AN EMPIRICAL STUDY OF WORD SENSE DISAMBIGUATION

M.Srinivas[1] and Dr. B.Padmaja Rani [2]

[1]Department of Computer Science & Engineering, Geethanjali College of Engineering & Technology, Hyderabad, India
[2]Department of Computer Science & Engineering, JNTU-CEH, Hyderabad, India

ABSTRACT

Word Sense Disambiguation (WSD) is an important area which has an impact on improving the performance of applications of computational linguistics such as machine translation, information retrieval, text summarization, question answering systems, etc. We have presented a brief history of WSD, discussed the Supervised, Unsupervised, and Knowledge-based approaches for WSD. Though many WSD algorithms exist, we have considered optimal and portable WSD algorithms as most appropriate since they can be embedded easily in applications of computational linguistics. This paper will also provide an idea of some of the WSD algorithms and their performances, which compares and assess the need of the word sense disambiguation.

KEYWORDS

Supervised, unsupervised, knowledge-based, WordNet, word sense disambiguation

1. INTRODUCTION

The creatures are using the language as their communication media. Through language information can be exchanged among the races. Verbal communication involves alphabets, words, sentences etc. In almost all natural languages, there are words having different meanings depending on the context. Those words are known as polysemous words making verbal communication ambiguous. Fortunately, human beings resolve the ambiguity instantly depending on the context with lot of ease. But, machines find it as a very difficult problem. This involves processing unstructured textual information to build the appropriate data structures. We determine the most appropriate meaning through analysing the data structures thoroughly. This is known as Word Sense Disambiguation, a common problem in Natural Language Processing (NLP).
This paper is organized as follows: In section II a brief description of WSD, presenting main approaches knowledge based, supervised, and unsupervised disambiguation in sections III, IV, V respectively. In section VI we elaborated on some evaluation measures for assessing WSD systems and conclusion in section VII followed by the references.

2. WORD SENSE DISAMBIGUATION

Word sense Disambiguation is the process of identifying the correct sense of a word that has several meanings in the context in a computational paradigm. Machine translation is one of the most former on growing research topic in computational linguistics. The problem WSD is as complex as most of the difficult problems in Artificial Intelligence and hence it is deemed as an AI complete problem. In 1940's WSD was developed as discrete field in computational

linguistics to help the research in machine translation. In 1950's Weaver identified that context is crucial and hence, statistical semantic studies have been undertaken as a necessary primary step. The automatic disambiguation of word senses has been given an utmost priority from the earliest days of computer treatment of languages in the 1950's. WSD depends heavily on knowledge sources like dictionaries, thesauri, ontology's, collocations, WordNet etc. WSD can be described as a method of providing the most appropriate sense to all or some words in the text where T is a sequence of words (w_1, w_2, ..., w_n). It is called as All-words WSD when it attempts to disambiguate all words in a text such as nouns verbs, adjectives, adverbs¸ etc. Otherwise Targeted WSD as it disambiguates some restricted words only. It consists of mainly discovering the mapping M from words to senses such that $M(k) \subseteq Senses_D(w_k)$ where M(k) is the subset of senses of w_k which are appropriate in the text T and $Senses_D(w_k)$ is the set of senses in dictionary D for word w_k. The mapping M can assign multiple senses to w_k belonging to T but finally the most appropriate sense is selected. Hence, WSD can be seen as a classification task where word senses form the classes and a method classifies each occurrence of the word to multiple classes by exploiting information available from the context and external knowledge sources such as dictionary, thesauri, ontology's, collocations, wordnet, unlabelled or annotated sense corpora.
The input text is pre processed to build a structured format suitable for our WSD system. It consists of the following steps in sequence:

a. Tokenization - Dividing the text into basic units (tokens) called as words.
b. Part -of-Speech Tagging - Determining the appropriate grammatical category for each word.
c. Lemmatization - Performs morphological analysis to provide the root words.
d. Chunking - Partitioning the text in syntactically correlated parts.
e. Parsing- Provides the parse tree of sentence structure.
Following the above pre processing, each word is represented as a feature vector making the assignment of the appropriate sense easy by the WSD system.

3. KNOWLEDGE BASED APPROACHES

The objective of knowledge based approaches is to make extensive use of knowledge resources like WordNet to determine the senses of words in context. These methods have lower performance than their counterpart supervised methods, but they can be applicable to a wider range. The knowledge based approaches to word sense disambiguation started initially in the year 1979 and 1980, conducting experiments on extremely limited domains. The unavailability of large-scale computational resources restrained scaling up these works due to difficulty in performing proper evaluation, comparison and exploration of these methods especially when applied to end-to-end applications.

We present the following essential knowledge-based techniques: the overlap of sense definitions, selectional restrictions, and structural approaches. Many of these exploit information from WordNet or other resources.

