University of Sussex

Graph-Based Approaches to Word Sense Induction

David Richard Hope

Submitted for the degree of Doctor of Philosophy
University of Sussex
December 2014

Declaration

I hereby declare that this thesis has not been and will not be submitted in whole or in part to another University for the award of any other degree.

Signature:

David Richard Hope

UNIVERSITY OF SUSSEX

David Richard Hope, Doctor of Philosophy

Graph-Based Approaches to Word Sense Induction

Summary

This thesis is a study of Word Sense Induction (WSI), the Natural Language Processing (NLP) task of automatically discovering word meanings from text. WSI is an open problem in NLP whose solution would be of considerable benefit to many other NLP tasks. It has, however, has been studied by relatively few NLP researchers and often in set ways. Scope therefore exists to apply novel methods to the problem, methods that may improve upon those previously applied. This thesis applies a graph-theoretic approach to WSI. In this approach, word senses are identified by finding particular types of subgraphs in word co-occurrence graphs. A number of original methods for constructing, analysing, and partitioning graphs are introduced, with these methods then incorporated into graph-based WSI systems. These systems are then shown, in a variety of evaluation scenarios, to return results that are comparable to those of the current best performing WSI systems. The main contributions of the thesis are a novel parameter-free soft clustering algorithm that runs in time linear in the number of edges in the input graph, and novel generalisations of the clustering coefficient (a measure of vertex cohesion in graphs) to the weighted case. Further contributions of the thesis include: a review of graph-based WSI systems that have been proposed in the literature; analysis of the methodologies applied in these systems; analysis of the metrics used to evaluate WSI systems, and empirical evidence to verify the usefulness of each novel method introduced in the thesis for inducing word senses.

Acknowledgements

Firstly, many thanks to my thesis supervisor Bill Keller, notably for his patience and ability to phrase complex problems in simple terms. Thanks also to my parents, and to those who helped in different ways along the way: Jonathon, Dan, Catherine, Isabelle, Diva, Steven, Diana, David, Simon, John, Rob, Patrick, Brian, and Adam - in no particular order.

Contents

List of Tables **xiv**

List of Figures **xviii**

1 Introduction **1**

1.1 Word Sense Induction . . . 3

1.1.1 Graph-Based Word Sense Induction . . . 4

1.1.1.1 A Graph Model of Word Context . . . 4

1.1.1.2 Identifying Word Senses . . . 5

1.1.1.3 Assigning Induced Senses to Words . . . 6

1.2 Motivation . . . 7

1.2.1 Lexicography . . . 7

1.2.2 Information Retrieval . . . 7

1.2.3 Machine Translation . . . 8

1.2.4 Information Extraction . . . 8

1.2.5 Social and Information Network Analysis . . . 10

1.2.6 Motivation: Summary . . . 11

1.3 Research Objectives . . . 11

1.4 Contributions of the Thesis . . . 12

1.5 Outline of the Thesis . . . 12

I Graphs, Association Measures and Clustering **15**

2 Graphs **16**

2.1 Undirected and Directed Graphs . . . 16

2.1.1 Undirected Graphs . . . 16

2.1.2 Directed Graphs . . . 17

2.2 Weighted Graphs . . . 18
2.3 Connectivity in Graphs . . . 19
2.3.1 Connected Graphs and Connected Components . . . 19
2.3.2 Complete Graphs . . . 20
2.3.3 Cliques . . . 21
2.3.4 Vertex Degree . . . 22
2.4 Vertex Neighbours . . . 23
2.4.1 Vertex Cohesion . . . 24
2.4.2 $n^{\text{th.}}$ Order Neighbours . . . 26
2.5 Additional Graph Properties . . . 27
2.5.1 Graph Centrality . . . 27
2.5.2 Global Measures . . . 28
2.6 Summary . . . 30

3 Measures of Vertex Association **31**
3.1 Introduction . . . 31
3.1.1 Vertex Measures and Graph Measures . . . 31
3.2 Association Measures . . . 33
3.2.1 Frequency . . . 33
3.2.2 Conditional Probability . . . 33
3.2.3 The Log Likelihood Ratio . . . 34
3.2.3.1 Log Likelihood Ratio Definition . . . 35
3.2.4 Alternative Association Measures . . . 36
3.3 The Clustering Coefficient . . . 37
3.3.1 The Local Clustering Coefficient . . . 37
3.3.2 A Generalisation for Directed Graphs . . . 38
3.4 Summary of the Chapter . . . 39

4 Weighted Clustering Coefficients **40**
4.1 Introduction . . . 40
4.2 A Review of Existing Weighted Generalisations . . . 40
4.2.1 Barrat et al. . . . 40
4.2.2 Lopez-Fernandez et al. . . . 41
4.2.3 Onnela et al. . . . 42
4.2.4 Zhang and Horvath . . . 43

4.2.5 Opsahl and Panzarasa . . . 43
4.2.6 Summary . . . 45
4.3 Three Novel Generalisations to the Weighted Case . . . 45
4.3.1 Generalisation 1 . . . 46
4.3.2 Generalisations 2 and 3 . . . 46
4.3.2.1 A Generalisation for Graphs Representing Probability Spaces 47
4.3.2.2 A Generalisation for Integer Weighted Graphs . . . 48
4.4 Summary of the Chapter . . . 49

5 Clustering **50**
5.1 Introduction . . . 50
5.2 Clustering Solution Types . . . 51
5.3 Two Commonly Applied Clustering Methods . . . 52
5.3.1 Prototype-Based Clustering . . . 53
5.3.2 Density-Based Clustering . . . 54
5.3.2.1 DBSCAN . . . 54
5.3.2.2 Shared Nearest Neighbours (SNN) . . . 56
5.3.3 Summary . . . 57
5.4 Graph-Based Soft, Fuzzy, and Parameter-Free
Clustering Methods for WSI . . . 58
5.4.1 Graph-Based Soft Clustering for WSI . . . 59
5.4.1.1 Hypergraphs . . . 59
5.4.1.2 Graphs of Collocations . . . 60
5.4.1.3 Soft Clustering of Semantically Similar Words . . . 62
5.4.2 Graph-Based Induction of Fuzzy Word Senses . . . 62
5.4.2.1 Borin and Forsberg (2010) . . . 62
5.4.2.2 Oliveira and Gomes (2011) . . . 63
5.4.2.3 Inducing Fuzzy Semantic Relations . . . 65
5.4.3 *Chinese Whispers*: a Parameter-Free Clustering Algorithm . . . 66
5.4.3.1 *Chinese Whispers* . . . 66
5.4.3.2 *Chinese Whispers* and Non-Determinism . . . 68
5.4.3.3 Analysis of *Chinese Whispers* . . . 69
5.4.3.4 *Chinese Whispers* and an 'Ideal' Clustering Algorithm . . . 70
5.5 The *MaxMax* Clustering Algorithm . . . 71
5.5.1 *MaxMax* . . . 71

5.5.2 Correctness of *MaxMax* . . . 73
5.5.3 Time Complexity of *MaxMax* . . . 74
5.5.4 Testing *MaxMax*: Graph Partitioning . . . 75
5.5.5 Testing *MaxMax*: Graph Clustering . . . 75
5.5.5.1 Introduction . . . 75
5.5.5.2 The Test Set: Shapes . . . 76
5.5.5.3 Applying the Clustering Algorithms to the Shapes Test Set 77
5.5.5.4 The *Affinity Propagation* Clustering Algorithm . . . 78
5.5.5.5 Evaluation Measures . . . 79
5.5.5.6 Results . . . 80
5.5.5.7 Results Discussion . . . 82
5.5.5.8 Conclusions . . . 85
5.5.6 *MaxMax*: Discussion . . . 85

II Word Sense Induction 88

6 Word Sense Induction 89
6.1 Word Sense . . . 89
6.2 Word Sense Induction . . . 92
6.2.1 Word Sense Disambiguation Systems . . . 93
6.2.2 Word Sense Induction Systems . . . 93
6.2.3 Graph-Based Word Sense Induction Systems . . . 94
6.2.3.1 Discussion . . . 96

7 A Review of Existing Approaches to Word Sense Induction 98
7.1 Introduction . . . 98
7.1.1 Lexical Co-occurrence Graphs . . . 98
7.2 Dorow and Widdows . . . 99
7.2.1 Dorow and Widdows (2003a) . . . 99
7.2.2 Dorow et al. (2005) . . . 101
7.3 *HyperLex* - Véronis (2004) . . . 102
7.4 Agirre et al. . . . 103
7.4.1 Summary and Discussion . . . 105
7.5 Klapaftis et al. . . . 105
7.5.1 Klapaftis (2008) . . . 105

7.5.2 Klapaftis and Manandhar (2008) . . . 106
7.5.3 Summary and Discussion . . . 108
7.6 *SquaT++* and *B-MST* - Di Marco and Navigli (2013) . . . 108
7.6.1 *SquaT++* . . . 108
7.6.2 *B-MST* . . . 109
7.6.3 A Web Search Result Clustering Task . . . 109

III Systems and Evaluations 112

8 Preliminary Evaluations 113
8.1 Evaluation 1: A Language Separation Task . . . 113
8.1.1 Language Separation . . . 114
8.1.2 Evaluation Data . . . 114
8.1.3 The Test Set . . . 115
8.1.4 The Test Set Graph . . . 115
8.1.5 Mapping Languages to Clusters . . . 116
8.1.6 Evaluation Measures . . . 117
8.1.7 Evaluation Results . . . 119
8.2 Evaluation 2: A Word Sense Induction Task . . . 120
8.2.1 Two Approaches to Word Sense Induction . . . 121
8.2.2 Dorow's Approach . . . 122
8.2.3 The *MaxMax/Chinese Whispers* Approach . . . 124
8.2.4 Evaluation Results . . . 125
8.2.5 Evaluation Results: Discussion . . . 131
8.3 A Summary of the Preliminary Evaluations . . . 132

9 Clustering Coefficients and Word Sense Induction 133
9.1 Introduction . . . 133
9.1.1 Evaluations . . . 133
9.1.2 Approaches . . . 133
9.1.3 Aim . . . 134
9.1.4 Graph Generation . . . 134
9.2 Clustering Coefficients and Ambiguity . . . 136
9.3 Clustering Coefficient Approaches to WSI . . . 139
9.3.1 Dorow's Approach . . . 139

9.3.1.1 Using Graph Percolation to Find a Clustering Coefficient Threshold . . . 139
9.3.1.2 Dorow's Method for Inducing Word Senses . . . 142
9.3.1.3 Problems with Dorow's Approach . . . 144
9.3.2 Two Novel Approaches . . . 147
9.3.2.1 A Strong-Weak Clustering Coefficient (SWCC) Approach . 147
9.3.2.2 An Edge Weight Thresholding (EWT) Approach . . . 149
9.4 Evaluation Methodology . . . 150
9.4.1 Mapping Clusters to Senses . . . 150
9.4.1.1 A Method to Map Clusters to Senses . . . 151
9.4.2 Evaluating Clusters . . . 151
9.5 Evaluation Results . . . 155
9.5.1 Strong-Weak Clustering Coefficient Approach Results . . . 155
9.5.1.1 Evaluation of Six Weighted Clustering Coefficient Measures in the Strong-Weak Clustering Coefficient Approach . 158
9.5.2 Edge Weight Thresholding Approach Results . . . 160
9.6 Conclusions . . . 163

10 The SemEval Word Sense Induction Task 165

10.1 The SemEval-2010 Word Sense Induction and Disambiguation Task . . . 165
10.2 Evaluation Measures . . . 166
10.2.1 The V-Measure . . . 166
10.2.2 The Paired F-Score . . . 167
10.3 Three Novel WSI Systems Applied to the SemEval-2010 Word Sense Induction & Disambiguation Task . . . 171
10.3.1 The Shared Nearest Neighbours Systems . . . 171
10.3.2 The Word Overlap System . . . 173
10.3.3 Assigning Clusters to Target Word Instances . . . 174
10.4 Evaluation Results . . . 175
10.4.1 V-Measure Results . . . 175
10.4.1.1 V-Measure Results Analysis . . . 176
10.4.1.2 V-Measure Results Discussion . . . 177
10.4.2 Paired F-Score Results . . . 179
10.4.2.1 Paired F-Score Analysis . . . 179

10.4.2.2 Paired F-Score Results Discussion 180
10.4.3 Using Different Clustering Algorithms in the SNN and Word Overlap Systems 182
10.4.3.1 V-Measure Results 182
10.4.3.2 Paired F-Score Results 183
10.4.3.3 Key Points and Conclusions 184
10.4.4 Supervised Evaluation Results 185
10.4.5 Using Clustering Coefficient Measures to Induce Word Senses in the Test Set 189
10.5 Conclusions 191

11 Summary, Conclusions and Future Research 194
11.1 Summary and Conclusions 194
11.2 Future Research 197

List of Tables

3.1 Log Likelihood Ratio (LLR) significance. 35
3.2 Observed frequencies. 35
3.3 Expected frequencies. 35

5.1 *Chinese Whispers* and an 'ideal' clustering algorithm. 70
5.2 The Shapes test set. 77
5.3 Jain results. 80
5.4 R15 results. 80
5.5 D31 results. 81
5.6 Aggregation results. 81
5.7 Flame results. 81
5.8 Compound results. 81
5.9 Path-Based results. 81
5.10 Spiral results. 82
5.11 The four algorithms ranked by average Precision, Recall, and F-Score over the eight datasets in the Shapes test set. 82
5.12 Aspects of the *MaxMax* clustering algorithm. 86

7.1 Two senses of *cherry*. 100
7.2 Contingency table for target word **network** and context word ***cable*** 107

8.1 The number of sentences in the test set *AW*. 115
8.2 An illustration of the mapping method proposed in Biemann (2007). 116
8.3 Evaluation results. 119
8.4 A possible mapping between clusters in $G_{\neg cherry}$ and senses of *cherry*. . . . 121
8.5 Cherry *Dorow-MCL*. 126
8.6 Cherry *MaxMax/Chinese Whispers* 126
8.7 Jersey *Dorow-MCL*. 127

8.8 Jersey *MaxMax/Chinese Whispers* . . . 127
8.9 Oil *Dorow-MCL*. . . . 128
8.10 Oil *MaxMax*. . . . 128
8.11 Head *Dorow-MCL*. . . . 129
8.12 Head *MaxMax*. . . . 129
8.13 Squash *Dorow-MCL*. . . . 130
8.14 Squash *MaxMax/Chinese Whispers* . . . 130
8.15 Lemon *Dorow-MCL*. . . . 131
8.16 Lemon *MaxMax*. . . . 131

9.1 Correlation between WordNet 1.7.1 polysemy counts and word frequency, vertex degree, and the clustering coefficient (CC). . . . 136
9.2 Correlation between WordNet 3.0 polysemy counts and word frequency, vertex degree, and the clustering coefficient (CC). . . . 137
9.3 Correlation strength (Cohen, 1988). . . . 138
9.4 Examples of clusters returned by Dorow's method. . . . 143
9.5 Clusters mapped to WordNet senses for the target word *orange*. . . . 153
9.6 *orange* calculation. . . . 153
9.7 Clusters mapped to WordNet 3.0 senses for the target word *mouse*. . . . 154
9.8 *mouse* calculation. . . . 154
9.9 Results reported in Pantel and Lin (2002) and Dorow (2007). . . . 155
9.10 Results for Dorow's approach using WordNet 3.0. . . . 155
9.11 Strong-weak clustering coefficient approach, $Precision_{Strict}$. . . . 156
9.12 Strong-weak clustering coefficient approach, $Precision_{Relaxed}$. . . . 156
9.13 $Precision_{Strict}$ results for the clustering coefficient measures. . . . 158
9.14 $Precision_{Relaxed}$ results for the clustering coefficient measures. . . . 159
9.15 Edge Weight Thresholding approach, $Precision_{Strict}$. . . . 160
9.16 Edge Weight Thresholding approach, $Precision_{Relaxed}$. . . . 161

10.1 Paired F-Score example. . . . 168
10.2 Misclassification relative to cluster size. . . . 170
10.3 Systems ranked by V-Measure. . . . 178
10.4 Systems ranked by Paired F-Score. . . . 181
10.5 Ranked V-Measure clustering results. . . . 182
10.6 Ranked Paired F-Score clustering results. . . . 184

10.7 Systems ranked by Supervised Recall (SR). 186
10.8 Systems ranked by Supervised Recall (SR). 187
10.9 Average number of clusters generated 188
10.10Results for the seven clustering coefficient measures. 190

List of Figures

1.1 A word co-occurrence graph for the target word *orange*. 2
1.2 A word co-occurrence graph for the target word *cherry*. 5
1.3 Sense clusters in the word co-occurrence graph of Figure 1.2. 6

2.1 An undirected graph. 16
2.2 An undirected edge $e_{\{v_i,v_j\}}$ between an unordered vertex pair $\{v_i, v_j\}$. . . . 17
2.3 A directed graph. 17
2.4 A directed edge $e_{(v_i,v_j)}$ between an ordered vertex pair (v_i, v_j). 17
2.5 Two arcs $e_{(v_i,v_j)}$ and $e_{(v_j,v_i)}$. 18
2.6 A weighted undirected edge. 18
2.7 A weighted directed edge. 18
2.8 Clustering using edge weights. 19
2.9 A connected and disconnected graph. 20
2.10 Complete graphs. 21
2.11 Two cliques in G. 21
2.12 Two 'senses' of orange. 22
2.13 Vertex degree. 23
2.14 Neighbours of a vertex v. 23
2.15 A triplet and a triangle (a 3-clique). 24
2.16 Vacuous triplets. 24
2.17 Non-vacuous triplets. 25
2.18 Square and diamond forms. 25
2.19 *sienna*'s first and second order neighbours. 26
2.20 *pomelo*'s first and second order neighbours. 26
2.21 Two representations of the same graph. 27
2.22 Two unweighted, undirected graphs: G_1 (a connected graph) and G_2 (a complete graph). 29

3.1 Conditional probability as edge weight in a directed graph. 34
3.2 An example of the use of conditional probability as edge weight in a directed graph. 34
3.3 The local clustering coefficient. 38
3.4 The local clustering coefficient for directed graphs. 39

4.1 Edges between neighbours of v are assigned an identity weight of 1. 41
4.2 A peripheral edge of v's neighbourhood. 42
4.3 Triplet values. 44
4.4 A frustrated triangle. 45
4.5 Distributing uniform weight (probability). 47
4.6 An integer weighted graph. 48

5.1 An input graph G. 51
5.2 Possible hierarchical and partitional clustering solutions for G. 51
5.3 Hard, soft, and fuzzy clustering solutions. 52
5.4 Prototype-based clustering. 53
5.5 DBSCAN: *Eps* of v. 55
5.6 DBSCAN: if *MinPts* is set to 4 then c is a core vertex, b is a border vertex and n is a noise vertex. 55
5.7 Shared Nearest Neighbours example. 57
5.8 A hypergraph. 60
5.9 Transformation of a graph G to a collocation (link) graph G'. 61
5.10 Assigning the predominant [class] label to vertex v. 67
5.11 *Chinese Whispers* clustering example. 67
5.12 *Chinese Whispers* and non determinism. 68
5.13 Tetrahedron graph. 69
5.14 A 3-bipartite-clique graph. 69
5.15 A weighted graph G transformed to an unweighted directed graph G'. . . . 72
5.16 Two clusters in G': $\{r, s, t, u, v\}$ and $\{t, w, x\}$. 73
5.17 A weighted 3-bipartite-clique graph. 75
5.18 The Shapes test set. 76
5.19 *DBSCAN*: 4th nearest nearest neighbour distance plot for the Spiral dataset. 78
5.20 The two *GS* clusters in the Flame dataset. 84
5.21 *MaxMax*: clusters in the vertex sequences of the Spiral dataset. 84

5.22 *MaxMax*: soft clustering example. 86
5.23 *MaxMax*: maximal affinity clustering example. 87
5.24 *MaxMax*: uniform edge weights example. 87

6.1 G_{orange}. 95
6.2 $C_{G_{orange}}$, a clustering solution. 95
6.3 Labelling clusters. 96
6.4 Labelling the 'fruit' clusters in $C_{G_{orange}}$. 96

8.1 G_{cherry}, a word co-occurrence graph for the polysemous word *cherry*. 120
8.2 $G_{\neg cherry}$: G_{cherry} with the *cherry* vertex deleted. 121
8.3 Widdows' class labelling algorithm. 123

9.1 Generating $G_{WordNet}$: nouns that co-occur in a coordination pattern are connected. 135
9.2 Correlation between WordNet 3.0 polysemy counts and the unweighted clustering coefficient (CC), and polysemy counts and the weighted clustering coefficient (WCC). 137
9.3 Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient (CC) threshold θ. 140
9.4 The number of clusters $|clusters|_{rel}(\theta)$ returned as a function of the clustering coefficient (CC) threshold θ. 141
9.5 Compute clustering coefficient (CC) scores for all vertices in the graph. . . 142
9.6 Delete vertices with CC scores $< \theta = 0.30$. 142
9.7 Expansion of core clusters. 143
9.8 Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient threshold θ (WCC). 144
9.9 Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient threshold θ (WCC). 145
9.10 The number of clusters $|clusters|_{rel}(\theta)$ returned as a function of the clustering coefficient threshold θ (WCC). 145
9.11 $n^{\text{th.}}$ order infections. 146
9.12 A strong-weak clustering coefficient approach. 147
9.13 A maximal clique. 148
9.14 A strong-weak clustering coefficient approach for the target word *mouse*. . . 149

9.15 The edge weight thresholding approach applied to the target word *orange*. The threshold T_w is set to 3. 150

9.16 F-Scores for $Baseline_{Cliques}$, WCC_{Counts}, and *Dorow* for all values of the similarity threshold σ. 157

9.17 F-Scores ($Precision_{Strict}$) for the highest scoring measures in the strong-weak clustering coefficient (SWCC) approach and the edge weight thresholding (EWT) approach, all values of the cluster-sense similarity threshold σ. 162

9.18 F-Scores ($Precision_{Relaxed}$) for the highest scoring measures in the strong-weak clustering coefficient (SWCC) approach and the edge weight thresholding (EWT) approach, all values of the cluster-sense similarity threshold σ. 163

10.1 The standard F-Score and Paired F-Score for % misclassified items. 169

10.2 Misclassification relative to cluster size. 170

10.3 V-Measure results: verbs. 175

10.4 V-Measure results: nouns. 176

10.5 Paired F-Score results: verbs. 179

10.6 Paired F-Score results: nouns. 180

Chapter 1

Introduction

This thesis is a study of how word meanings can be discovered directly from text through automated analysis of the contexts in which words are used. In particular, this thesis posits that word meanings can be defined solely in contextual terms, that is, without recourse to ostensive, taught, or dictionary definitions of what words mean; a hypothesis which presumes that: "The complete meaning of a word is always contextual, and no study of meaning apart from context can be taken seriously" (Firth, 1935, p.37), alternatively stated, in less dogmatic terms, as: "You shall know a word by the company it keeps" (Firth, 1957, p.11).

Finding the company a word keeps is approached in this thesis from a graph-theoretic perspective. In this approach, contexts of a word, e.g. sentences containing the word, are modelled using a word co-occurrence graph: words occurring in contexts are represented as *vertices* and word pairs that co-occur in contexts are connected by an *edge* (as illustrated in Figure 1.1 for the target word *orange*). A clustering algorithm is then applied to the word co-occurrence graph to partition words to groups of words, each of which defines a distinct use of the target word with this use taken to represent a discrete sense of the target word.

