

# **Prepositional phrase attachment ambiguity resolution using word sense hierarchies**

**Kailash Nadh**  
**M00067302**

Supervisor: Dr. Christian Huyck

Undergraduate thesis project  
Middlesex University, London, U.K  
February - April 2008

# Declaration

I, ..... declare that this thesis and the work presented in it are my own and has been produced by me as a result of my own original research work. This work has not been previously submitted anywhere and all sources of information used have been duly acknowledged.

Signature: .....

Date: .....

# Abstract

In recent years, many methods have been proposed for resolution of the prepositional phrase (PP) attachment ambiguity prevalent in English language in natural language parsing. This paper proposes a method of using partially supervised algorithms to use word sense hierarchies (hierarchy of hypernyms of words) to construct complex graphs of ambiguous components in a sentence, analyse their relationship with supervised data and try to resolve the ambiguity. This experiment uses the Penn Treebank text corpora for constructing the training and test datasets and WordNet (a large lexical database) as the source for the semantic data (word senses) and achieves an accuracy of 87.23% in resolving the PP-attachment ambiguity.

# Acknowledgements

I wish to express my gratitude, which cannot be overstated, to the immensely knowledgeable Dr.Christian Huyck, my supervisor, for his exceptional support, guidance and input throughout the project. I thank Dr.Christian for accepting me as his student and letting me work on this thesis for my under graduation, which would otherwise have not been possible.

I thank Illustrator's Lounge for providing the high performance computer required for the experiment.

My special thanks to Deepa for proof reading and pointing out errors and inconsistencies in the writing.

Gratitude would not equal what I wish to express to my parents and my younger brother for their wondrous support. I am forever indebted to my father who made possible my education in the U.K.

# Table of contents

Title	Page
<b>Declaration</b>	Ii
<b>Abstract</b>	Iii
<b>Acknowledgements</b>	Iv
<b>1 Introduction</b>	1
1.1 The PP-attachment ambiguity	1
1.2 Aims	2
<b>2 Literature Review</b>	3
2.1 The corpus	3
2.2 Word sense hierarchies – WordNet	4
2.3 PP-attachment ambiguity resolution systems	5
2.3.1 Ratnaparkhi’s Maximum Entropy Model	6
2.3.2 Zhao and Lin’s Nearest-Neighbor Method	6
2.3.3 Nakov and Hearst’s method using the Web as a training set	7
2.4 Syntactic ambiguity resolution by humans	7
2.5 Machine learning (ML)	8
<b>3 Implementation and Findings</b>	9
3.1 Selection of sentences with PP ambiguity from the corpus	9
3.2 WordNet parser	11
3.3 Quadruples, Training and Test data	11

3.3.1	Graphs of quadruple word sense hierarchies	13
3.4	The test	17
3.4.1	Intersection of graphs	17
3.4.2	Attachment decision	19
3.5	Results	20
3.6	Comparison with previous work	21
<b>4</b>	<b>Conclusion and discussion</b>	<b>22</b>
<b>Appendix 1:</b>	<b>Acronyms</b>	<b>23</b>
<b>Appendix 2:</b>	<b>Sample quadruples</b>	<b>24</b>
<b>Appendix 3:</b>	<b>Programs and scripts</b>	<b>25</b>
<b>Appendix 3.1:</b>	<b>PP-attachment parser – sample Perl script</b>	<b>26</b>
<b>Appendix 3.2:</b>	<b>WordNet parser – sample Ruby script</b>	<b>28</b>
	<b>References</b>	<b>31</b>

## Figures

Title	Page	
2.2	Sample word sense hierarchies	5
3.1	Results of the selection process run on the WSJ corpus	10
3.3a	Sample quadruples generated from the WSJ corpus	11
3.3b	Distribution of verb/noun attachments in the dataset	12
3.3c	Training dataset and Test dataset	12
3.3d	Distribution of verb/noun attachments in the training and test datasets	13
3.3.1a	A sample hierarchy of the verb ‘see’	14

3.3.1b	Constituents of a sample quadruple graph ( <i>see-girl-with-telescope</i> )	15
3.3.1c	A sample quadruple graph ( <i>see-girl-with-telescope</i> )	16
3.3.1d	Nodes of a quadruple graph ( <i>see-girl-with-telescope</i> )	16
3.4.1a	Intersection of graphs	18
3.3.1b	Sample attachment decision	19
3.5	Final test results	20
3.6	Comparison with previous work	21

# Introduction

In natural language processing (NLP), the presences of syntactic ambiguities make attaining fidelity in language parsing an arduous task. The PP-attachment ambiguity is one such syntactic ambiguity, which this paper proposes a method to resolve.

## 1.1 The PP-attachment ambiguity

PP-attachment ambiguity arises when a sentence contains a prepositional phrase (PP), after a verb complemented by a noun, making it syntactically ambiguous to determine what the PP attaches to, the verb or the noun and often tend to semantic errors in NLP (Dekang Lin, 1998).

A ubiquitous instance of PP-attachment ambiguity is the following sentence:

*“I saw the girl with the telescope”*

This sentence could be perceived in different ways, depending on how the PP is attached. If the PP is interpreted to have attached to the noun ‘*girl*’, the sentence implies that the girl was seen holding a telescope. If the PP is considered to have attached to ‘*saw*’, the sentence implies that the girl was seen through the telescope.

There have been a number of possible techniques used for solving the PP-attachment problem. These usually include training on corpora of colossal sizes (M.Banko and E.Brill, 2001) and constructing statistical models for predicting PP-attachment decisions (Ratnaparkhi, Reynar and Roukos, 1994).

## 1.2 Aims

To evaluate the semantic relationship of a phrase with PP-attachment ambiguity with samples whose attachment information is known and predict the attachment based on that. The system I developed for this project constructs a hierarchy of word senses (a collection of different hypernyms of a word) for the verb-noun1-preposition-noun2 quadruple in an ambiguous sentence and counts the number of intersections that might occur in the constructed hierarchies of known data to obtain the number of times the PP attaches to the verb and the noun using partially supervised algorithms. Word senses are obtained from the sense database of WordNet. This technique is applied to a large corpora of ambiguous sentences obtained from the Penn Treebank, in order to create a training dataset. Once enough data is obtained, results are recorded and the precision of the method at resolving PP-attachment ambiguities evaluated.

# Literature Review

The purpose of this section is to present a review of a few pieces of research that have proposed and taken practical approaches to solving the PP-attachment ambiguity. This segment would also provide an overview of a few related concepts and the corpus, testing and training data used in this project.

## 2.1 The corpus

This project uses the Wall Street Journal (WSJ) text corpus from the Penn Treebank corpora for analysis. The WSJ text corpus contains over a million words in phrases annotated in the Treebank II format, collected from a few thousand select stories from WSJ material of 1989. (M.P. Marcus, B. Santorini, and M.A. Marcikiewicz, 1995). The corpus has a standard annotated tree structure, manually annotated by lexicographers. The Penn Treebank corpus features a highly comprehensive tagging style which depicts parts-of-speech tags and also includes semantic and linguistic information of the contents. The trees in the Penn Treebank corpora follow a standard balanced bracketing scheme, tagging sentences as a hierarchy, preserving the relationship of various elements in the annotated material.

A sample Penn Treebank style annotation.

