Word Sense Disambiguation and Text Similarity Measurement Using WordNet

Pushpak Bhattacharyya[1] and Narayan Unny

Department of Computer Science and Engineering,
Indian Institute of Technology,
Bombay 400 076, INDIA.
{nue,pb}@cse.iitb.ernet.in

Abstract

This article advocates the use of lexical knowledge and semantics to improve the accuracy of information retrieval. A system of measuring text similarity is developed, which attempts to integrate the meaning of texts into the similarity measure. The work hinges on the organization of the synsets in the WordNet according to the semantic relations of *hypernymy/hyponymy*, *meronymy/holonymy* and *antonymy*. A unique measure using the *link distance* between the words in a *subgraph of the WordNet* has been evolved. This needs word sense disambiguation which in itself is a complex problem. We have developed an algorithm for word sense disambiguation exploiting again the structure of the WordNet. The results support our intuition that including semantics in the measurement of similarity has great promise.

1 Introduction

Today's search engines like *google* do an admirable job of information retrieval. However, the goal of *retrieving all and only the most relevant information* is still a far cry. One step towards realizing this ideal goal is the *detection of similarity of texts, i.e., judging how close in meaning two given texts are.* Fundamentally, the similarity of two objects is measured by the number of features the objects have in common. This idea is applied to text similarity detection also. However, approaches differ on the notion of what the features of a text object are. In contempoary IR, the words of a text are taken as features. The more features the two texts share, the more similar they are to each other.

1.1 Inclusion of meaning

One obvious shortcoming of the *bag of words* approach is that it does not at all consider the meaning of texts. Two phenomena that need to be considered in this context are *polysemy* and *synonymy. Polysemy* refers to the same word form having different meanings in different contexts, while *synonymy* refers to different word forms having the same meaning. Consider, for example, two texts each using the term *board* extensively. But in one, the term means *wood, plank, etc.* and in the other it means *committee.* Failure to detect this polysemy leads to *overextimating the similarity value.* On the other hand, consider two texts, one using the term *wood* and the other using *plank.* The similarity measure not accounting for the synomymy here leads to *underestimating the similarity value.*

[1]contacting author

These problems with the conventional approach necessitate considering the semantics of the text. The idea is to *expand a term so that it covers not only its synonyms but also the words which are closely related to it. This is the basic intuition behind our algorithm and is implemented using the WordNet*.

This article is organized as follows. In section 2 we discuss the WordNet. Section 3 describes the common similarity measures. In section 4 we give the block diagram of the system and describe the top level design. The word sense disambiguation module along with its evaluation is discussed in section 5. The description of the actual similarity estimation is given in section 6. Detailed evaluation of the system is given in section 7. We conclude in section 8.

2 WordNet

The lexical resource WordNet [2] plays the central role in the work described in this article. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different semantic relations link these synonym sets.

As discussed in the previous section, synonymy and polysemy make it impossible to have a one to one mapping of word forms to the meanings. This was the inspiration behind the organization of WordNet into *synsets*. A synset is a set of synonymous words whose primary function is to capture a unique meaning. For example, the word *board* can have two different meanings, *viz.* a *piece of lumber* and *a group of people*. The synonym sets, {*board, plank*} and {*board, committee*} can serve as unambiguous designators of these two meanings of *board*. These synonym sets do not explain what the concepts are; they merely signify that these unique concepts exist.

The most important relation for WordNet is **synonymy** which denotes the similarity of meaning, since the ability to judge that relation between word forms is a prerequisite for the representation of meanings. The definition of synonymy can be given as follows: *two expressions are synonymous in a linguistic context* **C** *if the substitution of one for the other in* **C** *does not alter the truth value*. For example, the substitution of *plank* for *board* will seldom alter truth values in carpentry contexts.

The next familiar relation is **antonymy** which for a given word is a word with opposite meaning. Antonymy provides a central organizing principle for the adjectives and adverbs in WordNet. One important observation is that the words from the same synset have definite preference for words from the synset with opposite meaning. For example, *rise* takes *fall* as the antonym and *ascend* takes *descend* as the antonym even though {*rise, ascend*} and {*fall, descend*} are two synsts.

Unlike synonymy and antonymy, which are lexical relations between word forms, **hypernymy/hyponymy** is a semantic relation between word concepts, *i.e.*, synsets. This semantic relation arranges the synsets in a hierarchy. For example, {*dust, debris, junk, rubble*} is a hyponym of {*rubbish, trash*} which in turn is a hyponym of {*substance, matter*}. Much attention has been devoted to hyponymy/hypernymy (variously called subordination/superordination, subset/superset, or the IS-A relation). Hyponymy is transitive and asymmetrical, and since there is normally a single superordinate, it generates a hierarchical semantic structure in which a hyponym is said to be below its superordinate.

The semantic relation *part-whole* or *HAS-A* is known to as **meronymy/holonymy**. The meronymic relation is transitive (with qualifications) and asymmetrical, and can be used to construct a part hierarchy (with some reservations, since a meronym can have many holonyms). For example, $\{house\}$ has as meronym $\{study\}$ which in turn has as meronym $\{door\}$.

The semantic relations represent associations that form a complex network. Knowing where a word is situated in that network is an important part of knowing the word's meaning. This network of synsets forms the WordNet.

3 Similarity measures

In this section, we look at some of the similarity measures that have been used extensively in the field of information retrieval. These measures are discussed in two parts: (1) Text similarity measures [12] and (2) Conceptual similarity measures [10]. In this article we have studied the use of the latter for solving the former problem.

3.1 Text Similarity

Text similarity measures essentially take the bag-of-words representation of text for measuring similarity. Discussion on some of these measures follows:

1. **Cosine**
   This is the most widely used measure. The popularity stems from its simplicity. It is calculated from the cosine of the vectors corresponding to the two texts being compared. The vector of a text is formed by using the frequency of occurrence of distinct words in the text as the components. Thus this measure gives the intersection of the two texts weighted by the respective frequencies of occurrence. Mathematically, it is denoted as,

   $$Cos(d,q) = \frac{\sum_{(t \in q) \wedge (t \in d)} f_{d,t} f_{q,t}}{\sqrt{(\sum_{t \in q} f_{q,t}^2)(\sum_{t \in q} f_{d,t}^2)}}$$

   where $f_{x,t}$ is the frequency of term t in document x.

2. **Dice**
   The dice coefficient is defined by the binary model of documents. Here the components of the document vector are binary values corresponding to the occurrence (or non occurrence) of the term in the document. If X and Y are the documents we need to compare, then the *Dice Similarity* is defined as:

   $$Dice(X,Y) = \frac{2 \mid X \cap Y \mid}{\mid X \mid + \mid Y \mid}$$

   where,

   $\mid X \cap Y \mid$ is the number of words that are common in the documents X & Y

   $\mid X \mid$ is the number of terms in document X

   $\mid Y \mid$ is the number of terms in document Y

3. **Jaccard**
   This similarity measure is a slight variant of the dice coefficient and is also based on the commonality between the two documents that we want to compare. The *Jaccard Coefficient* for two documents X and Y is given by:

   $$Jaccard(X,Y) = \frac{\mid X \cap Y \mid}{\mid X \cup Y \mid}$$

This definition of the coefficient gives the ratio of the common terms to the total number of terms between the two documents X and Y. This formulation can be adapted to the frequency model of the document as well as the binary one.

3.2 Conceptual Similarity

Given a hierarchical graph of IS_A relation among concepts, it is observed that *the similarity between two concepts in this has inverse relationship with the distance between them*, and this distance is termed as the *conceptual distance.* There are two basic appraoches based on this theme.