This approach to finding word meanings is based on the *distributional hypothesis* (Harris, 1954), the notion that words occurring in similar contexts tend to have similar meanings. Presumption of a correlation between distributional similarity and semantic similarity therefore allows one to utilise the former in order to estimate the latter (Sahlgren, 2008). In graph-theoretic terms, this translates to an assumption that subgraphs of a word co-occurrence graph with high vertex intra-connectivity contain semantically similar words, thus can be used to represent word meanings. For example, in the word co-occurrence graph shown in Figure 1.1, the fully connected subgraph containing the vertices: apple, banana, peach, and pear could be used to represent the FRUIT sense[1] of the target word *orange*. This approach to finding word meanings is therefore one in which "difference in meaning correlates with difference in distribution" (Harris, 1954, p.43) and where a word co-occurrence graph is used to exemplify "a network of [word] relations, and these relations really hold in the data investigated" (Harris, 1954, p.36).

Finding the degree to which relations between words 'really hold' requires quantifi-

[1] SMALL CAPITAL typeface is used throughout the thesis to name word senses.

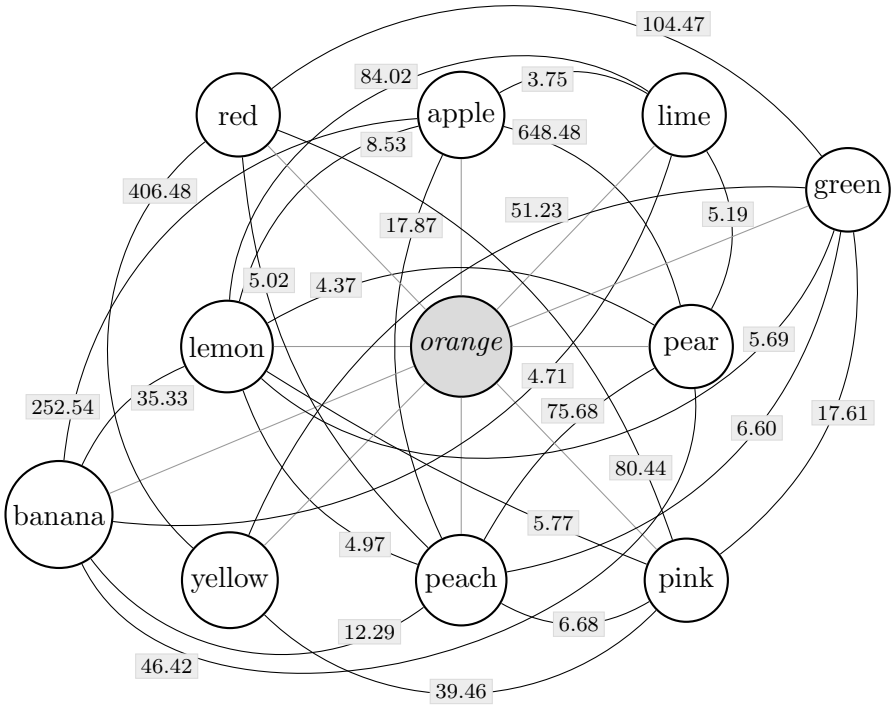


Figure 1.1: A word co-occurrence graph for the target word *orange*.

cation of the strength of these relations. This can be achieved by weighting relations between words. For example, in Figure 1.1 weights on edges quantify the significance of word pair co-occurrence relations, where higher values indicate greater significance of the co-occurrence relation between two words. Effectively, edge weights quantify how semantically similar word pairs are. Thus knowing, for example, that lemon and lime are semantically more similar than lemon and pink should be of more use in finding word meanings of the target word *orange* than simply knowing that these word pairs co-occur with *orange*.

By introducing a measure of the significance of word pair co-occurrence into graphs, the distributional hypothesis can be extended to that of the *statistical semantics hypothesis* (Weaver, 1955; Furnas et al., 1984), the notion that statistically significant patterns of word co-occurrence identify word meanings. In graph-theoretic terms, this translates to an assumption that subgraphs containing statistically significant word co-occurrence pairs define statistically significant patterns of word usage, thus can be used to represent word meanings.

The second hypothesis of this thesis is that parameter-free systems for finding word meanings in text are more suited to the task than those that are parameterised. Many of the systems applied to the task consist of a series of parameterised procedures (Pantel and Lin, 2002; Véronis, 2004; Agirre et al., 2006a; Klapaftis and Manandhar, 2010a,b; Di Marco and Navigli, 2013). In these systems, words are required to fit the parameters of these procedures. Returning the best set of word meanings therefore requires parameter

tuning of these procedures. Moreover, parameter tuning may be required each time the disambiguation scenario changes, for example, when applying a parameterised system to text in a previously unseen domain. Parameter-free systems, in comparison, can find the number of word meanings in the text under analysis automatically (Jurgens and Klapaftis, 2013; Hope and Keller, 2013b).

1.1 Word Sense Induction

Having outlined the major hypotheses of this thesis in the previous section, this section presents a formal introduction to the task of automatically identifying word meanings from text, a task known in Natural Language Processing (NLP) as Word Sense Induction (WSI)[2]. This section first illustrates why WSI, as an unsupervised method for identifying word meanings, may be preferable to approaches that are supervised by external knowledge. An introduction to graph-based WSI then follows: the approach to WSI that is applied in this thesis.

WSI is an open problem in NLP with various approaches having been applied to its solution (surveys of which are given in Apidianaki and Van de Cruys (2011); Navigli (2012); Nasiruddin (2013) and reviewed in Chapters 7 and 6). Though shown to identify word senses, these approaches have yet to better those of supervised approaches (Navigli, 2009). However, WSI is a relatively new research topic in NLP, therefore there is scope to improve upon existing approaches. Furthermore, WSI is a completely unsupervised process, thus would be of utility in NLP tasks that require word senses identified but that have no recourse to supervised methods to do so. For example, a WSI system could be applied to texts drawn from particular domains in order to find domain specific word senses: senses that may be undefined in standard dictionaries or in the lexical resources used by supervised systems. More generally, WSI has the potential to provide a huge breakthrough in the semantic analysis of the vast quantity of text that is now available to NLP researchers. For example, consider the very large corpora used in NLP, corpora such as the 1.9 billion[3] word ukWaC[4] corpus and Google's trillion[5] word Web 1T corpus[6]. Since it would be impracticable to manually disambiguate word meanings in corpora of this size, automated methods for identifying word senses would be required.

As noted in the introduction to this chapter, WSI adopts the distributional hypothesis of meaning (Harris, 1954; Firth, 1957), the hypothesis that words occurring in similar contexts tend to have similar meanings. In so doing, word senses can be induced through automated analysis of the distribution of word use, a straightforward example of which is to: count word co-occurrences in some predefined context (e.g. sentences); group together pairs of words whose co-occurrence is frequent (or found to be statistically significant); take each group of words to represent a word sense. For example, in Figure 1.1 words in

[2]The task has also been referred to as automatic word sense discrimination (Schütze, 1998) and unsupervised word sense disambiguation (Navigli, 2012).

[3]1,914,150,197 words.

[4]`http://wacky.sslmit.unibo.it/doku.php?id=corpora#english`

[5]1,024,908,267,229 words.

[6]`http://catalog.ldc.upenn.edu/LDC2006T13`

the pairs: (green, red), (pink, red), (red, yellow) could be taken to represent the COLOUR sense of the target word *orange*, with words in the pairs: (apple, pear), (apple, banana), (peach, pear) taken to represent *orange*'s FRUIT sense. Words in the remaining pair (lemon, lime) could then be taken to represent the finer-grained CITRUS FRUIT sense of *orange*. This simple model of word co-occurrence therefore returns representations of the senses of *orange* and also gives some indication of their granularity, an approach to finding word meanings that can be completely automated and that requires no recourse to external knowledge resources, e.g. sense inventories such as dictionaries and thesauri, nor to example-based learning resources such as sense annotated corpora: resources that contain only a limited number of senses words take and that are time consuming and expensive to produce.

This issue of acquiring and accessing sufficient resources to cover the needs of a task is known as the *knowledge acquisition bottleneck* (Gale et al., 1992b), an issue that affects many NLP tasks (Agirre and Edmonds, 2007), notably one that is highly related to WSI: Word Sense Disambiguation (WSD). WSD is the task of determining which of the senses of an ambiguous word best fits a particular use of the word. However, WSD, unlike WSI, is a supervised process that uses external knowledge to disambiguate word meanings. For example, it is common for a WSD system to use a fixed list of senses, contained in some sense inventory, e.g. WordNet (Miller et al., 1990). Words, therefore, can only be disambiguated to senses listed in the inventory used (Ide and Véronis, 1998; Manning and Schütze, 1999; Navigli, 2009). This assumption, however, that words have a fixed number of senses is an oversimplification of the complexity of word meaning. Words are not well behaved units of meaning: their meanings change to accommodate their use in language. Therefore, it is only by looking at how words are used in context that their meanings are fully realised, and this is exactly what WSI does. Thus, WSI, unlike WSD, can identify the senses of existing words and of those newly introduced into language: senses that may best fit their contextual use.

1.1.1 Graph-Based Word Sense Induction

The WSI systems described in this thesis use graphs to model word senses. This section presents a simple example of this approach to WSI, an approach that is expanded upon in later chapters.

1.1.1.1 A Graph Model of Word Context

A polysemous word without context is ambiguous. For example, the polysemous word *cherry*, on its own, could take any of its senses: *cherry* as a FRUIT, *cherry* as a TREE, as a COLOUR, a FLAVOUR, and so on. It is only by observing the context in which *cherry* is used that its sense becomes apparent. For example, *cherry* in the context:

cherry, plum and apricot tarts

is clearly *cherry* in its FRUIT sense as plum and apricot are observed. Though plum and apricot can, as *cherry* can, take COLOUR senses, they also take FRUIT senses; furthermore,

plum and apricot can, as *cherry* can, be used as FRUIT fillings in tarts.

One way to model the contexts of a polysemous word is to use a word co-occurrence graph, where vertices represent words occurring in contexts of the polysemous word and edges (links between vertices) represent word pair co-occurrence. For example, Figure 1.2 shows a word co-occurrence graph for the target word *cherry*, where vertices in this instance are adjectives and nouns occurring in contexts of *cherry*.

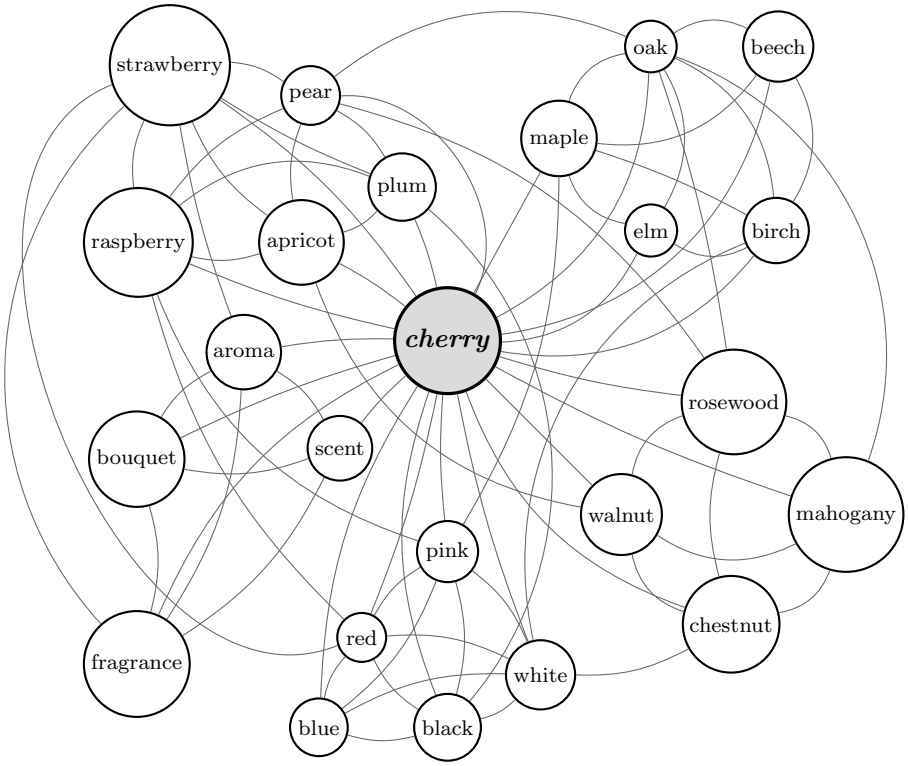


Figure 1.2: A word co-occurrence graph for the target word *cherry*.

1.1.1.2 Identifying Word Senses

The senses of a target word are identified by applying a clustering algorithm to partition the words in the target word's word co-occurrence graph to a set of *sense clusters*. For the unweighted graph shown in Figure 1.2, the clustering algorithm applied would use the connectivity structure of the graph to find subgraphs of highly interconnected vertices; for the weighted graph shown in Figure 1.1, the clustering algorithm applied would use the graph's edge weights to partition vertices in the graph to a set of sense clusters. Words in each sense cluster are then taken to represent a distinct sense of the target word. For example, Figure 1.3 shows that five sense clusters are found for the target word *cherry*, where the label assigned to a cluster, e.g. WOOD for the cluster: chestnut, mahogany, rosewood, walnut, is obtained by mapping the words in the cluster to a synset (sense) in WordNet (Miller et al., 1990), a method proposed in Widdows (2003) and described in full in Chapter 8.

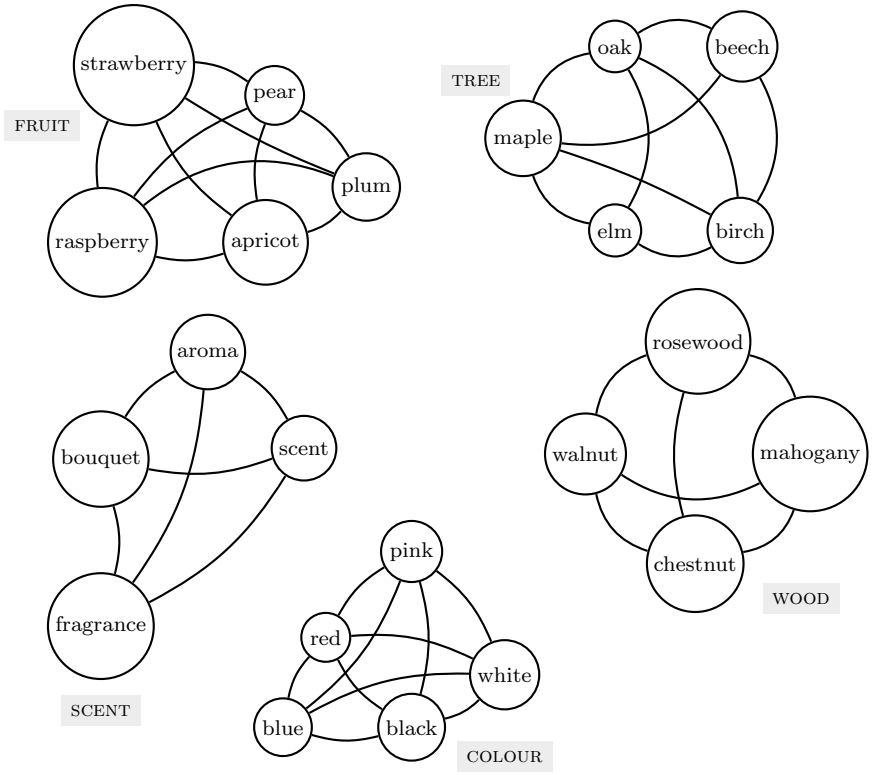


Figure 1.3: Sense clusters in the word co-occurrence graph of Figure 1.2.

1.1.1.3 Assigning Induced Senses to Words

Once obtained, sense clusters of a target word can be used to disambiguate the target word in further contexts. Effectively, this provides a way in which to carry out unsupervised word sense disambiguation. A straightforward method of doing this is to use the label of the sense cluster that has the greatest number of words in common with the context of the target word. For example, if *cherry* is the target word in the context:

cherry, **strawberry** and **raspberry** yoghurt,

cherry would be tagged, correctly in this instance, with its FRUIT sense as the FRUIT sense cluster in Figure 1.3 has the highest word overlap with the context. Things, however, are rarely this simple. For example, *cherry* in the context:

> I worked one to death yesterday, of a **cherry** tree against **blue** sky, the young shoots of the leaves were orange and gold, the clusters of flowers **white**[7],

would be incorrectly tagged with its COLOUR sense. Nevertheless, this approach of finding clusters in word co-occurrence graphs is shown in evaluations to return results that are comparable with those of the best-performing WSI systems (Dorow, 2007; Biemann, 2007), and it is this approach to WSI that is used as the basic model for many of the approaches described in this thesis.

[7]Letter from Vincent van Gogh to Emile Bernard. Arles, Thursday, 19th. April 1888. "J'en ai ereinté une hier, d'un cérisier contre ciel bleu, les jeunes pousses des feuilles étaient de l'orangé et de l'or, les touffes de fleurs blanches." `http://vangoghletters.org/vg/letters/let599/letter.html`

1.2 Motivation

Knowing what words mean, or what they identify, is a key requirement in many NLP tasks. To date, WSI has been shown to be of use in only a limited number of NLP tasks, for example, in: Question-Answering (Véronis, 2004; Di Marco and Navigli, 2013), Machine Translation (Apidianaki, 2008a), Taxonomy Learning (Klapaftis and Manandhar, 2010b), Named Entity Recognition (Artiles et al., 2010), and Novel Sense Detection (Dorow, 2007; Lau et al., 2012). However, WSI has the potential to assist many other NLP tasks. Indeed, any NLP task that requires words disambiguated could benefit from WSI. This section argues the case for the use of WSI in NLP by providing overviews of a number of tasks that have either benefited from WSI or that might benefit from its use.

1.2.1 Lexicography

WSI systems provide empirical evidence of word senses, returning clusters of words, each of which represents a particular word sense. Values that quantify semantic similarity between cluster words also provide statistically significant indicators of word sense (Navigli, 2009). Analysis of the clusters returned by a WSI system may therefore assist lexicographers in finding new words that take new or existing senses and in finding new senses of existing words; an analysis that may alleviate the time consuming process of having to go through text concordances in order to identify word senses. Furthermore, as WSI systems automatically extract the senses of words that are present in the text under consideration they can be applied to domain specific text to find domain specific senses of words or applied to text in very large corpora to find rarer senses of words (Pantel and Lin, 2002): in both cases, to find senses of words that may be undefined in general dictionaries,

Lexicological tasks that have directly benefited from unsupervised word sense discovery techniques include: synonym identification (Baroni and Bisi, 2004; Dorow, 2007; McCarthy et al., 2010), ontology learning (Klapaftis and Manandhar, 2006; Cohen and Widdows, 2009), lexical substitution (McCarthy and Navigli, 2007), lexical acquisition (Widdows and Dorow, 2002; Senellart and Blondel, 2008), finding predominant senses of words (McCarthy et al., 2007; Boyd-Graber and Blei, 2007; Tejada-Cárcamo et al., 2010), and finding graded senses of words (Erk et al., 2012; Jurgens and Klapaftis, 2013). WSI is shown to be of use in the detection of novel senses of words in Dorow (2007) and Lau et al. (2012).

1.2.2 Information Retrieval

Information Retrieval (IR) is the task of retrieving information from an information source that is relevant to an information request (Manning et al., 2008; Baeza-Yates and Ribeiro-Neto, 2011). The most common example of IR is that of querying a search engine: a user submits a query to the search engine; the search engine returns documents containing the terms (words) in the query. For example, given the query Pluto a search engine might return documents on Pluto as a planet, a Greek god, or a Disney character: an ambiguous set of documents containing different senses of the word Pluto. Retrieving documents that are relevant to the intended meaning of the user's query therefore requires accurate

disambiguation of the words in the query and of those in the documents stored by the IR system.

IR systems have, until recently, relied on the user to provide enough context (enough words) in the query to disambiguate its intended meaning. For example, a user wanting to retrieve documents on Pluto as a celestial body might submit the query Pluto solar system. No semantic analysis is applied to the query terms, simply a matching of the query terms with those contained in stored documents; the query, therefore, is semantically ambiguous with respect to the information resource. However, IR systems have recently begun to apply semantic analysis to user input queries, analysis known in IR as Semantic Search (Guha et al., 2003; Mäkelä, 2005). For example, Google's *Knowledge Graph*[8] and *Hummingbird* algorithm[9] and Microsoft's *Satori*[10] all apply some form of semantic analysis to user input queries. However, these are proprietary IR systems, and though some of the basic detail of the Semantic Search processes applied in these systems is known, the specifics are not. Nevertheless, WSI is shown to be of use in Semantic Search in Véronis (2004), which describes how WSI can be applied in a semantic analysis of user input queries in order to return documents related to the meanings of these queries, and in Di Marco and Navigli (2013), where WSI is used to rank and diversify search engine results.

1.2.3 Machine Translation

Machine Translation (MT) is the task of automatically translating text or speech from one language (the source language) to another language (the target language). MT is straightforward for words that have a one to one source to target language translation. For example, the English word kitten translates directly to kissanpentu in Finnish and killing in Danish. However, the right lexical choice must be made for source language words that have different translations for different senses in the target language (Koehn, 2010). For example, a French to English MT system must be able to recognise that the French noun feuille can, depending on the context in which it used, translate to the English noun leaf or paper (amongst other possible translations). Thus feuilles in the French context feuilles d'automne should be translated to its English equivalent: autumn leaves. Agirre and Edmonds (2007) notes that MT systems rarely incorporate a separate word sense disambiguation component into the translation process, however this is shown to be of use in Vickrey et al. (2005); Carpuat and Wu (2007), and Chan et al. (2007), with Apidianaki (2008a) showing how clusters returned by a WSI system can be used for lexical selection in MT systems.

1.2.4 Information Extraction

Information Extraction (IE) is the task of automatically extracting information from unstructured data, e.g. text documents, or semi-structured data, e.g. tables (Feldman and

[8] http://www.google.co.uk/insidesearch/features/search/knowledge.html
[9] http://www.bbc.co.uk/news/business-24292897
[10] http://www.bing.com/blogs/site_blogs/b/search/archive/2013/03/21/satorii.aspx

Sanger, 2007; Miner et al., 2012). Extracting the right information, however, requires accurate disambiguation of the source data. For example, a medical IE system may require information extracted from documents that contain the word coke in its drug sense as opposed to coke in its soft drink sense, a task that requires contextual disambiguation of the word coke. Indeed, many IE tasks can be viewed as disambiguation tasks, for example:

- **Terminology Extraction** requires the identification and extraction of terms in domain specific documents, e.g. in Bioinformatics genes and their proteins have the same name (Lesk, 2008), thus these terms must be disambiguated if they are to be correctly classified.

- **Named Entity Recognition** requires named entities (e.g. individuals, organisations, locations etc.) identified, e.g. that Timothy D. Cook is head of Apple Inc., and that Elizabeth Taylor can refer to two different people: Elizabeth Taylor the actress and Elizabeth Taylor the author.

- **Co-reference Resolution** requires disambiguation of co-reference and anaphoric links, e.g. that BBC and British Broadcasting Corporation refer to the same entity, and that she in the context Daisy felt sick because she'd eaten too much refers to Daisy.

- **Relationship Extraction** requires identification of the relationships between entities, so must identify entities involved in a relationship and disambiguate the relationship between these entities, e.g. the relationship Company acquires Company can be drawn from examples such as Microsoft agrees to buy Nokia if the entities Microsoft and Nokia are identified as companies and the relationship agrees to buy is disambiguated as acquires.

- **Automatic Summarisation** is the task of automatically producing a concise summary of the essential points, facts, and statements contained in a document. Summaries can then be used, for example, as web snippets in search engines, or in news wire reports and news aggregation applications. A standard approach is to apply a surface-level analysis of the text using key phrase extraction or text segmentation techniques (Mitkov, 2003). However, a deeper, semantic, analysis of documents may produce better summaries, ones that align more closely with those produced by humans. Such an analysis would require disambiguation of the meanings of words in documents.

Though little research, to date, has been carried out on applying WSI to IE, unsupervised disambiguation of terms is shown in Chang et al. (2006) to be of use in Web Information Extraction Systems, with induction of word senses shown to be of use in Artiles et al. (2010), a paper that reports on a Web People Search (WePS) task: a Named Entity Recognition task that requires disambiguation of named individuals.

