Using Fuzzifiers to Solve Word Sense Ambiguation in Arabic Language

Madeeh Nayer El-Gedawy
Computer Center
Institute of Public Administration (IPA) – Jeddah

ABSTRACT

Text mining techniques confront many challenges when dealing with the Arabic language including lexical disambiguation because Arabic is a highly inflectional and derivational language, most of the Arabic texts are devoid of diacritics especially Modern Standard Arabic (MSA), thus, it is a must to depend on the ambiguous word context under study. Two fuzzy logic classifiers have been built and compared to a supervised corpus-based Naïve Bayes classifier. The study concludes that the results that have been obtained from our fuzzy logic classifiers are more accurate and promising.

Keywords

Fuzzifiers - Word Sense Disambiguation – Jaccard Similarity – Sigmoid Function – Root Extraction.

1. INTRODUCTION

Word Sense Disambiguation (WSD) is one of the trickiest tasks in text mining and this task gets more difficult in Arabic because the Arabic Language has a loose word order (WO), and word discretization marks are usually absent from Modern Standard Arabic (MSA) [1]. Most Arabic words have dozens of different meanings; some of them are closely related (polysemy), and interestingly there are some words that can mean something and its opposite.

The paper is treating WSD as a pure text classification problem by marking the correct sense using keywords in context (KWIC) [2]. Our WSD relies on an Arabic sense inventory which feeds the classifier with the training clues that denote every sense. We need to consider a good classifier that allows classes to overlap as the senses do not have hard margins because words tend to appear in different senses with different densities.

To fulfill the task of Word Sense Disambiguation, we firstly begin by some preprocessing tasks such as: deleting existing stop words in the corpus, doing feature selection to limit the words getting analyzed by depending on a simple term frequency (TF) measure and then we chose to do root extraction using Al-Shalabi algorithm due to its simple implementation. We adopt a fuzzy logic classifier for marking the different senses. The fuzzy logic depends on a sense inventory that we created by exploiting the English WordNet complete synsets and their corresponding glosses and examples which were extended by doing a query expansion using Google APIs, all these keywords are fed to the fuzzy logic for training senses.

2. PREPROCESSING IN ARABIC

Preprocessing tasks are data preparation procedures that should be done before starting dealing with different text mining techniques. We will discuss two important processing tasks in this section: the removal of stop words (functional terms that affect the accuracy of text mining) and doing root extraction and stemming.

2.1 Stop Word Removal

The majority of text classifiers remove stop words; this removal process could be very aggressive to the extent that 90 percent of all the features are eliminated [2][3]. The removal of stop words also reduces the size of the corpus generally up to 25%, which leads to better accuracy [3]. Stop words can also be domain specific [4], for example, the word "دم" may be a common term in a corpus addressing diseases, but for sure it will not be a stop word in Quran as in the verse: “ حرمت عليكم الميتة والدم ولحم الخنزير”.

In literature, a stop word is defined using 2 criteria: First, it must have a high frequency of documents (DF) or term frequency. Second, the terms correlations with categories should be small. Mostly, the chi-square test is used to measure the correlation between a term and a specific category [3].

An entropy based approach has been adopted [5] to create an Arabic stop word list for our WSD system as explained by [5][6][7]. The dataset used for extracting the stop word list is a sample of the NEWSWIRE corpus that contains 17,000 articles.

Step 1: Word frequency is the number of times a word appears in a document. The list is sorted in descending order of frequency.

Step 2: we measure the likelihood Li,j of the term wj in document Di:

$$Li,j = \frac{frequency\ in\ the\ document\ Di}{the\ total\ number\ of\ words\ in\ document\ Di}$$