**Text:** *I saw the girl with the telescope.*

**Annotated version:**

```
(  
  (S  
    (NP (PRP I))  
    (VP (VBD saw)  
      (NP (DT the) (NN girl))  
      (PP (IN with)  
        (NP (DT the) (NN telescope))))  
  (. .)
```

)

)

In the above example, all the words of the phrase have been bracketed and marked with the Penn Treebank II tags. *NP* stands for Noun Phrase, *VP* stands for Verb Phrase. There are about eighty tags in the Penn Treebank II specification.

## 2.2 Word sense hierarchies - WordNet

WordNet (WN) is a large lexical database for the English language, which is comprised of various elements of it grouped into sets of cognitive synonyms or synsets (G.Miller, 1990). WordNet was conceived and is in constant development at Princeton University, New Jersey. The WordNet database as of now consists of over one hundred and fifty thousand words, with about two hundred and seventy thousand word-sense pairs. The word sense hierarchies required for the construction of sense trees of the verb-noun1-preposition-noun2 quadruple of ambiguous sentences for this experiment are extracted from the extensive WordNet (WN) sense database.

A word sense hierarchy is a lexical tree formed by a sequence of hypernyms in different levels, with each level being trailed by the synset of a superordinate term (hypernym) to a limited depth. That is, the last level being the most generic superordinate term of the term in the first level. This is illustrated in Figure 2.2. Each level might consist of a single hypernym or a synset of hypernyms. A synset of a term is the group of words that are semantically analogous to it. Smeaton and Quigley (1996) defined synsets in WordNet as logical groupings which consist of a list of synonymous word forms and semantic pointers that describe relationships between various semantically related synsets. Thus, multiple words or terms occurring in the same level are synonyms of each other and hypernyms of the preceding level.

**Word:** *anticipate* (verb)

*expect, anticipate*

=> *evaluate, pass judgment, judge*

=> *think, cogitate, cerebrare*

**Word:** *improvement* (noun)

*improvement*

=> *change of state*

=> *change*

=> *action*

=> *act, human action, human activity*

=> *event*

=> *psychological feature*

=> *abstraction*

=> *abstract entity*

=> *entity*

**Figure 2.2 : Sample word sense hierarchies**

In the above example for the verb *anticipate*, ‘*expect, anticipate*’ that occur in the first level are synonyms of each other. ‘*evaluate, pass judgment, judge*’ in the second level are hypernyms of words above them, i.e. ‘*expect, anticipate*’.

The WordNet database has six different senses for *anticipate* and three for *improvement*. The first occurring sense of a word is used for constructing hierarchies in this experiment.

## 2.3 PP-attachment ambiguity resolution systems

In the recent years, many pieces of research have attempted to tackle the PP-attachment ambiguity problem from different angles. Some constructed mathematical models, for instance the Maximum Entropy model

(Ratnaparkhi, Reynar and Roukos, 1994) and some like Nakov and Hearst (2005) tried solutions entirely based on semantic contextual learning.

### **2.3.1 Ratnaparkhi's Maximum Entropy Model**

Ratnaparkhi, Reynar and Roukos (1994) proposed a Maximum Entropy (ME) model which attempted to predict the probability of attachment decisions by constructing statistical models. Their model made use of only the lexical information within verb phrases, and did not depend on any external semantic knowledge base. They extracted verb-phrases with PP-attachment ambiguities from the Penn Treebank WSJ corpus and the IBM-Lancaster Treebank including the attachment information, and constructed the test and training datasets. The dataset which consisted of 27,937 quadruples has since established itself as a benchmark dataset. Due to the unavailability of the Ratnaparkhi dataset, I constructed a custom dataset from WSJ corpus for the experiment. The model which was based on models of exponential family constructed using the Maximum Entropy Principle, then assigned a probability to either of the possible attachments. The ME model produces a probability distribution for the PP-attachment decision using only the information from the ambiguous verb phrases in question. The experiment produced satisfactory results with the ME model predicting PP-attachments with 78.0% accuracy, compared to an average lexicographer performing at 88.2% accuracy. For comparison, they obtained PP-attachment resolution performances of three treebanking experts on a set of three hundred randomly selected test events from the WSJ corpus.

### **2.3.2 Zhao and Lin's Nearest-Neighbor Method**

More recently, Zhao and Lin (2004) proposed a nearest-neighbour method that did not rely on any manually constructed knowledge bases, but instead worked by computing distributional word similarities. This was based on the 'Distributional Hypothesis' in linguistics which states that words appearing in the same context tend to have similar meanings (Harris, 1968). Most of the techniques that compute distributional similarity between words represent them by feature vectors where features corresponded to types of context in which the words appeared. Zhao and Lin chose syntax based feature representation for their experiment, which constructed feature vectors of words by extracting all the features of all the words in the corpus. Their training set comprised of 4-tuples of ambiguous sentences (verb-noun1-preposition-noun2) with attachment information, which were extracted from the Ratnaparkhi (1994) dataset, a subset of the Penn Treebank WSJ corpus. Given a

4-tuple with no attachment information, the training set was searched for its top priority nearest neighbors and the PP-attachment determined based on the known classifications of the nearest neighbors. They used the Cosine Similarity Measure that measured similarity between two words based on the feature vectors of words. The experiment yielded a high resolution accuracy of 86.5% and they concluded that cosine of point wise mutual information was better than most commonly used word similarity measures.

### **2.3.3 Nakov and Hearst's method using the Web as a training set**

Nakov and Hearst (2005) proposed a method to resolve PP-attachment ambiguity using unsupervised algorithms which cleverly exploited the WWW as a very large corpus, making use of its surface features and paraphrases. This was based on the assumption that phrases found on the WWW are sometimes disambiguated and annotated by content creators. They used the Ratnaparkhi (1994) dataset as the test data. Their experiment used n-gram models, where statistics were obtained by querying exact phrases including inflections and all possible variations derived from WordNet, against WWW search engines. Nakov's experiment made use of WordNet in a similar way to this experiment, where synonyms from word sense hierarchies were used to construct all possible variations of a given phrase. The evaluation of these statistics concluded the experiment which produced an average accuracy of 83.82%.

## **2.4 Syntactic ambiguity resolution by humans**

Language processing in humans is a complex, multi-leveled process. When parsing a sentence, first comes the translation of the physical form, i.e., written text, speech etc., into a form the brain can interpret, followed by the lexical and semantic level of processing, where the information is parsed and its meaning extracted. There is another level called the structural or syntactic level of processing which determines the relation of different words in the sentence, followed by the discourse level which analyses the larger context of which the sentence is a part of (Haupt, 2006).

Altmann (1985) studied a number of syntactic resolution experiments so as to determine whether ambiguity resolution in humans is based on syntactic information alone or whether they are resolved on some other basis, for example the presence of contextual information, which is still an ongoing controversy. He found that syntactic ambiguity resolution in humans is largely based on existing knowledge. Even an isolated sentence with no context, that has for example a PP-attachment ambiguity, could be resolved based on prior knowledge.

Altmann stated that the notion of syntax being confined to the relations in single isolated sentences without discourse, are short sighted. Altman concluded that computational models of syntactic ambiguity resolution which ignore contextual considerations are not true models of natural language processing, hence stressing the importance of semantic models in NLP and not solely statistical.

