# A Comparative Study of Rocchio Classifier Applied to supervised WSD Using Arabic Lexical Samples

Soha M. Eid[*1] Almoataz B. Al-Said[***] Nayer M. Wanas[*2]
Mohsen A. Rashwan[**] Nadia H. Hegazy[*3]

[*]*Informatics Department, Electronics Research Institute, Cairo, Egypt*
[1]Sohaeid@gmail.com
[2]nwanas@gmail.com
[3]nhegazy@mcit.gov.eg
[**]*Electronics and Electrical Communications Department, Faculty of Engineering, Cairo University, Cairo, Egypt*
mrashwan@rdi-eg.com
[***]*Cairo University, Cairo, Egypt*
moataz@cu.edu.eg

***Abstract*—This work studies the possibilities of using the Rocchio classifier to solve the Word Sense Disambiguation problem in a supervised manner through the usage of the lexical samples of five Arabic words. The performance of the Rocchio classifier is compared to three supervised machine learning algorithms, namely the Most Frequent Sense (MFS), Naïve Bayesian Classifier (NBC) and the Support Vector Machine (SVM) representing the baseline and state-of-the-art algorithms for WSD. Results indicate that the Rocchio classifier outperforms the other classification approaches by reducing the error by over 14% compared to the best performing NBC due to its superior ability in feature selection.**

## 1. Introduction

Arabic, similar to most Natural Languages, is ambiguous since many words might have multiple senses. The correct meaning of an ambiguous word depends on the context in which it occurs. The speaker of a language can usually resolve this ambiguity without difficulty. However, identification of the specific meaning of a word computationally, through Word Sense Disambiguation (WSD), is not an easy task. Although WSD may not be considered a standalone approach by itself, it is an integral part of many applications such as Machine Translation [1,2], Information Retrieval [3,4] and Information Extraction [5].

Both supervised and unsupervised learning approaches have been investigated for WSD [6]. The supervised approaches depend on the existence of previously tagged examples to associate the most appropriate senses with words in context. Since the acquisition of manually tagged data is usually expensive, unsupervised algorithms are used to learn from unlabeled data. Supervised approaches are usually preferred for their higher performance. Moreover, unsupervised approaches usually make use of various knowledge sources such as dictionaries [7].

## 2. Supervised Learning Techniques for WSD

Supervised WSD may be seen as a categorisation task, once word occurrences context are viewed as documents and word senses as categories. Supervised learning methods represent linguistic information in the form of features. Each feature informs of the occurrence of certain attribute in a context. Different supervised Machine Learning (ML) approaches have been employed in supervised WSD by training a classifier using a set of labelled instances of the ambiguous word and create a statistical model [6,8,9]. The generated model is then applied to unlabeled instances of the ambiguous word to decide their correct sense. In this work, various supervised learning techniques, which are used for solving WSD problem, are considered. These are the Most Frequent Sense baseline (MFS), Naïve Bayes Classifier (NBC) and Support Vector Machine (SVM).

### *A. Most Frequent Sense baseline (MFS)*

Most Frequent Sense (MFS) is a simple baseline classification method. As suggested by the name, it relies on estimating the most frequent sense by counting number of examples of each sense in the training set. In turn, it assigns the most frequent sense to all the examples in the test set.

### *B. Naïve Bayes Classifier*

Naïve Bayes Classifier (NBC) is a simple probabilistic classifier based in the application of Bayes' theorem [10]. In spite of the independence assumption between features, which represents the naïve part, the method compares well with other supervised methods [11,12]. Moreover, some studies uses NB classifiers as a baseline for their approaches [13]. Furthermore, Naïve Bayes method was found to be the most adequate classifier for disambiguating words having a few examples in a comparative study that considered different training conditions [14].

Using Naïve Bayes classifier for supervised WSD relies on estimating the conditional probability of each sense $S_i$ of a word *w* given a feature $f_j$ in the context. The sense with the maximum conditional probability $P(S_i|f_1, \ldots, f_m)$ is chosen as the most appropriate sense in context, where $P(S_i|f_1, \ldots, f_m)$ is given by

$$P(S_i \mid f_1, \ldots, f_m) = \frac{P(f_1, \ldots, f_m \mid S_i)\, P(S_i)}{P(f_1, \ldots, f_m)} \quad (1)$$

Based on the naïve assumption that the features are conditionally independent given the sense, we need only to maximize $P(S_i|f_1, \ldots, f_m)$ based on the following equation

$$P(S_i \mid f_1, \ldots, f_m) = P(S_i) \prod_{j=1}^{m} (P(f_j \mid S_i) \quad (2)$$

The probabilities of $P(S_i)$ and $P(f_j|S_i)$ are estimated as the relative occurrences frequencies in the training set of sense $S_i$ and the feature $f_j$ in the presence of $S_i$, respectively. To handle the cases of zero counts, where a feature ($f_j$) does not exist in a sense, the Laplace smoothing technique is used and is defined as

$$P(f_j \mid S_i) = \frac{N(f_j, S_i) + 1}{\sum_f N(f, S_i) + |V|} \quad (3)$$

Where $N(f_j,S_i)$ is occurrence frequency of feature $f_j$ in all training examples of class $S_i$ and *V* is the number of all features.