1. **Information Theoretic**
   There are a number of algorithms which apply the principle of conceptual similarity to the WordNet graphs. [11] uses a measure based on the information content that the two concepts share. The basic intuition behind this approach is that the more the information the two concepts share, the more similar they tend to be. The information shared by two concepts is proportional to the information content of the concepts that subsume them in the taxonomy. Formally, define

   $$sim(c_1, c_2) = max_{c \in S(c_1, c_2)}[-\log p(c)]$$

   where $S(c_1, c_2)$ is the set of concepts that subsume both c_1 and c_2. Here $p(c)$ is the probability of encountering an instance of concept c. Notice that although similarity is computed by considering all upper bounds for the two concepts, the information measure has the effect of identifying minimal upper bounds, since no class is less informative than its superordinates. For example, in WordNet, *NICKEL* and *DIME* are both subsumed by *COIN*, whereas the most specific superclass that *NICKEL* and *CREDIT CARD* share is *MEDIUM OF EXCHANGE* (as shown in figure 1). Note that this approach gives only the similarity between

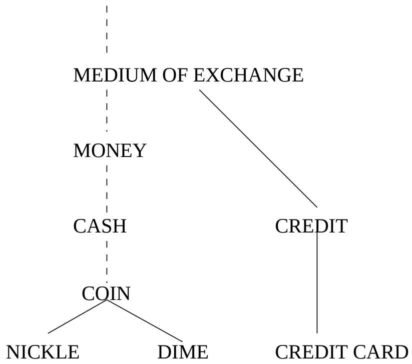


Figure 1: Figure showing the hierarchical relation in the WordNet

   two concepts and not between the two texts.

2. **Link based**
   The simple measure of link distance between two concepts in the WordNet as an estimate of the similarity between them has been used extensively [6]. In this approach, the number of relational links separating the two synsets is taken as a inverse measure of the similarity between them. For example, the distance between the synsets {*car, auto*} and {*vehicle*} is 2 in the WordNet graph, while the distance between {*car,auto*} and {*table (as a furniture)*} is 7. This means that the former pair of synsets are more similar in meaning than the latter.

4 Our system

The block giagram showing the design of our system is given in figure 2. The has 4

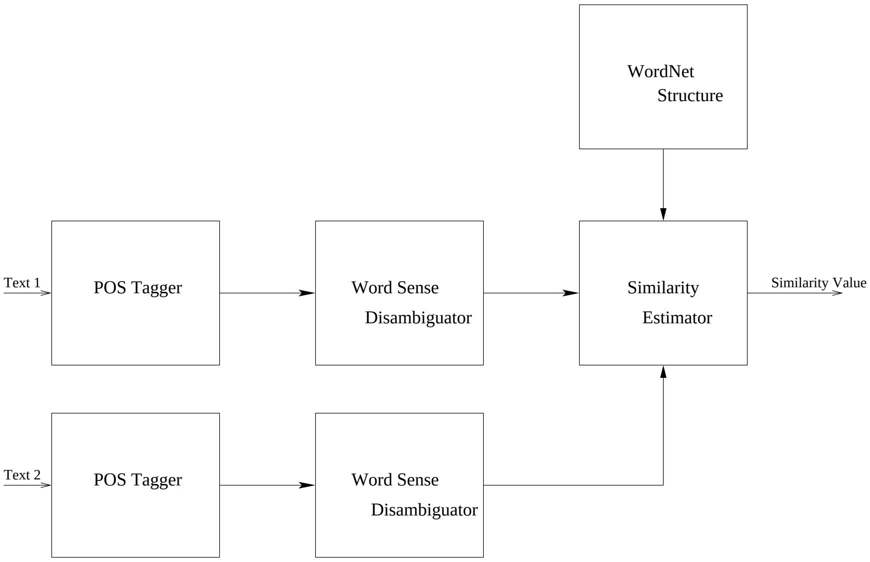


Figure 2: Block diagram of the system

independent modules- *Part-Of-Speech (POS) tagger*, *word sense disambiguator*, *WordNet* and *similarity estimator*. The system works by first extracting the nouns from the text using a part-of-speech tagger which in this case is the java based *Qtag*[7] (*vide* tables 1, 2 and 3 for two sample texts, the corresponding tagged outputs and the nouns extracted there from respectively). These nouns are then fed into the next module- the word sense disambiguator which maps the nouns to the corresponding synsets in the WordNet. These are then passed to the similarity estimator module which uses the WordNet graph and *link distance* to obtain the similarity value.

There are aesthetic and recreational reasons for an interest in fishes. Millions of people keep live fishes in home aquariums for the simple pleasure of observing the beauty and behaviour of animals otherwise unfamiliar to them. Sportfishing is another way of enjoying the natural environment, also indulged in by millions of people every year. Interest in aquarium fishes and sportfishing support multimillion - dollar industries throughout the world.	Fishes are of interest to humans for many reasons, the most important being their relationship with and dependence on the environment. A more obvious reason for interest in fishes is their role as a moderate but important part of the world 's food supply. This resource, once thought unlimited, is now realized to be finite and in delicate balance with the biological, chemical, and physical factors of the aquatic environment. Overfishing, pollution, and alteration of the environment are the chief enemies of proper fisheries management, both in fresh waters and in the ocean.

Table 1: Sample texts T_1 and T_2

There/EX are/BER aesthetic/JJ and/CC recreational/JJ reasons/NNS for/IN an/DT interest/NN in/IN fishes/NNS ./. . . .	Fishes/NNS are/BER of/IN interest/NN to/TO humans/NNS for/IN many/DT reasons/NNS ,/, the/DT most/RBS important/JJ being/BEG their/PP$ relationship/NN with/IN and/CC dependence/NN on/IN the/DT environment/NN ./. . . .

Table 2: POS tagged words from texts T_1 and T_2

reasons, interest, fishes, Millions, people, fishes, aquariums, pleasure, beauty, behaviour, animals, way, environment, millions, people, year, Interest, aquarium, fishes, support, multimillion, dollar, industries, world.	Fishes, interest, humans, reasons, relationship, dependence, environment, reason, interest, fishes, role, part, world, food, supply, resource, balance, factors, environment, pollution, alteration, environment, enemies, fisheries, management, waters, ocean.

Table 3: Nouns from texts T_1 and T_2

5 Word Sense Disambiguator module

The first landmark attempt at word sense disambiguation [14] used a supervised approach based on *Roger's Thesaurus*. It worked on a training corpus to learn the correspondence between the sense of a word and the context in which it occurs. After that, the research on word sense disambiguation received a boost with the development of the WordNet which provided a complex and well organised database of words and its senses. The fundamental Idea behind the use of the WordNet for WSD is first described.

5.1 WordNet based word sense disambiguation

Texts represent flow of ideas through structured sentences. The words which convey the main ideas are commonly known as *keywords*. A common hypothesis in all WSD work is that *although all the words are necessary in a text, it is the nouns the carry the main burden of the expression*. Also important is the observation that given the topic of a text, there is a high probability that most of the words in the text are very closely *related* to the words used for describing the topic. For instance, in a text about *car* we expect to find a lot of words related to *car*, like *steering*, *wheels* etc. This implies that **most of the synsets corresponding to the words in a text lie close to each other when mapped on to the WordNet graph**. This type of constraint imposed on the senses of a group of contiguous words to disambiguate them has been used in many algorithms. The first such algorithm was by Michael Sussna [13] which takes a group of contiguous words in a window, and the sense combination for the words in the window is chosen such that the sum of distances between the synsets corresponding to the word forms and word sense combinations is minimized. Another algorithm that considers this proximity of synsets is the one by Eneko Agirre and German Rigau [1]. In this, they define a *measure of conceptual density* as the number of words in the context window that lie in the sub hierarchy of synsets of the candidate word. The synset of a word is chosen such that the conceptual density is maximized.