## **2.5 Machine learning (ML)**

Machine learning, a broad subfield of Artificial Intelligence (AI), refers to autonomous changes in systems that perform tasks associated with AI. Alpaydin (2004) gives a better explanation, defining machine learning as programming computers to optimize a performance criterion using example data or past experience. In ML capable systems, a model is defined up to some parameters and machine learning is the execution of a computer program to optimize the parameters of the model appropriately, based on training data or past experience. ML techniques, which automatically learn linguistic information from text corpora, have been extensively applied to various problems in natural language processing (M.Banko and E.Brill, 2001), not to mention, the PP-attachment problem.

Unsupervised learning, as the name suggests, is a type of autonomous machine learning that only requires partial human input in the learning process, unlike supervised learning. UL techniques are invaluable in language engineering applications where manually tagged corpora are not available (Basili, Marziali, Pazienza and Velardi, 1996). Many of the proposed PP-attachment resolution methods have been based on unsupervised learning techniques, a notable one being Ratnaparkhi's Maximum Entropy Model (Ratnaparkhi, Reynar and Roukos , 1994).

# Implementation and Findings

## 3.1 Selection of sentences with PP ambiguity from the corpus

The WSJ corpus is comprised of a multitude of structurally, semantically and grammatically annotated material. To collect sentences with PP ambiguities, the Penn Treebank II corpus can be traversed, searching annotation tags looking for tags that define the prepositional phrases. The Penn Treebank tag that represents PP-attachments is *PP*. It might occur as is, or might be suffixed by a tag describing the semantic nature of the word tagged.

### Examples:

(*PP-CLR from*) - The prefix *CLR* stands for ‘closely related’

(*PP with*) - No prefix

I devised an algorithm for the automated traversal of the corpus searching for presence of sentences containing the PP tag based on a rule which leaves provision for suffixes, i.e., ‘PP-\*’ where ‘\*’ is the place holder of the suffix.

The resulting sentences in the Penn Treebank annotated format are then piped through another program I wrote, that implements a recursive decent parser that parses the parenthesised tree of Penn Treebank tags and words. The result is a stripped version of the annotation, eliminating the parenthesised nesting, retaining only the tags, corresponding words, position and attachment information.

(	3	<i>PRP</i>	<i>I</i>
( <i>S</i>	3	<i>VBD</i>	<i>saw</i>
( <i>NP (PRP I)</i> )	4	<i>DT</i>	<i>the</i>
( <i>VP (VBD saw)</i> )	4	<i>NN</i>	<i>girl</i>
( <i>NP (DT the) (NN girl)</i> )	4	<i>IN</i>	<i>with</i>
( <i>PP (IN with)</i>	5	<i>DT</i>	<i>the</i>
( <i>NP (DT the) (NN telescope)</i> ))			

(. .)

5 NN telescope

)

)

**Annotated form**

**Parsed form**

The parsed data is then further stripped of all elements except for the verb-noun1-preposition-noun2 tuple, as seen in examples provided on page 11, based on the following criteria.

Noun = (NN | NP | NX | NNS | NNP | NNPS | CD)

Verb = (VB | VBD | VBG | VBN | VBP | VBZ | VP | ADVP)

Preposition = (PP | IN | TO)

The following figure shows the results of the execution of the programs on the WSJ corpus.

Sentences analysed	:	49,208
Sentences containing PP* tag	:	42,186
Sentences with PP-attachment ambiguity	:	7810

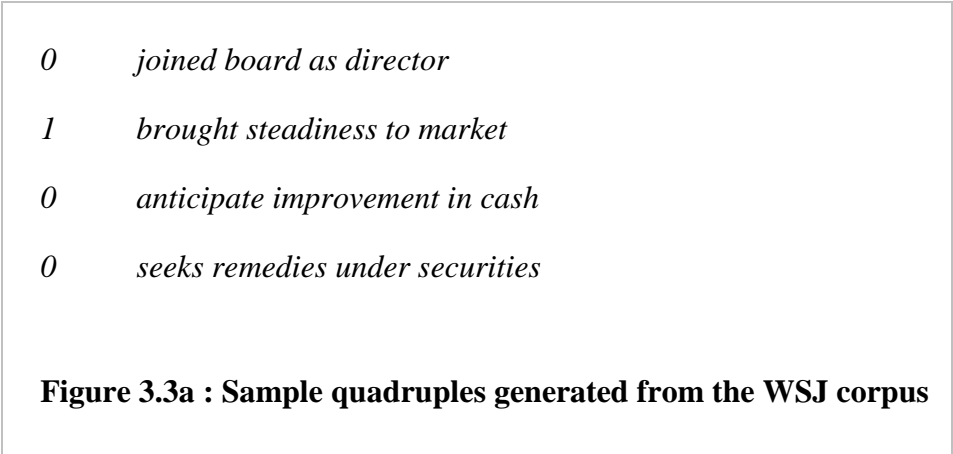
**Figure 3.1 : Results of the selection process run on the WSJ corpus**

## 3.2 WordNet parser

The synset hierarchy of words (Section 2.2) produced by the WordNet binary is captured and parsed into tree structures by a program I scripted. The program captures the output and uses textual pattern matching to convert the flat representation into high level trees in the program, preserving the structure and depth information of nodes.

## 3.3 Quadruples, Training and Test data

The multitude of parsed sentences produced in the selection process were converted into (verb-noun1-preposition-noun2) quadruples, similar to the Ratnaparkhi (1994) dataset with each quadruple being accompanied by a numerical flag (0, 1) specifying the sentence's attachment information. A total of 7810 quadruples were obtained (Figure 3.1). It is important to note that phrases with the preposition 'of' which accounted to 2868 quadruples were excluded in the process. Ratnaparkhi (1998) and Nakov (2005) predicted all quadruples with 'of' to have noun attachments without actually testing, considering it a reliable heuristic. Thus the exclusion was made so as to test only the phrases whose attachment decisions could not be assumed.



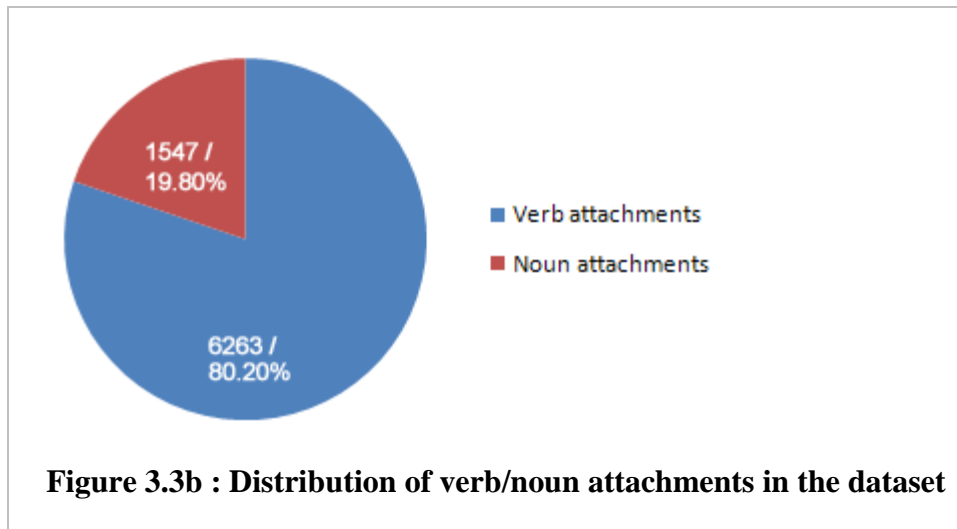
0     *joined board as director*  
1     *brought steadiness to market*  
0     *anticipate improvement in cash*  
0     *seeks remedies under securities*

**Figure 3.3a : Sample quadruples generated from the WSJ corpus**

The above figure shows a sample set of quadruples, preceded by the numerical flag where:

- 0     -     Attaches to Verb (verb1)
- 1     -     Attaches to Noun (noun1)

The 7810 quadruples consisted of 6263 verb attachments (0) and 1547 noun attachments (1). Interestingly, the number of verb attachments were very high (80.20%) compared to noun attachments that constitute only 19.80% as shown in the following figure.