*C. Support Vector Machines*

The Support Vector Machine (SVM) is a classification technique that is based on statistical learning theory and was employed in text classification by Joachims [14]. SVM defines a decision surface that separates training data into two classes of positives and negatives. This is achieved by a hyper-plane that optimally splits the training set. The Optimal Separating Hyper-plane (OSH) is the boundary where the distance to the closest points, either positive or negative, is the maximum. To obtain the optimal separating hyper-plane, the SVM model uses a mapping function Φ that transforms the input vectors into points in new space of higher dimensions than the original one. In this new space, the learning algorithm estimates the plane with maximum margin separation between it and all training.

The training algorithm selects the points of the training examples, the support vectors, which define the separation hyper-plane, with the maximum distance between the class and its compliment.

Cabezas et al. present an SVM-based word sense tagger system which was designed to perform WSD independent of the language of lexical samples [15]. Lee et al., on the other hand, used the SVMs to perform WSD using multiple features available from different knowledge sources such as Part of Speech (POS), morphological root forms and collocations [16]. In a comparative study on the medical domain, SVM has outperformed a number of supervised learning approaches when applied to WSD in the medical domain [17].

## 3. Rocchio Classifier for Supervised WSD

Rocchio relevance feedback is a learning algorithm that was originally developed for Information Retrieval [18]. In this algorithm, the documents and request are represented by a multidimensional property vectors. These are feature vectors with weighting terms by *tf/idf*, where tf is the Term Frequency and the idf is the Inverse Document Frequency [19]. Term Frequency is the number of times a certain term *k* occurs in document d, while the Inverse Document Frequency is the number of times the term *k* occurs in the used corpus. The retrieval and/or searching process is done by matching a request vector against the set of document vectors where the cosine of two vectors is used as the matching function.

Although the Rocchio algorithm was well introduced in document classification [19,21,22], its application in WSD has not been considered [6,9]. The Rocchio classifier produces an understandable classifier in that it produces category profile interpretable by human reader. Both training and testing examples are represented as vectors of numeric weights using the *tf/idf* weighting technique. The weight vector $v^e$ of the example *e* is defined as $v^e = (v_1^e, v_2^e, \ldots\ldots v_t^e)$, where *t* is the number of indexing terms and the weight $v_k^e$ given by Equation 4

$$v_k^e = \frac{f_k^e \log(N_D / n_k)}{\sum_{j=1}^{t} f_j^e \log(N_D / n_j)} \quad (4)$$

where $N_D$ is the number of examples, $n_k$ is the number of examples in which the indexing term $k$ appears; and $f_k^e$ is given by Equation 5

$$f_k^e = \begin{cases} 0 & l=0 \\ \log(l)+1 & otherwise \end{cases} \quad (5)$$

where $l$ is the number of occurrences of term $k$ in the example *m*. In the Rocchio's classifier, each class is presented by a prototype vector $\tilde{v}^c$ which has the dimension of the original weight vectors of the example. For class *c*, the *k*th term in its prototype is defined to be

$$\tilde{v}_k^c = \max\left\{0, \frac{\beta}{|R_c|}\sum_{e\in R_c} v_k^e - \frac{\gamma}{|\overline{R}_c|}\sum_{e\in \overline{R}_c} v_k^e\right\} \quad (6)$$

where $R_c$ is the set of all examples class *c* and $\overline{R}_c$ is the set of all documents not in *c*.

The parameters $\beta$ and $\gamma$ control the relative contribution of the positive and negative training examples in the prototype vector. As recommended by Buckley et al., we used the values of 16 and 4 for $\beta$ and $\gamma$, respectively [19]. This equation shows that each prototype vector $v^c$ maximizes the mean similarity of the positive training examples with class *c* minus the mean similarity of the negative training examples with this class. Additionally, Rocchio's algorithm requires that negative elements in the class *c* prototype vector, $v^c$, be set to 0. The resulting setoff prototype vectors, one vector per class, represents the learned model. After representing the test example with weight vector *v'*, it is compared with the prototype vector $v^c$ of each class using the cosine similarity. Then, it is assigned to the class with which it has the highest cosine value.