The main drawback of these algorithms is that they assume the words lying close to each other in the text to have similar senses. This might not be necessarily true. It has been observed that the reference of a word can extend up to a large distance in the text.

Use of anaphors is a good example in point. Hence, though we can be sure that the text is populated with similar words pertaining to the main topic of the text, we cannot make any such assumption about the *contiguous words* in the text. The proof of such a conclusion lies in the result obtained in [1]. In the evaluation of their algorithm it was seen that the precision of disambiguation increased with the increase in the window size used for disambiguation. This indicates that the constraint of proximity in the WordNet graph works better as the context size increases.

Another flaw in the above description of a text structure is that a text can have more than one theme. This factor must be taken into consideration when designing the WSD algorithms. The idea of capturing the flow of ideas in the text has been previously used in lexical chains [9][4]. The aim in [4] is automatic hyperlinking. Our work has been inspired by lexical chain like structures.

The discussion so far can be summarized by an illustration of how the words in a text can represent the ideas when they are mapped onto the synsets of the WordNet graph.

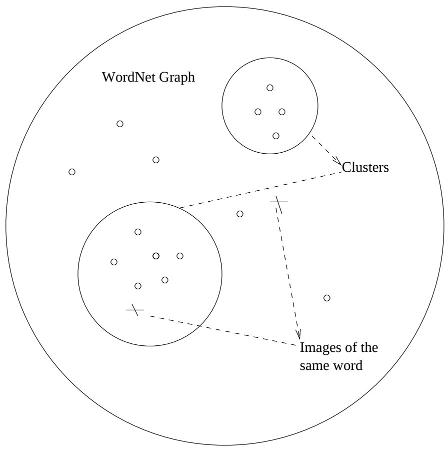


Figure 3: Mapping of words in a text onto WordNet

In figure 3, the large circle represents the WordNet graph. The smaller circles inside them correspond to the clusters formed by putting together similar words in the text. The points within the smaller circles are the synsets corresponding to the words in the text. Each cluster stands for an idea in the text.

When a particular word is not disambiguated there is more than one cluster for the word in the WordNet graph. If our previous assumption of clustering- the fact that *words in the text are within close semantic distance from one another*- is true, then we should be able to choose the sense of the word in such a way that we get a closely packed cluster. The question now is - where do we start from to obtain the clusters in the given text?

5.2 Our approach

We start building the clusters from the Monosems which are words having a single sense. Thereafter, the algorithm works by building a directed acyclic graph (DAG) over the words in the text and using the properties of the *conceptual distance*. We call the DAG the *semantic graph*. In this *semantic graph* the nodes are synsets of WordNet and the links represents the fact that there is a path in the WordNet, between the two synsets within the *search_depth* radius. The steps of the algorithm are now enumerated:

1. **Collect the Monosems:** Monosems form the roots of the semantic DAG that gets built as the WSD algorithm proceeds. For a single word in the text we have all the senses of the word as nodes in the DAG. The number of roots of the DAG becomes equal to the number of monosems in the text.

2. **Initialize the scores of the DAG:** For each node in the DAG, it has a score corresponding to the probability of the word taking that sense. To start with, the roots of the DAG are monosems. These are initialized with a score of 1.

3. **Find the link distance to other words:** Each node in the DAG is the synset corresponding to a word in the text. We take a word from the text that has not been added to the DAG yet and for all synsets corresponding to the word, we find the link distance between the synsets corresponding to the current node of the DAG and the synsets of the selected word from the text. This involves searching the WordNet graph starting from one of the synsets and proceeding in a breadth first fashion till we reach the other synset. As for the links, all kinds of relation links in the WordNet are considered.

   To decrease the time and space complexity of a breadth first search we restrict the search to a cut-off radius which we call **search_depth**. The value of this variable decides the depth upto which the search proceeds. Any word at a link distance more than the *search_depth* is ignored. It is easy to see that the more the *search_depth*, the more is the precision of disambiguation. The *Search_depth* also controls the value of the similarity assigned to pairs of synsets.

4. **Form the semantic DAG:** After finding the distance to the synsets of a word we compare the distances that we have obtained. The word along with all its senses to which a path has been found is added as the child of the DAG node. Two situations arise in this case:

   (a) The new DAG node is already present: This condition arises when the combination of the word and the sense was previously found by a sibling of the current node. In such a case an addtional parent pointer for the existing node is created.

   (b) The new DAG node is not present: In this case the node is created and added as a child of the current DAG node.

   The frontier of the DAG is expanded one level at a time in accordance with the breadth first search. After a particular level has been added to the DAG, it goes onto expanding the next level. When a DAG node is chosen as the current node, the word corresponding to the current selection is permanently removed from the list of words in the text. Therefore *a node corresponding to a particular word at a particular position in the text can occur only once in the DAG*. This is done to avoid cycles. Note that, this does not prevent the same word occurring at different points in the text from appearing more than once in the DAG at different levels.

5. **Pass the score of the parent to the child:** Having initialized the scores of the roots to 1, we need to assign scores to the subsequent nodes at each level. Let us denote a node at level i and sense j as W_j^i. The score of W_j^i depends on the following three parameters.

(a) *Score of parent*: Since a DAG node is added based on its proximity to the synset of the corresponding parent DAG node, the probability of the node calculated as

$$Score(W_j^i) \propto Score(W_j^{i-1})$$

(b) *Distance between the child and the parent*: The score of a new DAG node also depends on its distance from the parent node. The intuition behind this is that similar words are distributed quite close to each other. This phenomenon is evident in texts having multiple paragraphs. Mostly, the topic of discussion changes during the transition from one paragraph to another. Similar words can be found inside a paragraph rather than between paragraphs. Therefore,

$$Score(W_j^i) \propto \frac{1}{Dist(W^i, W^{i-1})}$$

(c) *Link distance in the WordNet*: The score of a word also depends on the link distance of the parent word from the current word in the DAG. The lesser the distance between them, the more similar the child node is to the parent node. We are then more certain about the sense of the child node. The score then is

$$Score(W_j^i) \propto \frac{1}{Link\_dist(W^i, W^{i-1})}$$

where *Link_dist* is the numerical value of the distance between the parent and child nodes in terms of the number of links separating them in the WordNet graph. If this distance is larger than the parameter *search_depth* then the *link_dist* is taken as infinite.

Finally combining,

$$Score(W_j^i) = \frac{Score(W_k^{i-1})}{Dist(W_j^i, W_k^{i-1}) \times Link\_dist(W_j^i, W_k^{i-1})}$$

6. **Judge the sense for a particular word:** Having assigned the scores to all the nodes of the DAG, we compare the scores of all the senses of a word. The sense with the highest score selected. *Since we remove the word from the text after having added it to the DAG, all the senses of a particular word occur at the same level of the DAG.*

The DAG produced by the above algorithm is a subgraph of the WordNet. This subgraph contains most of the words of the given text. A property of the created DAG is that the accuracy of disambiguation decreases with the depth of the nodes. This is because of the *cascading effect of errors* in the disambiguation process. To start with, one is sure about the sense of the words which are monosems. But as the score is passed from one level to another, the certainty about the proportionality between the score and the senses decreases. This means that if an error occurs at a level and a wrong sense of a word gets a high score, then this error can percolate down.

One technique to improve the precision is to maintain a cut-off at some level of the DAG. All the levels below this level are excluded from the disambiguation process.

5.3 An Example to illustrate the Algorithm

We demonstrate the steps of the algorithm using an example of a document from the Semcor corpus shown below:

Nevertheless, "we feel that in the future Fulton_County should receive some portion of these available funds", the jurors said. "Failure to do this will continue to place a disproportionate burden" on Fulton taxpayers. The jury also commented on the Fulton ordinary's court which has been under fire for its practices in the appointment of appraisers, guardians and administrators and the awarding of fees and compensation.

