Word Sense Ambiguity: A Survey

Mohd Shahid Husain

Department of IT,
Integral University,
Lucknow, India
Email: Mshahidhusain {at} ieee.org

Mohd Rizwan Beg

Department of IT,
Integral University,
Lucknow, India

Abstract—***In natural language processing (NLP), word sense disambiguation (WSD) is defined as the task of assigning the appropriate meaning (sense) to a given word in a text or discourse. Natural language is ambiguous, so that many words can be interpreted in multiple ways depending on the context in which they occur. The computational identification of meaning for words in context is called word sense disambiguation (WSD). In this paper we will discuss about the ambiguity and some important measures to deal with ambiguous statements.***

Keywords- WSD; Ambiguity; Word Discrimination; Supervised Ambiguity; Unsupervised Ambiguity

I. INTRODUCTION

Word sense ambiguity is a vital hitch for many established Human Language Technology applications. This provides motivation to many international research groups for working on WSD, applying a wide range of approaches. On the other hand, to date, no large-scale, broad-coverage, accurate WSD system has been built [1]. With current state-of-the-art accuracy in the range 60–70%, WSD is one of the most significant open problems in NLP. Even though most of the approaches / techniques for WSD more often than not are presented as stand-alone techniques, it is our conviction that full-fledged lexical ambiguity resolution will require to integrate several information sources and techniques.

Word sense ambiguity means a single word or sentence is interpreted differently by different users. The main reason for this is that a single
word has more than one meaning (exact meaning depends on the context).

As an example, consider the following three sentences:

1. Many cruise missiles have fallen on Baghdad.
2. Music sales will fall by up to 15% this year.
3. U.S. officials expected Basra to fall early.

Any system that tries to find out the meanings of the three sentences will need to represent somehow three different senses for the verb "fall". In the first sentence, the missiles have been launched on Baghdad. In the second sentence, the sales will decrease, and in the third the city will surrender early. WordNet 2.0 (Miller, 1995; Fellbaum, 1998) contains *thirty-two different senses for the verb fall as well as twelve different senses for the noun fall.* Note also that the first and third sentence belong to the same, military domain, but use the verb fall with two different meanings. Thus, a WSD system must be able to assign the correct sense of a given word, in these examples, fall, depending on the context in which the word occurs. In the example sentences, these are, respectively, senses 1, 2 and 9, as listed below.

1. Fall—"descend in free fall under the influence of gravity" ("The branch fell from the tree"; "The unfortunate hiker fell into a crevasse").
2. Descend, fall, go down, come down—"move downward but not necessarily all the way" ("The temperature is going down"; "The barometer is falling"; "Real estate prices are coming down").
9. Fall—"be captured" ("The cities fell to the enemy").

Providing innovative technology to solve this problem will be one of the main challenges in language engineering to access advanced knowledge technology systems.

A. Some Basic Types of Ambiguity

i. *Lexical ambiguity* occurs when a word has several meanings. For example, bank

ii. *Syntactic ambiguity*, also called *structural ambiguity*, occurs when a given sequence of words can be given more than one grammatical structure, and

each has a different meaning. In the terminology of compiler construction, syntactic ambiguity occurs when a sentence has more than one parse. For example

-- The Tibetan history teacher can be read as The (Tibetan history) teacher or The Tibetan (history teacher). *[Analytical ambiguity]*

--The police shot the rioters with guns. *[Attachment ambiguity]*

--I saw Peter and Paul and Mary saw me. *[Coordination ambiguity]*

--Perot knows a richer man than Trump. *[Elliptical ambiguity]*

iii. *Semantic ambiguity* occurs when a sentence has more than one way of reading it within its context although it contains no lexical or structural ambiguity. For example, all linguists prefer a theory.

iv. *Pragmatic ambiguity* occurs when a sentence has several meanings in the context in which it is uttered.
For example, every student thinks she is a genius.

v. *Vagueness* For example, fast response time, is often vague, because there is no precise way of describing and measuring it.