3.1. Overlap of Sense Definitions

It is a straightforward knowledge based approach that consists of calculating the word overlap between the sense definitions of two or more target words as a primary step. This approach is termed as gloss overlap or the LESK algorithm [1]. Given a two-word context (w_1, w_2), the senses of target words having the highest overlap in their definitions are chosen as the correct ones. For two words w_1 and w_2, we determine score for each pair of word senses $S1 \in Senses(w_1)$ and $S_2 \in Senses(w_2)$:

$$\mathbf{Score}_{\mathbf{Lesk}}(S_1, S_2) = |\mathbf{gloss}(S_1) \cap \mathbf{gloss}(S_2)|,$$

where gloss(S_i) is the set of words in the textual definition of sense S_i of w_i. The senses with maximum value for the above formula are assigned to the respective words. As it requires exponential number of steps, a variant of Lesk is currently used which identifies the sense of a word w whose textual definition has the highest overlap with the words in the context of w. For a target word w, corresponding score is computed for each sense S of w as follow:

$$\textbf{Score}_{\textbf{LeskVar}}\textbf{(S)} = \textbf{| context (w)} \cap \textbf{gloss(S) |,}$$

where context (w) is the set of all content words in a context window around the target word w. Lesk's approach is very sensitive to the exact wording of definitions, so the absence of a particular word can drastically change the results.

A measure of extended gloss overlap as discussed in [2], which expands the glosses of the words being compared to include glosses of concepts that are known to be related through explicit relations in the dictionary (e.g.,hypernymy, meronymy, pertainymy, etc.). The range of relationships used to extend the glosses is a parameter, and can be selected from any combination of WordNet relations.

For each sense S of a target word w, we estimate its score as

$$\textbf{score}_{\textbf{ExtLesk}}\textbf{(S)} = \sum_{\textbf{S}^{\textbf{I}}:\textbf{S}\rightarrow\textbf{S}^{\textbf{I}}\ \textbf{or}\ \textbf{S}\equiv\textbf{S}^{\textbf{I}}} \textbf{\textit{rel}} \quad \textbf{| context(w)} \cap \textbf{gloss(S) |,}$$

where context(w) is the bag of all content words in a context window around the target word w and gloss(S) is the bag of words in the textual definition of a sense S which is either S itself or related to S through a relation *rel*. The overlap scoring mechanism is also parameterized and can be modified to take into account gloss length or to include function words as

$$\textbf{score}_{\textbf{ExtLesk}}\textbf{(S)} = \sum_{\textbf{S}^{\textbf{I}}:\textbf{S}\rightarrow\textbf{S}^{\textbf{I}}\ \textbf{or}\ \textbf{S}\equiv\textbf{S}^{\textbf{I}}} \textbf{\textit{rel}} \quad \textbf{| context(w)} \cap \textbf{gloss(S) |,}$$

3.2. Selectional Preferences

It is one which exploits selectional preferences to constrain the number of meanings of a target word occurring in context. Selectional preferences or restrictions restrict the semantic type that a word sense imposes on the words with which it combines in sentences. For instance, the verb drink expects an animate entity as subject and a potable entity as its direct object. We can differentiate selectional restrictions and preferences in that the former rule out senses that violate the constraint, whereas the latter tend to select those senses which better satisfy the requirements.

The simple way to learn selectional preferences is to find the semantic appropriateness of the association provided by a word-to-word relation through frequency count. Given a pair of words w_1 and w_2 and a syntactic relation R (e.g., subject-verb, verb-object, etc.), this method counts the number of instances (R, w_1, w_2) in a corpus of parsed text, obtaining a measure Count(R, w_1, w_2) as in [3]. Conditional probability, another measure of the semantic appropriateness of a word-to-word relation, of word w_1 given the other word w_2 and the relation R:

$$\textbf{P (} w_1 \textbf{/} w_2 \textbf{ , R) = Count(}w_1\textbf{,}w_2\textbf{,R)/Count(} w_2 \textbf{ , R)}$$

WordNet helps to derive a mapping from words to conceptual classes. Various techniques have been devised for measure of Selectional association [4, 5], to tree cut models using the minimum description length [6, 7], hidden markov models [8], etc. Almost all these approaches exploit large corpora and model the selectional preferences of predicates by combining observed frequencies with knowledge about the semantic classes of their arguments. Finally disambiguation is performed with different means based on the strength of a selectional preference towards a certain conceptual class.

3.3. Structural Approaches

Multiple structural approaches have been developed to analyze and exploit the structure of the concept network available through computational lexicons like WordNet. The recognition and measurement of patterns, both in a local and a global context, can be accumulated in the field of structural pattern recognition [9], which intend to classify data based on the structural interrelationships of features. We discuss two prime approaches here: similarity-based and graph-based methods.