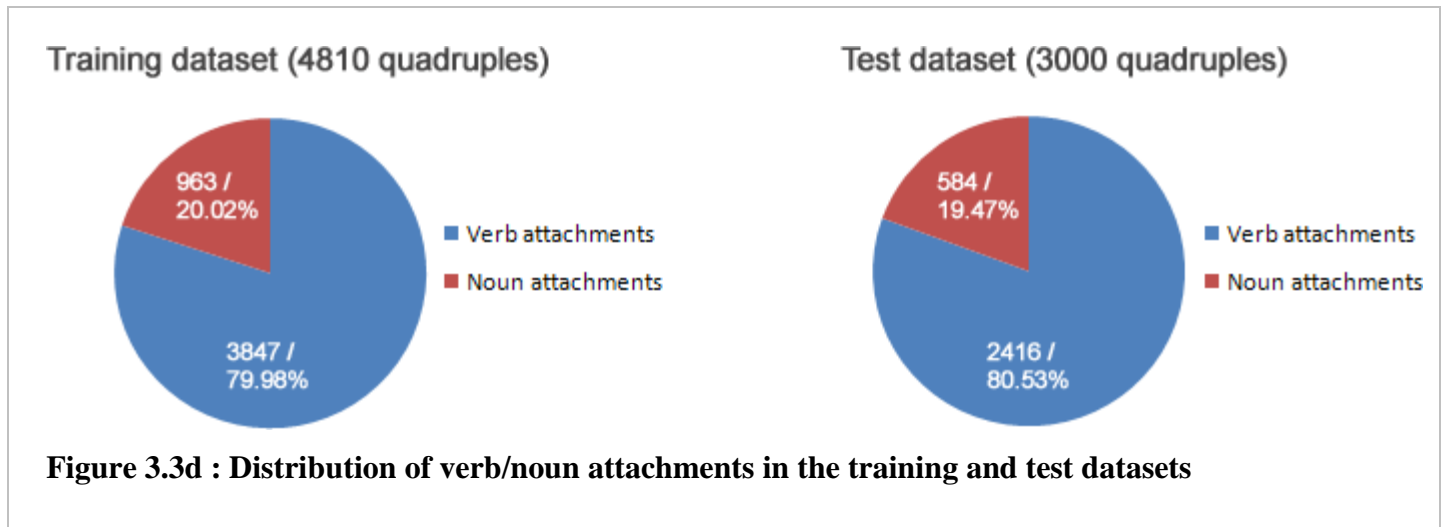


These quadruples were divided into two sets, the training set and the test set in no particular order, with the first 4810 quadruples forming the training set and the rest, test, as shown in the following figure.

Training dataset	:	4810 quadruples
Test dataset	:	3000 quadruples

**Figure 3.3c : Training dataset and Test dataset**

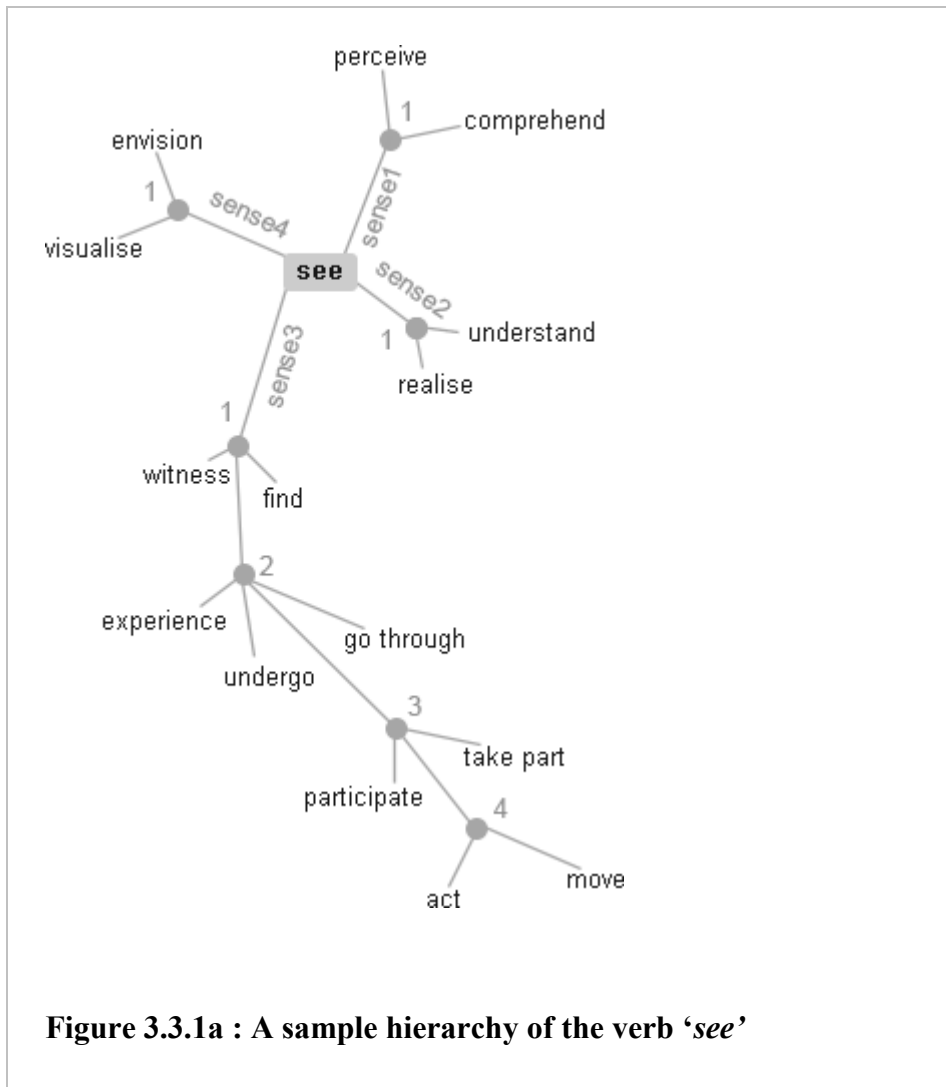
The training dataset consisted of 3847 (79.98%) verb attachments and 963 (20.02%) noun attachments and the test dataset consisted of 2416 (80.53%) verb attachments and 584 (19.47%) noun attachments as shown in the figure below.