Then we calculate entropy that measures the information value of the word wj:

$$H(w_j)=\sum Li,j \ * \ log(1/\ Li,j)$$

2.2 Root Extraction and Word Stemming

Arabic has about 10,000 roots, 5000 of them are still commonly used [8]. A root can generate hundreds of lexical forms of different meanings. Arabic words have 2 problems with stemming: inflection (more letters are attached to the word without changing the meaning: “عامل – عاملة - عمال”) and derivation (attached words cause different meanings: “عمل – عامل -عمولة”). There are 2 types of stemmers: root extractors and light stemmers. Root extractors are aggressive stemmers that confront the problem of over-stemming where many words of different meanings can be conflated to the same root. For example, in the verse: “ فسينفقونها ثم تكون عليهم حسرة ثم يغلبون”; we notice that the word ‘فسينفقونها’ if expressed in a 3 letters root ‘نفق’, then we will have 4 different meanings: ‘أنفق المال’,’نفق أى مات’,نفق لعبور السيارات‘, and ‘نفاق ومنافق’. So, over-stemming leads to many candidates that should be examined carefully and that leads to a more complex analysis. On the other hand, light stemmers try to find the shortest possible path without compromising the meaning, so it limits the candidates as much as possible but it sometimes fails to deal with affixes and broken (irregular) plurals.

2.2.1 Light Stemming

Many light stemmers have been proposed for Arabic. Larkey tried to use an n-gram model but observed that it does not suit the Arabic language as it does for English. Light 10 is reported to be the best light stemmer [9]. Also, both Aljalayel3 and Berkley stemmer are both reported to be excellent light stemmers [9] [10]. Both Xu and Croft have done superb work by using a corpus-based stemmer which can be adapted to a certain domain [11], but it needs a reasonable corpus size to give both a word and its stem the possibility to be mentioned. Also, it is noteworthy to mention Al-Beltagy stemmer which extends the light10 by utilizing a corpus.

2.2.2 Root Extraction

Aggressive stemmers have larger clusters of forms that could have different meanings, so these forms within the clusters are not quite homogeneous. Many attempts have been exerted to devise good root extractors. Khoja made a root extractor in 1999 that removes diacritics, stop word list, and some specific letters such as: “ال - و”; then the longest prefix or suffix is removed and then matched with a list of patterns [12]. This approach has been enhanced by Taghva in 2005 [13]. Al-Shalabi has devised a very straightforward root extractor that depends heavily on heuristics. We will discuss this approach in details as it is used by our WSD system; the main advantage of this approach is that it is simple in implementation which was coded in c#.

The first step for using Al-Shalabi [14] is to check the number of letters in the word, if it is less than or equal to 3, then the word will be taken without any further processing, else the following simple steps are followed:

1. For each letter in the term (from right to left) apply weight and rank values.
2. Measure the product of the rank and weight for each letter.
3. Keep only the letters with the smallest first three values.

Al-Shalabi did not clarify the theoretical foundation these ranking and weighting; only these classes and values were chosen after deep experimentation.

3. WORD SENSE DISAMBIGUATION TECHNIQUES

There are two main philosophies for dealing with WSD: deep approaches and shallow approaches; Shallow approaches don't try to understand the text. They just consider the surrounding words. It depends on the rule of: "one sense per discourse" as a generalization for "one sense per collocation" rule [15] [16]; where words are syntagmatically related as they tend to appear together in same syntagma (sentence) [17].

This approach uses a training corpus of words tagged with their word senses. Actually, it gives better results in practice, but of course it can be confused by tricky sentences like the verse: "وعلى الذين يطيقونه فدية طعام مسكين"; actually the word " يطيقونه " can mean either: able to or not able to and the context allow both meanings. This type of ambiguation is entitled "Pun" where 2 or more different senses can be swapped and the sentence will still be grammatically and semantically correct. Comparing and evaluating different WSD approaches is difficult because of the different training sets, test sets, and knowledge resources adopted. WSD is very important in many Information Retrieval (IR) aspects: filtering results, better ranking, giving suggestions, and query expansion [18]. WSD affects the recall and precision of any text mining (TM) classifier.