3.3.1 Similarity Measures

We disambiguate a target word w_i in a text $T = (w_1, \ldots, w_n)$ by selecting the sense S of w_i with maximum value for the following sum:

$$\hat{S} = \underset{S \in Senses_D(w_i)}{\operatorname{argmax}} \sum_{w_j \in T, w_j \neq w_i} \max_{S' \in Senses_D(w_j)} score(S, S').$$

Given a sense S of our target word w_i, the formula adds the contribution of the most appropriate sense of each context word $w_j = w_i$. The sense with the highest sum is selected. Similar disambiguation strategies can be applied [10]. We now present well-known measures of semantic similarity in the literature.

A simple metric depending on determining the shortest distance in WordNet between pairs of word senses is introduced in [11]. The hypothesis is that, for given a pair of words w and w^I occurring in the same context, the most proper sense is determined through selecting the senses that maximize the distance between w and w^I. It is defined as follows:

$$\mathbf{score_{Rada}}\ (S_w, S_w^I) = d\ (S_w, S_w^I),$$

where d (S_w, S_w^I) is the shortest distance between S_w and S_w^I is obtained by counting the number of edge of the shortest path in the lexicon network. The shortest path determined on the WordNet taxonomy includes only hypernymy edges.

An approach in [12] is based on ascertaining that concepts deep in taxonomy are more closely related to each another than those in the upper part of the same taxonomy. An edge in the WordNet noun taxonomy is seen as a pair of two directed edges representing inverse relations (e.g., kind-of and has-kind). The measure is defined as follows:

$$score_{Sussna}(S_w, S_{w'}) = \frac{w_R(S_w, S_{w'}) + W_{R^{-1}}(S_{w'}, S_w)}{2D}$$

where R is a relation, R^{-1} its inverse, D is the overall depth of the noun taxonomy, and each relation edge is weighted based on the following formula:

$$w_R(S_w, S_{w'}) = max_R - \frac{max_R - min_R}{n_R(S_w)},$$

where max_R and min_R are a maximum and minimum weight that we want to assign to relation R and n_R (S_w) is the number of edges of type R emerging from S_w.

A similarity measure based on the distance between two senses S_w and S_w^I is developed in [13]. They concentrated on hypernymy links and gradually incrementing the path length by the overall depth D of the taxonomy:

$$score_{Lch}(S_w, S_{w'}) = -log\frac{d(S_w, S_{w'})}{2D}.$$

The disadvantage of distance-based measures is not considering the density of concepts in a sub tree whose root is a common ancestor. The variant of Conceptual density that computes the density of senses of a word context is demonstrated in [14]. For a synset S, assuming m senses of words to be disambiguated in its sub hierarchy a mean number of hyponyms per sense nhyp is also taken into account. Then, the conceptual density S is determined as follows:

$$CD(S, m) = \frac{\sum_{i=0}^{m-1} nhyp^{i^{0.20}}}{descendants(S)},$$

where descendants(S) is the total number of descendants in the noun hierarchy of S and 0.20 is a smoothing exponent, whereas i ranges over all possible senses of words in the subhierarchy of S. For all senses of nouns in the given context, the conceptual density S is determined for all of their respective hypernyms. The set of sense choices is obtained from the synset associated with highest conceptual density. The senses included in its sub hierarchy are considered as respective words interpretation in context. The remaining senses of those words are removed from the hierarchy, and the procedure is repeated for the other ambiguous words.

4. SUPERVISED DISAMBIGUATION

Machine-learning techniques are applied to construct a classifier from manually sense-annotated data set in supervised word sense disambiguation approaches. The classifier or word expert chooses a word at a time and accomplish a classification task in an attempt to determine the appropriate sense for instance of that word. The training set consists of numerous examples having the given target word manually tagged with a sense from the sense inventory of a reference dictionary. This is used to learn the classifier. We discuss here well-known machine learning techniques namely: Decision lists, Decision trees, Naive Bayes and Neural networks in the following sections.

4.1. Decision Lists

A decision list [15] is an ordered set of rules for classifying test instances. It is a list of weighted if-then –else rules, arranged in decreasing order of their score. A set of features are built using a training set. Then rules of the form (F, S, T) where F denotes feature-value, S the sense and T its score are obtained.

First, a feature vector representing the given target word w is obtained. Then, the decision list is examined in order to find the feature with the maximum score matching the input vector. The winning feature identifies the proper word sense for word w.

$$\hat{S} = argmax_{S_i \in Senses_D(w)} score(S_i)$$

Yarowsky [16], modified the criteria to improve the performance. In this score of sense S_i is taken to be the maximum among the feature scores. The score of a feature f is determined through the logarithm of the probability of sense S_i given feature f divided by a value that is obtained accumulating the probabilities of the other senses given feature f :

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

Smoothing and pruning are performed to remove zero counts and avoiding not reliable rules with less weight respectively.