### 3.3.1 Graphs of quadruple word sense hierarchies

The first and the most important part of the experiment is the construction of graphs of quadruples comprising of word sense hierarchies of each of the terms in the quadruples. I devised an interpreter that takes quadruples and constructs graphs using word sense hierarchies obtained from the WordNet parser. The quadruples generated (Section 3.3) are separated into verb, noun1, preposition and noun2 respectively by the interpreter. The interpreter makes use of the WordNet parser and obtains the word sense hierarchies of each of these terms. These three word sense hierarchies are then joined together to form the final graph of the quadruple, which is retained in memory by the interpreter. Each node of the graph is a triplet comprising of one word from the sense hierarchies of each of the terms in the quadruple and the graph would contain every possible triplet combination of the three hierarchies. This is illustrated and explained in detail in the following sections.

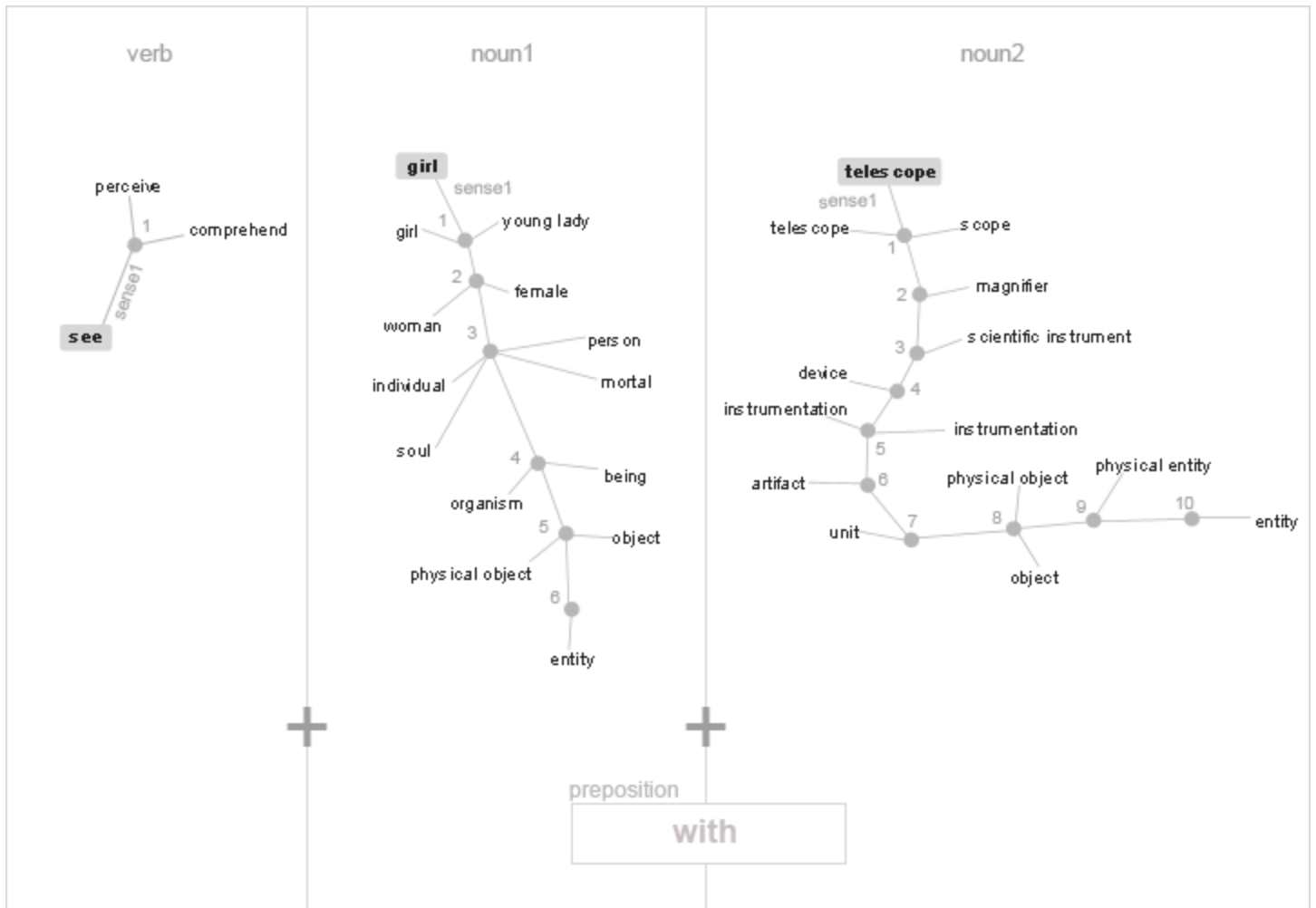
Shown below is the graphical representation of a sample hierarchy of the verb 'see', truncated at the four senses (WordNet database has twenty four senses for the verb 'see').



**Figure 3.3.1a : A sample hierarchy of the verb 'see'**

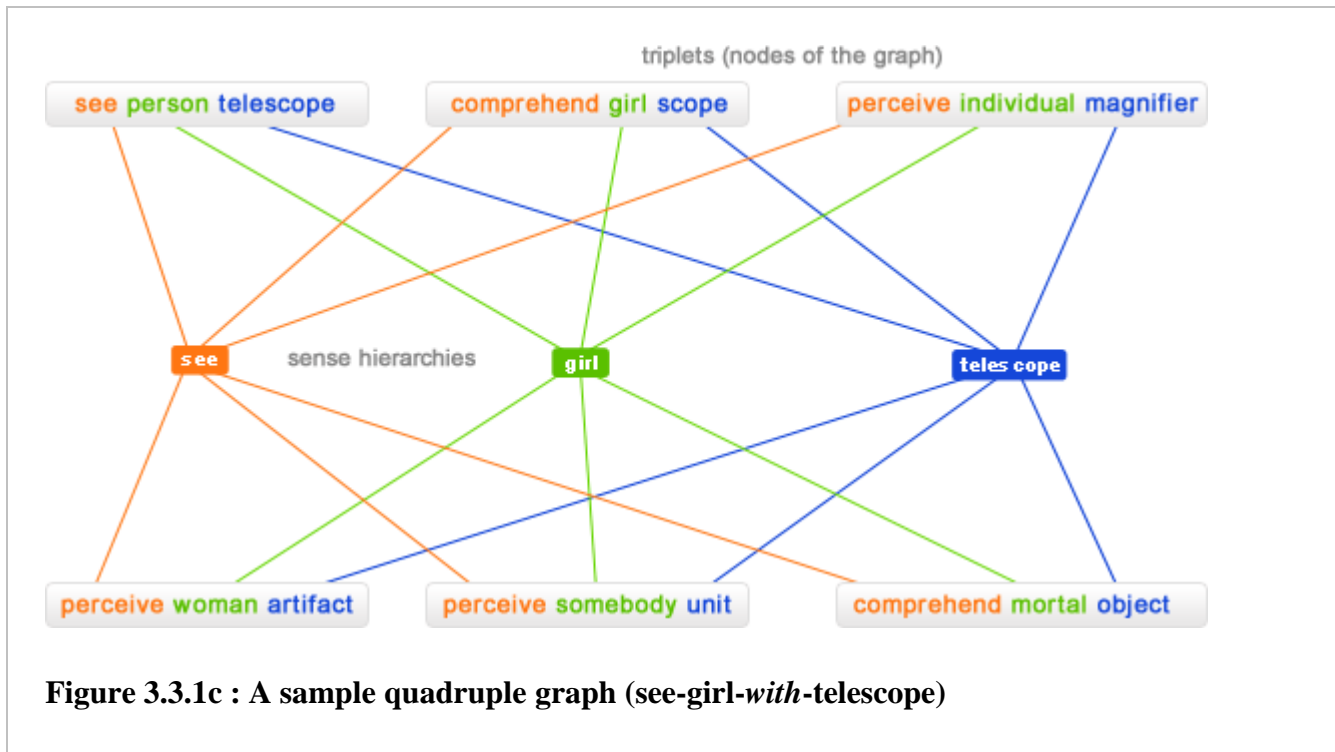
The experiment only uses the first of all the sense hierarchies of a term and ignores the rest. In effect, every quadruple graph is composed of three hierarchies (one sense each), one of each of the terms in the quadruple (verb, noun1, noun2). Terms in the quadruples with no word sense hierarchies in the WordNet database were assumed to be subordinates of the abstract class 'object' and numbers of the class 'number'.