B. Problem Description

Word sense disambiguation is the ability to computationally determine which sense of a word (having multiple meanings) is activated by its use in a particular context. WSD is usually performed on one or more texts (although in principle bags of words, i.e., collections of naturally occurring words). If we disregard the punctuation, we can view a text *T* as a sequence of words (*w*1, *w*2, . . . , *wn*), and we can formally express WSD as the task of assigning the appropriate sense(s) to all or some of the words in *T*, that is, to make out a mapping *A* from words to senses, such that *A*(*i*) ⊆ *Senses D*(*wi*), where *Senses D*(*w*i) is the set of senses encoded in a dictionary D for word wi and A(i) is that subset of the senses of wi which are appropriate in the context T. The mapping A can assign more than one sense to each word wi ∈ T, although typically only the most appropriate sense is selected, that is, | A(i) |= 1.

II. APPROACHES TO WSD

Main approaches to WSD can be categorized as:

i. *Supervised WSD:* these approaches use machine learning techniques to learn a classifier from labelled training sets, that is, sets of examples encoded in terms of a number of features together with their appropriate sense label (or class).

ii. *Unsupervised WSD:* these methods are based on unlabeled corpora, and do not exploit any manually sense-tagged corpus to provide a sense choice for a word in context.

Another criterion for categorization of approaches is use of knowledge (dictionary):

i. *Knowledge-based (knowledge-rich or dictionary based):* rely on the use of external lexical resources, such as machine-readable dictionaries, thesauri, ontologies, etc.

ii. *Corpus-based (or knowledge-poor):* this approach does not make use of any of these resources for disambiguation.

Finally, we can categorize WSD approaches as token-based and type-based. **Token based approaches** associate a specific meaning with each occurrence of a word depending on the context in which it appears. In contrast**, type-based disambiguation** is based on the assumption that a word is consensually referred with the same sense within a single text. Consequently, these methods tend to infer a sense (called the predominant sense) for a word from the analysis of the entire text and possibly assign it to each occurrence within the text.

A. Supervised Disambiguation

Supervised WSD uses machine-learning techniques for inducing a classifier from manually sense-annotated data sets. Usually, the classifier (often called *word expert*) is concerned with a single word and performs a classification task in order to assign the appropriate sense to each instance of that word. The training set

used to learn the classifier typically contains a set of examples in which a given target word is manually tagged with a sense from the sense inventory of a reference dictionary.

A.1.Decision Lists

A *decision list* [2] is an ordered set of rules for categorizing test instances (assigning the appropriate sense to a target word). It can be seen as a list of weighted "if-then-else" rules. A training set is used for inducing a set of features. As a result, rules of the kind (*feature-value*, *sense*, *score*) are created. The ordering of these rules, based on their decreasing score, constitutes the decision list.
Given a word occurrence *w* and its representation as a feature vector, the decision list is checked, and the feature with maximum score that matches the input vector selects the word sense to be assigned:

S= $\text{argmax}_{Si \in SensesD\,(w)}$ score (Si).

According to Yarowsky [3], the score of sense *Si* is calculated as the maximum among the feature scores, where the score of a feature *f* is computed as the logarithm of the probability of sense *Si* given feature *f* divided by the sum of the probabilities of the other senses given feature *f:*

$$score(S_i) = \max_f \log\left(\frac{P(S_i \mid f)}{\sum_{j \neq i} P(S_j \mid f)}\right)$$

A.2. Decision Trees

A *decision tree* is a predictive model used to represent classification rules with a tree structure that recursively partitions the training data set. Each internal node of a decision tree represents a test on a feature value, and each branch represents an outcome of the test. A prediction is made when a terminal node (i.e., a leaf) is reached.

A.3. Naive Bayes

A *Naive Bayes classifier* is a simple probabilistic classifier based on the application of Bayes' theorem. It relies on the calculation of the conditional probability of each sense *Si* of a word *w* given the features *f j* in the context. The sense *ˆS* which maximizes the following formula is chosen as the most appropriate sense in context:

$$\hat{S} = \underset{S_i \in Senses_D(w)}{\text{argmax}}\ P(S_i \mid f_1, \ldots, f_m) = \underset{S_i \in Senses_D(w)}{\text{argmax}}\ \frac{P(f_1, \ldots, f_m \mid S_i) P(S_i)}{P(f_1, \ldots, f_m)}$$

$$= \underset{S_i \in Senses_D(w)}{\text{argmax}}\ P(S_i) \prod_{j=1}^{m} P(f_j \mid S_i),$$

Where *m* is the number of features, and the last formula is obtained based on the naive assumption that the features are conditionally independent given the sense.