Our WSD system is classified as a knowledge-based and supervised approach. One important component in our system is WordNet which is a lexical knowledge base depends on conceptual lookup as it organizes terms lexical information in word meanings manner rather word forms. Actually, the main use of WordNet in natural language processing (NLP) and text mining is WSD. WordNet is created in Princeton University in 1985 [19]. It resembles the human nature that has a hierarchical lexical nature for storing related nouns [20]. Words in WordNet are not only paradigmatically related as in any other lexicon, but it is also syntagmetically related. WordNet is a relational semantic resource where the senses are its backbone. A WordNet sense which entitled 'synset' is a set of definite words that mean the same thing. The order of the set is obliged to the popularity of each of the words for expressing this meaning. Following the WordNet synsets is much better than engineering new ontologies as WordNet synsets follow very tough and restricted rules for synsets creation: minimality, coverage, and replacability [20 [21]. WordNet is a hierarchical resource that depends on simple semantic relations.

4. FUZZY LOGIC CLASSIFIER

The idea is to use a fuzzier for comparing glosses, by comparing the words contained in the glosses by semantic word similarity rather than simple term matching. In order to measure the similarity of word senses in glosses, it is very helpful to disambiguate the words in context, so that the similarity of only the ambiguous word senses need to be determined more easily, that is, without needing to measure similarity between many different senses of every word. This feature of the algorithm makes it recursive, because in order to disambiguate a sentence, we must also disambiguate the glosses of all the words in the sentence. This recursion depth must be limited in order to avoid disambiguating the entire set of glosses found in

WordNet, for a single input sentence. The task of disambiguating all the glosses and example sentences in WordNet is something carried out as a separate process, providing auxiliary information to WordNet, which could be loaded later to save time during subsequent computations [22].

Lot of work was done in the area of fuzzy text classification and fuzzy word sense disambiguation. [23] used a fuzzier where they construct a feature vector in the form of a membership degree with respect to every class. The feature vector is defined as:

$$\text{Feature vector (i)} = \\ < \mu\ (fi, c1), \mu\ (\text{fi}, \text{c2}), \mu\ (\text{fi}, \text{c3}), \dots, \mu\ (\text{fi}, \text{cn}) >$$

They use a simple formula for obtaining μ (fi, ce) as proposed by (67) which is defined as:

$$\mu\ (\text{fi},\ c_e) = \frac{\sum \text{Term Frequency in a document} * \text{flag}}{\sum \text{Term Frequency in a document}}$$

Where flag equals 1 if document belongs to class “e” and 0 otherwise.

[22] used a fuzzy logic approach originated from Rocchio algorithm. Specifically, a cluster center in Rocchio is created for each category from training documents and the similarity between a test document and a category is measured using cosine similarity. In the fuzzy similarity approach, a fuzzy term-category relation is developed, where the set of membership degree of words to a particular category represents the cluster prototype of the learned model. Based on this relation, the similarity between a document and a category's cluster center is calculated. The membership of a term t to a specific category c is calculated from the total number of term occurrences in category divided by the total number of term frequency in all categories, a similar approach is adopted by [23].

[24] preferred to construct fuzzy synsets and debated that dictionary definitions are incomplete, they used co-occurrence graphs to extract these synsets from dictionaries by clustering synonymous words. The fuzzy synsets are discovered using these steps:

1. Create an empty sparse matrix M (N*N).
2. Fill each cell Mij with the similarity between the adjacency vectors of the words ni and nj.
3. Normalize the columns of M, so that the values in each column, Mj, sum up to 1.
4. Extract a fuzzy cluster Fi from each row Mi, consisting of the words nj where Mij > 0. The value in Mij is used as the membership degree of the word nj to Fi.

5. OVERALL METHODOLOGY AND PROPOSED SYSTEM

We make use of a comprehensive resource of words associated with their senses, then we build a classifier to detect the most probable sense for an ambiguous word in a sentence. Fuzzy logic deals with the word vagueness as well as the word disambiguation [25]. Fuzzy logic can deal with vagueness by implementing ‘hedges’ [26].

The following 10 ambiguous words will be tested: ’, هم’,’ عجل’,’ مر’,’ ذكر’,’ حر’,’ فجر’,’ علم’,’ عملية’,’مارس‘ and ‘مال’. Each word of them will be translated into English words representing the different senses of this word, and then we get the exemplar sentences from WordNet that denote these senses, and translate back these sentences into Arabic. We used Google and Bing APIs to translate from Arabic to English and vice versa.