A graphical representation of these three hierarchies which constitute a graph is presented in the following figure.



**Figure 3.3.1b : Constituents of a sample quadruple graph (see-girl-with-telescope)**

The above figure shows the constituents of a typical quadruple graph, i.e., three hierarchies with one sense each. Similarly, graphs are generated for every quadruple in the training set and the test set. The graphical representation of the triplets that form the nodes of the graph, formed by the combination of nodes of all the three hierarchies is presented in the following figure.



The above figure shows six randomly selected nodes of the quadruple graph. Each node is a unique triplet formed by the combination of words (nodes) of each of the three sense hierarchies (Figure 3.3.1b) which constitute the graph. A graph would have all possible combination of terms in the three sense hierarchies as its nodes, thus forming a complex mesh.

For instance, consider the first sense hierarchy of each of the terms *see-girl-telescope*.

Term	Number of words in the first sense tree of the hierarchy (from WordNet)
<i>v. see</i>	2
<i>n. girl</i>	31
<i>n. telescope</i>	16
<b>2 x 31 x 16 = 992 nodes in the graph</b>	

**Figure 3.3.1d : Nodes of a quadruple graph (see-girl-with-telescope)**

Based on the figures in the above table, the graph of the quadruple *see-girl-telescope* has a total of 992 triplets, i.e., nodes in total. The numbers of nodes in graphs depend on the number of terms in the word sense hierarchies with some spanning to several thousands.

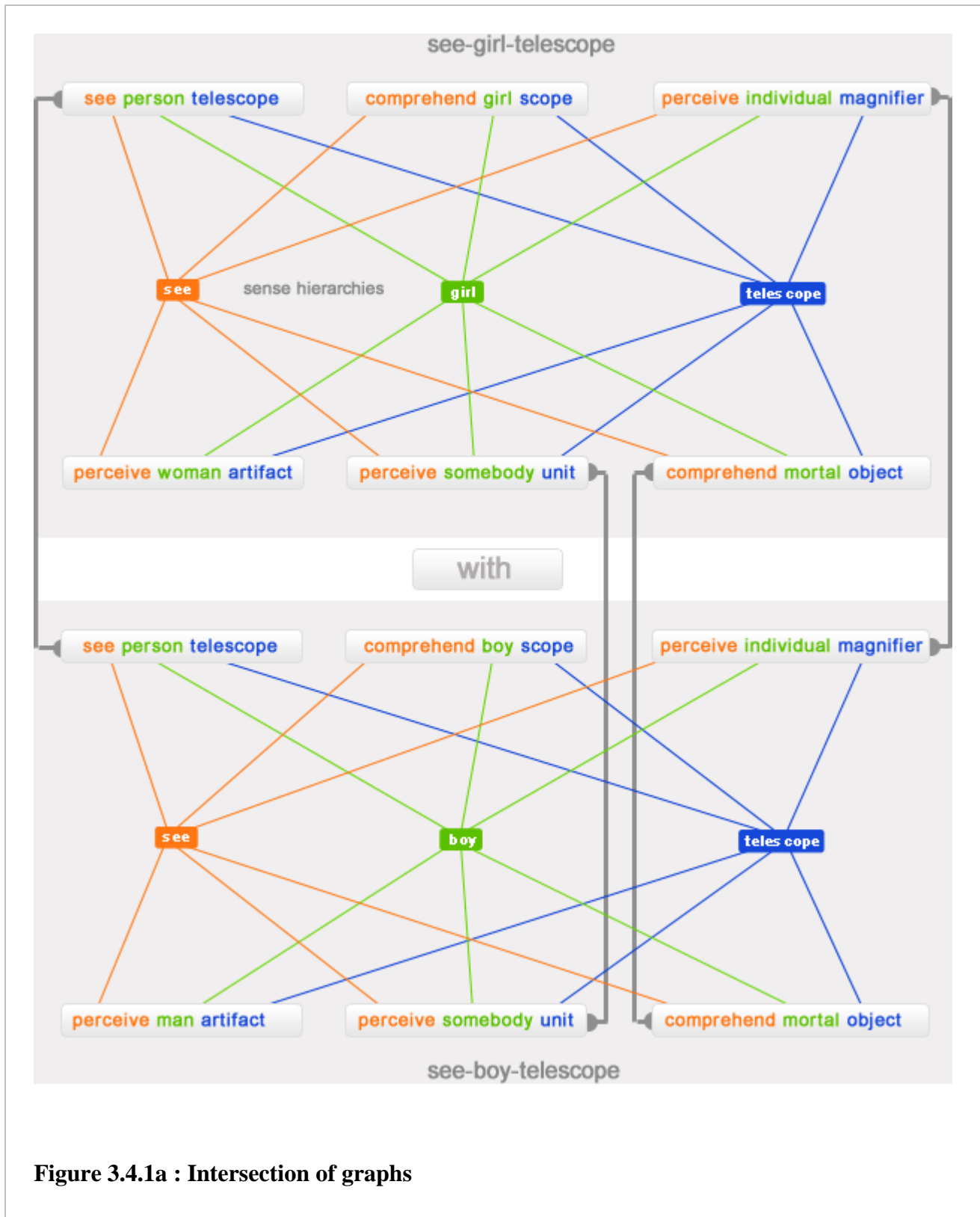
## **3.4 The Test**

This section details the process of the actual test, i.e., training the PP-attachment ambiguity resolution program I wrote with the training dataset and testing the test dataset against it. As detailed in Section 3.3.1, at the beginning of the execution of the test program, graphs of all quadruples in the training dataset are created and retained in memory. Following that, quadruples in the test dataset are read and their graphs created and compared against all the training dataset graphs in memory one after the other and evaluated in the process.

### **3.4.1 Intersection of graphs**

The program generated 4810 graphs in total (Figure 3.3c) for all the quadruples in the training dataset. The first step in predicting the PP-attachments of quadruples in the test dataset was to match each of their graphs to all the graphs of the training set one by one, and measures their similarity. The similarity measure was obtained by a simple algorithm which counted the number of ‘intersections’ between the nodes of graphs of all the test and training quadruples.

A visualisation of intersection of portions of two sample quadruple graphs is shown in the following figure.



**Figure 3.4.1a : Intersection of graphs**

The above figure shows the intersection of two sample graphs of the quadruples:

1. *see girl with telescope*
2. *see boy with telescope*

The sample quadruples are of high semantic similarity and large number of intersecting nodes has been chosen deliberately for illustrative purposes. The graphs shown above have been pruned to a few select nodes to show the intersections. Since the nouns *boy* and *girl* are nodes of the same superordinate classes *individual*, *individual*, *somebody*, *mortal*, they intersect (as marked in the figure). Actual graphs of two random quadruples chosen from the datasets may only have a few intersections or none at all, depending on their semantic similarity.