A.4. Neural Networks

We can implement neural networks to represent words as nodes: the words activate the concepts to which they are semantically related and vice versa. The activation of a node causes the activation of nodes to which it is connected by excitory links and the deactivation of those to which it is connected by inhibitory links (i.e., competing senses of the same word).

A.5. Exemplar-Based or Instance-Based Learning

Exemplar-based (or *instance-based*, or *memory-based*) *learning (for example KNN- one of the highest performing methods in WSD)* is a supervised algorithm in which the classification model is built from examples. The model retains examples in memory as points in the feature space and, as new examples are subjected to classification, they are progressively added to the model.

A.6. Support Vector Machines (SVM)

This method is based on the idea of learning a linear hyper plane from the training set that separates positive examples from negative examples. The hyper plane is located in that point of the hyperspace which maximizes the distance to the closest positive and negative examples (called *support vectors*). In other words, *support vector machines* (SVMs) tend at the same time to minimize the empirical classification error and maximize the geometric margin between positive and negative examples.

A.7. Ensemble Methods

Sometimes different classifiers are available which we want to combine to improve the overall disambiguation accuracy. Combination strategies—called *ensemble methods*—typically put together learning algorithms of different nature, that is, with significantly different characteristics.

Ensemble methods are becoming more and more popular as they allow one to overcome the weaknesses of single supervised approaches.

i. *Majority Voting:* Given a target word *w*, each ensemble component can give one vote for a sense of *w*. The sense ˆ*S* which has the majority of votes is selected. In case of tie, a random choice can be made among the senses with a majority vote.

ii. *Probability Mixture:* Supposing the first-order classifiers provide a confidence score for each sense of a target word *w*, we can normalize and convert these scores to a probability distribution over the senses of *w*. These probabilities (i.e., normalized scores) are summed, and the sense with the highest overall score is chosen.

iii. *Rank-Based Combination:* Supposing that each first-order classifier provides a ranking of the senses for a given target word *w*, the rank-based combination consists in choosing that sense ˆ*S* of *w* which maximizes the sum of its ranks output by the systems*C*1, . . . , *Cm* (negate ranks so that the best ranking sense provides the highest contribution).

iv. *AdaBoost:* AdaBoost or adaptive boosting is a general method for constructing a "strong" classifier as a linear combination of several "weak" classifiers. The method is adaptive in the sense that it tunes subsequent classifiers in favour of those instances misclassified by previous classifiers. AdaBoost learns the classifiers from a weighted training set (initially, all the instances in the data set are equally weighted). The algorithm performs m iterations, one for each classifier. At each iteration, the weights of incorrectly classified examples are increased, so as to cause subsequent classifiers to focus on those examples (thus reducing the overall classification error).

B. UNSUPERVISED DISAMBIGUATION

Unsupervised methods have the potential to overcome the knowledge acquisition bottleneck, that is, the lack of large-scale resources manually annotated with word senses. These approaches to WSD are based on the idea that the same sense of a word will have similar neighbouring words. They are able to induce word senses from input text by clustering word occurrences, and then classifying new occurrences into the induced clusters. They do not rely on labelled training text and, in their purest version, do not make use of any machine–readable resources like dictionaries, thesauri, ontologies, etc. However, the main disadvantage of fully unsupervised systems is that, as they do not exploit any dictionary, they cannot rely on a shared reference inventory of senses.

The main approaches to unsupervised WSD are namely: methods based on context clustering, word clustering, and cooccurrence graphs.

B.1. Context Clustering

In this approach each occurrence of a target word in a corpus is represented as a context vector. The vectors are then clustered into groups, each identifying a sense of the target word.

This approach is based on the idea of *word space*, that is, a vector space whose dimensions are words. A word *w* in a corpus can be represented as a vector whose j^{th} component counts the number of times that word w_j cooccurs with *w* within a fixed context (a sentence or a larger context). A context vector is built as the centroid (i.e., the normalized average) of the vectors of the words occurring in the target context, which can be seen as an approximation of its semantic context. Finally, sense discrimination can be performed by grouping the context vectors of a target word using a clustering algorithm. Schutze proposed an algorithm, called *context-group discrimination*, which groups the occurrences of an ambiguous word into clusters of senses, based on the contextual similarity between occurrences. Clustering is performed with the Expectation Maximization algorithm, an iterative maximum likelihood estimation procedure of a probabilistic model. A different clustering approach consists of agglomerative clustering [Pedersen and Bruce]. Initially, each instance constitutes a singleton cluster. Next, agglomerative clustering merges the most similar pair of clusters, and continues with successively less similar pairs until a stopping threshold is reached.