It is noticed that the examples size associated with senses in WordNet is not large. These examples keywords are the training set of the fuzzy logic classifier; so we need to enrich these keywords to feed sufficient cues denoting different senses. Thus, we will do a tricky thing here; we will pass each of these English training examples as a query to Bing and Google search engines using their search APIs coded in C#. The APIs fetch the most appropriate English sentences that match the query. We add the most frequent contextual words in the top 10 sentences to the classifier; thus, for every sense, the classifier takes into account the words extracted from WordNet examples and the contextual frequent words fetched. The query expansion method used here is a local analysis method [27] [28] and it uses local feedback not relevance feedback [29]; thus, we assume that the top ranked documents are the most likely relevant documents [30] [31].

To expand a WordNet exemplar sentence using the APIs:

Step 1: In our C# program, by use the English WordNet sentence exemplar as a query passed to the APIs.

Step 2: We will only consider contextual frequent words in the retrieved sentences that have frequency>=3.

Step 3: The fetched group of contextual words are annotated with the sense of the word used in retrieval and added to the sense inventory; the following 2 tables shows a sample of words statistics stored for the word 'عملية'.

Table 1: Contextual words statistics

Translated	Source	Frequency	Sense
إنزال	WordNet	1	Military
إنزال	WordNet	0	Operation
إنزال	WordNet	0	Practical
إنزال	Query	3	Military
إنزال	Query	0	Operation
إنزال	Query	0	Practical
نجاح	WordNet	1	Military
نجاح	WordNet	2	Operation
نجاح	WordNet	0	Practical
نجاح	Query	3	Military
نجاح	Query	1	Operation
نجاح	Query	1	Practical
قلب	WordNet	0	Military
قلب	WordNet	1	Operation

قلب	WordNet	0	Practical
قلب	Query	0	Military
قلب	Query	4	Operation
قلب	Query	0	Practical

Table 2: words statistics merged

Contextual	Frequency	Sense
إنزال	4	Military
إنزال	0	Operation
إنزال	0	Practical
نجاح	4	Military
نجاح	3	Operation
نجاح	0	Practical
قلب	0	Military
قلب	5	Operation
قلب	0	Practical

6. DETAILS OF THE FUZZY CLASSIFIER COMPONENT

We propose a fuzzy classifier to solve word sense disambiguation, we depend on WordNet to train the classifier with the senses, and thus, it is a knowledge-based approach. Two methods are proposed for fuzzifying the classifier: Jaccard fuzzy similarity approach and Sigmoid Fuzzy approach.

6.1. Fuzzy Jaccard-based similarity

In this method, the relationship between a term'ť' and a sense's' can be formulated as:

$$\mu\,(\mathrm{t},\mathrm{s}) = \frac{\text{total number of term frequency in this sense}}{\text{total number of term frequency in all senses}}$$

Where:

- µ(t,s): the truthfulness degree that the term 't' belongs to a specific sense 's'.
  (Note that $\mu\ (\mathbf{t},\mathbf{s})$ is calculated using WordNet examples)
- (TF): the term frequency of the context word in the training set.

For example, let us suppose that we have only four training examples as shown in Table 3; each has its sense marked and example's words frequency is listed per row.

Table 3: WordNet examples expressed as vectors of term frequencies

WordNet	Term						Sense
	t1	t2	t3	t4	t5	t6	
e1	2	1	2	0	0	0	S1
e2	3	2	0	0	0	1	S1
e3	0	0	1	2	3	0	S2
e4	0	0	0	3	1	1	S2

By calculating the term/sense probability (the truthfulness degree that the term't' belongs to specific sense), we get the statistics in Table 4.

Table 4: $\mu\ (\mathbf{t},\mathbf{s})$ calculated

	Sense	
Term	<u>S1</u>	<u>S2</u>
t1	1	0
t2	1	0
t3	0.67	0.33
t4	0	1
t5	0	1
t6	0.5	0.5

Now, suppose we have a test example that has an ambiguous word which needs to be disambiguated; then we apply the following fuzzy Jaccard-based similarity:

$$sim(test\ example, sense\ 'k') = \frac{\sum[\mu\ (\mathbf{t},\mathbf{s})]\ \text{conjunction [TF in testset]}}{\sum[\mu\ (\mathbf{t},\mathbf{s})]\ \text{disjunction [TF in testset]}}$$

Where:

- TF: denotes the term frequency in the test example.
- Conjunction and disjunction: operators will be substituted by four functions: Algebraic, Minimum-Maximum, Hamacher, and Einstein in chapter 5.