4.2. Decision Trees

In the decision tree classification rules are represented as a tree structure having the virtue of recursively partitioning the training data set. Every internal node indicates a test on a feature value. Branch followed specifies the outcome of the test. For a given word to assign the proper sense, we start from the root performing a sequence of test following the corresponding branches until a leaf node is reached, where sense assignment is done if it is not empty denoted by - .
Algorithms known for better performance for learning decision trees are C4.5 algorithm [17], ID3 algorithm [18]. When comparing the performance of decision tree obtained with the C4.5 algorithm with other machine learning algorithms, it is surfaced that the former results in less performance. Though it has the advantage of representing the predictive model in a compact and Human readable way, it suffers from issues like data sparseness because of features with more number of values, unreliable predictions because of small training set, etc. An example of a decision tree for WSD is described in Figure 1. For example, if the noun bank must be classified in the sentence "I deposited money in my account" the tree is traversed starting from the root, after following the branches, the choice of sense bank/ Finance is made.

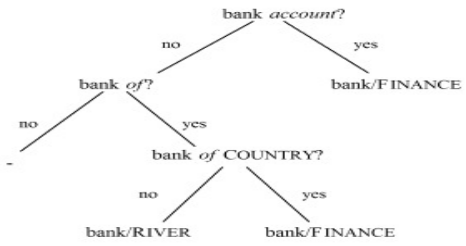

Fig1.An example of a decision tree

4.3. Naive Bayes

A Naive Bayes classifier is a straightforward probabilistic classifier obtained through the use of Bayes' theorem. We determine the conditional probability of each sense S_i of a word w for the set of features f_j in the context. Then we calculate the following formula for each sense.

$$\hat{S} = \underset{S_i \in Senses_D(w)}{\operatorname{argmax}} P(S_i \mid f_1, \ldots, f_m) = \underset{S_i \in Senses_D(w)}{\operatorname{argmax}} \frac{P(f_1, \ldots, f_m \mid S_i) P(S_i)}{P(f_1, \ldots, f_m)}$$
$$= \underset{S_i \in Senses_D(w)}{\operatorname{argmax}} P(S_i) \prod_{j=1}^{m} P(f_j \mid S_i),$$

The sense S maximizing the above is identified as the most appropriate sense in the context. Smoothing is performed to eliminate the impact of zero counts as in [19]. But, this solution causes probabilities to have value more than 1. We can address this problem through Back off or interpolation strategies.

4.4. Neural Networks

A neural network [20] is an interconnected group of artificial neurons. It processes data using a computational model based on a connectionist approach. The pairs consisting of (inut feature, desired response) are accepted as input to the learning program.

The key objective is to using the input features to divide the training contexts into disjoint sets associated with desired response. When the network is added new pairs, link weights are adjusted properly so that outcome unit specifying the desired response has more activation than any other outcome unit. Neural networks must be trained adequately so that the outcome of the unit specifying the desired response is more than the outcome of any other unit for every training example. Weights in the network are either positive or negative, accumulating the evidence supporting or denying a sense choice Cottrell [21] used neural networks where nodes represent words: the words activate the concepts to which they are semantically related and vice versa. The activation of a node results in the activation of nodes to which it is connected through excitory links and the deactivation of those to which it is connected through inhibitory links.

It is shown in [22, 23] that neural networks outperforms other supervised methods based on experiments performed on few number of words. The complex interpretation of results, the requirement of vast training data, and adjusting parameters like threshold, decay, etc are the major drawbacks in using the neural networks.

5. UNSUPERVISED DISAMBIGUATION

The performance of supervised disambiguation approaches suffers from knowledge acquisition bottleneck. But, unsupervised methods overcome the lack of large-scale resources annotated with word senses manually by virtue as discussed in [24]. These are developed based on the observation that word with a particular sense will always be surrounded by the similar words. These approaches identify word senses from input text by dividing word occurrences. New occurrences are clustered based on the induced clusters. Hence, these methods don not rely on labelled training text and even not using the knowledge of machine-readable resources like dictionaries, thesauri, WordNet, etc.

Unsupervised WSD performs word sense discrimination rather than assigning an appropriate sense to a target word like in knowledge based or supervised approaches. It accomplishes it by partitioning the occurrences of a word into multiple classes by identifying for any two occurrences whether they correspond to the same sense or not as discussed in [25]. The clusters obtained in these approaches may not be equivalent to the regular senses in a dictionary sense inventory. This makes the evaluation more complex which may require human intervention or indirect evaluation using the clusters in end-to-end applications.

However, sense discrimination and sense labelling are two sub problems of the WSD task as discussed in [26], but are related closely. Hereafter, we discuss the key approaches to unsupervised WSD, namely: methods based on context clustering, word clustering, and co occurrence graphs.

5.1. Context Clustering

We have an unsupervised approach based on the notion of context clustering. Context vector is used to represent each occurrence of a target word in a corpus. Then, these vectors are partitioned into groups, each helps determining a sense of the target word.

A historical approach in this category is based on the idea of word space also known as vector space, whose dimensions are words. A word w in a corpus is represented as a vector whose i^{th} component gives the total number of times that words w_i and w appear within a fixed context.