B.2. Word Clustering

Methods which aim at clustering words which are semantically similar and can thus convey a specific meaning.

A well-known approach to word clustering [Lin] consists of the identification of words *W* = (*w*1, . . . , *wk*) similar (possibly synonymous) to a target word *w*0. The similarity between w_0 and w_i is determined based on the information content of their single features, given by the syntactic dependencies which occur in a corpus (such as, e.g., subject-verb, verb-object, adjective-noun, etc.). The more dependencies

the two words share, the higher the information content. To discriminate between the senses, a word clustering algorithm is applied. Let W be the list of similar words ordered by degree of similarity to w0. A similarity tree T is initially created which consists of a single node w0. Next, for each i ∈ {1, . . . , k}, $w_i \in W$ is added as a child of w_j in the tree T such that wj is the most similar word to wi among $\{w_0, \ldots, w_i-1\}$. After a pruning step, each subtree rooted at w_0 is considered as a distinct sense of w_0.

In a subsequent approach, called the *clustering by committee (CBC)* algorithm [Lin and Pantel], a different word clustering method was proposed. For each target word, a set of similar words was computed as above. To calculate the similarity, again, each word is represented as a feature vector, where each feature is the expression of a syntactic context in which the word occurs. Given a set of target words (e.g., all those occurring in a corpus), a similarity matrix *S* is built such that $S_{i\,j}$ contains the pair-wise similarity between words w_i and w_j.
As a second step, given a set of words *E*, a recursive procedure is applied to determine sets of clusters, called *committees*, of the words in *E*. To this end, a standard clustering technique, that is, average-link clustering is employed. In each step, residue words not covered by any committee (i.e., not similar enough to the centroid of each committee) are identified. Recursive attempts are made to discover more committees from residue words. Notice that, as above, committees conflate senses as each word belongs to a single committee.
Finally, as a sense discrimination step, each target word $w \in E$, again represented as a feature vector, is iteratively assigned to its most similar cluster, based on its similarity to the centroid of each committee. After a word *w* is assigned to a committee *c*, the intersecting features between *w* and elements in *c* are removed from the representation of *w*, so as to allow for the identification of less frequent senses of the same word at a later iteration.

B.3. Cooccurrence Graphs

These approaches are based on the notion of a cooccurrence graph, that is, a graph G = (V, E) whose vertices V correspond to words in a text and edges E connect pairs of words which cooccur in a syntactic relation, in the same paragraph, or in a larger context.
Given a target ambiguous word *w*, a local graph G_w is built around *w*. By normalizing the adjacency matrix associated with G_w, we can interpret the graph as a Markov chain. The Markov clustering algorithm [van Dongen] is then applied to determine word senses, based on an expansion and an inflation step, aiming, respectively, at inspecting new more distant neighbours and supporting more popular nodes.
Subsequently, V´eronis proposed an ad hoc approach called ***HyperLex***. First, a cooccurrence graph is built such that nodes are words occurring in the paragraphs of a text corpus in which a target word occurs, and an edge between a pair of words is added to the graph if they cooccur in the same paragraph. Each edge is assigned a weight according to the relative cooccurrence frequency of the two words connected by the edge.
Formally, given an edge $\{i, j\}$ its weight $_{wi\,j}$ is given by $w_{i\,j} = 1 - max\{P(w_i \mid w_j), P(w_j \mid w_i)\}$,

where $P(w_i \mid w_j) = \frac{freq_{ij}}{freq_j}$,

and $freq_{i\,j}$ is the frequency of cooccurrence of words w_i and w_j and $freq_j$ is the frequency of w_j within the text. As a result, words with high frequency of cooccurrence are assigned a weight close to zero, whereas words which rarely occur together receive weights close to 1. Edges with a weight above a certain threshold are discarded.
As a second step, an iterative algorithm is applied to the cooccurrence graph: at each iteration, the node with highest relative degree in the graph is selected as a *hub* (based on the experimental finding that a node's degree and its frequency in the original text are highly correlated). As a result, all its neighbours are no longer eligible as hub candidates. The algorithm stops when the relative frequency of the word corresponding to the selected hub is below a fixed threshold. The entire set of hubs selected is said to represent the senses of the word of interest. Hubs are then linked to the target word with zero-weight edges and the minimum spanning tree (MST) of the entire graph is calculated. Finally, the MST is used to disambiguate specific instances of our target word.