6.2. Fuzzy Classifier with a sigmoid function

The relationship between the terms in the example and sense 'y' can be expressed as a degree of memberships that formulate a fuzzy set for this sense. Thus, each sense is expressed by a fuzzy set, as follows:

$$\mathrm{FS}\ \ (\mathrm{S_y})\ = \mu\,(\mathrm{w1},\mathrm{Sy})\ +\ \mu\,(\mathrm{w2},\mathrm{Sy})\ +\ \dots\dots + \mu\,(\mathrm{wk},\mathrm{Sy})$$

Where:

- FS (Sy): the fuzzy set of sense 'y'.
- w1: the first word in the example.
- k: number of words in the example.
- µ (w1): the weight or the membership that expresses the degree of truthfulness.
- µ (w1,sy): a fuzzy logic terminology; it means: [how much the word 'w1' should be allocated to the sense 'y' for the word 'w']. it is calculated by this formula [32] [33]:

$$\mu(w1,s_y) = 0.3 + 0.7\,\frac{1}{1+e^{-2\,(TF\ in\ WordNet\ for\ this\ word\ and\ for\ this\ sense)}}$$

Figure 1 illustrates the pseudo-code used for choosing the most appropriate sense using the sigmoid fuzzy classifier.

```
Input: WordNet information, the 'n' different senses to be disambiguated along with their examples

For y= 1 to sense 'n' do
        Set membership of context 'c' for sense y 'μ(Sy)'to 0
        For i=0 to word k in context 'c'
            Set flag=Exists (training set (WordNet), $t_i$)
            If (flag=true) then
              $\mu(t_i,s_y) = 0.3 + 0.7 \frac{1}{1+e^{-2\,(\text{TF in WordNet for this word and for this sense})}}$
            $\mu(Sy) = \mu(Sy) + \mu(t_i,s_y)$
          End if
        End for
End for
```

Figure 1: The sigmoid fuzzy classifier pseudo code

7. EXPERIMENTATIONS

We specified the ambiguous words that would be marked by scanning many papers tackling the same problem; we settled on the 10 ambiguous words listed in Table 5.

Table 5: Ambiguous words studied

Ambiguous word
مارس
عملية
علم
فجر
حر
ذكر
مر
عجل
هم
مال

First we compared the two proposed fuzzy methods with no query expansion; We proposed 2 fuzziers: one depends on Jaccard similarity and the other depends on a classical sigmoid function. The objective of this experiment is to compare these 2 methods.

7.1. Experiment Setup

The Table 6 shows the characteristics of the collected corpus from Google by searching for any of the ambiguous words and fetching the corresponding sentences. The sentences were fetched programmatically using Google Search API that returns the title, sentence and URL; then we remove inappropriate cases by hand.

Table 6: Corpus characteristics

	Training	Test set
Number of examples	146	1800
Number of	10	10
Average number of	2	2
Average number of	7.3	90

Using a knowledge-based approach in this experiment, we only assume that we know the words and associated examples but the correct sense is not given. Actually, the training is done using the examples in the sense inventory that contains any of the 10 ambiguous words. We tested both Jaccard similarity and sigmoid fuzzier:

1) using Jaccard similarity: similarity between a specific example and sense 'k': $sim(test\ example, sense\ 'k') = \frac{\sum[\mu\ (\mathbf{t},\mathbf{s})]\ \text{conjunction [TF]}}{\sum[\mu\ (\mathbf{t},\mathbf{s})]\ \text{disjunction [TF]}}$ Where:

$$\mu\ (t,s) = \frac{\text{total number of term frequency in this sense}}{\text{total number of term frequency in all senses}}$$

Where:

- μ(t,s): the truthfulness degree that the term 't' belongs to specific sense.