In the above, the underlined words are the nouns in the text and most of them have multiple senses. For example, *portion* has 6 WordNet senses as a noun.

(Step 1) Collect the monosems: Some of the monosems in the given text- as given in the WordNet- are *funds*, *jurors*, *guardians*, *awarding*, and *taxpayers*.

(Step 2) Initialize the scores of monosems: The words *funds*, *jurors*, *guardians*, *awarding* and *taxpayers* form the roots of the DAG and have a score of 1 each.

(Step 3) Find the link distance to other words: From each leaf of the DAG, the link distance is found to the words in the text. For example, a link distance of 2 is found from the sense no. 1 of *funds*- the only sense- to the sense no. 4 of *portion*. This is added as a child of *funds*.

(Step 4) Form the semantic DAG: Step 4 is repeatedly performed to grow the DAG structure. A part of the DAG structure is shown in figure 4.

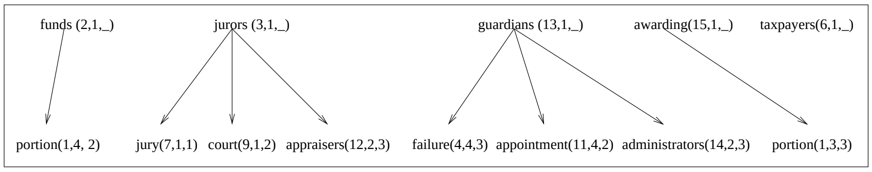


Figure 4: A partial DAG structure

In figure 4, monosems occupy the roots of the DAG. The three numbers given in brackets along with the words represent *the position of the word in the text, the sense number* and *the link distance to that sense from the parent node*, respectively.

(Step 5) Pass the score of the parent to the child: The nodes of the DAG are then assigned scores based on the relation,

$$Score(W_j^i) = \sum \frac{Score(W_k^{i-1})}{Dist(W_j^i, W_k^{i-1}) \times Link\_dist(W_j^i, W_k^{i-1})}$$

where the summation is over all the parents W^{i-1} of W^i. The DAG structure with the scores assigned to the nodes is shown in figure 5.

Figure 5 illustrates the final structure of the DAG that has been built. The numbers in brackets assigned to the nodes in the DAG represent the index of the word in the text, the sense number and the score of that particular node. For example, for the node *portion* the index is 1, the sense no. is 4 and the score is 0.5.

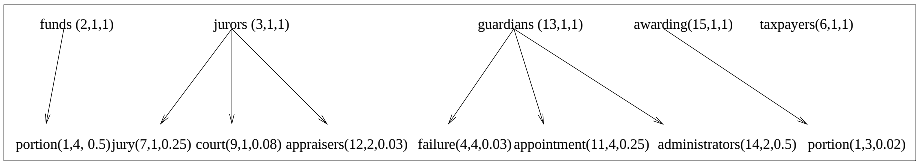


Figure 5: A partial DAG structure with scores

(Step 6) Judge the sense for a particular word: The sense for a word is the one with the highest score. For example, in figure 5, the 4th sense of the word *portion* with the score 0.5 is chosen. Though the actual sense of *portion* in the text is 3, sense 4 too is quite close. Thus the algorithm narrowed down the choice from the six senses to two in this case.

The other good example of disambiguation is *court* which has 9 senses.

5.4 Evaluation of the sense disambiguator module

The evaluation was done using three different parameters, *viz.*, **precision**[2], **recall**[3], and **coverage**[4]. For our experiments, we chose documents from the Semcor corpus [8]. This is a corpus in which the words have been tagged with their sense number in the wordnet, the part of speech and other lexical information.

In our experiment, we chose the Semcor corpus document **br-a01**. This text has been used by other WSD algorithms also [6]. Nouns extracted from this text and stripped of any sense tagging are passed to the proposed algorithm as input. The output is the same set of nouns with the WordNet sense marking. The senses obtained are compared with the actual senses of these nouns. The performance is then computed from the counts we obtain. We performed the experiment for three different values of the search_depth. The results are summarized below.

Search Depth	Precision	Recall	Coverage
3	62.87	32.30	51.38
4	57.50	42.46	73.84
5	51.91	45.84	88.30

Table 4: Results of the experiment

From the results we find a neat variation in the precision and recall values with the values of the search_depth parameter. This variation of precision can be explained by the fact that the search_depth restricts the distance upto which two synsets can be considered to be similar. If we increase the value of this parameter, even synsets that are far apart are considered similar, resulting in a dilution of the similarity measure. This in turn affects the precision of the disambiguation.

The results of our experiments compared with the results of the algorithm using *conceptual density* [6] have been summarized in table 5.

[2]Precision is defined as the ratio of the number of words disambiguated correctly, to the number of words disambiguated in total.

[3]Recall is defined as the ratio of number of words *correctly* disambiguated to the total number of words that were input to the algorithm.

[4]Coverage is defined as the ratio of number of disambiguated words to the total number of words that were input to the algorithm.

Algorithm	Precision	Recall	Coverage
Conceptual density	47.30	39.40	83.20
Semantic graph	62.87	32.31	51.38

Table 5: Comparison of performance

Considering the running example of the two texts in table 1, the sense disambiguation module gets the list of nouns in table 3. These are then disambiguated by our WSD module. The list of disambiguated words with their WordNet sense numbers are shown in table 6. Here, the value of *search_depth* was set to 4.

fishes(1), Millions(1), people(1), fishes(1), aquariums(1), animals(1), millions(1), people(1), aquarium(1), fishes(1), world(1), world(1)	Fishes(3), interest(7), humans(1), interest(7), fishes(2), part(4), world(5), food(1), enemies(3), fisheries(1), waters(1), ocean(1), ocean(1)

Table 6: List of disambiguated nouns

6 Similarity Estimation

We use the conceptual distance between constituent words to estimate the similarity between texts. Constructing a similarity measure between two documents from the similarity of words is a unique feature of our system. The output of the system is a similarity value in percentage.

Steps of the algorithm

The algorithm has three important steps:

- Get the list of synsets in each of the texts.
- For each synset S_1 in $text_1$ do,
  For each synset S_2 in $text_2$ do,
  For i = 0 to MAX do,
  Find all the synsets that are at a distance of i relational links from S_1in the WordNet graph.
  If S_2 is among them then increment count in L_i.
- Compute the similarity using the relations shown below.

L_i is number of links obtained between the synset of two documents at the distance of i.
MAX is the maximum specified radius of search on the WordNet graph. This means that the search for a synset in the neighbourhood of another synset is restricted only to a link distance of MAX. We define the parameter *Total_Weighted_links* as,

$$Total_Weighted\_links = \sum_{i=0}^{MAX-1} L_i \times [MAX - i]$$

Here, we compute the total number of links obtained at various radii of search on the WordNet graph, by weighting the number of links by the proximity of distance. This means that the links at a shorter distance gets more weightage than the links at larger distances. This is a direct result of the fact that conceptual distance is inversely related to the similarity between concepts.

If N_A and N_B are the number of synsets in the two documents A and B, then the maximum possible links is $N_A \times N_B$. Then the *link_similarity* is defined to be,

$$link\_similarity(A, B) = \frac{Total\_Weighted\_links}{maximum\ number\ of\ links}$$

$$= \frac{\sum_{i=0}^{MAX-1} L_i \times [MAX - i]}{N_A \times N_B}$$

This measure gives just the plain similarity in terms of the links between the two texts A and B. This similarity measure does not satisfy the property *link_similarity(A,A)= 1* because the numerator that stands for the total weighted number of links need not be equal to the denominator which is the maximum possible number of links. Hence, we need to normalise this measure with the self-similarity of the text documents themselves. The final measure of similarity between the two text documents is,

$$semantic\_similarity(A, B) = \frac{link\_similarity(A, B)}{\sqrt{link\_similarity(A, A) \times link\_similarity(B, B)}}$$

It is interesting to note the correspondence between this measure and the dot-product of the cosine similarity.