1.2.5 Social and Information Network Analysis

Social networks, such as Facebook[11] and Twitter[12], and information networks, the most obvious of which is the World Wide Web[13], are rich repositories of natural language data. Extraction and analysis of this data would therefore be of great value to many NLP tasks. Furthermore, these networks can be analysed in two distinct ways:

1. By applying a structural analysis to find connections between a network's *agents* (entities that play a role in the network, e.g. individuals, groups of individuals, organisations), with these connections then used to find the relative importance of agents in the network (Wasserman and Faust, 1994; Easley and Kleinberg, 2010; Kadushin, 2012).

2. By applying semantic (content) analysis to the the data stored on a network, analysis that may lead to a meaningful understanding of: the information generated by agents, the sentiments held by agents, and the linguistic interplay between agents (Feldman and Sanger, 2007; Bontcheva and Rout, 2012).

A joint analysis would, therefore, enable one to find *who said what to whom*, where: *who to whom* is found through structural analysis and *what* through content analysis. This raises possibilities for applying the graph-based systems that are introduced in this thesis to this analysis as structural and semantic properties of a network could be combined into a single graph representation: the semantic (content) properties of a network could be assigned to vertices in the graph; the structural properties of a network would be represented, naturally, as edges in the graph. Leaving such conjecture to one side, the following lists a number of network analysis tasks that might benefit from the use of graph-based WSI, as follows:

- **Content Analysis** - disambiguation of the content stored and shared on a network could lead to an understanding of the interactions between, and motives of, agents using the network. For example, disambiguation of the text in blogs may allow meaningful connections to be made between different bloggers or between blog posts that are written about particular subjects (Berendt and Navigli, 2006). Graph-based WSI has the potential be of use here by: 1.) linking blogs (or other types of articles); 2.) disambiguating both the readers' appraisal of an article's content (through analysis of the comments left by readers) and the sentiment the writer of the article holds (Read et al., 2007; Liu and Zhang, 2012).

- **Social Network Analysis** - graph-based WSI could be applied to social networks to find cliques of people that have some common interest, e.g. to find cliques of people that like some particular entity in some particular sense (Aleman-Meza et al., 2006; Mitzlaff et al., 2013), or used in topic tracking, e.g. to identify topics trending on Twitter (Ritter et al., 2010; Benhardus and Kalita, 2013). This information could

[11] `https://www.facebook.com/facebook`
[12] `https://about.twitter.com`
[13] `http://www.w3.org/Consortium`

then be used to target specific groups of people with particular information, e.g. adverts, recommendations, or news items.

- **The Semantic Web** - the idea proposed in Berners-Lee et al. (2001) of supplementing web pages with semantic content by assigning semantic tags to text entities. Given the scale of this task, automatic disambiguation of text entities to their context specific senses would be of huge benefit.

1.2.6 Motivation: Summary

This section's aim was to motivate an argument for the use of WSI in NLP by presenting a number of tasks that have either benefited from WSI or that might benefit from its use. It should be clear, however, that any NLP task that requires words disambiguated might benefit from WSI, notably those that have no recourse to supervised methods to do so. This section also suggested how graph-based WSI might be applied in an analysis of social and information networks by combining the structural and content properties of a network into a single graph representation.

1.3 Research Objectives

The research objectives of this thesis are as follows:

1. To introduce novel WSI solutions that improve upon those previously applied to the task. WSI is an open problem in NLP whose solution has potential benefits for many other NLP tasks, as illustrated in Section 1.2. However, the task has been studied by a relatively small band of researchers and often in set ways (Navigli, 2012; Nasiruddin, 2013). Therefore, there is scope to apply original solutions to the task by using different clustering algorithms, alternative feature sets, and novel measures of word co-occurrence and word cohesion.

2. To devise a graph-based clustering algorithm that is suited to the task of WSI. This algorithm should be:

    - Fast - in order to quickly partition a word co-occurrence graph to a set of clusters. Word co-occurrence graphs constructed from large corpora are likely to be large, therefore this algorithm should, ideally, run in time linear in the number of edges in the graph.
    - Unparameterised - graph vertices should not be constrained to fit a predetermined (parameterised) number of clusters. Vertices (words) should be left to self organise to word sense clusters, so allowing the number of clusters, hence the number of senses, in a word co-occurrence graph to be found automatically.
    - Deterministic - for a given input graph, the algorithm should always return the same clustering solution. Many clustering algorithms are parameterised, with changes to parameter values affecting the clustering solution: different values return different clustering solutions (Tan et al., 2006; Aggarwal and Reddy,

2013). Finding the best clustering solution using a parameterised algorithm therefore requires multiple runs of the algorithm; effectively, a trial and error approach to clustering (Jain and Dubes, 1988).

3. To analyse the measures used in WSI evaluations, measures that quantify the quality of a WSI system's clustering solution. This analysis should verify known issues with these measures (Meila, 2007; Klapaftis and Manandhar, 2013) though should also supply additional material and examples to illustrate why these measures are not well-suited to the evaluation of WSI systems. A further aim here is to consider the type of measure that might best evaluate a WSI system, notably with respect to how relationships between fine-grained and coarse-grained senses can be measured.

1.4 Contributions of the Thesis

This thesis introduces a number of novel methods for constructing, analysing, and partitioning graphs. These methods are incorporated into graph-based WSI systems which are then shown in various evaluations to return results comparable to those of the best-performing WSI systems. The main contributions of the thesis are: a novel parameter-free graph-based clustering algorithm that runs in time linear in the number of edges in the input graph, and novel generalisations of the clustering coefficient (Watts and Strogatz, 1998) to the weighted, directed case. The clustering algorithm returns quasi strongly connected components of the input graph: a soft clustering solution that allows vertices to belong to more than one cluster. Thus, if the edge weights in the input graph reflect the ambiguity of polysemous word it can be assigned to each of its various sense clusters. The clustering coefficient is an unweighted measure of vertex cohesion in graphs, shown to be of use in WSI (Dorow et al., 2005; Navigli and Crisafulli, 2010; Di Marco and Navigli, 2013). The assumption in this thesis is that weighted generalisations of this measure will better induce word senses. Further contributions of the thesis include: a review of the graph-based WSI systems that have been proposed in the literature; an analysis of the methodologies applied in WSI systems; an analysis of the measures used to evaluate WSI systems, and empirical evidence to show that each novel method introduced in this thesis is of utility in inducing word senses.

1.5 Outline of the Thesis

Chapter 2 sets out the graph-theoretic notation and concepts used throughout the thesis. The aim of this chapter is to provide the reader with the requisite knowledge to understand the graph models of language that are introduced in later chapters. This chapter first defines the notion of a graph, presenting examples of undirected and directed graphs, then illustrates how edge weights are introduced into graphs. This chapter also shows how particular patterns of vertex connectivity identify strong vertex cohesion, cohesion that can be used to find clusters of semantically related words in word co-occurrence graphs.

Chapter 3 defines a number of association measures, the values of which are used as edge weights in word co-occurrence graphs. This chapter begins by reviewing association measures that have been previously applied in NLP, finding that many of them are unsuited to graph-based WSI. This chapter also introduces the clustering coefficient (Watts and Strogatz, 1998), a measure of vertex cohesion in unweighted, undirected graphs.

Chapter 4 reviews weighted generalisations of the clustering coefficient that have been proposed in the literature. This review finds that the majority of these measures are not true generalisations to the weighted case. Noting this, novel weighted generalisations of the clustering coefficient are proposed.

Chapter 5 introduces clustering: the task of dividing a collection of data objects into groups. A number of clustering methods are outlined, notably those that are graph-based and that soft cluster vertices or are parameter-free. This chapter also introduces *MaxMax*, a novel parameter-free soft clustering algorithm. Empirical evidence is presented which shows that *MaxMax* returns clustering solutions that are comparable with those of three clustering algorithms representative of the same class as *MaxMax*.

Chapter 6 begins with a discussion regarding word sense in which various arguments are set out that attempt to define what word sense means. An introduction to Word Sense Induction (WSI) follows in which WSI is compared to the related task of Word Sense Disambiguation. This chapter also argues the case for a graph-based approach to WSI over other approaches that have been applied to the task.

Chapter 7 reviews graph-based approaches to WSI that have been proposed in the literature. The approaches described in this chapter are those considered in this thesis to be key works. Methods presented in these works are either adapted or extended in a number of the WSI systems that are introduced in this thesis..

Chapter 8 presents two preliminary evaluations of the *MaxMax* clustering algorithm. Though limited in scope, these evaluations clearly exemplify the viability of a graph-based approach to NLP. Results for the first evaluation show that *MaxMax* returns better results than *Chinese Whispers*, a clustering algorithm that has been shown to be of utility in several NLP tasks (Biemann, 2006a). Results for the second evaluation show that *MaxMax* returns results comparable to those of *Chinese Whispers*. This evaluation also shows that *MaxMax* induces a greater number of senses in a far more efficient manner than MCL (van Dongen, 2000), a state-of-the-art Markov model based clustering algorithm.

Chapter 9 reports on an evaluation that is far wider in scope than those presented in Chapter 8. This evaluation shows how the clustering coefficient measures, introduced in Chapters 3 and 4, can be used to induce the senses of nouns found in the British National Corpus. Two novel approaches are applied: the first uses a strong-weak clustering coefficient approach; the second uses an edge weight thresholding approach. A correlation

analysis, presented at the start of the chapter, indicates that the clustering coefficient and its weighted generalisation are poor predictors of word sense. However, this is not borne out in the evaluation where the clustering coefficient approaches are shown to return better results, and to have far greater coverage of word senses, than current, best-performing approaches to WSI.

Chapter 10 describes three novel WSI systems: two knowledge poor (unsupervised) systems; the third, knowledge enriched by grammatical relations. The systems are evaluated within the framework of the SemEval-2010 Word Sense Induction and Disambiguation task (Manandhar et al., 2010), with results compared to those of the twenty six participant systems. The evaluation shows that the novel systems return the best results if the V-Measure evaluation metric is applied (Rosenberg and Hirschberg, 2007), yet return the worst results for the Paired F-Score (Artiles et al., 2009). This leads to an analysis of the evaluation measures which finds them biased to favour degenerate clustering solutions, solutions that return higher scores than any participant system. The chapter concludes by presenting a number of suggestions that future evaluations might consider.

Chapter 11 summarises the thesis, noting contributions made to the study of Word Sense Induction. A number of conclusions are drawn and possibilities for future research outlined.

Part I

Graphs, Association Measures and Clustering

Chapter 2

Graphs

This chapter presents a number of elementary definitions from Graph Theory (Diestel, 2006; Bondy and Murty, 2011). The material covered here is straightforward; the aim, to provide the reader with sufficient knowledge to understand the graph models of language that are introduced in later chapters. Section 2.1 defines the notion of a graph. Section 2.2 shows how weights are introduced into graphs. Section 2.3 illustrates how particular patterns of vertex connectivity in graphs define unity amongst vertices, with Section 2.4 showing how these patterns can be used to measure semantic cohesion between words. Section 2.5 reviews graph properties that have been applied in graph-based approaches to Natural Language Processing (NLP), noting why these properties are not well-suited to graph-based Word Sense Induction (WSI).

2.1 Undirected and Directed Graphs

2.1.1 Undirected Graphs

A graph is a data structure used to represent relationships among a set of entities. In this thesis entities are words, with the relationships among a set of words defined by some measure of association between words, typically this will be a measure of the co-occurrence of word pairs in sentences.

To illustrate: Figure 2.1 shows a graph G in which a set of entities $\{a, b, c\}$ are represented as *vertices* $\{v_a, v_b, v_c\}$, with the relationships among these entities represented by *edges* connecting vertex pairs: $\{v_a, v_b\}, \{v_a, v_c\}, \{v_b, v_c\}$.

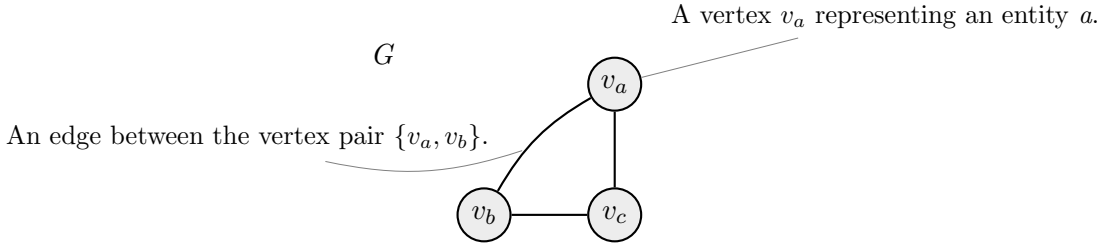


Figure 2.1: An undirected graph.

Formally, a graph $G = (V, E)$ consists of a set of vertices V and a set of edges E that connect vertex pairs: $\{\{v_i, v_j\} | v_i, v_j \in V\}$.

Undirected Edge: An undirected edge $e \in E$ between a vertex pair $\{v_i, v_j\}$ is denoted $e_{\{v_i,v_j\}}$. Note that set theoretic notation is used here to indicate that no direction is implied: v_i connects to v_j and v_j connects to v_i, a graphical representation of which is shown in Figure 2.2.

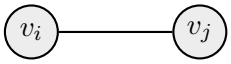


Figure 2.2: An undirected edge $e_{\{v_i,v_j\}}$ between an unordered vertex pair $\{v_i, v_j\}$.

Adjacent Vertices: Vertices v_i and v_j in Figure 2.2 are the *end points* of the edge $e_{\{v_i,v_j\}}$, said to be *adjacent* vertices of each other.

Order and Size: The number of vertices $|V|$ in a graph G is said to be the *order* of G. The number of edges $|E|$ in G is said to be the *size* of G. Thus, in Figure 2.1, the order and size of G is 3: three vertices; three edges.

2.1.2 Directed Graphs

A directed graph G contains edges with implied direction, as illustrated in Figure 2.3.

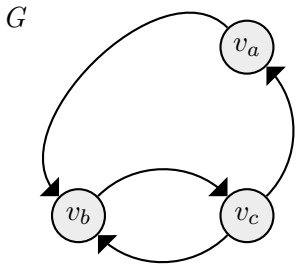


Figure 2.3: A directed graph.

Directed Edge: An edge between an ordered vertex pair (v_i, v_j) is denoted $e_{(v_i,v_j)}$, where the ordering of vertices indicates direction: v_i connects to v_j; v_j does not connect to v_i, a graphical representation of which is shown in Figure 2.4.

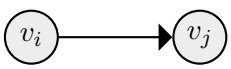


Figure 2.4: A directed edge $e_{(v_i,v_j)}$ between an ordered vertex pair (v_i, v_j).

Symmetry: Formally, a directed edge is said to be an *arc*. For two vertices v_i, v_j, if two arcs exist $e_{(v_i,v_j)}$ and $e_{(v_j,v_i)}$ then the relationship between v_i and v_j is said to be *symmetric*; otherwise, non-symmetric.

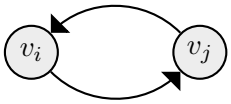


Figure 2.5: Two arcs $e_{(v_i,v_j)}$ and $e_{(v_j,v_i)}$.

In undirected graphs all relationships between vertex pairs are symmetric. In directed graphs relationships may be non-symmetric. As Figure 2.4 shows, v_i has a relationship to v_j that is not reciprocated, thus the relationship between the two vertices is non-symmetric. As Figure 2.5 shows, a symmetric relationship exists between between v_i and v_j. Note that as both edges are unweighted in Figure 2.5, this implies that the relationship between v_i and v_j is of equal value to each vertex.

2.2 Weighted Graphs

As illustrated in the previous section labels can be assigned to vertices, with these labels the nomenclature for the entities represented by the vertices. Similarly, each edge in a graph can be assigned a label, with the label indicating the relationship type or relationship strength between adjacent vertices.

Edge labels in this thesis represent the strength of the relationship between adjacent vertices, with the strength calculated by one of three association measures (defined in Chapter 3). The notation used is as follows.

Edge Weight Notation

A weighted undirected edge $w_{\{x,y\}}$ between an unordered vertex pair $\{x, y\}$ is shown in Figure 2.6, where the weight 0.5 is returned by an association measure, $AM(x, y) = 0.5$.

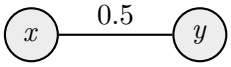


Figure 2.6: A weighted undirected edge.

A weighted directed edge $w_{(x,y)}$ between an ordered vertex pair (x, y) is shown in Figure 2.7, where the weight 0.5 is conditional probability, $p(y|x) = 0.5$: the association measure that is used in this thesis as edge weight in weighted directed graphs.

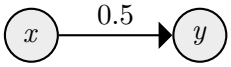


Figure 2.7: A weighted directed edge.

Edge Weights and Clustering

As will be discussed in Chapter 5, edge weights are used by clustering algorithms to partition graphs. Figure 2.8 gives a simple illustration of how edge weights are used to partition the graph G to two clusters.

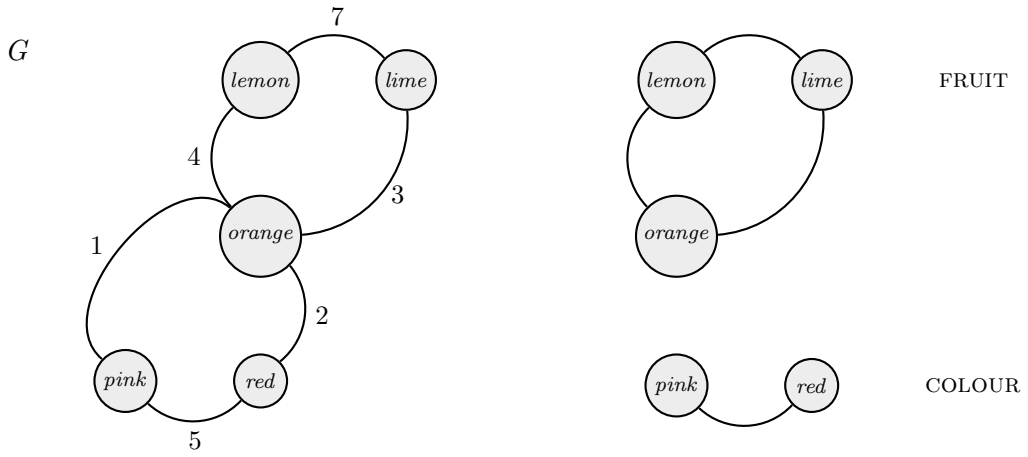


Figure 2.8: Clustering using edge weights.

Relating this example to word sense induction, the words in each cluster of Figure 2.8 can be taken to represent a word sense. Thus, if *orange* is the target word in G (the senses of which are to be induced) then the two clusters can be interpreted as representing *orange*'s FRUIT and COLOUR sense. Furthermore, the weights in G show that *orange* has higher edge weight to its 'FRUIT sense cluster' which might indicate that FRUIT is the predominant sense of *orange*. Admittedly, this is a highly rudimentary example of sense induction; it does, however, outline a number of key ideas that will be discussed throughout the thesis.

2.3 Connectivity in Graphs

2.3.1 Connected Graphs and Connected Components

Connected Graphs

A graph $G = (V, E)$ is said to be a *connected graph* if there is a *path* between every vertex pair $v_i, v_j \in V$; otherwise G is said to be a *disconnected graph*.

Path: A path is a sequence of adjacent edges in G. Whereas adjacent vertices share an edge, adjacent edges share a vertex. A path may traverse each edge, thus each vertex, in G once.

Shortest Path: A *shortest path*, also known as a geodesic path, is the shortest path between two vertices. Shortest paths are used as a measure of the *distance* between vertices.

Figure 2.9 shows a connected graph and a disconnected graph.

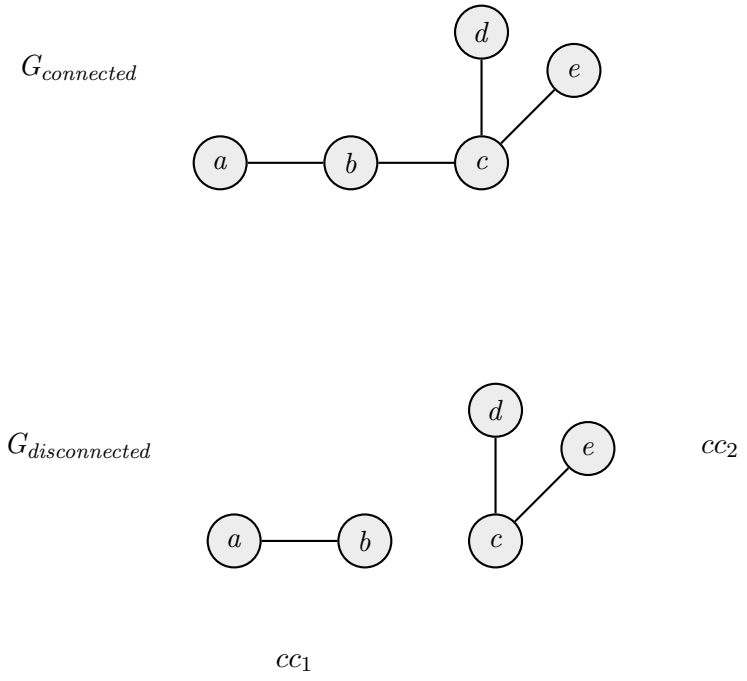


Figure 2.9: A connected and disconnected graph.

Connected Components

A *connected component* of a graph G is a connected *subgraph* of G. As Figure 2.9 illustrates $G_{disconnected}$ has two connected components cc_1 and cc_2.

Subgraph: A graph $G' = (V', E')$ is a subgraph of a graph $G = (V, E)$ if V' is a subset of V and E' is a subset of E. The subgraph relationship is defined as $G' \subseteq_S G$.

2.3.2 Complete Graphs

A complete graph is a fully connected graph, as shown in Figure 2.10.

Complete Graph: An undirected graph is said to be a *complete graph* if there is a edge between every vertex pair in the graph.

Complete Directed Graph: A directed graph is said to be a *complete directed graph* if, for every vertex pair v_i, v_j in the graph there is a directed edge from v_i to v_j and from v_j to v_i.

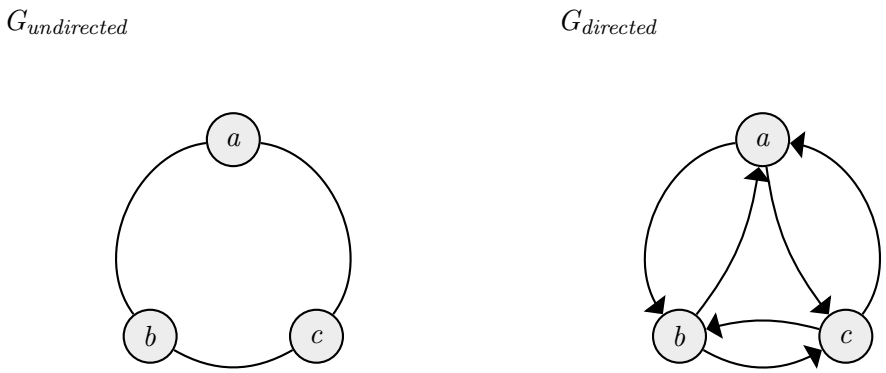


Figure 2.10: Complete graphs.

2.3.3 Cliques

A *clique* in a graph G is a complete subgraph of G. Figure 2.11 shows two cliques of G.