## 4. Experimental Setup

### *A. Dataset*

The data set used is a lexical samples corpus of five Arabic nouns. Each noun has two or three senses. The lexical samples corpus is a corpus of sentences and phrases. Each of these samples represents an example of the occurrence of an ambiguous noun. The examples were collected from different Arabic resources. Most of these resources are literature publications of several authors including Al-Khansaa poetry from Algazeera and Elhawy fi Alteb from Persia, which is a publication in medicine. They represent different domains, times and geographical distributions. .

Table (I)
Statistics of the Corpus \Dataset

| Noun | Senses | Examples | Average Examples per sense | Distribution of examples per sense | | |
|---|---|---|---|---|---|---|
| | | | | Sense1 | Sense2 | Sense3 |
| السرطان | 3 | 757 | 252.3 | 193 | 338 | 226 |
| الجدول | 2 | 684 | 342 | 225 | 658 | |
| المشروع | 2 | 1184 | 592 | 416 | 768 | |
| المرسوم | 2 | 1385 | 679 | 623 | 735 | |
| الحاجب | 3 | 1139 | 379.7 | 297 | 615 | 277 |

The data was selected and annotated by a professional linguist with different number of average examples per sense (Table I). It is worth noting that not all the tagged senses are derived from an Arabic dictionary (For instance, the noun الحاجب is annotated with three senses which are a brow, a doorman, or a proper noun). Obviously, the first two senses are consistent with dictionary definitions while the third depends on a corpus usage. Moreover, not all nouns have a balanced distribution of examples. The noun المرسوم has two senses, which are something drawn and a decree, with more or less balanced data. However, the unbalanced data may be justified, since it represents the normal usage of the noun in daily life. For instance, the word السرطان has unbalanced data for its three senses; an animal, a zodiacal constellation, and a disease, where the minimum examples are provided for the first sense. The remaining two words are المشروع (something legal and a project) and الجدول (water stream and a table). For both words, the second sense has a more presence in life and larger number of examples.

### *B. Word Sense Disambiguation Approaches*

The previously described dataset is used to compare the performance of different WSD approaches. The comparison will include four supervised algorithms. The supervised algorithms are: (i) Most Frequent Sense (MFS), (ii) Naïve Bayes classifier (NBC) and (iii) Support Vector Machines (SVM) and (iv) Rocchio classifier. The first classifier (MFS) is a baseline classifier while NBC and SVM are considered as the state of the art in the supervised WSD area. However, there is no clear agreement on which one is more preferable than the other. Although, the Rocchio algorithm is a well popular in text categorization, it has not been experimented for WSD.

*C. Performance Evaluation*

The evaluation of the Rocchio classifier is carried on based on its performance compared to the other machine learning algorithms. The results reported in terms of the standard measure of accuracy. In the supervised techniques, the available data set is partitioned into training and test sets. The training set is used for learning while the test set is used for evaluation. To insure that the results are unbiased to a certain train/test split, an N-fold cross validation is used. In N-folds cross-validation, the corpora are split in N parts of similar size. A single part is used as the testing gold-standard, and the remaining *(N-1)* segments arts for training the system. The final result is the average of the *N* executions. The corpora are partitioned in a way to keep the same proportion of word senses in each of the folds. Since the available number of examples is small, the results reported are averaged over five different test/training splits that are partitioned in a way to keep the same proportion of word senses in each split. In each split, 33% of the data is used for testing.

## 5. Results and Discussion

Each classifier is tested over the five nouns. Obviously, each classifier exhibits different performance for each noun. The performance of Rocchio classifier outperforms the other classifiers for السرطان, المرسوم and الحاجب. On the other hand, the SVM demonstrates the best performance for الجدول and the NBC for المشروع. Nevertheless, all classifiers achieve their best classification accuracy in classifying the word المرسوم since it has both balanced data and the highest number of examples per sense. Generally, the Rocchio classifier has the higher performance with average accuracy of 88.3% over the five words, followed by the NBC with an average accuracy of 86.14%.

Table (II)
Comparison of Different Classifiers

| Noun | MFS | NBC | SVM | Rocchio |
|---|---|---|---|---|
| السرطان | 44.6 | 86.5 | 78 | **89** |
| الجدول | 72 | 82.7 | **85** | 84.4 |
| المشروع | 65 | **87** | 84 | 86 |
| المرسوم | 54 | 92 | 88 | **92.8** |
| الحاجب | 57 | 82.2 | 79 | **89.3** |
| Average | 57.5 | 86.14 | 82 | 88.3 |

## 6. Conclusion

This work explores the possibility of using the Rocchio classifier to solve the Word Sense Disambiguation problem. Its performance is compared to different supervised approaches. The results suggest that the Rocchio classifier is very promising as a supervised approach for WSD. The Rocchio classifier outperforms the other classification approach with an overall accuracy of 88% (86% for NBC, 82 for SVM and 57.5% for MFS) with best performance in three words (out of five) that are considered in this study. Moreover, its performance is close to the best classifier in small margin (not more than 1%) for the other two words. This high performance is mainly due to its superior ability in feature selection.