As is apparent from the above algorithm, the similarity measure between two documents is found from a similarity measure of the constituent words.

6.1 An example

We illustrate the working of this module using the running example of previous sections. The list of disambiguated words given in table 6 is passed as input to this module. It produces as output the number of links at each of the distances less than *MAX*. These are then used to find the similarity between the two texts. Here, the value of *MAX* used was 3.

people $\Longrightarrow$ human(1)
people $\Longrightarrow$ world(1)
people $\Longrightarrow$ enemy(1)
animal $\Longrightarrow$ human(2)

Table 7: The links between the two texts

In the table 7 the left hand side of the arrow corresponds to word from the first text and the right hand side corresponds to those from the second text. The number given in the bracket gives the link distance between the two words. Table 7 gives the links found between text1 and text2. Tables 8 and 9 give the self similarity links for the two texts.

million $\Longrightarrow$ million (0)
people $\Longrightarrow$ people(0)
aquarium $\Longrightarrow$ aquarium(0)
animal $\Longrightarrow$ animal(0)

Table 8: The self similarity links for text1

fish $\Longrightarrow$ fish(0)
fish $\Longrightarrow$ food(2)
interest $\Longrightarrow$ interest(0)
human $\Longrightarrow$ human(0)
human $\Longrightarrow$ world(2)
human $\Longrightarrow$enemy(2)
part $\Longrightarrow$ part(0)
world $\Longrightarrow$ human(2)
world $\Longrightarrow$ world(0)
world $\Longrightarrow$ enemy(2)
food $\Longrightarrow$ fish(2)
food $\Longrightarrow$ food(0)
enemy $\Longrightarrow$ human(2)
enemy $\Longrightarrow$ world(2)
enemy $\Longrightarrow$ enemy(0)
fishery $\Longrightarrow$ fishery(0)
water $\Longrightarrow$ water(0)

Table 9: The self similarity links for text2

From table 7 we find that there are 3 links at distance 1 and 1 link at distance 2. The number of synsets in text1 and text2 are 12 and 13 respectively. Therefore,

$$link\_similarity(text1, text2) = \frac{3 \times 1 + 2 \times 3 + 1 \times 0}{12 \times 13}$$

$$link\_similarity(text1, text2) = 0.057$$

Similarly we compute *link similarity(text1, text1)* and *link similarity(text2, text2)*. These values are then plugged into the formula for *semantic_ similarity* to get its final value as **61.69%**.

7 Evaluation of the System

The efficacy of the similarity measure that we have evolved has been tested against the universally used cosine similarity measure. This is consistent with the practice adopted in the field of text clustering research.

The experiments have been performed in three different settings.

1. The first is the one in which just the basic task of *computing and comparing the similarity measures* is performed. Ninety nine documents from 17 classes are taken and pairwise inter-class similarity is computed using both our measure and the cosine similarity measure.

2. The second setting is *classification*. These 99 documents are split into training and testing sets and are put into classes using both our measure and the cosine similarity measure. The accuracy of the classification on the test set throws light on the effectiveness of the similarity measure.

3. The third setting is *clustering*. The 99 documents are clustered using both our measure and the cosine similarity measure.

In the first setting we wanted to test the *raw effectiveness* of the proposed similarity measure. Since multiple senses of words interfere, we have performed the tests with 100% correct sense tagged texts. We have used the *Semcor* corpus which is a subset of the famous *Brown corpus* tagged with WordNet sense number of the words. We call this the ideal situation.

In the second setting- which is more realistic- the text tagged with only part-of-speech but no sense tagging has been taken through the word sense disambiguation stage followed by classification using the $K - NN$ method. The underlying similarity measures for classification are our measure and the cosine similarity measure. The accuracy of the classification is expected to stand witness to the effectiveness of the similarity measure.

In the third setting the text tagged with only part-of-speech but no sense tagging has been taken through the word sense disambiguation stage followed by clustering. The underlying similarity measures for clustering are our measure and the cosine similarity measure. The accuracy of the clustering is expected to throw light on the effectiveness of the similarity measure.

All the experiments are performed using only the nouns in the documents. The reason- it is reiterated- is that nouns carry the burden of information in the text.

7.1 Corpus description

We have used the documents of the Semcor corpus for the purpose of evaluation. Semcor corpus is a subset of the *Brown Corpus*[3] consisting of texts that have been tagged with POS and the WordNet senses. Semcor consists of about 186 documents classified into 20 classes. We have chosen a subset of the documents and the classes as shown in the table 10.

S.No.	Class Names	Document identification in Brown corpus	Number of documents
0	Political	br-a01- a02,b20,g01,h21,j37-38,j42	8
1	Law	h9,12,16,17	4
2	Taxation	h24	1
3	Science and Technology	e25-26,g11,d22,j01-20,j53	25
4	Science Fiction	m01-02	2
5	Arts	c04,g16,j59	3
6	Fiction(Mystery)	l01-l18	11
7	Fiction(Adventure)	n05,n09-17,n20	10
8	Fiction(Romance)	p01,p07-10,p12,p24	6
9	Fiction(Humour)	r04-09	6
10	Literature	g12,g15,g28,g44	4
11	Psychology	j29,j31	2
12	Management	g20,j30	2
13	Linguistics	j32-35	4
14	History	j54	1
15	Religion	d01-04,g21	5
16	Sports	a11-a15	5

Table 10: Classes and their respective documents

When the experiments are performed with WSD module on, this data is passed on after stripping the sense information.

7.2 Setting-1: Basic comparison of similarity measures

To get the feel of the similarity values of the documents in the corpus, we used the sense tags of the words given in the Semcor corpus. The nouns along with their senses were fed into the similarity estimator module. This corresponds to the *fully correct sense* criterion as discussed earlier. The same nouns were then used to obtain the similarity values as given by the cosine similarity measure.