An alternative graph-based algorithm for inducing word senses is ***PageRank.*** PageRank is a well-known algorithm developed for computing the ranking of web pages, and is the main ingredient of the Google search engine. In its weighted formulation, the PageRank degree of a vertex $v_i \in V$ is given by the following formula:

$$P(v_i) = (1-d) + d \sum_{v_j \to v_i} \frac{w_{ji}}{\sum_{v_j \to v_k} w_{jk}} P(v_j),$$

Where $v_j \to v_i$ denotes the existence of an edge from v_j to v_i, w_{ji} is its weight, and d is a damping factor (usually set to 0.85) which models the probability of following a link to *vi* (second term) or randomly jumping to *vi* (first term in the equation). The PageRank of each vertex is iteratively computed until convergence.

In the adaptation of PageRank to unsupervised WSD (due to Agirre et al.), w_{ji} is, as for HyperLex, a function of the probability of cooccurrence of words *wi* and *wj*. As a result of a run of the PageRank algorithm, the vertices are sorted by their PageRank value, and the best ranking ones are chosen as hubs of the target word.

C. Knowledge-Based Disambiguation

The objective of knowledge-based or dictionary-based WSD is to exploit knowledge resources (such as dictionaries, thesauri, ontologies, collocations, etc. to infer the senses of words in context. These methods usually have lower performance than their supervised alternatives, but they have the advantage of a wider coverage, thanks to the use of large-scale knowledge resources.

C.1. Overlap of Sense Definitions

A simple and intuitive knowledge-based approach relies on the calculation of the word overlap between the sense definitions of two or more target words. This approach is named *gloss overlap* or the *Lesk* algorithm after its author [Lesk]. Given a two word context (*w*1, *w*2), the senses of the target words whose definitions have the highest overlap (i.e., words in common) are assumed to be the correct ones. Formally, given two words *w*1 and *w*2, the following score is computed for each pair of word senses
*S*1 ∈ *Senses* (*w*1) and *S*2 ∈ *Senses* (*w*2):
scoreLesk (*S*1, *S*2) =| *gloss* (*S*1) ∩ *gloss*(*S*2) |,
Where *gloss* (*Si*) is the bag of words in the textual definition of sense *Si* of w_i. The senses which maximize the above formula are assigned to the respective words. However, this requires the calculation of | *Senses* (*w*1) | · | *Senses*(*w*2) | gloss overlaps.

Given the exponential number of steps required, a variant of the Lesk algorithm is currently employed which identifies the sense of a word *w* whose textual definition has the highest overlap with the words in the context of w. Formally, given a target word w, the following score is computed for each sense S of w:
scoreLeskVar (S) =| context(w) ∩ gloss(S) |,
Where context (w) is the bag of all content words in a context window around the target word w.

The original method achieved 50–70% accuracy (depending on the word), using a relatively fine set of sense distinctions such as those found in a typical learner's dictionary Unfortunately, Lesk's approach is very sensitive to the exact wording of definitions, so the absence of a certain word can radically change the results. Further, the algorithm determines overlaps only among the glosses of the senses being considered. This is a significant limitation in that dictionary glosses tend to be fairly short and do not provide sufficient vocabulary to relate fine-grained sense distinctions.

C.2. Selectional Preferences

A historical type of knowledge-based algorithm is one which exploits Selectional preferences to restrict the number of meanings of a target word occurring in context. Selectional preferences or restrictions are constraints on the semantic type that a word sense imposes on the words with which it combines in sentences (usually through grammatical relationships). For instance, the verb eat expects an animate entity as subject and an edible entity as its direct object. We can distinguish between selectional restrictions and preferences in that the former rule out senses that violate the constraint, whereas the latter (more typical of recent empirical work) tend to select those senses which better satisfy the requirements.