  (Note that $\mu\ (\mathbf{t},\mathbf{s})$ is calculated using WordNet examples)

- (TF): the term frequency in the context example.

  (For example, suppose that the ambiguous word " has a context word " in the training corpus. This context word was mentioned in the training example 2 times, and then we will substitute for the TF in the equation with 2. On the other hand, this context word is mentioned 7 times with the ambiguous word examples in WordNet but only 2 of these 7 examples denote the sense 'k', then we will substitute for the $\mu\ (t,s)$ with 2/7.)

Table 7 shows the different conjunction and disjunction formulas used to measure the fuzzy Jaccard similarity. They are compared to each other in the results section.

Table 7: Conjunction and Disjunction operators

Method	Conjunction (a,b)	Disjunction (a,b)
Algebraic product	$a * b$	$a + b - a * b$
Minimum-Maximum	Min{a,b}	Max(a,b)
Hamacher Product	$\frac{a * b}{a + b - a * b}$	$\frac{a + b - 2 * a * b}{1 - a * b}$
Einstein Product	$\frac{a * b}{2 - [a + b - a * b]}$	$\frac{a * b}{1 + a * b}$

using sigmoid membership function:

$$\mu(t,sy) = 0.3 + 0.7 \frac{1}{1 + e^{-2\,(TF)}}$$

Where:

- (TF): the term frequency in WordNet examples of the ambiguous word for a specific sense 'y' where the context word is mentioned (discussed in 4.4.2).

7.2. Results and discussion

Table 8 shows the recall, precision, f-measure and micro-average for different window sizes.

Table 8: Measurements for different window sizes

	Jaccard Similarity fuzzier				Sigmoid
	Algebraic	*Mi*	*Hamacher*	*Einstein*	
Recall	0.76	0.58	0.67	0.86	0.84
Precision	0.7	0.54	0.65	0.81	0.78
f-measu	0.72	0.55	0.65	**0.83**	**0.8**
Micro-avg	0.73	0.56	0.66	0.83	0.81

You can notice that there is one Jaccard similarity fuzzier version (the Einstein fuzzier) outperforms the sigmoid fuzzier. The sigmoid fuzzier outperform the Algebraic, Hamacher, and Min-Max products. I believe there is a good reason why some Jaccard similarity measures can outperform the sigmoid fuzzier; the reason is that the average length of the context examples tested is little longer than the WordNet examples and there is a good opportunity that the surrounding words get repeated in the same context example, where the Jaccard similarity takes into account the frequency of word repetition in the context and the sigmoid fuzzier does not take into account this type of frequencies, it is only interested in the frequency of the words in the WordNet examples which are not a lot for each sense.

And we repeated the experiment but we performed query expansion and here are the results:

Table 9: Sigmoid fuzzier with different query expansion parameters

	Top 10			Top 20			Top 30		
	T	T	T	T	T	T	T	T	T
Re	0.	0.	0.	0.	0.	0.	0.	0.	0.
Pr	0.	0.	0.	0.	0.	0.	0.	0.	0.
f-	0.	**0.**	0.	0.	0.	0.	0.	0.	0.
Mi	0.	0.	0.	0.	0.	0.	0.	0.	0.

Table 10 shows the recall, precision, f-measure and micro-average of Jaccard similarity fuzzier with Einstein variation using different number of sentences and term frequency thresholds.

Table 10: Einstein Jaccard similarity fuzzier with different query expansion parameters

	Top 10 sentences			Top 20 sentences			Top 30 sentences		
	TF =2	TF =3	TF =4	TF =2	TF =3	TF =4	TF =2	TF =3	TF =4
Recall	0.86	0.88	0.85	0.88	0.86	0.85	0.84	0.82	0.82
Precision	0.83	0.83	0.85	0.77	0.81	0.82	0.73	0.74	0.68
f-measure	0.84	**0.85**	0.85	0.82	0.83	0.83	0.78	0.77	0.74
Micro-average	0.84	0.85	0.85	0.82	0.83	0.83	0.78	0.78	0.75

8. CONCLUSIONS

This paper has presented an approach for making use of fuzziers for building a better classifier that can be used for the task of Arabic Word Sense Disambiguation. The approach uses a fuzzy logic classifier which uses WordNet as a knowledge base for all the senses. The WordNet synsets have been translated into Arabic using Google APIs and Microsoft Bing APIs. The fuzzy logic proved to be an excellent classifier for dealing with the fuzzy nature of natural languages; in other words, fuzzy logic allows dealing with vagueness as long as ambiguity. Moreover, fuzzy logic takes into consideration the natural overlapping among different senses.