Then, the similarity between two words v and w is computed geometrically by the cosine between the corresponding vectors v and w:

$$\mathrm{sim}(v, w) = \frac{v \bullet w}{|v||w|} = \frac{\sum_{i=1}^{m} v_i w_i}{\sqrt{\sum_{i=1}^{m} v_i^2 \sum_{i=1}^{m} w_i^2}}$$

where m is the number of features in each vector. A vector is determined for every word in a corpus.

We obtain a cooccurrence matrix by combining the set of vectors for each word in the corpus. To reduce the dimensionality of the resulting multidimensional space through singular value decomposition (SVD), latent semantic analysis (LSA) is applied [27]. SVD identifies the major axes of variation in the word space. The input to the dimensionality reduction is set of word vectors in the high dimensional space and output is representation in a lower- dimensional space. The dimensions corresponding to terms having similar meanings may be merged.

Following the dimensionality reduction, contextual similarity between two words is determined in terms of the cosine between respective vectors. Next, our objective is clustering context vectors. A context vector is constructed as the centroid of the vectors of the words appearing in the target context that accounts for its semantic context.

Finally, sense discrimination is achieved by grouping the context vectors of a target word applying a standard clustering algorithm. Context-group discrimination algorithm was proposed by Schutze that categorize the occurrences of an ambiguous word into clusters of senses depending on the contextual similarity between occurrences. Contextual similarity is calculated as above. In this, clustering is performed with the Expectation Maximization algorithm based on the variation of the probabilistic model [28]. A different clustering approach consists of agglomerative clustering [29]. Initially, each instance consists of a cluster with single element. Next, agglomerative clustering combines the most similar pair of clusters and repeated for resulting less similar pairs until the corresponding threshold is not acceptable. Its performance was assessed in the biomedical domain.

The difficulty in building the context vectors is that a vast amount of training data is necessary for identifying a remarkable distribution of word co-occurrences. This problem can be resolved by augmenting the feature vector of every word with the content words appearing in the glosses of its senses [30]. A more subtle issue to be resolved is the fact that different context vectors are not assigned to distinct word senses. In [31], it is resolved through a well trained supervised classifier. Word senses are determined through the use of multilingual context vectors in [32]. Here, a word occurrence in a multilingual corpus is represented as a context vector that takes into account every possible lexical translation of the target word w, whose value is 0 if the specific occurrence of w cannot be translated properly, otherwise it is 1.

5.2. Word Clustering

In the previous method, first- or second-order context vectors are used to represent word senses. A different approach for determining appropriate word senses is based on word clustering techniques, whose objective is clustering words that are semantically similar and thereby supporting a specific meaning.

A popular approach to word clustering involves the identification of words W=((w_1, . . . , wk) that are similar to a target word w_0. For each w_i its similarity with $w_0$is measured using the information content of their single features given by the syntactic dependencies like subject-verb, verb-object, adjective-noun etc. The information content is high, when two words share dependencies. Now, word clustering algorithm is applied to discriminate between the senses. Suppose that W is the set of similar words ordered based on the degree of similarity to w_0. A similarity tree T is created which initially has a single node w_0. Next, for each $i \in \{1, \ldots, k\}$, $w_i \in W$ a child is added to w_j in the tree T where w_j is the most similar word to w_i among $\{w_0, \ldots, w_{i-1}\}$. It is followed by pruning – deleting unnecessary nodes. Now each sub tree with root w_0 is considered as a distinct sense w_0.

A different word clustering technique, known as Clustering By Committee (CBC) algorithm, was discussed in [33]. In this, for every target word, a set of similar words is determined as above. To determine the similarity, each word is represented as feature vector, where every feature is the expression of syntactic context having the word.

For a given set of target words, first a similarity matrix S is constructed where S_{ij} contains the pair wise similarity between words w_i and w_j. Next step invokes a recursive procedure to determine set of clusters known as committees, for each word in the given set of words E. One of the standard clustering technique, average-link clustering is employed. In each step, words not covered by any committee are discovered. Recursive attempts are done to identify more committees from residue words. Finally, for each target word $w \in E$, which is represented as a feature vector, is iteratively allocated to its most similar cluster, depending 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 deleted from the representation of w, allowing the identification of less frequent senses of the same word in subsequent iterations.

5.3. Cooccurrence Graphs

The word sense discrimination relying on graph-based approaches provides a different view, which has been recently explored providing fairly good performance. Cooccurrence graph g-(V,E), where V the set of vertices represents words in a text and edges E connect two words having a syntactic relation, in the same paragraph, or in a larger context, is used in developing graph-based approaches.

Cooccurrence graph built based on grammatical relations between words in context is described in [34]. A local graph G_w is constructed around w for each target ambiguous word w. The graph can be seen as a Markov Chain, after normalizing the adjacency matrix representing G_w. Then, the Markov clustering algorithm [35] is applied to identify word senses, depending on an expansion and an inflation step, focusing, respectively, at inspecting new more distant neighbours and including other popular nodes.