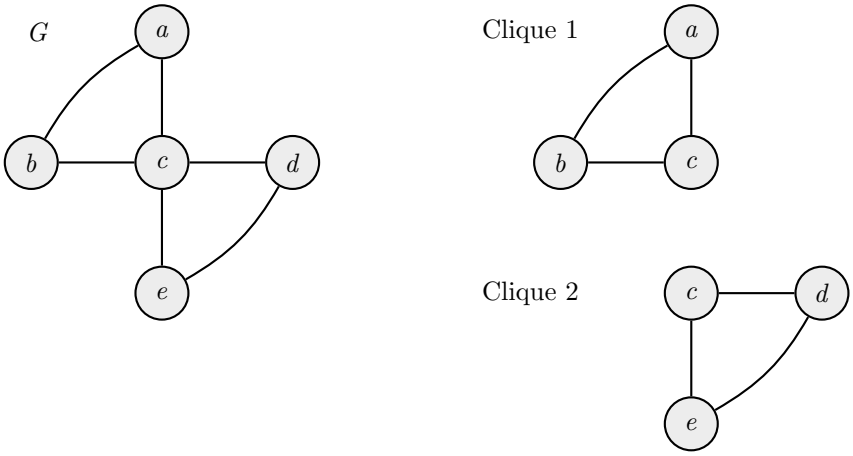


Figure 2.11: Two cliques in G.

***k*-clique:** A clique consisting of a k number of vertices is said to be a *k-clique*. The two cliques shown in Figure 2.11 contain three vertices, thus are said to be *3-cliques*.

Triangle: In graph theory, a 3-clique is commonly referred to as a *triangle*.

A Soft Clustered Vertex: The cliques in Figure 2.11 both contain vertex c. In terms of clustering, a vertex that is clustered to more than one cluster is said to be *soft clustered*. Germane to this thesis is the notion that soft clustered vertices may indicate ambiguity in language. For example, in Figure 2.12 the vertex *orange* is a member of two 3-cliques. If the words in the cliques are taken to represent *orange*'s senses, then two senses of *orange* may be inferred: *orange* in its COLOUR sense: $\{orange, pink, red\}$ and *orange* in its FRUIT sense: $\{orange, apple, pear\}$.

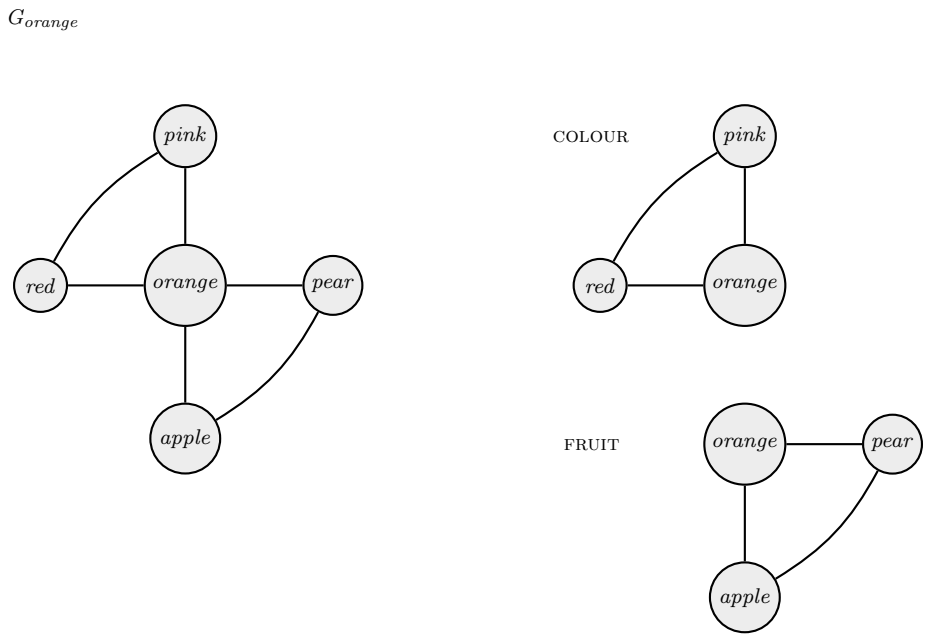


Figure 2.12: Two 'senses' of orange.

Note that the soft clustered vertex *orange* in Figure 2.12 acts as a hub for the words that define its senses: a "*semantic wormhole*" (Widdows, 2004, p.54), traversal of which leads from one distinct semantic space to another.

2.3.4 Vertex Degree

The *degree* of a vertex v, denoted $deg(v)$, is the number of edges attached to v. An edge attached to v is said to be *incident* with v.

In an undirected graph $deg(v)$ is the number of edges incident with v. In a directed graph, $deg(v)$ is defined by its *indegree* and *outdegree*. The indegree of a vertex v is the number of edges directed at v, denoted $deg^{in}(v)$. The outdegree of a vertex v is the number of edges directed away from v, denoted $deg^{out}(v)$. For example, in Figure 2.13 $deg(v)$ in $G_{undirected}$ is 3. In $G_{directed}$, $deg^{in}(v) = 3$ and $deg^{out}(v) = 2$.

A vertex with zero degree is known as a *singleton*. A vertex with $deg^{in} = 0$ and $deg^{out} \geq 1$ is called a *source*. A vertex with $deg^{out} = 0$ and $deg^{in} \geq 1$ is called a *sink*. Figure 2.13 illustrates these vertex types, where vertex o is shown to be a source and vertex i a sink.

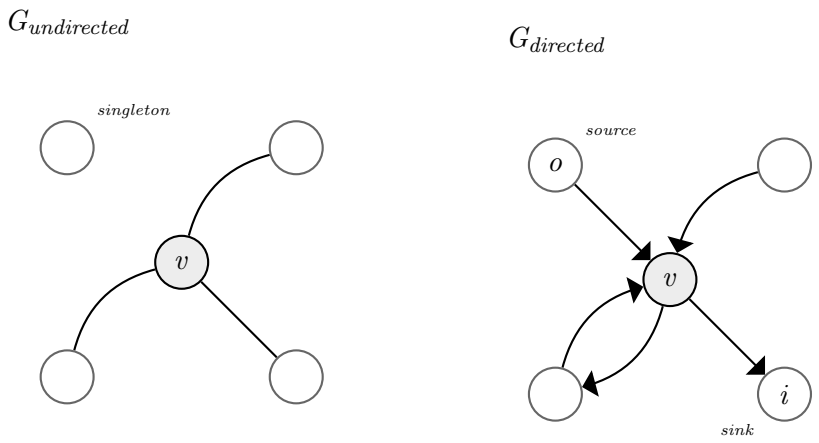


Figure 2.13: Vertex degree.

2.4 Vertex Neighbours

Vertices adjacent to a vertex v are said to be *neighbours* of v, denoted $\mathcal{N}(v)$. For example, in Figure 2.14 $\mathcal{N}(v)$ consists of the set of vertices $\{v_{n_1}, v_{n_2}, v_{n_3}\}$ as these vertices connect to v.

In an undirected graph $|\mathcal{N}(v)| = deg(v)$. In a directed graph $\mathcal{N}(v)$ consists of *out-neighbours* $\mathcal{N}^{out}(v)$ and *in-neighbours* $\mathcal{N}^{in}(v)$. $\mathcal{N}^{out}(v)$ is the set of neighbours to which v has an outgoing arc. $\mathcal{N}^{in}(v)$ is the set of neighbours having an arc incoming to v.

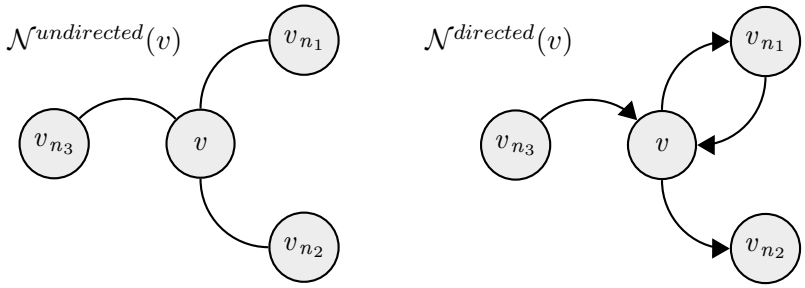


Figure 2.14: Neighbours of a vertex v.

As Figure 2.14 illustrates:

$\mathcal{N}^{undirected}(v)$ consists of vertices: $\{v_{n_1}, v_{n_2}, v_{n_3}\}$.

$\mathcal{N}^{directed}(v)$ consists of: $\mathcal{N}^{out}(v) = \{v_{n_1}, v_{n_2}\}$ and $\mathcal{N}^{in}(v) = \{v_{n_1}, v_{n_3}\}$.

Ego Network: The *ego network* of a vertex v consists of vertices v, $\mathcal{N}(v)$ and all edges between these vertices. The ego network of a vertex $v \in G$ is thus a subgraph of G centred on v.

2.4.1 Vertex Cohesion

Graph-based approaches to word sense induction require a measure of the cohesiveness of a set of words, with the value returned by the measure indicating the degree of semantic relatedness between words (Widdows, 2004; Navigli, 2012).

Given a target word tw and $\mathcal{N}(tw)$, the dyadic relationships in $\mathcal{N}(tw)$ between pairs of vertices are of limited use. For example, the dyadic relationships between a target word such as *orange* and its adjacent vertices {*red*, *pear*, *pink*, *apple*} do not delineate the senses of *orange*. However, inter-connectivity between ≥ 3 vertices in $\mathcal{N}(tw)$ can be used as a measure of cohesion, with ≥ 3-cliques in $\mathcal{N}(tw)$ taken as indicators of the target word's senses.

The primary units of cohesion used in this thesis are triangular in form. Given a vertex i and its neighbourhood $\mathcal{N}(i) = \{j, k\}$, the undirected edges $e_{\{i,j\}}$ and $e_{\{i,k\}}$ form a *triplet*, as shown in Figure 2.15. Adding the undirected edge $e_{\{j,k\}}$ closes the triplet, forming a *triangle* (a 3-clique). The edge $e_{\{j,k\}}$ in Figure 2.15 is said to be a *transitive* edge in $\mathcal{N}(i)$ as it links all three vertices in $\mathcal{N}(i)$, permitting a path from i to k via j and a path from i to j via k. This edge reinforces the relationships between i, j, and k. In terms of language, transitive edges reinforce semantic cohesion between triplets of words.

Figure 2.15: A triplet and a triangle (a 3-clique).

If $\mathcal{N}(i)$ is a directed space, the primary units of cohesion must be *non-vacuous* triplets (Wasserman and Faust, 1994). A *vacuous* triplet, centred on a vertex i, is where i acts solely as the source or sink for its neighbours. For example, Figure 2.16 shows that vertex i is a source in (a) and (b), as no neighbour connects to it, and a sink in (c) and (d), as i connects to neither of its neighbours. In all four instances, vertices i, j, and k cannot form a directed 3-clique .

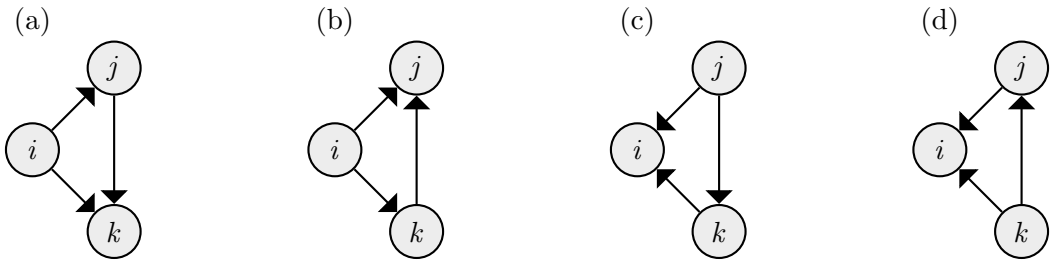


Figure 2.16: Vacuous triplets.

Figure 2.17 shows examples of non-vacuous triplets, where the vertices i, j, and k are

able to form directed 3-cliques. Relating this to language, non-vacuous triplets of vertices that represent words indicate semantic cohesion between word triplets.

Figure 2.17: Non-vacuous triplets.

Two other forms have been applied in graph-based WSI, namely *squares*, in Navigli and Crisafulli (2010), and *diamonds*, in Di Marco and Navigli (2013). These forms are illustrated in Figure 2.18.

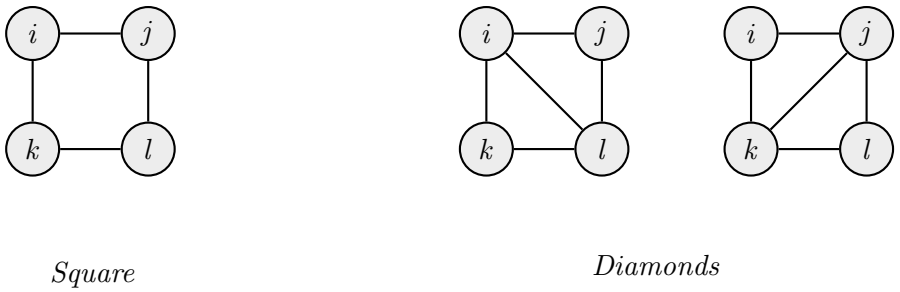


Figure 2.18: Square and diamond forms.

Squares and diamonds are used within the framework of a *Web Search Result Clustering and Diversification* task (Navigli and Vannella, 2013). This framework requires "a looser notion of sense to cope with ambiguous queries" (Di Marco and Navigli, 2013, p.40), thus contrasts with the aim in this thesis of identifying well-defined word senses. Di Marco and Navigli (2013) shows that squares and diamonds are better than triangles at identifying such 'looser' word senses. This finding suggests that triangles might be the best choice for identifying well-defined word senses, a choice that can be further justified by the following two observations. Firstly, in graph-based WSI local vertex connectivity represents meaning consistency. Edges in triangles, squares, and diamonds are therefore assumed to connect vertices (words) that define the same word sense. Note that triangles, unlike squares or diamonds, are complete subgraphs, therefore, arguably, represent regions in word co-occurrence graphs with stronger semantic cohesion and greater semantic stability. Conversely, as squares and diamonds are not fully connected subgraphs they arguably represent regions with weaker semantic cohesion/stability, thus represent looser definitions of word senses. For example, if the vertex pairs i, j and i, k in the triangle of Figure 2.15 represent pairs of semantically related words then the words represented by the vertex pair j, k are only likely to be semantically related if they are related to the same sense of i. However, as Figure 2.18 shows, this cannot be the case for all vertex triplets in squares and

diamonds as these forms are not fully connected subgraphs. Secondly, Widdows (2004) and Dorow (2007) present empirical evidence of a correlation between word senses and a measure that uses triangles as units of word cohesion, something that has not been shown for squares or diamonds.

2.4.2 $n^{\text{th.}}$ Order Neighbours

Given a vertex v, $\mathcal{N}(v)$ is said to contain the *first order* neighbours of v: vertices immediately adjacent to v. Expanding $\mathcal{N}(v)$ to include the neighbours of the first order neighbours of v returns a set of vertices called *second order* neighbours (Dorow, 2007). Figure 2.19 shows the first and second order neighbours of vertex *sienna* where, for example, *orange* is a first order neighbour of *sienna* and *lemon* is a second order neighbour of *sienna*.

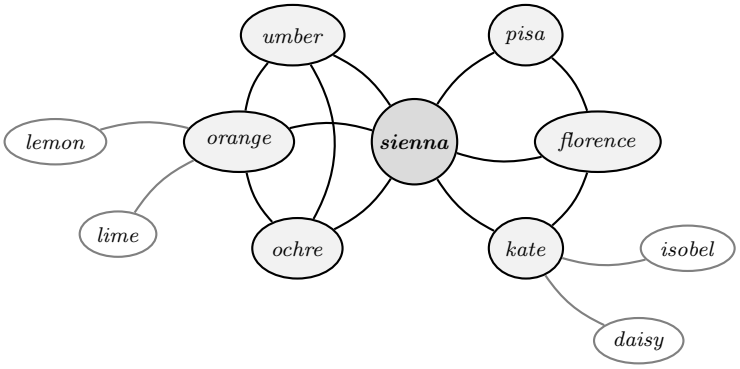


Figure 2.19: *sienna*'s first and second order neighbours.

Relating this to the problem of finding word senses, the expansion of the space around a vertex v from $\mathcal{N}(v)$ to second, or greater, order neighbours can venture out of the space that defines v's senses. However, in Natural Language Processing (NLP) there is often a paucity of data that can be used to disambiguate a word's meanings. Thus, one possibility is to expand $\mathcal{N}(v)$, venturing further out to some $n^{th.}$ order set of vertices around a target word.

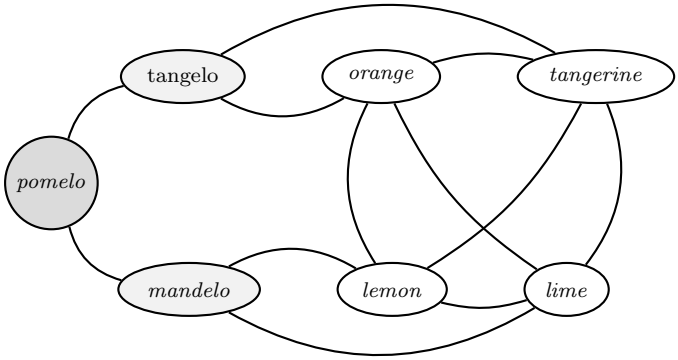


Figure 2.20: *pomelo*'s first and second order neighbours.

For example, in Figure 2.20 the target word to be disambiguated is *pomelo*[1]. The first order neighbours of *pomelo*, $\mathcal{N}(pomelo) = \{tangelo, mandelo\}$, are also assumed for the purposes of this example to be ambiguous, therefore do not assist in defining a sense of *pomelo*. However, expansion of $\mathcal{N}(pomelo)$ to second order neighbours finds the clique {*tangerine*, *orange*, *lemon*, *lime*} containing what might be considered to be prototypical examples of citrus fruits; thus, an inference might be made here that *pomelo* is also a citrus fruit.

2.5 Additional Graph Properties

A *graph property* is an invariant property of the abstract structure of a graph not of any specific representation. For example, the *diameter* of a graph (the longest shortest path in a graph) is a graph property as its value is the same however the graph is drawn (Diestel, 2006). As Figure 2.21 shows, the diameter in two representations (a) and (b) of the same graph is identical: diameter(a) = diameter(b) = 3.

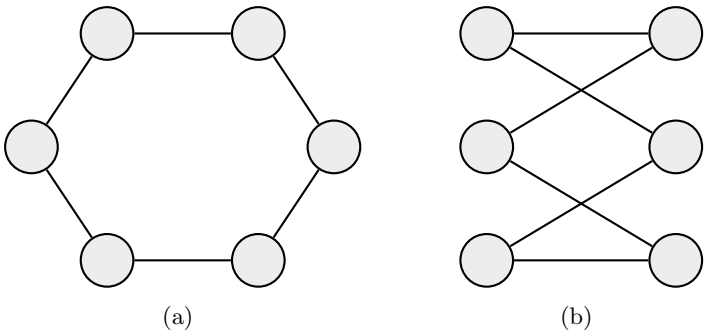


Figure 2.21: Two representations of the same graph.

Similarly, graph measures such as the clustering coefficient are graph properties as they return the same values for different representations of the same graph.

This chapter defines the graph properties that are applied in this thesis. However, there are other graph properties that could be applied in a graph-based approach to Natural Language Processing (NLP). This section briefly defines a number of additional graph properties that have been applied to NLP tasks (Navigli and Lapata, 2007; Sinha and Mihalcea, 2007; Korkontzelos et al., 2009), noting why these properties are not best suited to WSI, thus not explored in this thesis.

2.5.1 Graph Centrality

Graph centrality is a property of individual vertices in a graph. A measure of graph centrality assesses the importance of a particular vertex in a graph. Four measures are defined here: degree centrality, closeness centrality, betweenness centrality (Freeman, 1979), and PageRank (Brin and Page, 1998). The measures are defined for a vertex v in an unweighted, undirected graph $G = (V, E)$.

[1] A large yellow citrus fruit similar to a grapefruit.

- **Degree Centrality** is the number of edges incident to v, defined as:

$$DC(v) = deg(v) = |\{\{u, v\} \in E : u \in V\}|. \tag{2.1}$$

- **Closeness Centrality** measures how close, on average, v is to other vertices in G, defined as:

$$CC(v) = \frac{1}{\sum_{u \in V} s(v, u)}, \tag{2.2}$$

  where $s(v, u)$ is the shortest path between v and u.

- **Betweenness Centrality** is the number of shortest paths σ in G that pass through v divided by the number of shortest paths in G, defined as:

$$BC(v) = \sum_{i,j \in V} \frac{\sigma_{i,j}(v)}{\sigma_{i,j}}. \tag{2.3}$$

- **PageRank** measures the importance of a vertex v using a recursive, convergence process. Vertices connected to a vertex v are said to cast a vote for v (Sinha and Mihalcea, 2011). The higher the number of votes cast for v, the higher the importance of v. Furthermore, the importance of each vertex casting a vote determines the value of the vote. The PageRank score of a vertex v is defined in Navigli and Lapata (2010) as:

$$PR(v) = \frac{(1 - \alpha)}{|V|} + \alpha \sum_{\{u,v\} \in E} \frac{PR(u)}{deg(u)}, \tag{2.4}$$

  where α is a damping factor set to 0.85. Each vertex in G is initialised to an arbitrary PR value. PR values are then repeatedly computed using Equation (2.4) - a recursive process that terminates when PR values cease to change. Note that in practice PageRank is often set to terminate after a predefined number of iterations, or to iterate until convergence below a given threshold is achieved (Mihalcea et al., 2004b).

The graph centrality measures defined above use the graph properties of vertex degree or geodesic distance. These properties alone cannot measure the cliquishness of a vertex, therefore cannot find clusters of highly interconnected vertices: clusters that may best demarcate word senses. The vertex measure applied in this thesis (the clustering coefficient) returns a value that reflects the cliquishness that exists between a vertex v and its adjacent vertices $\mathcal{N}(v)$. A measure of cliquishness is better suited to WSI as it allows one to find highly cohesive clusters of words (vertices), clusters that can then be used as representations of word senses.

2.5.2 Global Measures

Global measures consider the structure of a graph as a whole. Two global measures are defined here: graph entropy and edge density (Navigli and Lapata, 2007). These measures

use the graph properties of *order* (the number of vertices in a graph) and *size* (the number of edges in a graph).

- **Graph Entropy**: In information theory, entropy is a measure of the uncertainty (alternatively, the information content) of a random variable (Shannon, 1948). In graph-theoretic terms, high entropy in a graph indicates that many vertices are equally important whereas low entropy indicates that only a small number of vertices are important. Graph entropy for a graph $G = (V, E)$ is defined in Navigli and Lapata (2010) as:

$$H(G) = \frac{-\sum_{v \in V} p(v) \log (p(v))}{\log (|V|)}, \tag{2.5}$$

  where $p(v) = \frac{deg(v)}{2|E|}$ and $log(|V|)$ is the maximum entropy of G. $H(G)$ returns a value in the range $[0, 1]$. A value of 0 indicates that G is a completely disconnected graph (no vertex connects to any other vertex); a value of 1 indicates that G is a fully connected graph.

- **Edge Density** is the number of actual edges in G divided by the number of possible edges in G, defined as:

$$ED(G) = \frac{2|E|}{|V|(|V| - 1)} \tag{2.6}$$

  $ED(G) = 1$ if G is a fully connected graph; $ED(G) = 0$ if G is totally disconnected. A high score, close to 1, indicates that G is a *dense graph*; a low score, close to 0, indicates that G has only a few edges, thus is said to be a *sparse graph.*

As noted in Section 2.5.1, a measure of cliquishness is best suited to graph-based WSI. However, the global measures in (2.5) and (2.6) do not distinguish between connectivity and cliquishness.

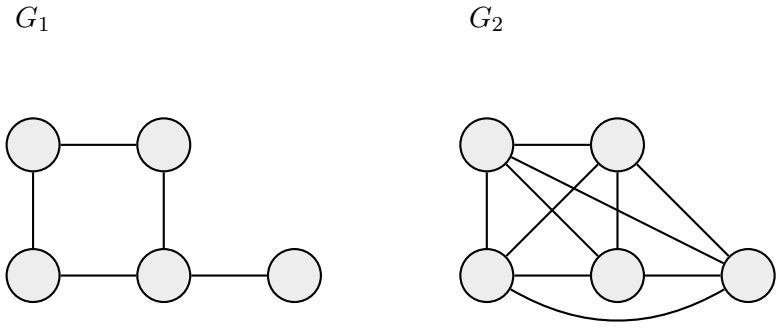


Figure 2.22: Two unweighted, undirected graphs: G_1 (a connected graph) and G_2 (a complete graph).

