senses of these words are determined by using Java programs that help to load WordNet browser.
For each of the senses of a given word, a score is displayed. This score is based on similarity value of all senses for a given word. This score is used to find the exact sense of a given word which is activated in a certain context.

The D-bees algorithm makes use of the following parameters and their values:

- a) the number of bees that are produced in a hive which corresponds to the number of senses = 3
- b) number of recruiters = 3
- c) total number of iterations = 10
- d) number of constructive movements in forward pass = 3.

The CS uses the following main parameters with values:

- a) number of nests = 4
- b) probability of host bird finding cuckoo’s egg (pa) = 0.25.

The FA uses the following parameter values:

- a) initial attractiveness $\beta_0 = 2 \times \text{rand}$ where rand is a random number
- b) the loss factor $\gamma = 1$
- c) for Levy distribution, the random parameter $\alpha = 1$.

5 Results

Swarm intelligence methods have led to better results because the agents are capable of maintaining memories about partial solutions. Moreover, they communicate with each other and exchange knowledge regarding the goodness of partial solutions.

The counts of true positive (TP), false positive (FP), false negative (FN) and true negative (TN) while running the programs for testing FA, CS and D-bees are shown in Table 1. The experimental results obtained correspond to a sample run for 800 target words. It illustrates that FA and CS are capable of producing more ‘TP’ values than D-bees. So, precision and recall values go high.
Word sense disambiguation using optimisation techniques

Table 1  Results for 800 target words

<table>
<thead>
<tr>
<th>Method name</th>
<th>True positive (TP)</th>
<th>False positive (FP)</th>
<th>False negative (FN)</th>
<th>True negative (TN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefly</td>
<td>624</td>
<td>7</td>
<td>95</td>
<td>74</td>
</tr>
<tr>
<td>Cuckoo search</td>
<td>531</td>
<td>17</td>
<td>190</td>
<td>64</td>
</tr>
<tr>
<td>D-bees</td>
<td>484</td>
<td>58</td>
<td>234</td>
<td>24</td>
</tr>
</tbody>
</table>

5.1 Accuracy

The accuracy is ratio between total number of TPs and TNs and the total number of true and false values for all sentences examined. Accuracy can be calculated from formula given as follows:

$$\text{Accuracy} = \frac{\text{True positive (TP)} + \text{True negative (TN)}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

(1)

Figure 6 shows the comparison of D-bees, CS optimisation, and FAs in terms of accuracy values in percentages. The results show that the FA approach provides higher accuracy than CS optimisation and D-bees algorithms.

Figure 6  Comparison of accuracy (see online version for colours)

5.2 Precision

Precision (Kent et al., 1955) is the number of correct senses got as results divided by the number of all returned senses.

$$\text{Precision} = \frac{\text{True positive (TP)}}{\text{True positive (TP)} + \text{False positive (FP)}}$$

(2)

Figure 7 shows the comparison of D-bees, CS optimisation, and FAs in terms of precision values. The results show that the FA approach provides higher precision than CS optimisation and D-bees algorithms. CS and FAs have almost 0.95 and 0.97 as precision.
values which are very close. These algorithms predict correctly the meanings of the words most of the time. D-bees also are having a good precision value of 0.9.

**Figure 7** Comparison of precision (see online version for colours)

5.3 Recall

Recall is defined as the statistical measure such that the fraction of relevant material is returned by the search (Kent et al., 1955).

\[
\text{Recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative (FN)}}
\]  

The recall values shown in Figure 8 indicate that the FA provides higher recall than CS optimisation and D-bees algorithms. The recall values of D-bees and CS have only a minor difference of 3%.

**Figure 8** Comparison of recall (see online version for colours)
5.4 F-measure

F-measure is derived from the values obtained for precision and recall (Kent et al., 1955). It is calculated as:

\[
\text{f-measure} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
\]  

(4)

Figure 9 shows the comparison of D-bees, CS optimisation, and FAs in terms of F-measure values. The FA approach provides higher F-measure than CS optimisation and D-bees algorithms.

5.5 Overall performance

FA has provided better results when compared to CS and D-bees methods in terms of precision, recall, accuracy and f-measure. FA provides an excellent convergence rate and a strong exploration capability. The results show that FA provides better results as it uses inverse-square law to guide interactions between all search processes. This helps for several localised interactions and as a result of this, multiple regions of search are formed. Search for word senses happens along multiple paths.

CS algorithm has performance that lies between those of firefly and D-bees algorithms. It is efficient in finding the global optima with high success rates. It is also quick in completing runs. D-bees algorithm and CS have only minor variations in the result parameters.

6 Conclusions and future work

In this work, two algorithms have been proposed and implemented for solving the WSD problem. First is CS, which is inspired by the meta-heuristic optimisation
algorithm, in which several cuckoos collaborate to solve the problem. Another algorithm is FA, where in the flashing behaviour of fireflies are used to solve the problem. D-bees algorithm is an existing optimisation method for WSD problem. All the three algorithms were evaluated on a standard SemEval 2016 task 11 complex word identification dataset. Based on the precision, recall, accuracy and F-measure, FA provides better results compared to CS and D-bees algorithms. CS has second ranking among the three methods and D-bees the last. As a future work, WSD by using FA could be used in semantic web which requires automatic annotation of documents.

References