Subsequently, an ad hoc approach known as HyperLex is proposed in [36]. In this, initially a cooccurrence graph is constructed where vertices V is the set of words occurring in the paragraph of a text corpus having the target word, and an edge between a pair of words is included in the

graph if they co-occur in the same paragraph as the first step. It is a weighted graph, where each edge is given a weight based on the relative cooccurrence frequency of the two words connected by the edge. So, for each edge (i, j) its weight w_{ij} is calculated as

$$w_{ij} = 1 - \max\{P(w_i \mid w_j), P(w_j \mid w_i)\},$$

where $P(w_i \mid w_j) = freq_{ij} / freq_j$, and $freq_{ij}$ is the frequency of cooccurrence of words w_i and w_j and $freq_j$ is the frequency of w_j within the text. Following the above, words which rarely occur together are assigned weight close to 1, whereas words with high frequency of cooccurrence are given weight close to zero.

The next step involves running an iterative algorithm on the cooccurrence graph built in the above. In each iteration, we select as a hub, a node with highest relative degree in the graph. It results in all its neighbours cannot be hubs. The algorithm ends when the relative frequency of the word corresponding to the chosen hub is less than a fixed threshold. The entire set of hubs selected so far, represent the senses of the word of interest. Then, these hubs are linked to the target word and edges are assigned zero-weight Hubs are then linked to the target word with zero-weight edges and the minimum spanning tree (MST) is built for the entire graph that is used to disambiguate particular instances of our target word. Suppose that $W = (w_1, w_2, \ldots, w_i, \ldots, w_n)$ is a context having w_i, our target word.

A score vector s is assigned to each $w_j \in W$ ($j = i$), such that its k^{th} component s_k represents the contribution of the k^{th} hub as follows:

$$s_k = \begin{cases} \frac{1}{1 + d(h_k, w_j)} & \text{if } h_k \text{ is an ancestor of } w_j \text{ in the MST} \\ 0 & \text{otherwise,} \end{cases}$$

where $d(h_k, w_j)$ is the distance between root hub h_k and node w_j.

Next, all score vectors assigned to all $w_j \in W$ ($j \neq i$) are added and the hub which is associated with the highest score is chosen as the most appropriate sense for w_i.

An alternative graph-based algorithm for determining word senses PageRank is discussed in [37]. PageRank is popular algorithm devised for determining the ranking of web pages. Not only the Google search engine depends more on page rank for its better performance, but also used in many research areas where the prime objective is determining the importance of entities where graph represents the relations among them. In its weighted formulation, the PageRank degree of a vertex $v_i \in V$ is determined using the following formula:

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

where $v_j \rightarrow v_i$ represents the presence of an edge from v_j to v_i, d is a damping factor and w_{ji} is its weight which models the probability of following a link to v_i or randomly jumping to v_i. The above formula is recursive: the PageRank of each vertex is iteratively computed until convergence.

6. EVALUATION METHODOLOGY

We discuss here the evaluation measures and baselines, borrowed from the field of information retrieval, namely coverage, precision, and recall. However, the main objective of WSD is to demonstrate that it improves the performance of applications such as machine translation, information retrieval, question answering systems, etc. The evaluation of WSD that is part of applications is called in vivo or end-to-end evaluation. We present this second kind of evaluation in the following section.

6.1. Evaluation Measures

Let $S = (w_1, \ldots, w_n)$ be a test set and T an "answer" function that associates with each word $w_i \in S$ the appropriate set of senses from the dictionary D (i.e., $T(i) \subseteq \text{Senses}_D(w_i)$). Then, given the sense assignments $T^I(i) \in \text{Senses}_D(w_i) \cup \{\epsilon\}$ provided by an automatic WSD system ($i \in \{1, \ldots, n\}$), then, coverage C is defined as the percentage of items in the test set for which the system identify a sense assignment. So

$$C = \text{\# answers provided / \# total answers to provide} = |\{i \in \{1, \ldots, n\} : T^I(i) = \epsilon\}| / n$$

Where ϵ denotes that system fails in providing an answer for a specific word w(i.e., in that case we assume that $T^I(i) = \epsilon$). The total number of answers is given by $n = |S|$.

The precision P of a system is calculated as the percentage of correct answers given by the automatic system, that is:

$$P = \text{\# correct answers provided / \# answers provided}$$

$$= |\{i \in \{1, \ldots, n\} : T^I(i) \in A(i)\}| / i \in \{1, \ldots, n\}: T^I(i) \neq \epsilon\}|$$

Recall R is defined as the number of correct answers provided by the automatic system over the total number of answers to be provided:

$$R = \frac{\text{\# correct answers provided}}{\text{\# total answers to provide}} = \frac{|\{i \in \{1, \ldots, n\} : A'(i) \in A(i)\}|}{n}.$$

According to the above definitions, we have $R \leq P$ by default. When coverage is 100%, we have P = R. In the WSD literature, recall is also called as accuracy, but these are two different measures in the information retrieval and machine learning literature.