The easiest way to learn selectional preferences is to determine the semantic appropriateness of the association provided by a word-to-word relation. The simplest measure of this kind is frequency count. Given a pair of words w1 and w2 and a syntactic relation R (e.g., subject-verb, verb-object, etc.), this method counts the number of instances (R, w1, w2) in a corpus of parsed text, obtaining a figure Count(R, w1, w2). Another estimation of the semantic appropriateness of a word-to-word relation is the conditional probability.

C.3. 2.3.3. Structural Approaches

Since the availability of computational lexicons like WordNet, a number of structural approaches have been developed to analyze and exploit the structure of the concept network made available in such lexicons. The

recognition and measurement of patterns, both in a local and a global context, can be collocated in the field of structural pattern recognition, which aims at classifying data (specifically, senses) based on the structural interrelationships of features. Two main approaches of this kind: similarity-based and graph-based methods.

III. EVALUATION MEASURES

Let T = (w1, . . . , wn) be a test set and A an "answer" function that associates with each word wi ∈ T the appropriate set of senses from the dictionary D (i.e., A(i) ⊆ SensesD(wi)). Then, given the sense assignments A'(i) ∈ SensesD(wi) ∪ {e} provided by an automatic WSD system (i ∈ {1, . . . , n}), we can define

i. *coverage C* as the percentage of items in the test set for which the system provided a sense assignment that is:

C = # answers provided / # total answers to provide,

ii. *precision P* of a system is computed as the percentage of correct answers given by the automatic system, that is:

P = # correct answers provided / # answers provided.

Precision determines how good are the answers given by the system being assessed.

iii. *Recall R* is defined as the number of correct answers given by the automatic system over the total number of answers to be given:

R = # correct answers provided / # total answers to provide

According to the above definitions, we have that R ≤ P. When coverage is 100%, we have that P = R.
In the WSD literature, recall is also referred to as accuracy.

Finally, a measure which determines the weighted harmonic mean of precision and recall, called the

iv. F1-measure or balanced F-score, is defined as F1 = 2PR / (P + R)

IV. CONCLUSION

In this survey paper, we have discussed the ambiguity, various methods used to deal with ambiguity and evaluation methods used to measure the effectiveness of these methods. By this survey work, one can conclude following:

i. Token-based approaches can always be adapted to perform in a type-based fashion by assigning the majority sense throughout the text to each occurrence of a word.
ii. Generally, supervised approaches to WSD have obtained better results than unsupervised methods.
iii. In several studies, neural networks have been shown to perform well compared to other supervised methods. However, these experiments are often performed on a small number of words. As major drawbacks of neural networks we cite the difficulty in interpreting the results, the need for a large quantity of training data, and the tuning of parameters such as thresholds, decay, etc.
iv. As SVM is a binary classifier, in order to be usable for WSD it must be adapted to multiclass classification (i.e., the senses of a target word). A simple possibility, for instance, is to reduce the multiclass classification problem to a number of binary classifications of the kind sense Si versus all other senses. As a result, the sense with the highest confidence is selected.

The methods discussed are generally used to deal with ambiguity; however, any method alone is not very efficient, so blend of these methods can be used to improve the accuracy of the disambiguation.

REFERENCES

[1] Benjamin Snyder and Martha Palmer. "The English all-words task". In Proceedings of SENSEVAL-3, pages 41–43, 2004.
[2] Ronald L. Rivest "Learning Decision Lists" August 2001. http://people.csail.mit.edu/rivest/Rivest-DecisionLists.pdf.
[3] David Yarowsky "Decision Lists for Lexical Ambiguity Resolution: Application to accent restoration in Spanish and French" http://acl.ldc.upenn.edu/P/P94/P94-1013.pdf.
[4] Arindam Chatterjee, Salil Joshi, Pushpak Bhattacharyya, Diptesh Kanojia, and Akhlesh Meena. "A study of the sense annotation process: Man v/s machine." International Conference on Global Wordnets, January. 2012.
[5] Lee Yoong K. and Chia. "Supervised word sense disambiguation with support vector machines and multiple knowledge sources." In Proceedings of Senseval-3: Third

International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, 2004, pages 137–140.

[6] Roberto Navigli. "Word sense disambiguation: A survey. In ACM Computing Surveys, . February 2009. Vol. 41, No. 2, Article 10.