For the 17 classes described in table 10 the mean and variation of similarity have been shown in tables 11 and 12.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
0	0.32,0.27																
1	0.22,0.08	0.44,0.33															
2	0.23,0.06	0.29,0.04	1.00,0.00														
3	0.13,0.10	0.13,0.10	0.15,0.10	0.20,0.19													
4	0.27,0.15	0.30,0.13	0.40,0.01	0.27,0.19	0.87,0.13												
5	0.16,0.10	0.16,0.08	0.18,0.05	0.16,0.12	0.34,0.15	0.46,0.39											
6	0.21,0.16	0.25,0.16	0.33,0.12	0.22,0.20	0.61,0.24	0.29,0.18	0.60,0.32										
7	0.19,0.15	0.22,0.15	0.29,0.11	0.21,0.19	0.58,0.21	0.27,0.17	0.53,0.29	0.57,0.28									
8	0.21,0.16	0.23,0.16	0.32,0.12	0.21,0.20	0.61,0.23	0.29,0.18	0.56,0.29	0.53,0.26	0.60,0.31								
9	0.22,0.13	0.25,0.14	0.32,0.10	0.22,0.18	0.59,0.21	0.31,0.13	0.53,0.26	0.49,0.25	0.52,0.25	0.57,0.29							
10	0.21,0.11	0.21,0.11	0.23,0.10	0.18,0.14	0.43,0.18	0.32,0.14	0.36,0.21	0.33,0.19	0.36,0.21	0.38,0.19	0.53,0.30						
11	0.21,0.08	0.20,0.06	0.24,0.02	0.19,0.10	0.41,0.05	0.26,0.09	0.37,0.11	0.33,0.10	0.35,0.12	0.39,0.11	0.31,0.09	0.64,0.36					
12	0.27,0.13	0.29,0.10	0.32,0.01	0.24,0.16	0.57,0.06	0.31,0.12	0.50,0.19	0.45,0.18	0.48,0.20	0.49,0.16	0.37,0.16	0.41,0.03	0.76,0.24				
13	0.16,0.09	0.18,0.07	0.20,0.03	0.17,0.11	0.41,0.06	0.21,0.11	0.32,0.14	0.30,0.12	0.31,0.14	0.32,0.11	0.27,0.10	0.27,0.04	0.36,0.06	0.48,0.31			
14	0.24,0.11	0.25,0.09	0.34,0.00	0.23,0.13	0.55,0.02	0.33,0.11	0.46,0.17	0.41,0.15	0.46,0.17	0.46,0.14	0.41,0.15	0.33,0.03	0.43,0.03	0.29,0.04	1.00,0.00		
15	0.15,0.09	0.13,0.10	0.13,0.11	0.11,0.10	0.21,0.19	0.19,0.12	0.16,0.18	0.14,0.17	0.16,0.18	0.19,0.17	0.23,0.13	0.19,0.09	0.21,0.16	0.13,0.10	0.23,0.14	0.35,0.33	
16	0.14,0.05	0.10,0.04	0.17,0.04	0.07,0.05	0.12,0.03	0.10,0.08	0.10,0.04	0.08,0.05	0.12,0.06	0.13,0.04	0.09,0.03	0.13,0.03	0.11,0.03	0.05,0.02	0.18,0.04	0.11,0.04	0.60,0.22

Table 11: Class pair similarity values for sense based similarity

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
0	0.20,0.30																
1	0.08,0.04	0.36,0.38															
2	0.07,0.04	0.13,0.03	1.00,0.00														
3	0.04,0.03	0.03,0.02	0.02,0.02	0.09,0.19													
4	0.08,0.03	0.05,0.03	0.07,0.00	0.05,0.03	0.59,0.41												
5	0.04,0.02	0.03,0.02	0.02,0.01	0.05,0.04	0.07,0.03	0.38,0.44											
6	0.05,0.03	0.04,0.02	0.07,0.04	0.04,0.03	0.13,0.04	0.07,0.04	0.28,0.23										
7	0.04,0.03	0.03,0.03	0.04,0.02	0.04,0.03	0.12,0.04	0.05,0.03	0.18,0.07	0.27,0.25									
8	0.05,0.03	0.03,0.02	0.05,0.03	0.04,0.03	0.13,0.03	0.08,0.04	0.17,0.06	0.16,0.07	0.31,0.32								
9	0.06,0.03	0.03,0.02	0.05,0.01	0.04,0.03	0.12,0.03	0.08,0.05	0.15,0.04	0.12,0.05	0.13,0.05	0.27,0.33							
10	0.06,0.05	0.02,0.01	0.02,0.01	0.04,0.03	0.10,0.03	0.12,0.06	0.08,0.03	0.07,0.04	0.10,0.05	0.10,0.03	0.37,0.36						
11	0.06,0.04	0.03,0.01	0.04,0.00	0.05,0.04	0.08,0.03	0.07,0.06	0.08,0.05	0.06,0.04	0.07,0.05	0.09,0.05	0.07,0.03	0.53,0.47					
12	0.08,0.03	0.04,0.02	0.05,0.02	0.06,0.03	0.10,0.02	0.06,0.03	0.11,0.04	0.08,0.04	0.10,0.03	0.09,0.02	0.08,0.01	0.09,0.02	0.55,0.45				
13	0.03,0.03	0.02,0.01	0.02,0.01	0.04,0.02	0.06,0.02	0.04,0.02	0.04,0.02	0.03,0.02	0.03,0.02	0.04,0.02	0.05,0.03	0.06,0.02	0.08,0.03	0.29,0.41			
14	0.07,0.03	0.02,0.01	0.02,0.00	0.04,0.03	0.06,0.00	0.09,0.04	0.05,0.02	0.04,0.02	0.06,0.04	0.07,0.03	0.14,0.03	0.05,0.01	0.06,0.00	0.03,0.01	1.00,0.00		
15	0.07,0.05	0.04,0.03	0.04,0.04	0.03,0.03	0.10,0.02	0.08,0.06	0.09,0.05	0.08,0.06	0.11,0.06	0.09,0.03	0.12,0.05	0.06,0.03	0.08,0.02	0.03,0.02	0.12,0.05	0.30,0.35	
16	0.06,0.03	0.04,0.02	0.10,0.02	0.03,0.02	0.08,0.02	0.05,0.04	0.08,0.03	0.07,0.03	0.11,0.06	0.07,0.02	0.04,0.02	0.04,0.01	0.05,0.02	0.02,0.01	0.05,0.02	0.05,0.02	0.50,0.26

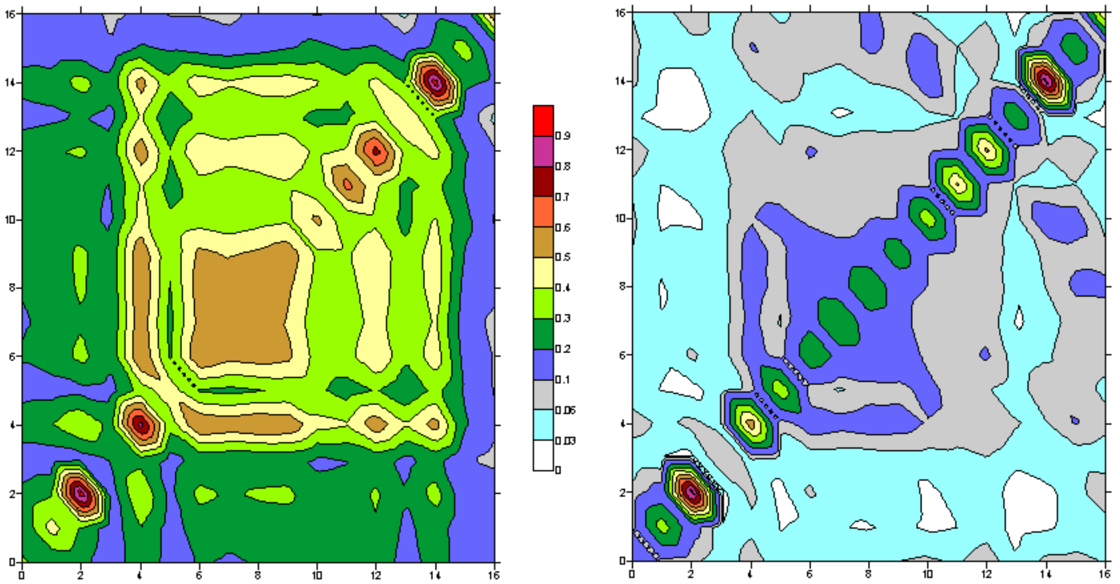
Table 12: Class pair similarity values for cosine based similarity

The former is for sense similarity while the latter is for cosine based similarity. The mean and variance of similarity for two given classes C_i and C_j, of documents are given by,

$$Mean(i,j) = \frac{\sum_{d_i,d_j \in C_i,C_j} sim(d_i,d_j)}{|C_i| \times |C_j|}$$

$$Variance(i,j) = \sqrt{\frac{\sum_{d_i,d_j \in C_i,C_j} \left(Mean(i,j) - sim(d_i,d_j)\right)^2}{|C_i| \times |C_j|}}$$

The tables 11 and 12 are represented by plotting the contour graphs and wireframe models. For each pair of classes we plot the contour graphs of the mean similarity values computed using both the cosine and the sense based similarity measures. In these graphs, the X and Y axes represent the classes and the colours represent the similarity gradients. Figure 6(a) represents the contour map for sense based similarity and figure 6(b) represents that for the cosine similarity values. Similarly, the figures 7(a) and 7(b) represent the wireframe graphs for the mean of classes based on sense based similarity and cosine based similarity respectively.