Finally, we have F_1-measure or balanced F-score, computed as

$$F_1 = 2PR/P + R$$

It is a measure calculating the weighted harmonic mean of precision and recall.

F_1-score is useful to compare systems providing coverage less than 100%. If simple arithmetic mean is used, we can easily build a system with P = 100% and almost-zero recall giving around 50% performance, but harmonic mean such as F_1- score significantly reports lower performance for low values of either precision or recall.

7. CONCLUSION

In this paper we presented an empirical study of the word sense disambiguation (WSD). WSD is as complex as an AI complete problem since it has to dealt with all the complexities of language in obtaining the semantic structure from the unstructured text. Moreover, the complexity of WSD is proportional to the granularity of the sense distinctions considered. We have presented various techniques in Knowledge-based, supervised, and unsupervised approaches. Knowledge –based approaches appear to be more successful due to highly enriched knowledge sources. But, it may fail completely if sufficient information is not present in the knowledge sources. Supervised approaches undoubtedly outperform other approaches. But, they require a large training corpora for every possibility which demands more human effort and time i.e these may suffer from knowledge acquisition bottleneck. Unsupervised approaches do not depend on knowledge sources. So these are most suitable for all word WSD even for languages with low resources. Following this, we would like to build a comprehensive WSD system for TELUGU language most spoken Dravidian based language in India.

REFERENCES

[1] LESK.M.1986. Automatic sense disambiguation using machine readable dictionaries: Hoe to tell a pine cone from an ice cream cone. In Proceedings of the 5th SIGDOC (New York, NY). 24-16

[2] BANERJEE, S. AND PEDERSEN, T. 2003. Extended gloss overlaps as a measure of semantic relatedness. In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI, Acapulco, Mexico). 805–810.

[3] HINDLE, D. AND ROOTH, M. 1993. Structural ambiguity and lexical relations. Computat. Ling. 19, 1, 103–120.

[4] RESNIK, P. S., Ed. 1993. Selection and information: A class-based approach to lexical relationships, Ph.D. dissertation. University of Pennsylvania, Pennsylvania, Philadelphia, PA.

[5] RESNIK, P. 1997. Selectional preference and sense disambiguation. In Proceedings of the ACL SIGLEXWorkshop on Tagging Text with Lexical Semantics: Why, What, and How? (Washington, D.C.). 52–57.

[6] LI, H. AND ABE, N. 1998. Generalizing case frames using a thesaurus and the MDL principle. Computat.Ling. 24, 2, 217–244.

[7] MCCARTHY, D. AND CARROLL, J. 2003. Disambiguating nouns, verbs and adjectives using automatically acquired selectional preferences. Computat. Ling. 29, 4, 639–654.

[8] ABNEY, S. AND LIGHT, M. 1999. Hiding a semantic class hierarchy in a Markov model. In Proceedings of theACL Workshop on Unsupervised Learning in Natural Language Processing (College Park, MD). 1–8.

[9] FU, K. 1982. Syntactic Pattern Recognition and Applications. Prentice-Hall, Engelwood Cliffs, NJ.

[10] PEDERSEN, T., BANERJEE, S., AND PATWARDHAN, S. 2005. Maximizing semantic relatedness to perform word sense disambiguation. Res. rep. UMSI 2005/25. University of Minnesota Supercomputing Institute, Minneapolis, MN.

[11] RADA, R., MILI, H., BICKNELL, E., AND BLETTNER, M. 1989. Development and application of a metric on semantic nets. IEEE Trans. Syst. Man Cybernet. 19, 1, 17–30.

[12] SUSSNA, M. 1993. Word sense disambiguation for free-text indexing using a massive semantic network.In Proceedings of the 2nd International Conference on Information and Knowledge Base Management(Washington D.C.). 67–74.

[13] LEACOCK, C. AND CHODOROW,M. 1998. Combining local context andWordNet similarity for word sense identification.In WordNet: An electronic Lexical Database, C. Fellbaum, Ed. MIT Press, Cambridge, MA,265–283.

[14] AGIRRE, E. AND RIGAU, G. 1996.Word sense disambiguation using conceptual density. In Proceedings of the 16th International Conference on Computational Linguistics (COLING, Copenhagen, Denmark). 16–22.

[15] RIVEST, R. L. 1987. Learning decision lists. Mach. Learn. 2, 3, 229–246.