# References:


1. Carpaut, M., and Wu, D., Word Sense Disambiguation vs. Statistical Machine Translation", in *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 387-394, 2005
2. Chan, Y., Ng, H., and Chiang, D., "Word Sense Disambiguation Improves Statistical Machine Translation", in *Proc. of the 45rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 33-40, 2007
3. Schütze, H., and Pedersen, J. "Information Retrieval Based on Word Senses", in *Proc. of Symposium on Document Analysis and Information Retrieval (SDAIR'95)*, pp. 161-175, 1995
4. Stokoe, C., Oakes, M., and Tait, J. "Word Sense Disambiguation in Information Retrieval Revisited", in *Proc. of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 159-166, 2003
5. Jacquemin, B., Brun, C., and Boux, C., "Enriching a Text by Semantic Disambiguation for Information Extraction", in *Proc. of the Workshop on Using Semantics for Information Retrieval and Filtering in the 3rd International Conference in Language Resources and Evaluation (LREC)*, 2002
6. Navigli, R., "Word Sense Disambiguation: A Survey", *ACM Computing Surveys*, 41(2), pp.1-10, 2009
7. Manning, C. and Schutz, H., "Foundations of statistical Natural Language Processing", *MIT Press*, 1999
8. McCarthy, D., "Word Sense Disambiguation: An Overview", *Language and Linguistics Compass*, 3(2), pp. 537-558, 2009
9. Marquez, L., Escudero, G., Martinez, D., and Rigau, G., "Supervised Corpus Based Methods for WSD*", in Aggirre and Edmonds Ed.: Word Sense Disambiguation*, 2006
10. Duda, R. and Hart, P. "Pattern Classification and Scene Analysis", *Wiley*, 1973
11. Mooney, R. "Comparative Experiments on Disambiguating Word Senses: An illustration of the role of bias in machine learning", in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP 1996)*, pp. 82-91, 1996
12. Pedersen, T., "A Simple Approach to Building Ensembles of Naive Bayesian classifiers for Word Sense Disambiguation", in *Proc. of the North American Chapter of the Association for Computational Linguistics (NAACL 2000)*, pp. 63-69, 2000
13. Rodríguez, A., Gómez, A., Pineda, L., and Rosso, P. "A mapping between Classifiers and Training Conditions for WSD", *in Proc. of the 6th International Conference on Intellegent Text Processing and Computational Linguistics (CICLing 2005)*, pp 241-244, 2005

14. Joachims, T. “Making Large-Scale SVM Learning Practical”, In: Advance in Kernel Methods – Support Vector Learning, B. Scholkopf, C. Burges, and A. Smola (ed.), *MIT Press*, 1999
15. Cabezas, C., Resnik, P., and Stevens, J. “Supervised Sense Tagging using Support Vector Machines.”, in *Proc. of SENSEVAL-2, Second Edition Workshop on Evaluating Word Sense Disambiguation Systems*, pp 59-62, 2001
16. Lee, Y., Ng, H., and Chia, T., “Supervised Word Sense Disambiguation with Support Vector Machines and Multiple Knowledge Source”, in Proc. *of SENSEVAL-3, Third International Workshop on the Evaluation Systems for the Semantic Analysis of Text*, 2004
17. Joshi, M., Pedersen, T., and Maclin, R. “A Comparative Study of Support Vector Machines applied to Word Sense Disambiguation Problem in the Medical Domain”, in *Proc. of the 2nd Indian International Conference on Artificial Intelligence (IICAI 2005)*, 2005
18. Rocchio, J., “Relevance Feedback in Information Retrieval”, in Salton: The SMART Retrieval System: Experiments in Automatic Processing, Chapter 14, pp 313-323, *Prentice-Hall*, 1971
19. Buckley, C., Salton, G., and Allan, j. “The Effect of Adding Relevance Information in a Relevance Feedback Environment”, in *Proc.International ACM SIGIR Conference*, pp 292-300, 1994
20. Itner, D., lewis, D., Ahn, D., “Text Categorization of Low Quality Images”, *in Proc. of 4th Annual Symposium on Document Analysis and Information Retrieval (SDAIR 1995)*, pp. 301-315, 1995
21. Cohen, W. and Singer, Y. “Context-Sensitive Learning Methods for Text Categorization”, *ACM Transaction on Information Systems*, Vol. 17, No. 2, pp. 141-173, 1999
22. Sebastiani, F., “A Tutorial on Automated Text Categorization”, in *Proc. of Argentine Symposium on Artificial Intelligence*, 1999