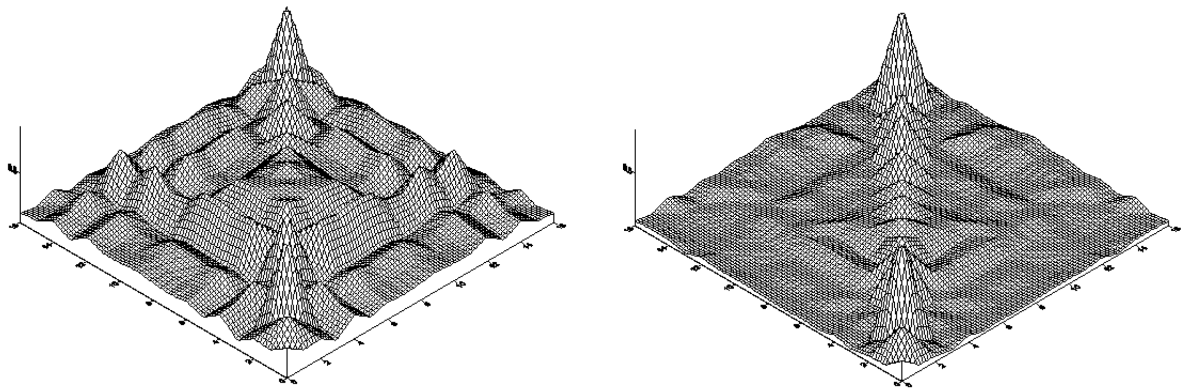
(a) Sense (b) Cosine

Figure 6: The contour plot for cosine vs sense based similarity

Discussion of Results for Setting-1 Experiments

Let us try to examine the properties of the matrices that we have obtained in tables 11 and 12. These lower triangular matrices represent the inter-class mean similarity values. For every tuple represented in a cell of the table, the first part stands for the mean and the second part for the variance.

It is seen from the diagonal of the matrix that the mean similarity values exceed the values in the same column and in the same row. This is expected since the diagonal values are self similarity numbers. We also note that the mean self similarity values is high in class 4 which is *Science Fiction* as well as in class 12 which is *Management*. This



(a) Sense (b) Cosine

Figure 7: The wireframe plot for cosine vs sense based similarity

is due to consistent use of almost the same set of nouns across documents within the class. The mean self similarity is pretty low in class 3 and class 15 which are *Science* and *Religion* respectively. The former is due to the fact that *Science*, as a class is extremely diverse with multiple disciplines in it which have their specific terminologies. In the case of *Religion*, nouns typically are used in very many senses. In all the cells we observe that the variance is small ranging from 0.05 to 0.3, which means that the similarity values are not widely dispersed. This gives us confidence in the mean similarity value as a measure.

Now, we make some interesting observations about the cross similarity values. Class 6 is an interesting point. The similarity among classes 6, 7, 8 and 9 are close to their self similarity values. These documents all belong to *Fiction* category- *Mystery*, *Adventure*, *Romance*, and *Humour* respectively. Works of fiction tend to use similar words and phrases. This fact leads to the high cross-similarity values. This observation is further vindicated by the behaviour of the classes 0, 1 and 2 which are *Politics*, *Law* and *Taxation* respectively. Documents in these areas tend to use similar terminology. Coming to low cross-similarity values we see this happening in the row 16 of table 11. Documents in class 16 pertain to *Sports* and hence share few terms with documents of other classes like *Religion*, *Literature*, *Science* etc.

The above observations based on numbers are brought home very effectively with the contour maps and the wireframe diagrams shown in figures 6 and 7. The region around the center of the contour map spanning coordinates 5-9 in both X and Y axes shows reasonably high similarity value in brown colour. These are the documents from literary classes as mentioned above. When one steps out of the region bounded between 3 and 15 on either axes, one observes a sharp decrease in the similarity values. This transition is into sports documents on one side, and politico-legal documents on the other side. The difference in literary styles and terminology choices are brought out by this factor.

Comparison

One immediate observation from the tables, contour maps and wireframes is that the similarity values are shown in a more enhanced and sharper fashion by the sense based similarity measure. We see more details in the contour map and the wireframe for sense based similarity, than in those for cosine similarity. For example, the terrain between $X = 7$ and $X = 11$ show more separate regions in the wireframe diagram for sense based similarity. This highlights difference among the similarity measures between classes like Psychology, Romance and Humour. On the other hand, if we follow the diagonal we see better separated regions in the cosine similarity measure based diagrams. This measure seems to better highlight the difference between self similarity values.

On comparing the two graphs we find that the similarity values are more uniformly distributed over the documents in the case of sense based similarity measure. Also, we find that distinct square regions of high similarity are formed among the documents of the same class. This is not so prominent in the case of cosine similarity.

7.3 Setting 2: K-NN classification

In setting-1, we just looked at the qualitative aspect of the similarity measures obtained and compared our similarity measure with that of cosine similarity. In this section we try to obtain a quantitative evaluation for our similarity measure.

From the documents given in table 10, the nouns were extracted. For one experiment we chose the senses of these nouns from the Semcor itself whereas in the other experiment we passed these nouns to the disambiguator module. These correspond to the situations of *fully correct sense* and *partially correct sense* respectively. The disambiguated nouns in each case were passed finally to the similarity estimator. From these we got two sets of inter-document similarity values.

A similar procedure was carried out to get the inter-document cosine similarity values. In order to meaure the goodness of these inter-document similarity values, we used these values to classify the documents using the K-Nearest Neighbourhood (K-NN) classification method. In this method some prototypes are fixed as representing a class. When a test case is encountered it is compared with the prototypes and K nearest prototypes are selected. The class of the test case is decided by the class to which the majority of its neighbours belong. Here, out of the 99 documents in the corpus, 75% were used as prototypes of classes and the rest as test cases. The result of these trials are given in figure 8.

The three cases that we had discussed have been shown in the graphs. The case where we have taken the senses directly from the tags in Semcor corresponds to the one marked as *Fully correct senses*. This is so, because the word senses provided are manually tagged ones and are expected to be 100% correct. The one marked *Partially correct senses* corresponds to the trial where we have disambiguated senses of the nouns using our algorithm for word sense disambiguation. The third one marked *cosine* corresponds to the cosine similarity.

Discussion of Results for Setting-2 Experiments

As we can see, the trend is not very clear from the graphs. For small values of K the accuracy is quite high, and both the measures perform equally well. For values of K between 20 and 50, the sense based similarity is better than the cosine similarity for the first case of fully correct senses. But after K exceeds 50, all the three plots plummet to a constant accuracy. This of course is due to the fact that the value of K far exceeds the cardinality of the training set. When K exceeds the cardinality of training set, all the documents in the training set are considered neighbours of the test document and there is no change with the increase in K beyond that point.