[16] YAROWSKY, D. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration inSpanish and French. In Proceedings of the 32nd Annual Meeting of the Association for ComputationalLinguistics (Las Cruces, NM). 88–95.
[17] QUINLAN, J. R. 1993. Programs for Machine Learning. Morgan Kaufmann, San Francisco, CA.
[18] QUINLAN, J. R. 1986. Induction of decision trees. Mach. Learn. 1, 1, 81–106.
[19] NG, T. H. 1997. Getting serious about word sense disambiguation. In Proceedings of the ACL SIGLEXWorkshop on Tagging Text with Lexical Semantics: Why, What, and How? (Washington D.C.). 1–7.
[20] MCCULLOCH, W. AND PITTS, W. 1943. A logical calculus of the ideas immanent in nervous activity. Bull.Math. Biophys. 5, 115–133.
[21] COTTRELL, G. W. 1989. A Connectionist Approach to Word Sense Disambiguation. Pitman, London, U.K.
[22] MOONEY, R. J. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In Proceedings of the 1996 Conference on Empirical Methods in Natural Language Processing (EMNLP). 82–91.
[23] TOWELL, G. AND VOORHEES, E. 1998. Disambiguating highly ambiguous words. Computat. Ling. 24, 1, 125–145.
[24] GALE, W. A., CHURCH, K., AND YAROWSKY, D. 1992b. A method for disambiguating word senses in a corpus.Comput. Human. 26, 415–439.
[25] SCHUTZE, H. 1998. Automatic word sense discrimination. Computat. Ling. 24, 1, 97–124.
[26] SCHUTZE, H. 1992. Dimensions of meaning. In Supercomputing '92: Proceedings of the 1992 ACM/IEEEConference on Supercomputing. IEEE Computer Society Press, Los Alamitos, CA. 787–796.
[27] GOLUB, G. H. AND VAN LOAN, C. F. 1989. Matrix Computations. The John Hopkins University Press,Baltimore, MD.
[28] DEMPSTER, A., LAIRD, N., AND RUBIN, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Statist. Soc. 39, 1–38.
[29] PEDERSEN, T. AND BRUCE, R. 1997. Distinguishing word senses in untagged text. In Proceedings of the1997 Conference on Empirical Methods in Natural Language Processing (EMNLP, Providence, RI). 197–207.
[30] PURANDARE, A. AND PEDERSEN, T. 2004. Improving word sense discrimination with gloss augmented feature vectors. In Proceedings of theWorkshop on Lexical Resources for theWeb andWord Sense Disambiguation(Puebla, Mexico). 123–130.
[31] NIU, C., LI, W., SRIHARI, R., AND LI, H. 2005. Word independent context pair classification model for wordsense disambiguation. In Proceedings of the 9th Conference on Computational Natural Language Learning(CoNLL, Ann Arbor, MI).
[32] IDE, N., ERJAVEC, T., AND TUFIS, D. 2001. Automatic sense tagging using parallel corpora. In Proceedings of the 6th Natural Language Processing Pacific Rim Symposium (Tokyo, Japan). 83–89.
[33] LIN, D. AND PANTEL, P. 2002. Discovering word senses from text. In Proceedings of the 8th ACM SIGKDDInternational Conference on Knowledge Discovery and Data Mining (Edmonton, Alta., Canada). 613–619.
[34] WIDDOWS, D. AND DOROW, B. 2002. A graph model for unsupervised lexical acquisition. In Proceedings of the 19th International Conference on Computational Linguistics (COLING, Taipei, Taiwan). 1–7.
[35] VAN DONGEN, S. 2000. Graph Clustering by Flow Simulation, Ph.D. dissertation. University of Utrecht,Utrecht, The Netherlands.
[36] V´ERONIS, J. 2004. Hyperlex: Lexical cartography for information retrieval. Comput. Speech Lang. 18, 3, 223–252.
[37] NAVIGLI, R. 2009. Word Sense Disambiguation: a Survey, ACM Computing Surveys, Vol. 41, No.2, ACM Press, pp. 1-69.
[38] Amiat Jain, Sudesh Yadav "Measuring Context-Meaning for Open Class Words in Hindi Language" Sixth International Conference on Contemporary Computing IEEE 2013. ISBN:978-1-4673-5114-0,Page(s)173-178

Authors

M.Srinivas holds B.Tech in Computer Science and Engineering from JNTU Hyderabad, M.Tech in Computer Science from JNTU Hyderabad and Pursuing Ph.D in Computer Science and Engineering from JNTU Hyderabad. His areas of research interest includes Natural Language Processing, Information Security, Data Mining, and Data Analytics. At present he is working as Associate Professor, Department of Computer Science & Engineering, Geethanjali College of Engineering & Technology, Hyderabad.

Dr.B. Padmaja Rani holds B.E in Electronics and Communication Engineering from Osmania University, Hyderabad, M.Tech in Computer Science from JNTU Hyderabad and She received a Doctoral Degree(Ph.D.) in Computer Science from JNTU Hyderabad. At present she is working as Professor, Department of Computer Science and Engineering, JNTUH College of Engineering, JNTUH University, Hyderabad. She is guiding couple of Ph.D Scholars in the area of Information Retrieval, Natural Language Processing and Information Security. Her area of research interest includes Information Retrieval, Natural Language Processing, Information Security, Data Mining and Embedded Systems. To the Credit she published 40 + research papers in various International/National Conferences and Journals. She is member of various professional bodies including CSI, IEEE, ISTE.