For example, Figure 2.22 shows a graph G_2 that is a fully connected. G_2, therefore, is a maximal clique of itself and $H(G_2) = 1.0$. However, G_1 in Figure 2.22 (a connected graph, but one that has no vertex with a fully connected neighbourhood) returns $H(G_1) = 0.97$, a value that is close to the maximal value that graph entropy can return. Edge density (ED) returns values of 1 for G_2 and 0.5 for G_1, indicating that G_2 is a dense graph

and that G_2 is middling between being a dense and a sparse graph. The problem with edge density is that the distinction between a dense and a sparse graph is not clearly defined in graph theory (Nešetřil and de Mendez, 2012), therefore values returned by edge density must be defined in relation to the task the measure is applied to. As with the graph centrality measures defined in Section 2.5.1, graph entropy and edge density do not measure cliquishness in graphs, thus are not suited to graph-based WSI. The points raised in this section are further discussed in Chapter 3.

2.6 Summary

This chapter defined the graph theoretical notation and terminology that is used throughout this thesis. The notion of a graph in its undirected, directed, and weighted forms was introduced. Connectivity in graphs was discussed, notably with regard to how particular patterns of vertex connectivity can be used to identify semantic relatedness in word co-occurrence graphs; patterns of connectivity that may be applied in a graph-based approach to WSI. A number of graph properties were also reviewed; properties that do not measure cliquishness in graphs, thus are not well-suited to WSI.

Chapter 3

Measures of Vertex Association

3.1 Introduction

Given a set of entities and some predefined relationship between entities, an association measure is used to signify the strength of relationships between entities. For example, if entities are vertices representing words and the relationship is word co-occurrence, values returned by an association measure signify the strength of word co-occurrence relationships. These values can then be used as edge weights in a word co-occurrence graph.

As edge weights affect how clustering algorithms partition graphs, selecting the most appropriate association measure to apply is of key importance. This chapter's aim is to select those association measures, out of the many proposed, that are best suited to a graph-based approach to Word Sense Induction (WSI). The chapter begins by reviewing a number of measures that have been previously applied in Natural Language Processing (NLP), finding them unsuited to the task of WSI. Section 3.2 defines three vertex association measures selected for use in the graph models proposed in Chapters 8, 9, and 10. This section also discusses why particular measures that are often applied in NLP are not chosen. Section 3.3 introduces the clustering coefficient (Watts and Strogatz, 1998), a measure of vertex cohesion in unweighted, undirected graphs. This section proposes a straightforward generalisation of this measure to the directed case.

3.1.1 Vertex Measures and Graph Measures

Numerous measures exist for quantifying connectivity in graphs. Many, though, are unsuited to finding cliquishness in graphs – the type of connectivity that may best demarcate word senses. Measures range from the relatively straightforward: local (vertex) measures such as *centrality degree*, *betweenness*, *closeness* (Freeman, 1979; Wasserman and Faust, 1994) and *eigenvector centrality* (Brin and Page, 1998), applied in Navigli and Lapata (2010, 2007); Mihalcea and Radev (2011), and global (graph) measures such as graph *density*, *entropy*, and *compactness*, applied in Navigli and Lapata (2010), to comparatively complex measures of 'community structure' such as the *Newman-Girvan Betweenness* measure (Girvan and Newman, 2002) and *Modularity* (Newman, 2006), applied in Chen et al. (2008); Bernhard (2010), and Pivovarov and Trunov (2011). However, none of these

measures are designed to find cliques in graphs.

Local measures assess the importance of a vertex v relative to its neighbouring vertices $\mathcal{N}(v)$, though only partially account for the interconnectivity between neighbours (cf. 2.5.1). Global measures consider the structure of a graph G as a whole, or are delimited to consider the structure of subgraphs $G(v) \subset_S G$ of a particular vertex $v \in G$. However, the distinction between a dense and a sparse graph in the *graph density* measure is not clearly defined in graph theory, thus must be defined relative to evaluation results returned for a specific task, and the measures of *graph entropy* and *graph compactness* can return values for connected graphs that are close to those returned for complete graphs, thus do not clearly define cliques. All of these measures, both local and global, indicate the degree of vertex connectivity in a graph/subgraph; however, they do not factor in the cliquishness between vertices, and this is exactly the type of vertex connectivity that may be of most use in graph-based WSI.

Considering the more complex measures: given a graph G, the *Newman-Girvan Betweenness* and *Modularity* measures detect 'community structure' in G by partitioning it to subgraphs. Each subgraph $G' \subset_S G$ is said to be a 'community' (*Newman-Girvan Betweenness*) or 'module' (*Modularity*) of G, each having dense intra connectivity and sparse inter connectivity to other subgraphs. However, both measures are computationally expensive to compute. *Modularity* requires an optimisation method to partition G, using for example, spectral optimization, simulated annealing, or a greedy search algorithm to find the optimal partition (out of all possible partitions) of G. *Newman-Girvan Betweenness* requires all shortest paths in G to be computed in each step of an iterative process, where shortest paths must be recalculated in each iteration. Both measures have also been criticised for failing to detect all possible communities/modules in a graph, with Kumpula et al. (2007) and Fortunato and Barthélemy (2007) showing (for *Modularity*) that the resolution limit of a graph decreases as its size increases. In simple terms: as the graph grows larger connectivity becomes 'blurred', resulting in the merging of small, highly cliquish modules. Fortunato and Barthélemy (2007) state:

> "We find that modularity optimization may fail to identify modules smaller than a scale which depends on the total size of the network and on the degree of interconnectedness of the modules, even in cases where modules are unambiguously defined." (Fortunato and Barthélemy, 2007, p.36).

Indeed, *Modularity* was applied in preliminary research for this thesis where it was found to merge senses to a greater extent than the comparatively simple methods that are applied in Chapters 8, 9, and 10. Kumpula et al. (2007) also find that both *Modularity* and *Newman-Girvan Betweenness* are prone to merge clusters, stating:

> "It is clear that the problem of the resolution limit is not restricted to the Newman-Girvan method of modularity optimization. Rather, it is a flaw which seems to be present in any community detection scheme based on global optimization of intra- and extra-community links." (Kumpula et al., 2007, p.5).

Of particular note are the conclusions drawn in these papers: Fortunato and Barthélemy

(2007, p.41) concludes that "a new theoretical framework that focuses on a local definition of community" is required; Kumpula et al. (2007, p.5) concludes that "small communities should be considered on a more local level". Local measures of this type are discussed in Section 3.3 and Chapter 4

3.2 Association Measures

An association measure quantifies the dependence between entities in some problem space. Relating this to WSI: entities are words; the problem space is a set of (initially) ambiguous contexts, and dependence between words is based on word co-occurrence.

Given a target word tw and a set of contexts S_{tw} (for example, sentences in which the target word occurs), associations between context word pairs $\{x, y\} \in S_{tw}$ are quantified by an association measure AM, with the values returned by AM used as edge weights in a word co-occurrence graph G_{tw}. Thus, if $AM(x, y) = 0.5$, the weight on the edge connecting the vertex pair x and y in G_{tw} is set to 0.5. A clustering algorithm is then applied to partition G_{tw}, with the words in each cluster taken to represent a sense of tw. Ideally, therefore, the words in an $\{x, y\}$ pair that return the highest association score for all $\{x, *\}, \{*, y\}$ pairs considered should be assigned to the same cluster.

Three association measures are applied in this thesis: Frequency, Conditional Probability, and the Log Likelihood Ratio. These measures are defined in Sections 3.2.1 - 3.2.3, where x and y in $\{x, y\}$ pairs represent any two context words, $x \neq y$, found in contexts S.

3.2.1 Frequency

Frequency is the number of times x and y co-occur in contexts:

$$Frequency(x, y) = \sum_{s \in S} \{x, y\} \in s. \tag{3.1}$$

High frequency of $\{x, y\}$ is taken to indicate a strong association between x and y. This is a naive measure of association. For example, given a set of contexts for the target word *orange*, the pair $\{orange, and\}$ is likely to have higher frequency than $\{orange, lemon\}$, yet the second pair has more utility for word sense induction. Frequency is used in this thesis as a baseline measure of association.

3.2.2 Conditional Probability

Conditional probability:

$$p(x|y) = \frac{p(x \cap y)}{p(y)} = \frac{Frequency(x, y)}{Frequency(y)} \tag{3.2}$$

is used as edge weight in directed graphs. As Figure 3.1 shows, conditional probability allows different weights to be placed on directed edges between two vertices x, y.

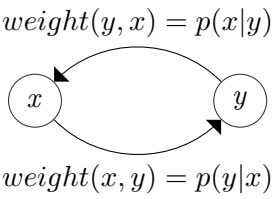


Figure 3.1: Conditional probability as edge weight in a directed graph.

Use of conditional probability as edge weight thus allows non-symmetric relationships between word pairs to be expressed. For example, in Figure 3.2 the association strength between the two vertices *orange* and *lemon* is imbalanced: *lemon* has a stronger connection to *orange* than *orange* has to *lemon*.

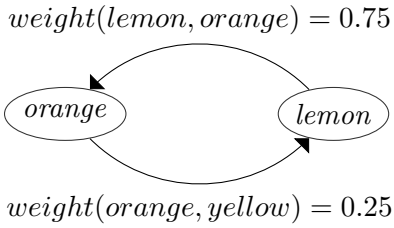


Figure 3.2: An example of the use of conditional probability as edge weight in a directed graph.

Relating this (toy) example to WSI, the edge weights in Figure 3.2 suggest that *lemon*, as a context word of the target word *orange*, is more indicative of *orange* in its FRUIT sense than *orange*, as a context word of the target word *lemon*, is for *lemon* in its FRUIT sense.

3.2.3 The Log Likelihood Ratio

The Log Likelihood Ratio (LLR) (Dunning, 1993) is the association measure that is most commonly applied in this thesis. Unlike a number of measures that have been applied in Natural Language Processing (discussed in Section 3.2.4), LLR is "a meaningful, sound and robust association measure" (Evert, 2005, p.21) that has "become a de facto standard in the field of computational linguistics for the purpose of measuring the statistical association between words"(Evert, 2005, p.137).

The Log Likelihood Ratio applies a statistical hypothesis test to establish if the co-occurrence of two entities x and y is greater than chance (entities here being words that co-occur in contexts). Two hypotheses are proposed:

- H_0, the null hypothesis; independence: $\{x, y\}$ is a random pair with probability $p(x, y) = p(x) \times p(y)$.
- H_1, the alternative hypothesis; dependence: $\{x, y\}$ co-occur greater than chance.

Given a predefined significance level, LLR returns a value indicating how much more likely one hypothesis is than the other. LLR measures a quantity: the significance of evidence obtained from the dataset (contexts) against the null hypothesis H_0. H_0 is rejected if the

probability that H_0 is true is less than some predefined significance level; otherwise, H_0 is accepted.

In Table 3.1, the lower the significance level, the higher the certainty of H_1. However, it is perhaps more intuitive to read that higher LLR values indicate stronger association. Thus, for example, if $LLR(x, y) \geq 3.84$, the null hypothesis H_0 that $\{x, y\}$ is a random pairing of words is rejected at a significance level of 0.05; alternatively stated: there is 95% 'certainty' that H_1 is true.

LLR value	Significance
3.84	0.05
6.63	0.01
7.88	0.005
10.83	0.001

Table 3.1: Examples of Log Likelihood Ratio (LLR) values and their respective significance.

3.2.3.1 Log Likelihood Ratio Definition

The definitions of the Log Likelihood Ratio given in the original paper (Dunning, 1993) and in Manning and Schütze (1999) and Bordag (2008) are somewhat cryptic. Evert and Krenn (2004) define the measure in a more transparent form, using contingency tables to compute LLR scores for $\{x, y\}$ pairs, as shown in Tables 3.2 and 3.3:

	y	$\neg y$	
x	O_{11}	O_{12}	R_1
$\neg x$	O_{21}	O_{22}	R_2
	C_1	C_2	N

Table 3.2: Observed frequencies.

	y	$\neg y$
x	$E_{11} = \frac{R_1 C_1}{N}$	$E_{12} = \frac{R_1 C_2}{N}$
$\neg x$	$E_{21} = \frac{R_2 C_1}{N}$	$E_{22} = \frac{R_2 C_2}{N}$

Table 3.3: Expected frequencies.

O_{11} is the joint frequency of x and y observed in the data; E_{11} is the expected frequency of x and y. Observed frequencies are used to calculate expected frequencies, under the null hypothesis that x and y are statistically independent. N is the number of all pairs in the dataset, and C_*, R_* are marginals: $C_1 = O_{11} + O_{21}$, similarly for C_2; $R_1 = O_{11} + O_{12}$, similarly for R_2.

Using the values in Tables 3.2 and 3.3, LLR can be computed as follows:

$$LLR_{\text{dunning}} = -2 \log \frac{L(O_{11}, C_1, r) \times L(O_{12}, C_2, r)}{L(O_{11}, C_1, r_1) \times L(O_{12}, C_2, r_2)}, \tag{3.3}$$

where:

$L(k, n, r) = r^k(1-r)^{n-k}$

$r = \frac{R_1}{N}, r_1 = \frac{O_{11}}{C_1}, r_2 = \frac{O_{12}}{C_2}$.

However, a far more straightforward definition is given in Evert and Krenn (2004):

$$LLR = 2\sum_{ij} O_{ij} \log \frac{O_{ij}}{E_{ij}}, \tag{3.4}$$

where ij index the values in Tables 3.2 and 3.3.

3.2.4 Alternative Association Measures

Two measures that are often applied in NLP are Pointwise Mutual Information (PMI) (Fano and Hawkins, 1961) and Chi-Squared ($\mathcal{X}^2$) (Pearson, 1900). In terms of Tables 3.2 and 3.3, PMI can be defined as:

$$PMI(x, y) = \log_2 \frac{O_{11}}{E_{11}} \tag{3.5}$$

and $\mathcal{X}^2$ as:

$$\mathcal{X}^2(x, y) = \frac{N(O_{11} - E_{11})^2}{E_{11}\ E_{22}}. \tag{3.6}$$

However, there are issues with these measures that make them unsuited to datasets in which the frequency of $\{x, y\}, x$, or y is low (as is often the case in the contexts used in this thesis):

- **PMI** overestimates association between low frequency pairs (Manning and Schütze, 1999; Weeds, 2003). Considering completely dependant pairs, for expository purposes, the rarer the pair the better the PMI score. For example, given two pairs:

  1. $\{x, y\}$, where $count(x, y) = count(x) = count(y) = 1$,

  2. $\{x', y'\}$, where $count(x', y') = count(x') = count(y') = 100$,

  $\{x, y\}$ will return a higher PMI score: $PMI(x, y) = 13.29$, $PMI(x', y') = 6.64$ for $N = 10,000$.

- $\mathcal{X}^2$: Dunning (1993) shows, for highly skewed contingency tables, where the observed frequency of $\{x, y\}$ is small and N is large, that LLR is a more accurate measure of association than $\mathcal{X}^2$.

A further measure, Fisher's exact test (Fisher, 1922), was applied in preliminary research for this thesis. In terms of Tables 3.2 and 3.3, this measure can be computed as follows:

$$Fisher(x, y) = \sum_{k=O_{11}}^{min\{R_1, C_1\}} \frac{\binom{C_1}{k} \times \binom{C_2}{R_1 - k}}{\binom{N}{R_1}}. \tag{3.7}$$

This measure gives a precise assessment of the association between x and y. However, the numerical complexity involved in its calculation makes it impractical to apply to large

sets of word co-occurrence counts. Moreover, LLR is shown in Evert (2005) to give an excellent approximation of the values returned by Fisher's exact test.

‘

3.3 The Clustering Coefficient

The clustering coefficient (Watts and Strogatz, 1998)[1] is a measure of cliquishness in undirected, unweighted graphs. Watts and Strogatz apply the measure in an analysis of connectivity in graphs, finding that many real world graphs (networks) are examples of the small world phenomenon (Karinthy, 1929; Milgram, 1967). A graph is said to be small world if: 1.) the average clustering coefficient (cliquishness) of vertices is significantly higher than that of a random graph constructed from the same vertex set; 2.) has an average shortest path length close to that of a random graph (Erdős and Rényi, 1959).

The measure has been applied in Bioinformatics (Lubovac et al., 2006; Kalna and Higham, 2007; Stingl et al., 2010), Computational Chemistry (Mazurie et al., 2010), Neuroscience (van den Heuvel et al., 2010; Wang et al., 2010; Hänggi et al., 2011; Douw et al., 2011), and Financial Analysis (Schiavo et al., 2010; Minoiu and Reyes, 2011; Squartini et al., 2011). In NLP, the measure has been applied to WSI (Widdows, 2004; Dorow, 2007; Navigli and Crisafulli, 2010; Anand et al., 2011) and in Ferrer-i-Cancho and Solé (2001) to the study of word co-occurrence.

The measure can be applied at either a global (entire graph) or local (single vertex) level. This section looks at the local clustering coefficient, a measure of vertex connectivity within the neighbourhood of a single vertex. This section also introduces a straightforward generalisation of the local clustering coefficient that is applicable to directed graphs.

3.3.1 The Local Clustering Coefficient

The local clustering coefficient is a measure of cliquishness within the neighbourhood of a single vertex. The measure is often described in terms of a social network (Wasserman and Faust, 1994; Newman et al., 2006) by stating that the probability of any two people knowing each other is increased if the two individuals have a friend in common. Thus, if person a has two friends c and b, the probability that c and b are also friends is greater than chance (Newman et al., 2006, p.287). Furthermore, if a, b, and c are friends of each other they are said to form a clique; in social network theory terms, a, b, and c are said to form a socially cohesive group. The measure also has a geometric interpretation, being the curvature of a graph at a given vertex (Eckmann and Moses, 2002; Dorow, 2007), and may also be viewed in terms of how embedded a vertex is within its local neighbourhood.

Given a graph $G = (V, E)$, the local clustering coefficient for a vertex $v \in V$ is defined as follows:

$$C(v) = \frac{2\,|e_{\{i,j\}}|}{k_v(k_v - 1)} : i, j \in \mathcal{N}(v), \tag{3.8}$$

where k_v is the degree of v and $e_{\{i,j\}}$ is an undirected edge between two neighbours i, j of v.

[1] A matrix formulation is given in Luce and Perry (1949).

The local clustering coefficient is thus the number of undirected edges between neighbours $\mathcal{N}(v)$ of v divided by the possible number of undirected edges between neighbours of v. $C(v)$ rises as connectivity in $\mathcal{N}(v)$ increases. As Figure 3.3 shows, if none of the neighbours of v are connected $C(v) = 0$; if all of the neighbours of v are connected $C(v) = 1$.

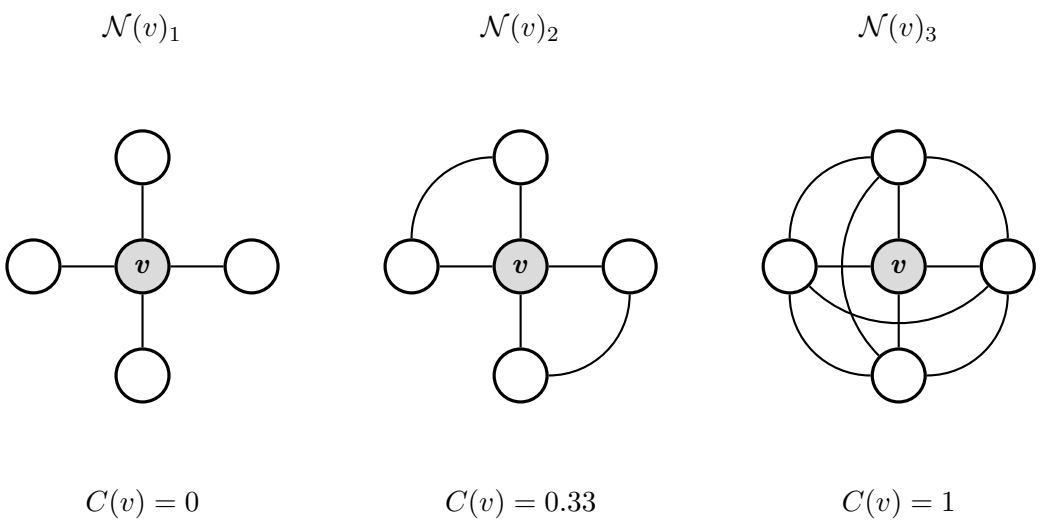


Figure 3.3: The local clustering coefficient.

Relating this to WSI, if connectivity in $\mathcal{N}(v)$ is based on word co-occurrence, a high clustering coefficient score for a vertex (word) v indicates high connectivity between words co-occurring with v, thus, according to the distributional hypothesis (Harris, 1954), implies strong semantic relatedness: words in $\mathcal{N}(v)$ may therefore be good candidates for defining a sense of v.

3.3.2 A Generalisation of the Clustering Coefficient to the Directed Case

A straightforward generalisation of the local clustering coefficient to the directed case is proposed in this thesis, defined as follows:

$$C_{\mathsf{directed}}(v) = \frac{|e_{(i,j)}|}{k_v(k_v - 1)} : i, j \in \mathcal{N}(v), \tag{3.9}$$

where $e_{(i,j)}$ is a directed edge between two neighbours i, j of v.

Note that the numerator in the undirected local clustering coefficient (3.8) doubles the count of unordered edges between neighbours of v in order to balance with the denominator. This allows the measure to return scores in the range $[0, .., 1]$, reflecting actual vertex connectivity in $\mathcal{N}(v)$ as a fraction of possible connectivity in $\mathcal{N}(v)$. The numerator in (3.9) is the number of directed edges between neighbours of v. Dropping the doubling of the edge count therefore gives a straightforward generalisation of the local clustering coefficient to the directed case. Figure 3.4 shows decreasing values of $C_{\mathsf{directed}}(v)$ as directed edges are removed from $\mathcal{N}(v)$.

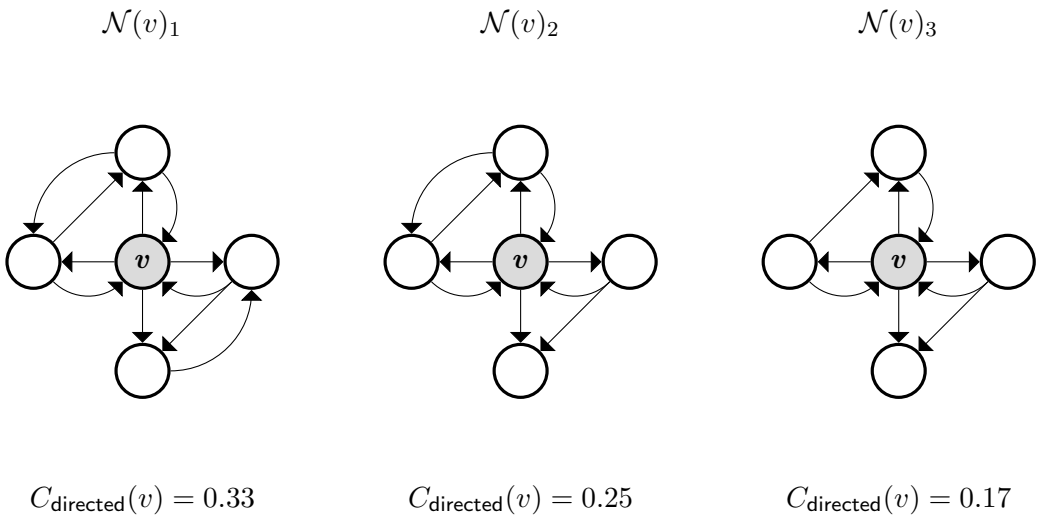


Figure 3.4: The local clustering coefficient for directed graphs.