The main contribution of this approach can be summarized in these points:

- The fuzzy logic membership function that is used in allocating words to senses.
- Creating an Arabic sense inventory out of the English WordNet instead of depending on ArabWordNet which has very poor coverage.
- Enriching the training set derived from the knowledge base by extending the sense inventory through query expansion.

Future work will mainly focus on making a domain-based WSD. We will also investigate the possibility of producing the Arabic sense inventory with synsets tagged by domains probabilities so that other researchers could depend solely on this inventory to get the senses and theirs domains.

REFERENCES

[1] Ali Farghaly, Khaled Shaalan (2009). "Arabic Natural Language Processing: Challenges and Solutions". ACM Transactions on Asian Language Information Processing (TALIP) , Volume 8 Issue 4, Article No. 14, NY, USA.

[2] Roberto Navigli (2009). "Word sense disambiguation: A survey". ACM Computing Surveys (CSUR), Volume 41 Issue 2, Article No. 10, New York, USA.

[3] Ronen Feldman, James Sanger (2006). "Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data". Cambridge University Press, NY, USA.

[4] Mehdi Khosrow-Pour (2008). "Encyclopedia of Information Science and Technology, 2 edition". Information Science Reference - Imprint of: IGI Publishing Hershey, PA.

[5] Zhou Yao, Cao Ze-wen (2011). "Research on the Construction and Filter Method of Stop-word List in Text Preprocessing". Proceedings of the 2011 Fourth International Conference on Intelligent Computation Technology and Automation, Volume 1.

[6] Feng Zou, Fu Lee Wang, Xiaotie Deng, Song Han, Lu Sheng Wang (2006). "Automatic construction of Chinese stop word list". Proceedings of the 5th WSEAS international conference on applied computer science, Pages: 1009-1014.

[7] A. Alajmi, E. M. Saad, R. R. Darwish (2012). “Toward an ARABIC Stop-Words List Generation”. International Journal of Computer Applications (0975 – 8887), Volume 46, Number 8.

[8] A. Nwesri (2008). "Effective retrieval techniques for Arabic text". PhD Thesis, School of Computer Science and Information Technology, RMIT University.

[9] Mohamed I. Eldesouki, Waleed M. Arafa, Kareem M. Darwish (2009). "Stemming techniques of Arabic Language: Comparative Study from the Information Retrieval Perspective". The Egyptian Computer Journal, Pages: 30-49.

[10] Samhaa R. El-Beltagy, Ahmed Rafea (2011). "An accuracy-enhanced light stemmer for arabic text". ACM Transactions on Speech and Language Processing (TSLP), Volume 7, Issue 2, Article No. 2, New York, USA.

[11] Jinxi Xu, W. Bruce Croft (1998). "Corpus-based stemming using cooccurrence of word variants". ACM Transactions on Information Systems (TOIS), Volume 16, Issue 1, Pages: 61 - 81, New York, USA.

[12] Shereen Khoja, R. Garside (1999). "Stemming Arabic Text". Tech. rep. Computing Department, Lancaster University, Lancaster, U.K.

[13] K. Taghva, R. Elkhoury, J.S. Coombs (2005). "Arabic Stemming without a Root Dictionary". ITCC 1, Pages: 152-157.

[14] R. AI-Shalabi, G. Kannan, and H. AI-Serhan (2003). "New Approach for extracting Arabic roots". In Proc of 2003 International Arab conference on Information Technology, Alexandria, Pages: 42-59.

[15] D. Yarowsky (1993). “One sense per collocation”. In Proceedings of the ARPA Workshop on Human Language Technology, Princeton, Pages: 266-267.