### 3.4.2 Attachment decision

The attachment decision for a test quadruple was made by determining what the majority of intersections inclined to, quadruples with verb attachments or quadruples with noun attachments in the training dataset. Although not optimal, this method helped predict roughly the probability of the test quadruple's PP-attachment. If a particular test quadruple had the most number of intersections with quadruples with verb attachments in the training set, its preposition was assumed to attach to the verb and otherwise, the noun. Quadruples with zero intersections were assumed to have verb-attachments as that is the observed case most of the time, although better algorithms could have been applied to predict the probability of attachment based on the nature of the preposition.

An example result from the actual test is given below. In this particular case, since the number of attachments to verb phrases was greater than the number of noun attachments (zero in this case), the quadruple was predicted to attach to the verb which in fact is the right attachment decision.

**Quadruple:** *join board as director*

**Number of intersections** (with 4810 training quadruples)

**Noun = 0**

**Verb = 60**

**Figure 3.3.1b : Sample attachment decision**

### 3.5 Results

The test took over twelve hours to execute on a high performance computer. The test concluded with 2617 correct predictions out of 3000, resulting in an accuracy of 87.23%. The following table summarizes the results of the test.

Number of training quadruple graphs produced	<b>4810</b>
Total number of nodes in all the training graphs	<b>5,106,156</b>
Number of test quadruple graphs produced	<b>3810</b>
Total number of nodes in all the test graphs (approx.)	<b>3,150,000</b>
<b>Number of correct attachment predictions</b>	<b>2617</b> ----- $2617 / 3000 = \underline{\underline{87.23\%}}$
Number of correct verb-attachment predictions	<b>2224 / 2415 = 92.09%</b>
Number of correct noun-attachment predictions	<b>393 / 585 = 67.17%</b>

**Figure 3.5 : Final test results**

The above table shows that verb attachments were predicted with an accuracy of 92.09%. The fact that test quadruples with no similarity with any of the training quadruples were assumed to have word attachments, could have been a constituent in this and has to be subjected to extended analysis for accurate information. Another factor is that majority of the phrases in the WSJ corpus with PP-attachment ambiguities have verb attachments, as shown previously in Figure 3.3d. Same would be the reason for the lower prediction accuracy of noun attachments.

### 3.6 Comparison with previous work

Of all the reviewed work, this experiment attained comparatively high prediction accuracy. Due the unavailability of the Ratnaparkhi dataset which the reviewed experiments (Section 2.3) used, it is difficult to conduct a fair and accurate comparison. The following table presents the test results from the work discussed in the literature review (Section 2.3).

<b>Experiment</b>	<b>Result</b>
Maximum Entropy Model, WSJ Corpus (Ratnaparkhi, 1994)	78.0%
Nearest neighbour method, Ratnaparkhi dataset (Zhao & Lin, 2004)	86.5%
Method using web as a training set, Ratnaparkhi dataset (Nakov & Hearst, 2005)	83.82%
<b>Method using word sense hierarchies (WSJ Corpus)</b>	<b>87.23%</b>

**Figure 3.6 : Comparison with previous work**

## Conclusion and discussion

The PP-attachment disambiguation method using word sense hierarchies from WordNet I proposed in this thesis achieved comparatively higher accuracy than some of the most recent methods proposed. The experiment demonstrated that using simple, partially supervised algorithms to introduce semantic information (supervised information obtained from WordNet in this case) in PP-attachment disambiguation techniques produce results significantly better than models that are wholly statistical. It is to be noted that unlike other experiments that included and assumed phrases with the preposition 'of' to have noun attachments without testing, this experiment excluded them, which if tested would significantly improve the final result. In addition, this experiment has left room for the proposed model to be bettered in many ways, for instance by mere introduction of better algorithms that compute semantic relationships more efficiently as stated in Section 3.4.2, which could have been implemented if it weren't for the time constraint. An important factor that affected the experiment as any other, was the sparse nature of the corpora which often leads to false attachment decisions. Unavailability of computing power required to execute the experiment in a timely manner was a factor that down paced the experiment.

Suggesting improvements, the algorithm which calculated the intersection of graphs could be modified to recognize various data types that occur in the quadruples such as numbers, dates, percentages, place names etc. and act accordingly and not merely treat all unrecognized terms as the abstract entity *object* as mentioned in Section 3.3.1. A scoring algorithm that measures the semantic distance between the intersections based on the depth of the intersection of nodes in a graph could be introduced to make the process much more accurate and produce better results.

One possible disadvantage of this method is that it relied solely on supervised training data (quadruples with attachment decisions from the WSJ corpus and WordNet word sense hierarchies), which would prove difficult in real world scenarios due to the limited availability of large, quality, annotated corpora. As mentioned in the beginning, the presence of data sparseness due to the diversity and non-generic nature of most available corpora affect the quality of experiments. As research in the field of natural language processing progresses and better annotated corpora become widely available, better solutions to challenges like the PP-attachment ambiguity come to light; but it is up to one to wonder whether systems capable of perfectly solving issues inherent in AI and NLP would come into existence any time soon.

# Appendix 1: Acronyms

AI	-	Artificial Intelligence
ME	-	Maximum Entropy
ML	-	Machine language
NLP	-	Natural language parsing
NP	-	Noun Phrase
PP	-	Prepositional phrase
PTB	-	Penn Treebank
PTB II	-	Penn Treebank II
WN	-	WordNet
WSJ	-	Wall Street Journal
WWW	-	World Wide Web
VP	-	Verb Phrase

## Appendix 2: Sample quadruples

A few sample quadruples generated from the WSJ corpus used for the training and test datasets.

join board as director

have information on users

has bearing on work

approved acquisition by royal

holding company with interests

been executive with chrysler

had space on machines

squeezed meetings at hotel

said exports at end

give discounts for ad

awards credits to advertisers

gaining circulation in years

leaves utilities as bidders

retired undersecretary on board

ordered edison to refunds

held hostage through round

upheld challenge by consumer

set rate on refund

faces refund on rate

setting month since march

been orders for cray

face competition from cray

calculate value at 475

claiming success in trade

growing realization around world

## Appendix 3: Programs and scripts

The programs and scripts used in this experiment to parse the corpus, produce quadruples, process data, run tests etc. were written in the popular programming languages Perl and Ruby.