The directed clustering coefficient in (3.9) is clearly derived from the local clustering coefficient (3.8). Like (3.8), (3.9) only considers graph structure (vertex connectivity), therefore takes no account of the strength of connection between vertices (edge weights). In contrast, the following chapter considers a different way of extending the local clustering coefficient, to the case of weighted graphs. Generalisations such as that proposed by Barrat et al. (2004) are thus quite distinct from that proposed here: whereas (3.9) is applicable to unweighted, directed graphs, Barrat et al.'s generalisation (along with others presented in Chapter 4) is applicable to weighted, undirected graphs.

3.4 Summary of the Chapter

This chapter's aim was to select vertex association measures, out of the many proposed, that are best-suited to graph-based WSI. The chapter showed that many of the measures presented in the literature are either unsuited to finding cliques in graphs (the type of vertex connectivity that is arguably best-suited to finding word senses in graphs), or are inappropriate for corpora with low word co-occurrence counts (as is often found in the corpora used in this thesis). These findings led to the selection of three measures of vertex association: Frequency (used as a baseline measure); Conditional Probability (for weighted, directed graphs), and the Log Likelihood Ratio. These measures are used as edge weights in the graph-based WSI systems presented in Chapters 8, 9, and 10. A measure of vertex cohesion, applicable to unweighted, undirected graphs, was also introduced, with a straightforward generalisation of this measure to the directed case proposed. This measure is further generalised to the weighted, directed case in the following chapter.

Chapter 4

Weighted Clustering Coefficients

4.1 Introduction

As illustrated in the previous chapter, the clustering coefficient only considers the structure (*topology*) of a graph, thus will return the same score for two weighted graphs of identical topology, even though the graphs may have different edge weights.

Given that many real world graphs (networks) have edge weights, it may be useful to generalise the clustering coefficient to the weighted case. Edge weights quantify some qualitative aspect between vertices, therefore incorporation of these weights into a generalisation of the unweighted clustering coefficient may capture qualities in the graph that the topological measure has no access to. As Dorow and Widdows (2003a); Dorow (2007); Navigli and Lapata (2010) show, the unweighted clustering coefficient has utility for Natural Language Processing (NLP) tasks. This thesis conjectures that a weighted generalisation of this measure may have greater utility for NLP, allowing a measure of vertex cohesion to be made in edge weighted graphs that model some aspect of language.

This chapter begins by reviewing generalisations of the unweighted clustering coefficient to the weighted case that have been proposed in the literature. This review finds that the majority of these measures do not sufficiently generalise the unweighted clustering coefficient to the weighted case. This finding leads to the introduction of three novel generalisations in Section 4.3.

4.2 A Review of Generalisations of the Local Clustering Coefficient to the Weighted Case

4.2.1 Barrat et al.

Barrat et al. (2004) proposed the first generalisation of the local clustering coefficient to the weighted case. This measure is defined for a vertex v as follows:

$$C^{Barrat}_{\mathsf{weighted}}(v) = \frac{1}{s_v(k_v - 1)} \sum_{i,j} \frac{w_{vi} + w_{vj}}{2} a_{vi} a_{vj} a_{ij}, \tag{4.1}$$

where s_v is the strength of v (the sum of the weights on incident edges of v); k_v is the

degree of v; i, j are neighbours of v, and $a_{vi} = 1$ if there is an edge between v and i; 0 otherwise.

This measure considers the amount of vertex strength associated with each possible triangle centred on v, where each triangle's contribution is weighted by the average weight of the two adjacent edges to v. The first term in the equation normalises the measure such that returned values are in the range $[0, .., 1]$. For the binary case, where edge weights are 1 or 0, the measure returns values equal to those of the unweighted clustering coefficient. However, there are three issues with this measure:

- The use of an underlying adjacency matrix for edge identity is not required for all edges in a triangle. The term $a_{vi}a_{vj}a_{ij}$ can be reduced to a_{ij} as edges vi, vj are known to exist.
- The measure looks at edge weights that do not participate in any triangle, thus are irrelevant to the computation.
- As shown in Figure 4.1, only those weights on the edges between v and vertices adjacent to v are considered.

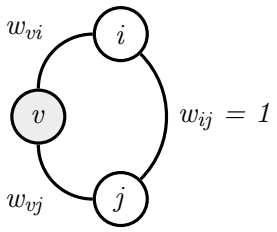


Figure 4.1: Edges between neighbours of v are assigned an identity weight of 1.

The measure, therefore, does not account for the weight on edges which complete triangles, these edges are simply assigned an identity weight of 1.

4.2.2 Lopez-Fernandez et al.

Lopez-Fernandez et al. (2004) define a weighted clustering coefficient measure for a vertex v as:

$$C_{\text{weighted}}^{Lopez-Fernandez}(v) = \sum_{i \neq j \in \mathcal{N}(v)} w_{ij} \frac{1}{k_v(k_v - 1)}, \tag{4.2}$$

where $\mathcal{N}(v)$ is the neighbourhood of v; w_{ij} is an edge weight between i, j neighbours of v, and k_v is the degree of v.

This measure is thus the sum of the edge weights between neighbours of v normalised by the unweighted clustering coefficient denominator. Lopez-Fernandez et al. interpret this generalisation as "a measurement of the local efficiency of the network around $[v]$", being "the total degree of relationship in the neighbourhood of v" (Lopez-Fernandez et al., 2004, p.3).

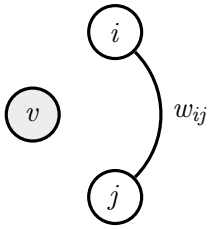


Figure 4.2: A peripheral edge of v's neighbourhood.

However, as shown in Figure 4.2, the measure only considers edge weights between neighbours of v, thus sums only the edge weights on the periphery of $\mathcal{N}(v)$.

The numerator in the unweighted clustering coefficient counts the number of i, j pairs adjacent to v that are connected, implicitly assuming the existence of edges between v, i and v, j. Note that as the measure is calculated over the space $\mathcal{N}(v)$, these edges do in fact exist. The Lopez-Fernandez et al. measure ignores the weights on v, i and v, j edges, thus, arguably, cannot be considered to be a valid generalisation of the unweighted clustering coefficient to the weighted case. Kalna and Higham, in their review of weighted clustering coefficients, state that the Lopez-Fernandez et al. measure considers "the total weight of relationship in the neighbourhood $N(v)$ of node v" (Kalna and Higham, 2006, p.2). However, if Lopez-Fernandez et al. meant to account for the total weight in $\mathcal{N}(v)$, the definition of the measure would show the summation of all weights in $\mathcal{N}(v)$ rather than just those between the i, j neighbours of v.

4.2.3 Onnela et al.

Onnela et al. (2005) propose a measure based on the notion of subgraph intensity. The intensity $I(g)$ of a subgraph g is the geometric mean of its weights, defined as:

$$I(g) = \left(\prod_{(ij) \in l_g} w_{ij} \right)^{1/|l_g|}, \tag{4.3}$$

where l_g is the set of edges in g.

Onnela et al. note that by using the geometric mean intensity may be low for subgraphs in which either all weights are low or just one weight is low. To remedy this issue, the authors introduce a measure of subgraph coherence $Q(g)$ in order to differentiate between these two possibilities. $Q(g)$ is the ratio of the geometric mean over the arithmetic mean of the weights in g, defined as:

$$Q(g) = \frac{I(g)|l_g|}{\sum_{(ij) \in l_g} w_{ij}}. \tag{4.4}$$

Thus, if the weights in g are of similar values, $Q(g)$ is close to unity. The generalisation of the clustering coefficient to the weighted case can then be defined as:

$$C^{Onnela}_{\mathsf{weighted}}(v) = \frac{2}{k_v(k_v - 1)} \sum_{i,j} (\tilde{w}_{vi} \tilde{w}_{ij} \tilde{w}_{jv})^{1/3}, \tag{4.5}$$

where the term $(\tilde{w}_{vi}\tilde{w}_{ij}\tilde{w}_{jv})^{1/3}$ is the subgraph intensity (the geometric mean) of each triangle in $\mathcal{N}(v)$, where weights are scaled (normalised) by the largest weight in the graph $\tilde{w}_{vi} = w_{vi}/max(w)$. This definition can be rewritten in terms of the relationship between the unweighted and weighted clustering coefficient by taking an average over the intensity of triangles around v,

$$\bar{I}(v) = (1/t_v) \sum_{g \in N(v)} I(g), \tag{4.6}$$

where t_v is the number of triangles that include v. The rewritten definition is then:

$$C^{Onnela}_{\mathsf{weighted}}(v) = \bar{I}_v C_v, \tag{4.7}$$

where C_v is the unweighted cluster coefficient. Note that in this definition the topological (unweighted) aspects of $\mathcal{N}(v)$ are renormalised by the average intensity of the triangles in $\mathcal{N}(v)$.

As with the Barrat et al. measure, this measure returns values in the range $[0, .., 1]$ and returns values for a binary weighted graph (a graph in which edge weights are either 0 or 1) that are equal to those of the unweighted clustering coefficient. Unlike the Barrat et al. measure, this measure considers all edges in each triangle, thus accounts for the weights on edges ij in triangles centred on v.

4.2.4 Zhang and Horvath

Zhang and Horvath (2005) define a weighted clustering coefficient measure for a vertex v as:

$$C^{Zhang}_{\mathsf{weighted}}(v) = \frac{\frac{1}{2}\sum_{i \neq v}\sum_{\{j|j \neq v, j \neq i\}} w_{vi}w_{ij}w_{jv}}{\frac{1}{2}\Big(\big(\sum_{i \neq v} w_{vi}\big)^2 - \sum_{i \neq v} w_{vi}^2\Big)}. \tag{4.8}$$

Zhang and Horvath note that the number of triangles t_v in $\mathcal{N}(v)$ can be written in terms of an adjacency matrix: $t_v = \frac{1}{2}\sum_{ij} a_{vi}a_{ij}a_{jv}$. The numerator in (4.8) is a straightforward weighted generalisation of this. The denominator in (4.8) considers the upper bound in the numerator in order to return values in the range $[0, ..1]$.

The measure is redefined in Saramäki et al. (2007) and in Kalna and Higham (2007) where its similarity to the definition given in Grindrod (2002) is noted. Indeed, Kalna and Higham (2007) give a derivation of (4.8) which shows that Grindrod's definition is equivalent[1]. Kalna and Higham in their review of weighted generalisations conclude that there is "one very promising candidate" (Kalna and Higham, 2006, p.6): the Zhang and Horvath's generalisation. However, it could be argued that this measure is not a valid generalisation of the unweighted clustering coefficient to the weighted case as the denominator in (4.8) does not account for edges ij that make up triangles in the numerator.

4.2.5 Opsahl and Panzarasa

Opsahl and Panzarasa (2009) define a weighted generalisation of the clustering coefficient

[1] As Grindrod was not attempting to define a weighted clustering coefficient, and noting that Zhang and Horvath's measure is shown to be equivalent, this measure is not defined here.

using triplets. A triplet is three connected vertices in a graph. A triplet can be closed or open. A closed triplet has three edges connecting three vertices to form a triangle. An open triplet has two edges connecting three vertices in series. The unweighted clustering coefficient for a vertex v is defined in Opsahl and Panzarasa as:

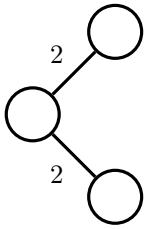
$$C^{OandP}(v) = \frac{\sum \mathcal{T}_{\triangle}}{\sum \mathcal{T}}. \tag{4.9}$$

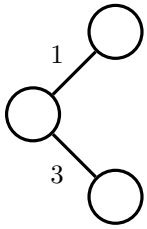
This is the number of closed triplets in $\mathcal{N}(v)$ divided by the number of closed and open triplets in $\mathcal{N}(v)$. The weighted generalisation of the unweighted measure is defined as:

$$C^{OandP}_{\mathsf{weighted}}(v) = \frac{\sum_{\mathcal{T}_{\triangle}} \mathcal{W}}{\sum_{\mathcal{T}} \mathcal{W}}. \tag{4.10}$$

This is the total *value* of closed triplets in $\mathcal{N}(v)$, measured as three separate open triplets (one for each vertex in the triangle) divided by the total value of triplets in $\mathcal{N}(v)$, where value is derived from the weights in a triplet.

The Value of a Triplet: As shown in Figure 4.3, the value of a triplet is defined in one of four ways: arithmetic mean, geometric mean, maximum edge weight, or minimum edge weight.





	Triplet 1	**Triplet 2**
Arithmetic mean	2	2
Geometric mean	2	1.73
Maximum edge weight	2	3
Minimum edge weight	2	1

Figure 4.3: Triplet values.

The geometric mean is used in this thesis as this is the value that is commonly applied in Opsahl (2009). The measure returns values in the range $[0, .., 1]$, with values equivalent to those of the unweighted clustering coefficient when applied to binary weighted graphs.

The weighted measure in (4.10) can be further generalised to the directed case, defined as:

$$C^{OandP}_{\mathsf{weighted,directed}}(v) = \frac{\sum_{nv\mathcal{T}_{\triangle}} \mathcal{W}}{\sum_{nv\mathcal{T}} \mathcal{W}}. \tag{4.11}$$

This is the total value of non-vacuous (nv) closed triplets in $\mathcal{N}(v)$ divided by the total value of non-vacuous triplets in $\mathcal{N}(v)$, where a non-vacuous triplet (as described in Chapter 2) is one in which the centre vertex of the triplet is neither a sink (all edges point at the vertex) nor a source (all edges point away from the vertex) for its neighbours. For example,

Figure 4.4 shows a 'frustrated triangle', in which c is a sink vertex and a is a source vertex. The ordered triplet (a, b, c) is non-vacuous as b is neither a sink nor a source of its neighbours whereas all other triplets in Figure 4.4 are vacuous.

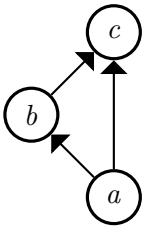


Figure 4.4: A frustrated triangle.

4.2.6 Summary

The review presented in this section finds no consensus as to which of the fives measures best generalises the unweighted clustering coefficient to the weighted case. Indeed, Onnela et al. (2005) in their review of weighted generalisations conclude that none of the measures suitably generalise the unweighted clustering coefficient to the weighted case as different generalisations capture different aspects of the graph space. The suggestion made by the authors is to apply the unweighted clustering coefficient to measure graph topology and then apply a weighted generalisation to study how edge weights affect the graph space. Kalna and Higham (2006) is more prescriptive, recommending the use of the Zhang and Horvath measure, a measure they deem to be a true generalisation of the clustering coefficient to the weighted case. However, it is arguable whether this measure is a valid generalisation as it does not account for all edge weights within the neighbourhood of the vertex measured; indeed, the only measure that does so is the Onnela et al. generalisation.

4.3 Three Novel Generalisations of the Clustering Coefficient to the Weighted Case

This section introduces three novel generalisations of the unweighted clustering coefficient to the weighted case. All three generalisations:

- Use all edge weights within the neighbourhood of the vertex measured.
- Return values in the range $[0, .., 1]$.
- Return the same values as the unweighted clustering coefficient when applied to a binary weighted graph.
- Read edge weights both ways, thus are applicable to both undirected and directed weighted graphs.

The first generalisation is straightforward. This measure is applied in the evaluation presented in Chapter 9. The two generalisations that follow are applicable to particular

types of graphs. As these types of graphs are not used in this thesis, these measures are presented here as suggestions for future research to consider.

4.3.1 Generalisation 1

A straightforward generalisation of the unweighted clustering coefficient to the weighted case is defined here as:

$$C_{\mathsf{weighted}}(v) = \frac{\sum_{i,j} w_{vi} w_{ij} w_{jv}}{k_v(k_v - 1)} : i, j \in \mathcal{N}(v). \tag{4.12}$$

The numerator in (4.12) is the sum of the products of edge weights that make up each triangle in $\mathcal{N}(v)$. The denominator is identical to that of the unweighted clustering coefficient, as defined in (3.8). This generalisation, therefore, does not attempt to fit the denominator to the weighted space of the graph, though it could be argued that this is equally true for the five generalisations reviewed in Section 4.2. Note that the use of $k_v(k_v - 1)$ in the denominator implies the possibility of a maximally connected weighted space in $\mathcal{N}(v)$ in which every edge weight equals one. A weighted graph, however, is unlikely to contain triangles in which this is the case. Furthermore, if the graph were normalised such that all edges incident to each vertex in the graph sum to 1 then these triangles could not occur as the graph would consist of just two vertices.

Note too that (4.12) is similar to Onnela et al.'s formulation (4.5). In particular, both measures compute the product of edge weights in triangles. However, (4.12) was devised without any knowledge of (4.5). I chose to use the product of edge weights in triangles rather than the (perhaps more obvious) sum of edge weights in triangles as the sum can be dominated by a single large edge weight. There are also clear differences between the two generalisations. Firstly, Onnela et al.'s measure is only applicable to undirected graphs, whereas (4.12) can be applied to both undirected and directed graphs. Secondly, (4.12) computes the sum of the products of edge weights in triangles, whereas Onnela et al.'s measure has to compute the sum of triangle intensities, i.e. the sum of the geometric means of the edge weights in triangles. However, there is no special reason for using the geometric mean. Thirdly, edge weights in Onnela et al.'s measure are scaled by the largest edge weight in the graph, a normalisation step that is not required if edge weights are in the range $[0, .., 1]$. These differences highlight the disadvantage of using Onnela et al.'s formulation instead of (4.12), i.e., that it is computationally more expensive to calculate. Additionally, note that the extra work of computing geometric means and scaling edge weights is unnecessary if, as is the case in this thesis, graph edge weights are in the range $[0, .., 1]$. Generalisation (4.12) is applied in the evaluation presented in Chapter 9.

4.3.2 Generalisations 2 and 3

The following two generalisations are applicable to particular types of graphs.

4.3.2.1 A Generalisation for Graphs Representing Probability Spaces

Probability spaces can be represented as graphs in which: vertices represent entities; edge weights represent probabilities between entities, and weights on edges incident to each vertex sum to 1 (Tijms, 2012). A weighted clustering coefficient for a graph of this type may be defined as:

$$C_{\text{weighted}}^{\text{Probability}}(v) = \frac{\sum_{i,j} w_{vi} w_{ij} w_{jv}}{\sum_{i,j} \left(\frac{1}{k_v}\right)^3} : i, j \in \mathcal{N}(v). \tag{4.13}$$

The numerator in this measure is identical to that in (4.12), this being the sum of the products of edge weights that make up triangles around v. The denominator's effect, however, is to distribute the maximum possible uniform weight (probability) over each actual and possible edge in $\mathcal{N}(v)$, as shown in Figure 4.5.

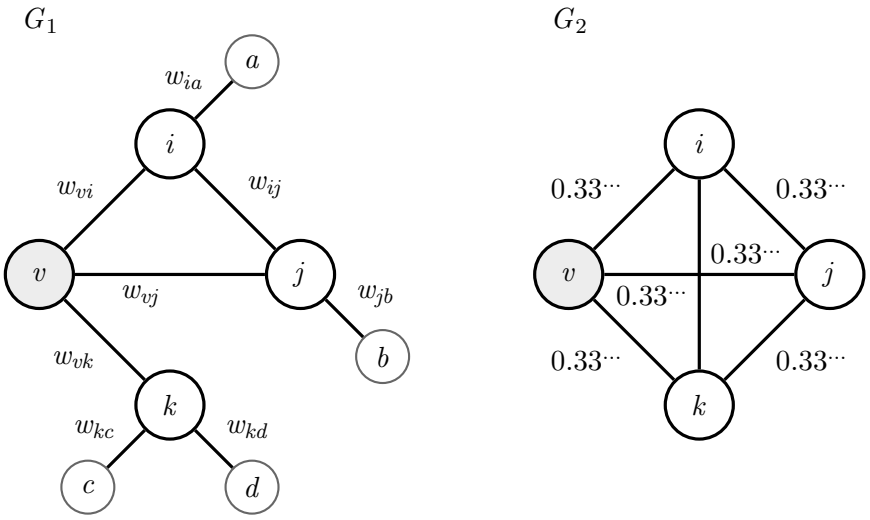


Figure 4.5: Distributing uniform weight (probability) over the possible connectivity within the neighbourhood of a vertex v.

The thinking behind this measure is as follows: given a weighted graph $\mathcal{N}(v)$ in which weights on edges between vertex pairs represent joint probabilities (or conditional probabilities if the graph is directed), what is the optimal way in which vertices could be connected? The denominator in (4.13) suggests that $\mathcal{N}(v)$ should be a fully connected graph in which every vertex connects to every other vertex, and in which the optimal distribution of edge weight is uniform probability over all possible edges in $\mathcal{N}(v)$; that is, the most equable distribution of the possible probability that could be assigned to each edge in $\mathcal{N}(v)$ if $\mathcal{N}(v)$ were a fully connected graph. Thus, if the actual probabilities between vertices (entities) in $\mathcal{N}(v)$ indicate certainty of the likelihood of these entities occurring together (that is, where none of the probabilities associated with vertices in $\mathcal{N}(v)$ fall outside of $\mathcal{N}(v)$) the generalisation will return its maximum score of 1.

Unlike generalisation (4.12), this generalisation attempts to give a true assessment of the weight ('probability') contained in $\mathcal{N}(v)$, this being the proportion of the weight that *could* be contained in $\mathcal{N}(v)$. This generalisation may therefore be of interest to

researchers who want to measure cohesion in probability spaces, allowing them to find probability subspaces centred on particular vertices of a graph in which there is a high (or low, depending on the research aim) probability of vertex (entity) association. If the denominator in (4.13) is accepted as a valid normalisation term in the measure (which is certainty debatable and may be a fruitful line of enquiry for future research to consider) this generalisation of the unweighted clustering coefficient to the weighted case reflects the likelihood of the vertex pairs in $\mathcal{N}(V)$ co-occurring.

4.3.2.2 A Generalisation for Integer Weighted Graphs

A weighted clustering coefficient for graphs in which edge weights are integer values ≥ 1 may be defined as:

$$C^{\text{Integer}}_{\mathsf{weighted}}(v) = \frac{\sum_{i,j} w_{vi} w_{ij} w_{jv}}{\sum_{i,j} \left(w_{v,max(v)} w_{i,max(i\neg(iv))} w_{j,max(j\neg(ji))} \right)} : i, j \in \mathcal{N}(v). \tag{4.14}$$

The numerator in this generalisation is identical to that of (4.12) and (4.13). The denominator, however, suggests that maximal cohesion occurs in the neighbourhood of a vertex v when each vertex in a v, i, j, v path connects to the next vertex in the path via its highest weighted incident edge. Thus, in the denominator of (4.14) $w_v, max(v)$ is the highest weight out of all weights on edges incident to v and $w_i, max(i\neg(iv))$ is the highest weight out of all weights on edges incident to i. Note that edges incident to a vertex include those in $\mathcal{N}(v)$ and those external to $\mathcal{N}(v)$. This measure is therefore checking to see if the highest edge weights of vertices in $\mathcal{N}(v)$ are actually in $\mathcal{N}(v)$. Note also that disallowing the selection of weights on edges back to the previous vertex in the path means that the same edge (weight) cannot be selected twice. This restriction allows the measure to be correctly normalised, with the measure returning a maximum score of 1 if the highest edge weights of vertices in $\mathcal{N}(v)$ are found in $\mathcal{N}(v)$. For example, in Figure 4.6 $C^{\text{Integer}}_{\mathsf{weighted}}(v) = 1$ as $\mathcal{N}(v)$ is a fully connected component in which the maximum edge weights of vertices in $\mathcal{N}(v)$ are found in $\mathcal{N}(v)$.