[16] Roberto Navigli (2009). "Word sense disambiguation: A survey". Computing Surveys (CSUR), Volume 41, Issue 2, Article No. 10, New York, USA.

[17] Igor A. Bolshakov, Alexander Gelbukh (2004). "A very large dictionary with paradigmatic, syntagmatic, and paronymic links between entries". Proceedings of the Workshop on Enhancing and Using Electronic Dictionaries Publisher: Association for Computational Linguistics, Pages: 53-56, Stroudsburg, USA.

[18] Robert Krovetz, W. Bruce Croft (1992). "Lexical ambiguity and information retrieval". ACM Transactions on Information Systems (TOIS), Volume 10, Issue 2, Pages: 115 - 141, New York, USA.

[19] C. Leacock, M. Chodorow (1998). “Combining local context and WordNet sense similarity for word sense identification”. The MIT Press, Pages: 265-283.

[20] R. Beckwith, C. Fellbaum, D. Gross, G. A. Miller (1991). "WordNet: A Lexical Database Organized on Psycholinguistic Principles". Hillsdale, Erlbaum.

[21] W. J. Black, S. Elkateb (2004). "A Prototype English Arabic Dictionary Based on WordNet". Proceedings of 2nd Global WordNet Conference, GWC2004, Czech Republic, Pages: 67-74.

[22] Edda Leopold, Jörg Kindermann (2002). "Text Categorization with Support Vector Machines. How to Represent Texts in Input Space". Journal OFMachine Learning, Volume 46, Issue 1-3, Pages: 423 - 444, MA, USA.

[23] Chun-Ling Chen, Frank S. Tseng, Tyne Liang (2009). "An Integration of Fuzzy Association Rules and WordNet for Document Clustering". Proceedings of the 13th Pacific-Asia Conference on

Advances in Knowledge Discovery and Data Mining, Pages: 147-159, Berlin, Heidelberg.

[24] Goncalo Oliveira, Paulo Gomes (2011). “Automatic Discovery of Fuzzy Synsets from Dictionary Definitions”. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, Pages: 1801-1806.

[25] Aitor Almeida, Diego López-de-Ipiña (2012). "Assessing Ambiguity of Context Data in Intelligent Environments: Towards a More Reliable Context Managing System". the 5th International Symposium on Ubiquitous Computing and Ambient Intelligence.

[26] Balamurugan, Senthamarai Kannan (2010)."A Framework for Computing Linguistic Hedges in Fuzzy Queries".The International Journal of Database Management Systems, Volume 2, Number 1.

[27] Bhawani Selvaretnam, Mohammed Belkhatir (2012). "Natural language technology and query expansion: issues, state-of-the-art and perspectives". Journal of Intelligent Information Systems, Volume 38, Issue 3, Pages: 709-740, MA, USA.

[28] Zhiguo Gong, Chan Wa Cheang, U Leong Hou (2005). "Web query expansion by wordnet". Proceedings of the 16th international conference on Database and Expert Systems Applications, Pages: 166-175, Berlin, Heidelberg.

[29] J. Bhogal, A. Macfarlane, P. Smith (2007). "A review of ontology based query expansion". Journal of Information Processing and Management: an International Journal, Volume 43, Issue 4, Pages: 866-886, Tarrytown, USA.

[30] Jiuling Zhang, Beixing Deng, Xing Li (2009). "Concept Based Query Expansion Using WordNet". Proceedings of the 2009 International e-Conference on Advanced Science and Technology, Pages: 52-55, IEEE Computer Society Washington, USA.

[31] Zhiguo Gong, Chan Wa Cheang, Leong Hou U (2006). "Multi-term web query expansion using wordnet". Proceedings of the 17th international conference on Database and Expert Systems Applications, Pages: 379-388, Berlin, Heidelberg.

[32] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery (2007). "Numerical Recipes 3rd Edition: The Art of Scientific Computing, 3 edition". Cambridge University Press, New York, USA.

[33] D. Zelterman (1987). "Parameter estimation in the generalized logistic distribution". Computational Statistics & Data Analysis, Volume 5, Issue 3, Pages: 177 - 184, Amsterdam, The Netherlands.