Perl - <http://www.perl.org>

Ruby - <http://www.ruby-lang.org>

# Appendix 3.1: PP-attachment parser – sample Perl script

## Referring to Section 3.1

```
#!/usr/bin/perl

# results directory
$_OUT_FILE = "./data/pps.dat";

# opening of a sentence
$_STR_OPEN = "( ";

# Check CL arguments
if($#ARGV < 0) {
    print "\n\tExtracts sentences with PP attachments\n\tfrom Penn Treebank
tagged corpora.\n\t\t - Kailash Nadh (kailashnadh.name)\n";
    print "\n\tUsage: $0 [directory to scan]\n\tOutputs to $_OUT_FILE\n";
    exit;
}

# source path from the argument
my $_PATH = @ARGV[0];

# length of the opening string
$_STR_OPEN_LEN = length($_STR_OPEN);

# read the directory
my @files = <$_PATH/*>;

# oops
if(!@files) {
    print "> Invalid or empty directory\n";
    exit;
}

# counters
$count_sentence = $count_pp = 0;

printf "> Parsing..\n";

open(PP_FILE, ">".$_OUT_FILE);
# scrounge the files
foreach my $file (@files) {

    my @fn = split(/\\/, $file);
    $filename = @fn[$#fn];

    # read the file
    open(FILE, $file);
    my @data = <FILE>;
    close(FILE);

    # pp-extractor
    parsePP(@data);
}
close(PP_FILE);
```

```

# screen output
print "> Parsed $count_sentence sentences from ".$#files + 1). " files\n>
Found $count_pp sentences with PP attachments\n> Results saved to $_OUT_FILE\n";
# =====

# extract PP-attachments
sub parsePP {
    # full text
    my @data = toSentences(@_);

    foreach my $sentence (@data) {
        $count_sentence++;
        if($sentence =~ m/\(PP( |-(.*) )/ig) {
            $count_pp++;
            print PP_FILE $filename."\t\t".$sentence."\n";
        }
    }
}

##### String functions

# Perl trim function to remove whitespace from the start and end of the string
sub trim($) {
    my $string = shift;
    $string =~ s/^\s+//;
    $string =~ s/\s+$//;
    return $string;
}

# adopted from Anoop Sarkar
sub toSentences {

    my @data = @_;

    my ($i, $lcount, $rcount);
    my @sexpr = ();
    my $partialsexpr = "";
    my @sentences = ();

    $i = $lcount = $rcount = 0;
    foreach $line (@data) {
        $lcount += ( $line =~ tr/(// ) ;
        $rcount += ( $line =~ tr/)// ) ;
        $partialsexpr .= $line ;

        if (($lcount - $rcount) == 0) {
            $lcount = 0 ; $rcount = 0 ;
            $sexpr[$i] = $partialsexpr; $i++ ;
            $partialsexpr = "" ;
        }
    }
    foreach (@sexpr) {
        s/ \n/ /go;
        s/\n/ /go;
        s/\s+/ /go;
        if(trim($_)) { push(@sentences, $_); }
    }
    return @sentences;
}

```

# Appendix 3.2: WordNet parser – sample Ruby script

## Referring to section 3.2

```
require 'Dumper'

class Senses
  # == class variables

  # wordnet binary
  @@WORDNET_BIN = "/wordnet/bin/wn"

  # X spaces = 1 tab
  @@WORDNET_TAB_LEN = 4;

  # available functions (noun, verb)
  @@WORDNET_POS = {"v" => 1, "n" => 1}

  # cache
  @@CACHE = {}

  # ===== get senses of a word from wordnet
  def get(word, pos)
    if not @@WORDNET_POS.has_key?(pos)
      return false
    end

    # number ?
    if word.match(/[-+]?[0-9]*\.[0-9]+/) != nil
      return [['number']]
    end

    if not @@CACHE.include?(word)
      senses = `#{@WORDNET_BIN} " " + word + " -hype" + pos`
      @@CACHE[word] = parse(ready(senses))
    end

    #print Dumper.dump(@@CACHE[word])
    #exit

    return @@CACHE[word]
  end

  # ===== parse the textual senses into array
  def ready(lines)

    start = false
    lines = lines.split("\n") # split into lines
    count = 0
    senses = []

    lines.each { |line|
      # get the sense into an array
      if start == false && /Sense [0-9]{1,3}/ =~ line
        start = true

        elsif start == true && /Sense [0-9]{1,3}/ =~ line
          count=count+1

        elsif start == true && (/ [0-9]{1,3}/ =~ line || /\// =~ line)
          # unwanted

        elsif start == true && line.strip != ""

```

```

senses[count] = senses[count].to_s + ( line.index("=>") == nil ? "=>
"+line : line ) + "\n"
    end
  }
  return senses
end

# ===== parse the textual senses into array
def parse(senses)

  if senses == nil || senses.length == 0
    return [['object']]
  end

  parsed = [""]
  level = sub = prev = -1

  # parse each block
  senses.each { |set|

    depth = sub = 0
    prev = -1
    set = set.split("\n")
    level += 1
    parsed[level] = []

    set.each { |line|

      /(.*)=>/.match(line)

      depth = $1.scan(/ /).length
      depth = (depth > 0 ? depth-3 : 0) / @@WORDNET_TAB_LEN;

      if prev >= depth
        level+=1
        parsed[level] = []
        sub = 0
      end
      prev = depth

      if not parsed[level][sub].instance_of?(Array)
        parsed[level][sub] = []
      end

      parsed[level][sub].concat(sanitize(line))
      sub+=1
    }
  }

  # cleanup
  if parsed != nil
    parsed.map! { |block| block.uniq}
  end

  return parsed
end

def sanitize(line)
  line = line.sub(/=>/, '').strip
  line = line.sub(/, /, ',')
  words = line.split(',')

```

```
        words.map! { |w| w.strip.downcase.gsub(/\s/, '_') }
        return words
    end

    private :sanitize, :parse, :ready
end
```

# References

- Basili, A., Marziali, A., Pazienza, M.T. and Velardi, P. 1996. Unsupervised Learning of Syntactic Knowledge: methods and measures. In *Conference on Empirical Methods in Natural Language Processing*, Philadelphia, Pennsylvania.
- Alpaydin, E., (2004). Introduction to Machine Learning. Cambridge: The MIT Press.
- Altmann, G. 1985. The resolution of local syntactic ambiguity by the human sentence processing mechanism. In *Proceedings of the second conference on European chapter of the Association for Computational Linguistics*.
- Banko, M and Brill, E. 2001. Scaling to very very large corpora for Natural Language Disambiguation. In *Proceedings of ACL*.
- Brill, E. and Resnik, P. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In COLING.
- Harris, Z.S. 1968. Mathematical Structures of Language. New York: Wiley.
- Houpt, J. 2006. Ambiguous Prepositional Phrase Resolution by Humans. Masters thesis. University of Edinburgh, pages 1-5.
- Lin, D. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of COLING-ACL98. Montreal, Canada*.
- Marcus, M.P., Santorini, B, and Marcikiewicz, M.A. 1995. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.
- Miller, G. 1990. WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4).
- Nakov, P. and Hearst, M. 2005. Using the web as an implicit training set: Application to structural ambiguity resolution. In *Proceedings of HLT-NAACL*.
- Ratnaparkhi, A., Reynar, J., and Roukos, S. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 250 – 252.
- Adwait Ratnaparkhi. 1998. Statistical models for unsupervised prepositional phrase attachment. In *proceedings of COLING-ACL*, volume 2, pages 1079–1085.
- Smeaton, A and Qigley, I. 1996. Experiments on using semantic distances between words in image caption retrieval. In *Proceedings of the ACM SIGIR '96 Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, August.

- Thompson, C.A., Califf, M.E. and Mooney, R.J. 1999. Active Learning for Natural Language Parsing. In *Proceedings of the Sixteenth International Machine Learning Conference*, pp.406-414, Bled, Slovenia, June 1999.
- Zhao, S and Lin, D. 2004. A Nearest-Neighbor Method for Resolving PP-Attachment Ambiguity. In *Proceedings of IJCNLP-04*.