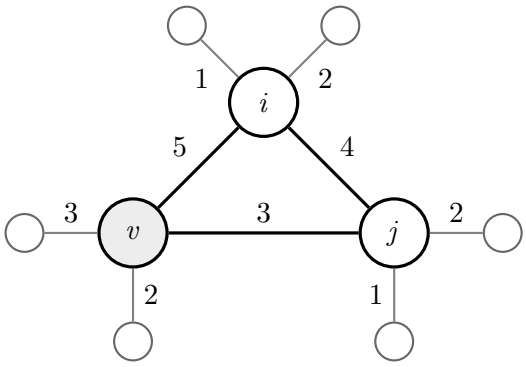


Figure 4.6: An integer weighted graph.

This generalisation may be of interest to researchers studying social networks (Wasserman and Faust, 1994) or to those studying contagion networks (Newman et al., 2006) as

it is directly applicable to the types of graph that are often applied in these studies: a weighted graph in which edge weights are counts (integers) quantifying some predefined relationship between *agents* (people). One might note that this generalisation is analogous to a greedy search algorithm (Dasgupta et al., 2006) as the locally optimal edge weight (the maximum edge weight for the current vertex in a path) is selected. Relating this to social networks, this generalisation might be used to measure the strength of association between agents who are connected to a *specific* agent (v); that is, used to find maximally cohesive social cliques that are centred on particular individuals. Relating this to contagion networks, this generalisation may be of use in finding the most likely source of an epidemic, or in finding cliques of agents who are highly susceptible to infection or those most likely to pass infection on to other agents. In a similar vein, this generalisation may also be of use in NLP where, for example, it might be applied to find sets of words that make up expressions or phrases that are centred on specific keywords.

4.4 Summary of the Chapter

This chapter began by reviewing generalisations of the unweighted clustering coefficient to the weighted case that have been proposed in the literature. This review found that there is little consensus amongst researchers as to which measure best generalises the unweighted clustering coefficient. The review also noted that the majority of these generalisations only account for a subset of the edge weights in the graph space, thus, arguably, do not sufficiently generalise the unweighted clustering coefficient to the weighted case. This observation led to the introduction of a straightforward novel generalisation, a measure that accounts for all edge weights in the graph space. Two further generalisations were also proposed, each applicable to particular types of graph.

Chapter 5

Clustering

5.1 Introduction

Clustering is the process of dividing a collection of data items into groups (clusters). Though numerous methods for clustering data items exist they can, broadly speaking, be divided into two types: those that provide some utility for other processes and those that provide an understanding of the data (Tan et al., 2006; Aggarwal and Reddy, 2013). This chapter is concerned with the second type: clustering methods that aim to divide an amorphous or ambiguous collection of data items into clusters that are intrinsically meaningful. Clustering methods of this type attempt to find natural orderings amongst the data items, using some measure of affinity between items to partition them to meaningful groups. In the clustering methods described in this thesis, data items are words and the aim is to partition contexts in which words occur into a set of clusters, each of which is taken to represent a word sense. In this scenario, each cluster, ideally, contains a set of words that have sufficient semantic relatedness to each other to identify a particular word sense. For example, the cluster {*lemon*, *lime*, *tangerine*, *grapefruit*} for the target word *orange* contains words that are semantically related to *orange* and semantically related to each other, thus could be used to represent the FRUIT sense of *orange*.

This chapter begins by defining the various types of clustering solutions that different clustering algorithms return. The following section then describes two well-known and widely applied clustering methods. The aim of this section is to show that commonly applied clustering methods, which return good results for many tasks, are not necessarily well-suited to the task of Word Sense Induction (WSI). Section 5.4 reviews clustering methods that are arguably more suited to WSI, methods that are graph-based and that return either a soft or fuzzy clustering solution. Section 5.4.3 describes *Chinese Whispers* (Biemann, 2007), a parameter-free graph-based clustering algorithm that has been shown to be of use in Natural Language Processing (NLP). Section 5.5 introduces *MaxMax* (Hope and Keller, 2013a), a novel parameter-free soft clustering algorithm that attempts to address various limitations of existing clustering algorithms.

5.2 Clustering Solution Types

This section gives a brief overview of the types of clustering solutions that different clustering algorithms return, defined here in graph-theoretic terms where G represents the input graph (the graph to be clustered) and $\mathcal{C}_G$ represents the clustering solution (the clusters returned by a clustering algorithm).

Hierarchical Clustering: Given a graph G, a hierarchical clustering algorithm returns a *tree* (a dendrogram) $\mathcal{C}_G$. The *leaves* of the tree are single vertices (singleton clusters) and each *branch* of the tree is the (possible) amalgamation of the vertices in clusters below it. Each vertex in $\mathcal{C}_G$ therefore belongs to its singleton cluster and to every branch cluster that subsumes the singleton cluster.

Partitional Clustering: Given a graph G, a partitional clustering algorithm returns a flat, non-hierarchical partitioning $\mathcal{C}_G$ of G in which each vertex belongs to one cluster. Possible hierarchical and partitional clusterings for the graph in Figure 5.1 are illustrated in Figure 5.2.

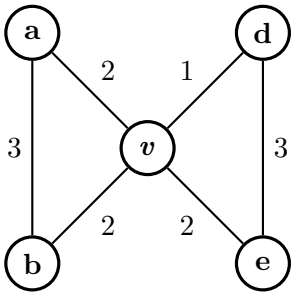


Figure 5.1: An input graph G.

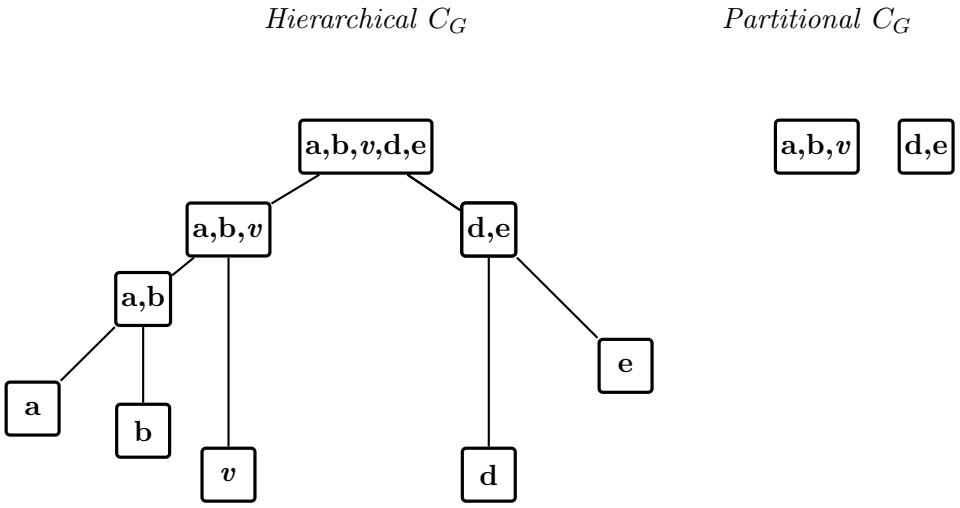


Figure 5.2: Possible hierarchical and partitional clustering solutions for G.

Exclusive, Non-Exclusive, and Fuzzy Clustering Solutions

A clustering algorithm returns either an exclusive (hard), non-exclusive (soft), or fuzzy clustering solution, as illustrated in Figure 5.3.

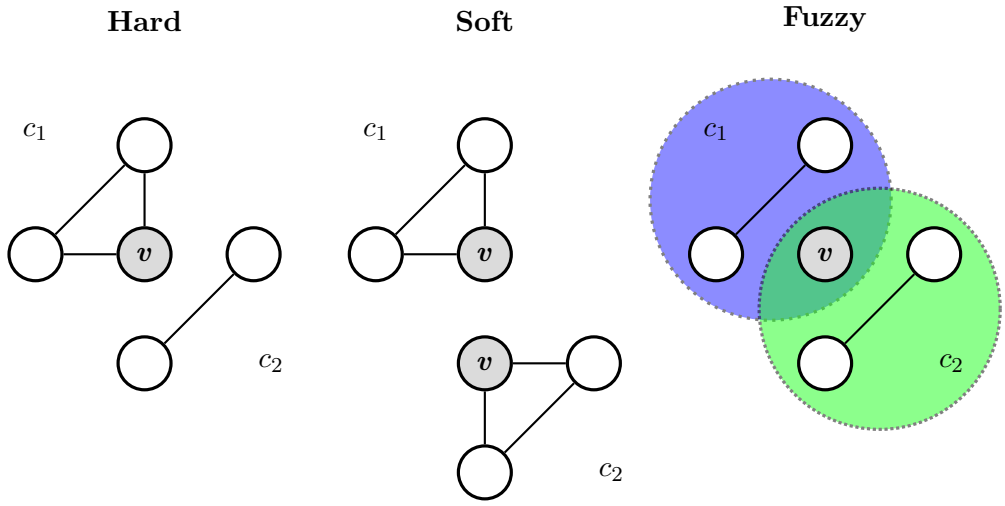


Figure 5.3: Hard, soft, and fuzzy clustering solutions.

A hard clustering algorithm returns a clustering solution in which each vertex v of a graph G is assigned to a single cluster. Soft and fuzzy clustering algorithms allow a vertex v to belong to more than one cluster, the difference being that a soft clustering algorithm will explicitly place v into ≥ 1 clusters whereas a fuzzy clustering algorithm uses some measure (probabilistic or otherwise) of the degree to which v belongs to each cluster in $\mathcal{C}_G$[1].

Complete and Partial Clustering Solutions

A complete clustering solution assigns every vertex in G to a cluster in $\mathcal{C}_G$. A partial clustering solution leaves particular vertices out of $\mathcal{C}_G$. For example, in weighted graphs, vertices with low similarity (edge weight) to all other vertices in G may be omitted from C_G in order to return well defined clusters containing vertices with high intra-cluster similarity.

5.3 Two Commonly Applied Clustering Methods

All clustering methods are based on a notion of prototype, density, or hierarchy (Tan et al., 2006; Aggarwal and Reddy, 2013). This section describes two well-known and widely applied clustering methods: the first prototype-based; the second density-based[2]. These particular clustering methods were chosen as they are good representatives of their

[1] Fuzzy clustering is analogous to fuzzy set theory cf. Zadeh (1965) and Klaua (1965).

[2] Hierarchical clustering is not considered in this section as partitional clustering methods are applied in this thesis. Tan et al. (2006) provides a highly readable account of agglomerative and divisive hierarchical clustering, with Klapaftis (2008) describing how (conceptual) hierarchical clustering can be applied to WSI.

classes: k-means/medoids for the prototype-based class of clustering algorithms; *DBSCAN* for density-based clustering. The aim of this section is to show that commonly applied clustering methods are not necessarily the right clustering methods to apply in WSI. The following two sections define the clustering methods in graph-theoretic terms where G represents the input graph and $\mathcal{C}_G$ represents the clustering solution.

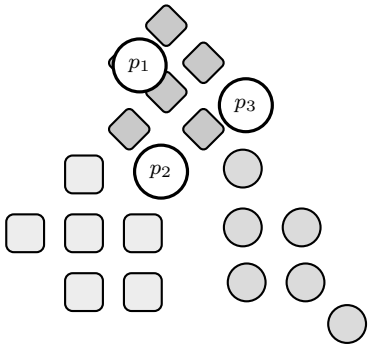
5.3.1 Prototype-Based Clustering

A prototype-based approach to clustering uses a notion of typicality. In this approach, each cluster c_i in $\mathcal{C}_G$ contains a 'typical vertex' p_{c_i}, said to be the prototype for c_i. Each vertex v_i in G is assigned to the cluster containing the prototype p_j it is most similar to.

The k-means (Steinhaus, 1956; MacQueen et al., 1967; Lloyd, 1982) and k-medoids (Kaufman and Rousseeuw, 1990) algorithms are two widely applied prototype-based clustering algorithms. k-means uses the notion of a *centroid* vertex to represent the prototype of a cluster. A centroid vertex p_{c_i} is the average of all vertex attribute values within cluster c_i. k-medoids applies a similar notion to categorical data, where a *medoid* vertex p_{c_i} is the most representative vertex out of all vertices within cluster c_i.

As illustrated in Figure 5.4 (a), a standard implementation of a k-type algorithm begins by randomly selecting k number of vertices as initial prototypes. Each vertex in G is then assigned to the cluster containing the prototype it is most similar to. The prototypes are then recomputed, with vertices reassigned to the updated prototypes. This process continues until convergence or until some pre-defined stopping condition is met (Figure 5.4 (b)).

(a) Initial prototypes



(b) Final prototypes

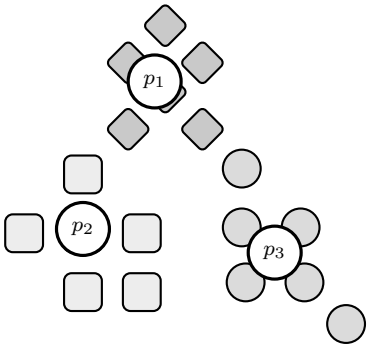


Figure 5.4: Prototype-based clustering.

k-means and k-medoids are straightforward to implement and compute. Both have a relatively efficient time and space complexity, with a run-time of $O(n)$ and a space complexity of $O((n + k)a)$, where: n is the number of vertices in G; k is the number of clusters required, and a is the number of attributes vertices take. However, k-type algorithms have a number of limitations, outlined as follows:

- A notion of prototypicality must exist in the input data for these algorithms to work.

- They can only find globular shaped clusters; clusters of varying shape, size, and density cannot be found.

- The output $\mathcal{C}_G$ is a hard clustering in which each vertex is assigned to a single cluster. Thus, if a vertex is equally attracted to ≥ 2 clusters, one must be selected at random. This issue may be alleviated by applying a soft clustering version of k-means: fuzzy k-means (formally known as fuzzy c-means (Dunn, 1973)). Fuzzy k-means assigns each vertex v_i in G to every cluster c_j in $\mathcal{C}_G$. A weight $w(v_i, c_j)$, ranging between 0 and 1, denotes the strength of v_i's membership in c_j[3]. However, aside from soft clustering vertices, fuzzy k-means retains all the limitations of k-means.

- Prototype-based algorithms are non-deterministic and sub-optimal. Initial prototypes are selected at random, thus multiple runs over the input data are usually required in order to find an optimal clustering solution; yet, as each run is non-deterministic there is no guarantee of finding an optimal solution. Requiring the user to pre-define the number of clusters k means that these algorithms cannot automatically find the number of clusters in the input space. In WSI, the number of senses a word has is not necessarily known in advance, and different words will have different numbers of senses, thus k-type algorithms are not well-suited to this task.

5.3.2 Density-Based Clustering

A density-based approach to clustering finds high density subgraphs in an input graph G that are surrounded by low density subgraphs. Various density-based clustering methods have been proposed in the literature (Tan et al., 2006; Kriegel et al., 2011), all of which work by counting the number of vertices found in particular subgraphs of the input graph. This section describes a widely applied density-based approach to clustering: DBSCAN (Ester et al., 1996). This section also describes the Shared Nearest Neighbours (SNN) algorithm (Tan et al., 2006), an algorithm that enables DBSCAN to find clusters of varying density. The methods described in this section are arguably the most applicable to word co-occurrence data, thus of most use in NLP. Indeed, the SNN algorithm is applied in Ferret (2004) to WSI and in Dorow (2007) to identify synonyms, with DBSCAN applied in Bogdanova (2010) to detect figurative language. SNN is applied in the novel WSI systems that are introduced in Chapter 10.

5.3.2.1 DBSCAN

Ester et al. (1996) propose a centre-based approach to density clustering: DBSCAN (Density Based Spatial Clustering of Applications with Noise). Ester et al. describe the algorithm in terms of points in a metric space. Relating this to graphs, the space is a graph G and points are vertices in G. A centre-based approach to density clustering places each vertex v in G at the centre of some pre-defined subspace of G. DBSCAN finds high density subgraphs of G by applying two user-defined parameters: *Eps* and *MinPts*. These parameters are used to classify a vertex as either a core, border, or noise vertex, *Eps* is

[3]See: Bezdek (1981); Jain and Dubes (1988); Tan et al. (2006) for further detail.

the length of the radius around a vertex and *MinPts* is the number of vertices within *Eps* that must be exceeded for a vertex to be classified as a core vertex. Figure 5.5 shows a vertex v with three neighbouring vertices: $\{n_1, n_2, n_3\}$ that fall within the *Eps* of v.

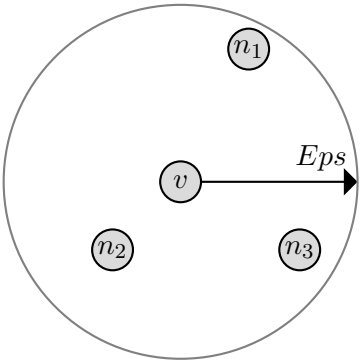


Figure 5.5: DBSCAN: *Eps* of v.

Each vertex $v \in G$ is classified as either a core, border, or noise vertex by counting the number of vertices that fall within the radius (*Eps*) of v (inclusive of v). If this number exceeds *MinPts*, then v is classified as a core vertex. A border vertex falls within the *Eps* of a core vertex though is not a core vertex itself. A noise vertex is neither a core nor border vertex, as illustrated in Figure 5.6 below.

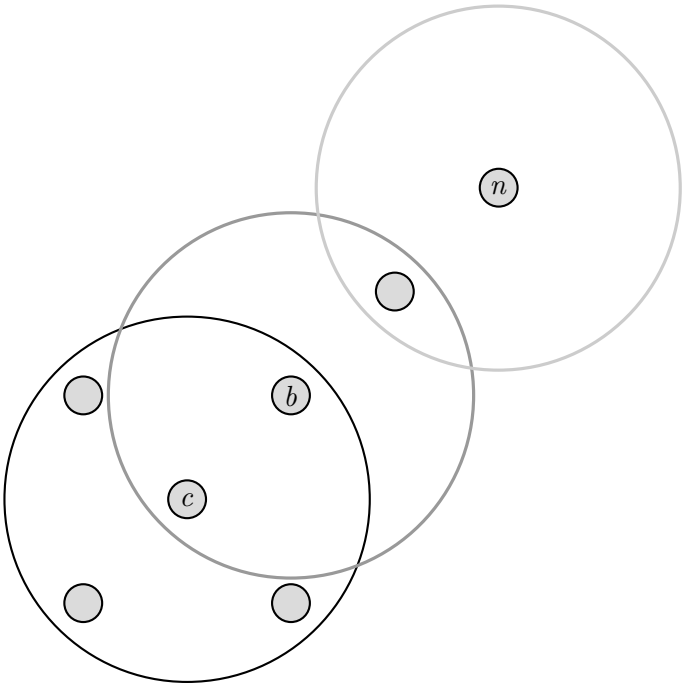


Figure 5.6: DBSCAN: if *MinPts* is set to 4 then c is a core vertex, b is a border vertex and n is a noise vertex.

The DBSCAN algorithm is defined in Algorithm 5.1.

Algorithm 5.1 DBSCAN

1: Classify each vertex in G as a core, border, or noise vertex.
2: Discard noise vertices.
3: Connect all core vertices that fall within *Eps* of each other.
4: Connect each border vertex to its associated core vertex/vertices.
5: **return** C_G: the connected components of G.

Given suitable values for the *Eps* and *MinPts* input parameters, DBSCAN will return high density subgraphs of G that are clearly delineated from each other by the low density subgraphs which surround them. DBSCAN appears to be better suited to WSI than prototype-based approaches as it automatically determines the number of clusters in the input graph and finds clusters of arbitrary shape and size, features not evident in k-type algorithms. However, DBSCAN has its own limitations. The algorithm returns a partial clustering, as noise vertices are discarded. Vertices are hard clustered, thus the ambiguous nature of vertices is lost. Such ambiguity may be of particular use in WSI, as a vertex belonging to ≥ 2 clusters might indicate that the word the vertex represents is ambiguous (polysemous). DBSCAN can be expensive to compute, particularly if the input graph is dense. The worst case run-time for the algorithm is $O(n^2)$. However, if G is sparse and an optimisation method such as a k-d tree (Bentley, 1975) is used to store and retrieve the data then the run-time can be reduced to $O(n\ log\ n)$. Finding a suitable value for the *Eps* parameter can be problematic: if *Eps* is set too high then every vertex v in G contains G in its radius; if *Eps* is set too low then every vertex v in G contains only v in its radius. The algorithm is also unable to find clusters of varying density. This issue is addressed in the following section.

5.3.2.2 Shared Nearest Neighbours (SNN)

As noted above, DBSCAN cannot find clusters of varying density. This section describes a method that can be applied to the input graph G prior to running DBSCAN, one that enables DBSCAN to find clusters of different densities.

The Shared Nearest Neighbours (SNN) algorithm (Tan et al., 2006) takes a graph G and for each vertex pair i, j in G computes the number of first order neighbours the pair have in common: their shared nearest neighbours. The intuition here is that two vertices sharing many neighbours are more similar to each other than two vertices sharing few, or no, neighbours. This idea is analogous to the notion in NLP of contextual similarity where words that share many context words are deemed to be semantically related: the distributional hypothesis (Harris, 1954). The SNN algorithm is defined in Algorithm 5.2 below.

Algorithm 5.2 The Shared Nearest Neighbours (SNN) Algorithm

1: For each vertex v in G find the k nearest neighbours of v.
2: For each pair of vertices i, j if i is in the k nearest neighbours of j and j is in the k nearest neighbours of i: similarity(i, j) = the number of shared neighbours; 0 otherwise.
3: **return** SNN scores for all i, j pairs in G.

Thus, if two vertices i and j are nearest neighbours of each other and connect to the same n number of nearest neighbours, the SNN score for i, $j = n$, as illustrated in Figure 5.7.

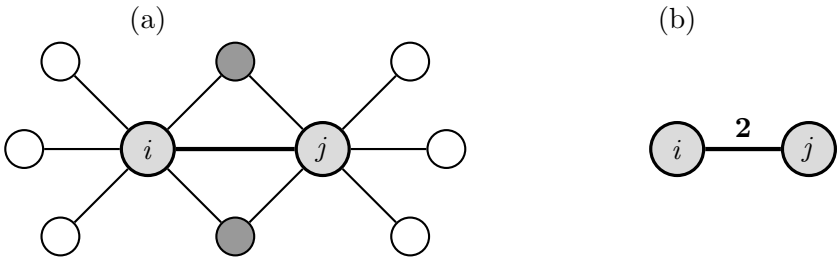


Figure 5.7: Shared Nearest Neighbours example.

SNN scores are then used to construct a SNN similarity graph G_{SNN}, a sparsified transformation of G in which each edge weight $w_{\{i,j\}}$ is the SNN score between vertices i and j. Note that SNN is not a clustering algorithm, its purpose is to reduce (sparsify) the graph space over which a clustering algorithm is applied.

SNN is a simple algorithm, however, its strength lies in its ability to group vertices of varying density. As Tan et al. note: "the stars in a galaxy are no less real clusters of stellar objects than the planets in a solar system" (Tan et al., 2006, p.624), a statement that is pertinent to WSI as word sense clusters may have varying densities, with words that define predominant senses of a target word found in large, dense subgraphs of the input graph and those that define rarer senses of the target word sparsely distributed across the graph in small, low density clusters.

5.3.3 Summary

This section presented an overview of prototype and density-based approaches to clustering. Two commonly applied clustering methods were used as exemplars of these approaches: k-means/medoids, representative of a prototype-based approach to clustering, and DBSCAN, which provides a good example of a density-based approach.

A prototype-based approach to clustering was shown to require a notion of typicality in the input data, a concept that is illusive to define for WSI. This approach to clustering also requires the number of clusters to be pre-specified, thus is not well-suited to WSI where the number of senses words take may be unknown in advance and where different words will have different numbers of senses. Varying density was shown to be an issue in the DBSCAN density-based approach; thus, in its standard implementation, is unsuited to WSI where the sense distributions of words may be skewed, with predominant senses of target words represented by closely-knit, high-density subgraphs and rarer senses loosely dispersed across the graph space in small, low density clusters. A solution to this problem was given by transforming the original input graph to a sparse SNN similarity graph, with the SNN graph then used as input to DBSCAN.