[7] AGIRRE, E. AND EDMONDS, P., Eds. "Word Sense Disambiguation: Algorithms and Applications." Springer, New York, NY. 2006.

[8] NIU, C., LI, W., SRIHARI, R., AND LI, H. :Word independent context pair classification model for word sense disambiguation." In Proceedings of the 9th Conference on Computational Natural Language Learning(CoNLL, Ann Arbor, MI). . 2005.

[9] AGIRRE, E. AND MARTINEZ, D." Exploring automatic word sense disambiguation with decision lists and the web." In Proceedings of the 18th International Conference on Computational Linguistics (COLING, Saarbr ¨ ucken, Germany). 11–19. 2000

[10] MARQUEZ, L., ESCUDERO, G., MART`INEZ, D., AND RIGAU, G. "Supervised corpus-based methods for WSD." In Word Sense Disambiguation: Algorithms and Applications, E. Agirre and P. Edmonds, Eds. Springer, New York, NY, 167–216. 2006.

[11]MARTINEZ, D. "Supervised word sense disambiguation: Facing current challenges, Ph.D. dissertation." University of the Basque Country, Spain. 2004.

[12] M. Kim and E. Hovy. "Determining the sentiment of opinions." In *(COLING 2004)*, pages 1267– 1373, Geneva, Switzerland. 2004.

[13] Snow, S. Prakash, D. Jurafsky, and A. Ng. "Learning to merge word senses." In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic. 2007.

[14] Su and K. Markert. 2008. "From word to sense: a case study of subjectivity recognition." In (COLING- 2008), Manchester.

[15] Wiebe and R. Mihalcea. "Word sense and subjectivity." In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia. 2006.

[16] A. Azzini, C. da Costa Pereira, M. Dragoni, and A. G. B. Tettamanzi," Evolving Neural Networks for Word Sense Disambiguation", Eighth International Conference on Hybrid Intelligent Systems, 2008.

[17] Dan Klein, Kristina Toutanova, H. Tolga Ilhan, Sepandar D. Kamvar and Christopher D. Manning," Combining Heterogeneous Classifiers forWord-Sense Disambiguation", Computer Science Department Stanford University, 2002.

[18] Rada Mihalcea," Word Sense Disambiguation", the 18th European Summer School in Logic, Language and Information 31 July - 11 August, 2006.

[19] Yee Seng Chan and Hwee Tou Ng, David Chiang,"Word Sense Disambiguation Improves Statistical Machine Translation", Department of Computer Science National University of Singapore, 2007.

[20] Andres Montoyo, Armando Su´arez, German Rigau,"Combining Knowledge- and Corpus-based Word-Sense-Disambiguation Methods", Journal of Artificial Intelligence Research ,2005.

[21] Antonio J Jimeno-Yepes, Alan R Aronson, "Knowledge-based biomedical word sense disambiguation: comparison of approaches", Jimeno-Yepes and Aronson BMC Bioinformatics 2010.

[22] P.Tamilselvi , S.K.Srivatsa ," Case Based Word Sense Disambiguation Using Optimal Features", IPCSIT vol.16, IACSIT Press, Singapore,2011.

[23] David Martinez, Oier Lopez de Lacalle, Eneko Agirre," On the Use of Automatically Acquired Examples for All-Nouns Word Sense Disambiguation", University of the Basque Country 20018, Journal of Artificial Intelligence Research 33,79-107,2008.

[24] M. Nameh, S.M. Fakhrahmad, M. Zolghadri Jahromi," A New Approach to Word Sense Disambiguation Based on Context Similarity", Proceedings of the World Congress on Engineering 2011 Vol I, July 6 - 8, 2011.

[25] A. R. Rezapour, S. M. Fakhrahmad and M. H. Sadreddini ," Applying Weighted KNN to Word Sense Disambiguation", Proceedings of the World Congress on Engineering 2011 Vol III WCE 2011, July 6 - 8, 2011.

[26] Sch¨utze, H.: "Automatic word sense discrimination," *Computational Linguistics*, **24**(1), 1998, 97–123.

[27] Pedersen, T.: "Unsupervised corpus-based methods for WSD," Word Sense Disambiguation. Algorithms and Applications. Edited by E. Agirre and P. Edmonds, Springer, the Netherlands, 2006.