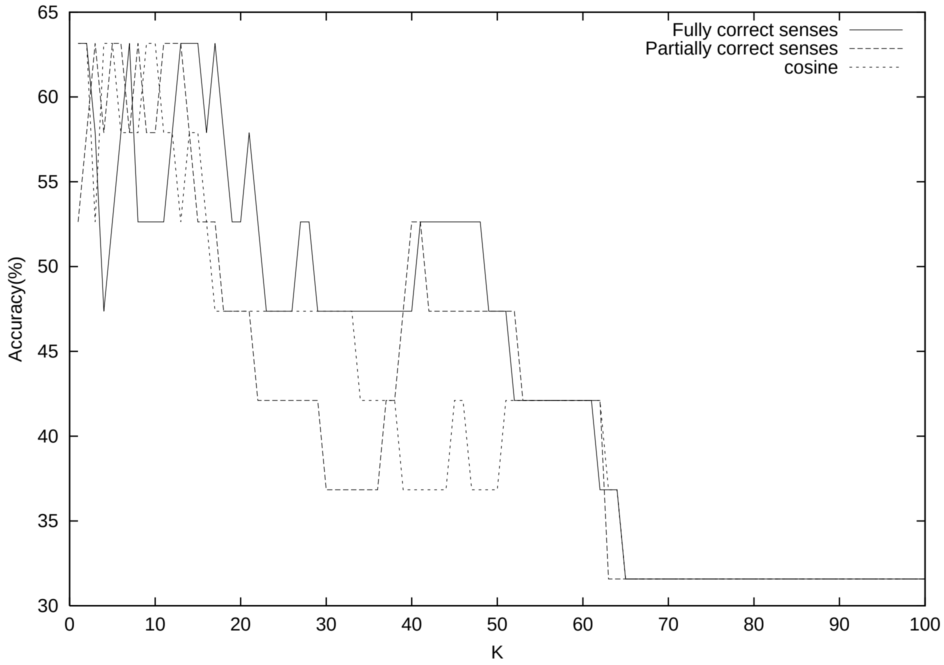


Figure 8: Plots of the K-NN classification

7.4 Setting-3: Clustering

Unable to draw any firm conclusions due to the uncertainty in the value to be chosen for K, we preformed clustering experiments. We employed *Hierarchical Agglomerative Clustering*[5]. It is a method of clustering in which clusters are built hierarchically by merging existing clusters with the highest similarity to form new clusters. The basic requirement for this clustering is that the similarities between the documents be available. The steps of the clustering technique are:

- Initialize clusters with the individual documents, *i.e*, the initial cluster consists of just a single document.
- Choose the cluster pair with the highest similarity and merge them into a single cluster.
- Update the similarity values between the newly formed cluster and the rest of the clusters.
- Repeat this process till the required number of clusters are formed.

The third step is the most important where we define inter-cluster similarity using the similarity values of the documents constituting the cluster. Various measures can be used for this. The most commonly used is the maximum similarity value between the constituents of the two clusters. Expressed mathematically it is,

$$Sim(A, B) = max(Sim(A \times B))$$

where A and B are two clusters and $A \times B$ is the cross-product of the two clusters.

This algorithm was used to cluster the corpus of 99 documents derived from Semcor under the three test criteria discussed earlier.

Though the clustering process is basically an unsupervised approach, to measure the accuracy of clustering, we need to label the clusters and find the number of documents

Algorithm	POS accuracy(%)	WSD accuracy(%)	Search_depth	MAX	Clustering accuracy
Fully correct sense	100	100	-	3	37.37
Fully correct sense	100	100	-	2	36.36
Partially correct sense	100	47.01	3	3	33.33
Partially correct sense	100	47.01	3	2	39.39
Cosine	-	-	-	-	35.35

Table 13: The accuracy of clustering

that have been wrongly classified. In the above algorithm this was done by stopping the hierarchical building process when there were 17 (the number of classes in the corpus) clusters. The clusters were labeled by the majority class of the documents in the cluster. The rest of the documents were then assigned the class corresponding to the cluster to which it belongs. This was then compared with the actual classes of the document. The accuracy of clustering is then based on the number of matches.

Discussion of Results for Setting-3 Experiments

From the above experiment we find that the clustering accuracy is the highest for the case where we have the correct word senses. The first entry in the table 13 represents the ideal case where we have cent percent accuracy for all the modules. Also, we see that by varying the parameter **search_depth** we can increase the accuracy of the word sense disambiguation and hence increase the accuracy of the similarity algorithm. The role of the parameter **MAX** in relation to the accuracy of the similarity algorithm is not very clear, but a few words are in place regarding the two parameters in the system and their influence on the working of the algorithm.

7.5 Influence of Search_Depth and Similarity_Radius

The two parameters *search_depth* and *MAX* stand for the same property and are related to the control of similarity in the WordNet graph. By specifying a maximum radius of influence of the similarity measure, we are able to control the definition of similarity. The more the value of *MAX* or *search_depth*, the more is the dilution of similarity, leading to a reduction in precision. This is very evident in the behaviour of *search_depth* in determining the precision-recall balance in word sense disambiguation. In the case of *MAX*, it influences the noisy links obtained during the similarity measurement. When we go on increasing this parameter there is a high probability that a path is found between two words. This brings in a lot of noise in the form of useless links. Although the links with higher path length are penalized, the sheer number of such links overwhelms the measure. On the other hand if we minimize the value of *MAX*, in the extreme case of $MAX = 0$ it reduces to the cosine similarity. It then becomes important for us to determine the optimum value for these parameters. Typically it can be assigned a value less than the average radius of the WordNet graph.

8 Conclusion and future work

From our evaluation report of the three part experimentation it is clear that the proposed similarity measure based on the *resources of the WordNet* is highly promising. It highlights the similarity where it should, leads to better K-NN classification for appropriate values of K and achieves superior clustering- all in comparison to the classical cosine similarity measure. Thus this work should be looked upon as an attempt to **introduce semantics into similarity determination**. By doing this we have tried to bridge the gap between the linguistics based text understanding and the task of information retrieval. Information retrieval has traditionally been dominated by stochastic methods for similarity measurement. Through the implementation of the system we have demonstrated the use of lexical knowledge and semantics in information retrieval.

The future work consists in:

- Improving the efficiency of the similarity computing algorithm through mechanisms of faster access to the WordNet data structures.
- Increasing the algorithm performance by involving words of parts-of-speech other than noun also. Here a verb centric sentence representation scheme like the Universal Networking Language[15] will be of great help.
- Enhancing the accuracy of the word sense disambiguation module. Any improvement in the WSD accuracy directly impacts the accuracy of the similarity measure.

References

[1] Eneko Agirre and German Rigau. A proposal for word sense disambiguation using conceptual distance. In *Proceedings of the First International Conference on Recent Advances in NLP*, 1995.

[2] Christiane Fellbaum, editor. *WordNet, An Electronic Lexical Database*. The MIT press, 1999.

[3] Francis and Kucera. *Computatinal Analysis of present day American English*. Brown University Press, 1967.

[4] S. Green. Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering*, 11(5):713–730, 1999.

[5] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs NJ, U.S.A., 1988.

[6] J. Lee, M. Kim, and Y. Lee. Information retrieval based on conceptual distance in isa hierarchies. *Journal of Documentation*, 49:188–207, 1993.

[7] Oliver Mason. Qtag: A portable parts of speech tagger. http://www.clg.bham.ac.uk/QTAG/, 1998.

[8] G. Miller, C. Leacock, T. Randee, and R. Bunker. A semantic concordance. In *proceedings of the 3rd DARPA Workshop on Human Language Technology*, pages 303–308, 1993.

[9] J. Morris and G. Hirst. Lexical cohesion, the thesaurus, and the structure of text. *Computational Linguistics*, 17(1):211–232, 1991.

[10] R. Rada, H. Milli, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 1(9):17–30, 1989.

[11] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research (JAIR)*, 11:95–130, 1999.

[12] Gerald Salton, editor. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[13] Michael Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *Proceedings of the Second International Conference on Information and Knowledge Management*, 1993.

[14] D. Yarowsky. Word sense disambiguation using statistical model of roger's categories trained on large corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1992.

[15] Meying Zhu and Hiroshi Uchida. Universal networking language specification. Technical report, UNU/IAS/UNL Center, 1998.