5.4 Graph-Based Soft, Fuzzy, and Parameter-Free Clustering Methods for WSI

The previous section described two well-known clustering methods, methods that have been widely applied and that have been shown to return good results in a variety of tasks (Borgelt, 2006; Parimala et al., 2011). The aim was to illustrate various drawbacks of these methods that make them poorly-suited to the task of WSI. These drawbacks, however, are not unique to these methods. Indeed, many other clustering methods have particular issues that make them poorly-suited to WSI. For example, Hierarchical Agglomerative Clustering (HAC) (Murtagh and Contreras, 2011) and Spectral Clustering (Nascimento and de Carvalho, 2011) have, in their standard implementations, a time complexity of $O(n^3)$, where n is the number of items to be clustered, thus are not suited to clustering large word co-occurrence graphs. Furthermore, these algorithms recursively compare pairs of items to find partitions in the data, pairwise computations that can place two items in the same cluster that would be better clustered to separate clusters. Clustering methods based on Dirichlet processes, e.g. Latent Dirichlet Allocation (Blei et al., 2003) and Expectation-Maximisation (Dempster et al., 1977), require parameters set by the user and/or estimated, thus cannot automatically find the number of clusters in the input space. Finding the optimal graph cut (partition) in Min-Cut clustering (Flake et al., 2004) is an NP-hard (Non-deterministic Polynomial-time hard) problem, shown in Biemann (2007) to be unsuited to NLP tasks.

Noting the points above, this section reviews clustering methods that are arguably best-suited to WSI. More precisely, this section reviews graph-based clustering methods that return either a soft or fuzzy clustering solution, or that require no input parameters. Soft and fuzzy clustering methods allow vertices (words) to belong to more than one cluster, thus can model word ambiguity by distributing words to their various sense clusters; parameter-free clustering methods allow vertices (words) to self organise to word sense clusters, thus can find the number of clusters, hence the number of word senses, in a graph automatically. As numerous examples of these types of clustering methods exist, it would be infeasible to review every clustering method that has been proposed in the literature. Indeed, Schaeffer (2007), which presents a comprehensive survey of graph-based clustering methods makes exactly this point, stating that: "As the field of graph clustering has grown quite popular and the number of published proposals for clustering algorithms as well as reported applications is high, we do not even pretend to be able to give an exhaustive survey of all the methods, but rather an explanation of the methodologies commonly applied and pointers to some of the essential publications related to each research branch" (Schaeffer, 2007, p.27). As this thesis is concerned with the use of graph-based methods for inducing word senses, the clustering methods reviewed in this section are: 1.) graph-based, thus of a similar class to the novel methods introduced in this thesis; 2.) applied in WSI; 3.) similar in some respect to the parameter-free soft clustering algorithm that is introduced in Section 5.5. Sections 5.4.1 and 5.4.2 review soft and fuzzy graph-based clustering methods that have been applied to WSI. Section 5.4.3 describes the only parameter-free

graph-based clustering algorithm that, to date, has been applied to induce word senses. Section 5.5 introduces a novel graph-based clustering algorithm that is both parameter-free and returns a soft clustering solution. A number of the methods that are reviewed in this section are further discussed in the literature review of graph-based WSI systems that is presented in Chapter 7.

5.4.1 Graph-Based Soft Clustering for WSI

This section describes three graph-based soft clustering methods that have been applied to WSI: the first uses hypergraphs, the second uses graphs of word collocations; the third uses the graph-theoretic measure of curvature. All three methods allow an ambiguous (polysemous) word to belong to more than one cluster, thus, in theory, allow polysemous words to be clustered to each of their various sense clusters.

5.4.1.1 Hypergraphs

A hypergraph is a generalisation of a graph to a set of vertex subsets. Whereas edges in a graph connect vertex pairs, edges in a hypergraph (*hyperedges*) can connect any number of vertices (Berge, 1984). Formally, a hypergraph H is a pair $H = (V, E)$ where V is a set of vertices and E is a set of hyperedges: a set of non-empty subsets of V. Figure 5.8 shows an example of a hypergraph where $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ and $E = \{e_1, e_2, e_3, e_4\} = \{\{v_1, v_2, v_3\}, \{v_2, v_3\}, \{v_3, v_5, v_6\}, \{v_4\}\}$. Note that vertices v_2 and v_3 are members of hyperedge e_1 and hyperedge e_2.

Korkontzelos and Manandhar (2010) implement a hypergraph based WSI system *UoY*, reasoning that hypergraphs should better induce word senses than single vertex graphs as: "a hyperedge is a more expressive representation than a simple edge, because it is able to capture the information shared by two or more words" (Korkontzelos and Manandhar, 2010, p.1). In *UoY*, a hypergraph is constructed for each target word: words in contexts of the target word are represented as vertices and sets of 2, 3, or 4 co-occurring context words are represented as a hyperedge. Target word graphs are then filtered and weighted using association rules (Tan et al., 2006), a process which requires six separate parameters tuned. Senses of a target word are then induced by iteratively selecting the vertex in the target word's hypergraph with the (current) highest degree and taking this vertex and its connected neighbours to represent a sense of the target word. This process continues until a stopping condition is met. Target words in contexts are then disambiguated by assigning to each induced cluster a score equal to the sum of weights of hyperedges found in the context of the target word and selecting the cluster with the highest score.

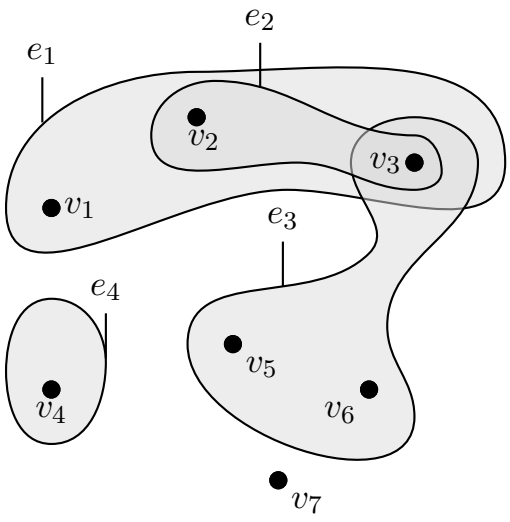


Figure 5.8: A hypergraph.

UoY is used to induce the senses of the target words in the SemEval-2007 Word Sense Induction and Discrimination task (Agirre and Soroa, 2007) and the SemEval-2010 Word Sense Induction and Disambiguation task (Manandhar et al., 2010). The system was found to be the worst performing system out of the six WSI systems that participated in the SemEval-2007 task. However, results were far better in the SemEval-2010 task where the system was shown to return the second best V-Measure score and the best Supervised Recall scores out of the twenty six participant systems.

5.4.1.2 Graphs of Collocations

A collocation or link graph is a graph in which vertices represent more than one word (Dorow, 2007). Figure 5.9 shows an abstract example of how a graph G, in which vertices represent single words, is transformed to a collocation graph G', in which vertices represent two words that were connected in the original graph G. For example, vertices v_1 and v_2 are linked in G, therefore are transformed to vertex v_{1-2} in G'. Note that this transformation from a single vertex graph to a collocation graph allows vertices to belong to more than one connected component in the collocation graph. For example, vertex v_2 in G is a member of two separate components in the collocation graph G', thus v_2 is, effectively, soft clustered.

Dorow et al. (2005)

Dorow et al. (2005) use link graphs to induce word senses by first building a noun co-occurrence graph $G = (V, E)$ in which each vertex in V represents a noun found in British National Corpus (BNC)[4] noun co-ordination patterns and edges in E connect noun (vertex) pairs that co-occur in co-ordination patterns. Word senses are then induced by transforming G into a link graph G' in which each vertex represents a two word collocation.

[4] `http://www.natcorp.ox.ac.uk/`

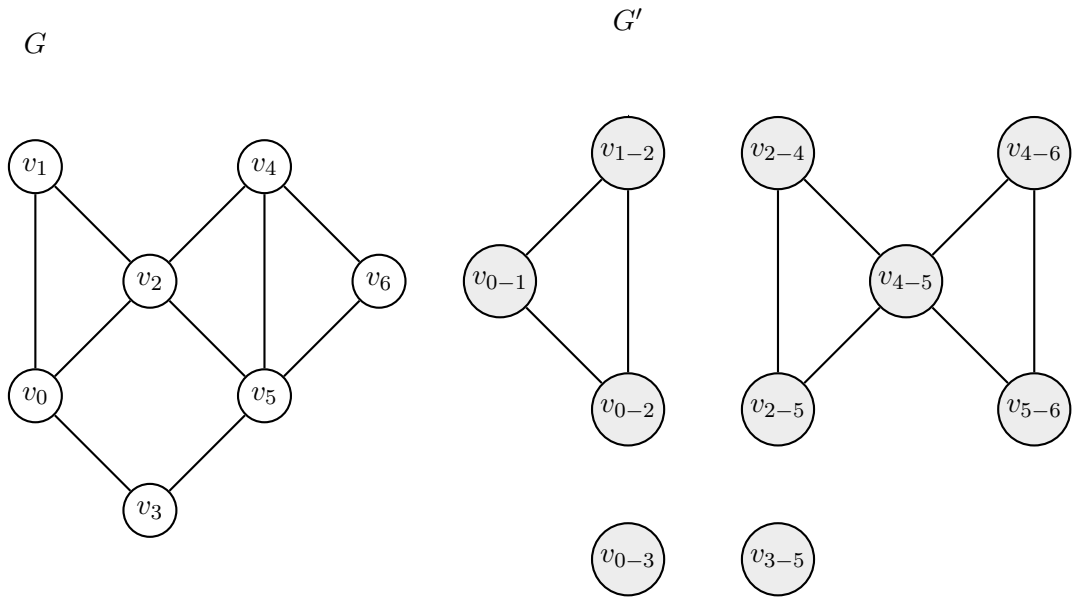


Figure 5.9: Transformation of a graph G to a collocation (link) graph G'.

Vertices in G' not appearing in triangles are discarded (e.g. vertices v_{0-3} and v_{3-5} in Figure 5.9 would be discarded), filtering out less important (non-transitive) connections between words. The link graph is then clustered using the *MCL* clustering algorithm (van Dongen, 2000). Cluster pairs whose shared information content (the negative logarithm of the probability of the words they have in common) exceeds 50% of the information contained in the smaller of the two clusters are merged, with *MCL* applied a second time to return the clustering solution. Dorow et al. (2005) claim that: "links contain more specific contextual information than nodes (vertices) representing words", further, that: "ambiguity is specifically addressed and accommodated by allowing a word to belong to several clusters" (Dorow et al., 2005, p.1). These claims, however, are not tested within a formal WSI evaluation, though link clustering is applied to a WordNet mapping exercise where it is shown to produce more accurate class labels than single vertex graphs.

Klapaftis and Manandhar (2008)

Klapaftis and Manandhar (2008) extend the approach presented in Dorow et al. (2005) by using graphs of collocations to induce word senses. In this approach, senses of a target word are induced by building an edge weighted BNC word co-occurrence graph. Vertices in the graph are collocations (two words, nouns only) that co-occur with the target word, and weights on edges between vertex pairs are calculated using the co-occurrence frequencies of the associated collocations. A smoothing technique is applied to the graph in order to find further edges between vertices, with the final graph clustered using the *Chinese Whispers* clustering algorithm (described in Section 5.4.3). Klapaftis and Manandhar evaluate this approach to WSI within the framework of the SemEval-2007 Word Sense Induction and Discrimination task where results show that the approach: 1.) produces less sense conflating clusters than a standard graph-based approach; 2.) achieves high

performance, relative to the other five participant systems, in both the supervised and unsupervised evaluations. This approach to WSI is described in greater detail in Chapter 7.

5.4.1.3 Soft Clustering of Semantically Similar Words

Dorow et al. (2005) and Dorow (2007) use *curvature*, a measure of vertex cohesion in graphs (equivalent to the clustering coefficient measure described in Chapter 3), to find classes of semantically similar words. This method uses the same word co-occurrence graph G that was used in the link clustering method described in Section 5.4.1.2. The method is described in detail in Chapter 9, it can, however, be summarised as follows:

1. Compute the curvature of each vertex in G.
2. Delete vertices in G with curvature less than 0.5.
3. Return the remaining connected components in G.

This process returns a hard clustering in which semantically fuzzy (ambiguous) low curvature words are not assigned to their respective sense clusters. Each hard cluster is therefore augmented with the low curvature vertices that are attached to it, which, effectively, results in the soft clustering of ambiguous words to their various sense clusters.

5.4.2 Graph-Based Induction of Fuzzy Word Senses

This section describes two graph-based methods for generating fuzzy clusters of word senses and a third method that finds fuzzy semantic relations between word senses, relations that can be used to vary the granularity of a target word's senses. The first two methods are to some extent influenced by the research presented in Velldal (2003) and Velldal (2005) in which words are soft clustered to create fuzzy clusters of Norwegian word senses. However, the methods Velldal applies are not graph-based, rather they use prototype-based clustering (fuzzy c-means and its derivatives (Bezdek, 1981)) and hierarchical agglomerative clustering (HAC) (Tan et al., 2006) to identify and merge word senses. As previously noted, these algorithms, arguably, are not well-suited to WSI: fuzzy c-means is parameterised to find a particular number of clusters; HAC has a prohibitively high run-time of $O(n^3)$, where n is the number of items (words) to be clustered, thus would not be suited to the large word co-occurrence graphs that are often used in WSI (Dorow, 2007). Furthermore, no systematic quantitative evaluation of these methods is given by Velldal as no broad coverage semantic resource for Norwegian exists that could be used as a gold standard.

5.4.2.1 Borin and Forsberg (2010)

Borin and Forsberg (2010) view synonymy, and more generally word sense, as a matter of degree that is best expressed in terms of uncertainty (fuzziness). In this work, the degree of synonymy amongst Swedish words is quantified by building fuzzy clusters (*fuzzy synsets*) of Swedish word senses. These synsets are constructed using the lexical resources

Synlex and SALDO. Synlex (Kann and Rosell, 2006) is a list of graded synonym pairs, created by asking users of an on-line Swedish-English dictionary (not cited) to rate the degree of synonymy of random, automatically generated pairs of words on a scale of 0 (not synonyms) to 5 (completely synonymous). Synlex contains all pairs with an average rating of 3 or higher. SALDO (Borin et al., 2008) is a Swedish lexical-semantic resource, similar to WordNet (Miller et al., 1990), containing associative relations (connections) between word senses. As with WordNet, SALDO can be viewed as a graph in which lexical-semantic relations (e.g. synonymy, hyponymy, hypernymy, antonymy) between words are encoded on graph edges. The basic units of SALDO are word senses; the basic units of Synlex are word pairs. Thus, in order to generate fuzzy clusters of word senses, a set of Synlex-SALDO word sense pairs is created in which either:

1. Each word in a Synlex pair occurs once in Synlex and has one sense in SALDO
2. Each word in a Synlex pair occurs in several pairs in Synlex and has one sense in SALDO

This yields a set of 9,236 pairs (*Pairs*). Fuzzy synsets are then generated by computing the transitive closure of word sense pairs, as follows. For every graded word sense pair (w_i, w_j) in $Pairs$ with a rating d_k greater than or equal to d_{cutoff}, the membership of the words in the current $Synsets$ set (the fuzzy synsets) is checked and adjusted based on each word's synset membership. The pseudocode for this process, as presented in Borin and Forsberg (2010), is as follows:

$Synsets = \{\}$

for $\langle\langle w_i, w_j\rangle, d_k\rangle \in Pairs$
  if $d_k \geq d_{cutoff}$
  case membership $(\langle w_i, w_j\rangle, Synsets)$ of
  $\langle S_1, S_2\rangle \Rightarrow Synsets.merge(S_1, S_2)$
  $\langle S_1, \{\}\rangle \Rightarrow Synsets.add(w_j, S_1)$
  $\langle \{\}, S_2\rangle \Rightarrow Synsets.add(w_i, S_2)$
  $\langle \{\}, \{\}\rangle \Rightarrow Synsets.new(w_i, w_j)$

return $Synsets$

This method therefore returns synsets of degree $\geq d_{cutoff}$ by assigning all words that are connected by some path of graded synonymy relations (where no relation has degree less than d_{cutoff}) to the same synset. Borin and Forsberg discuss various theoretic issues that arise from their method, however, no formal evaluation of the method is presented.

5.4.2.2 Oliveira and Gomes (2011)

Oliveira and Gomes (2011) apply a graph-based clustering algorithm to build fuzzy synsets of Portuguese word senses. The process begins by using a lexical pattern analyser (PAPEL[5]) to extract synonym pairs (*synpairs*) from dictionary definitions. Examples of syn-pairs, as presented in Oliveira and Gomes (2011), are as follows:

[5] `http://www.linguateca.pt/PAPEL/PJR.html`

- **mind**, n: *brain*, *head*, *intellect*

  synpairs = (*brain*, *mind*), (*head*, *mind*), (*intellect*, *mind*)

- **machine**, n: *the same as computer*

  synpairs = (*computer*, *machine*)

A synonym graph $G = (V, E)$ is constructed using the extracted synpairs: each vertex in the vertex set V represents a word in a synpair and each edge in the edge set E connects two words that are in the same synpair. Fuzzy synsets are then identified by applying the following graph clustering algorithm (as defined in Oliveira and Gomes (2011)):

1. Create a $|V| \times |V|$ matrix M.
2. Assign each m_{ij} (row, column) cell in M the cosine similarity score between the vectors $\vec{v_i}, \vec{v_j}$ that represent words v_i, v_j.
3. Normalise the columns in M to sum to 1.
4. Extract a fuzzy cluster F_i from each row m_i in M, consisting of words where the value in $m_{ij} > 0$. The value m_{ij} is *the membership degree* of the word v_j in F_i, denoted $\mu_{F_i}(v_j)$.
5. For each cluster F_i that has all words included in a larger cluster F_j ($F_i \cup F_j = F_j$ and $F_i \cap F_j = F_i$), merge F_i and F_j to a new cluster F_k. F_k consists of all words in F_j, with the membership degree of each word $v_j \in F_k$ the sum of the individual membership degrees in F_i and F_j: $\mu_{F_k}(v_j) = \mu_{F_i}(v_j) + \mu_{F_j}(v_j)$.

In step 2, values in a vector $\vec{v_i}$ are pointwise mutual information (PMI) scores (Fano and Hawkins, 1961) between the word v_i and each synpair word in the synpair set, with each value multiplied by a discounting factor to negate the bias PMI has for infrequent words (Lin and Pantel, 2002). As M represents the graph G, the value in each m_{ij} cell can therefore be interpreted as the weight w_{ij} on the graph edge e_{ij}.

The result of the clustering is a fuzzy thesaurus in which synsets model uncertainty (analogous to fuzzy sets (Zadeh, 1965)) and where the fuzzy membership of a word in a synset can be interpreted as:

1. The confidence level in using the word to indicate the meaning of the synset. That is, if $\mu_{F_i}(v_j) > 0$, the word v_j has a meaning in common with those of other words in F_i. The membership degree $\mu_{F_i}(v_j)$ is therefore the confidence level in using the word v_j with the meaning of the synset F_i.
2. The likelihood of a word v_j taking the sense of each synset it belongs to. That is, as $\sum_{i=1\ to\ |F|} \mu_{F_i}(v_j) = 1$, membership degrees of v_j can be interpreted as the possible senses v_j takes, hence the likelihood of v_j conveying each of its various senses.

Oliveira and Gomes note that their algorithm is better suited to large word co-occurrence graphs than fuzzy c-means as there is no requirement to keep two matrices in memory

(fuzzy c-means requires one matrix for weights, the other for centroids). Furthermore, there is no requirement, as there is in fuzzy c-means, to specify the number of clusters. Oliveira and Gomes manually evaluate their method against hand-crafted thesauri, finding that more than 73% of the fuzzy synsets are classified as correct.

5.4.2.3 Inducing Fuzzy Semantic Relations

Apidianaki (2010) shows how an unsupervised WSI method, proposed in Apidianaki (2008b), can be used to find fuzzy semantic relations between word senses in parallel corpora. This method induces the senses of polysemous source language (SL) words by clustering their translation equivalents (TEs). The clustering solution returned by this method reveals: 1.) semantic relations between the TEs of polysemous SL words; 2.) relations between their various senses.

The method is trained on a bi-lingual sentence aligned parallel corpus. Aligned sentences in which a polysemous SL word occurs are extracted and indexed by reference to each of its TEs. SL contexts are then analysed, with a frequency list constructed for each TE. Each list contains the lemmas of the content words that occur in contexts of the SL word whenever the SL word is translated by the TE. These SL content words form a feature set for the TE. Similarity between the feature sets of TE pairs is then computed using a weighted generalisation of the Jaccard coefficient (Apidianaki, 2008a). The hypothesis here is that TE pairs with high similarity scores are semantically similar.

Algorithm 5.3 *SEMCLU*

Input: a list, TE-List, of translation equivalents; a similarity table; a similarity threshold; an initially empty set of clusters C.

1: **for** each target word, translation equivalent $(tw, \text{TE} \in \text{TE-List})$ pair **do**
2: **if** similarity $(tw, \text{TE}) >$ similarity threshold **then**
3: (tw, TE) have a pertinent relation and form an initial cluster $c_{(tw,\text{TE})}$
4: add $c_{(tw,\text{TE})}$ to C
5: **end if**
6: **end for**
7: **for** each cluster $c \in C$ **do**
8: recursively enrich c by adding each TE $\in$ TE-LIST that has a pertinent relation with all members of c
9: **end for**
10: **return** clusters C: a set of fully connected components in which each member of each cluster has a pertinent relation to all other members of the cluster

The scores of the pairwise similarity calculations are then used in a clustering algorithm *SEMCLU* (Apidianaki, 2008a) which places translation equivalents that are highly similar to each other into the same cluster. Pseudocode for the *SEMCLU* algorithm is given in Algorithm 5.3 where the similarity threshold for a target word tw is the mean of the scores assigned to the pairs of its translation equivalents and a translation equivalent pair with a score greater than the similarity threshold is said to have a pertinent relation. *SEMCLU* returns a clustering solution in which cliques of words represent word senses. Note, however, that translation equivalents in (tw, TE) pairs can be clustered

to any number of clusters, a process that allows the algorithm to return an overlapping (fuzzy) clustering of word senses. Overlaps (fuzziness) between word senses can then be used to describe the relations between intersecting word sense clusters. For example, to differentiate between close and distant word senses or to modify the granularity of word senses. That is, as overlapping sense clusters often describe sub-senses of coarse-grained senses, merging sub-senses of a target word will return a description of the predominant senses that define the target word. Apidianaki (2009) shows how this WSI method can improve the performance of a Word Sense Disambiguation method in a bilingual context; Apidianaki et al. (2009) shows how the method can benefit Machine Translation.

5.4.3 *Chinese Whispers*: a Parameter-Free Clustering Algorithm

Chinese Whispers (Biemann, 2006a, 2007) is a graph-based, parameter-free clustering algorithm that has been shown to be of use in a number of NLP tasks (part of speech tagging (Biemann, 2006b); word sense disambiguation and induction (Navigli and Crisafulli, 2010; Di Marco and Navigli, 2013); summarisation (Medelyan, 2007), and lexical acquisition (Dorow et al., 2005), amongst others). This section describes the algorithm in some detail in order to show how various limitations of *Chinese Whispers* can be addressed by the novel clustering algorithm that is introduced in Section 5.5.

5.4.3.1 *Chinese Whispers*

Chinese Whispers partitions a weighted, undirected graph into a set of hard clusters. Each cluster is said to contain "vertices that broadcast the same message to their neighbours", thus "can be viewed as a simulation of an agent-based social network" (Biemann, 2007; Epstein, 2006). The algorithm is designed with NLP tasks in mind, with Biemann illustrating its utility for part of speech tagging (Biemann, 2006b), language identification (Biemann and Teresniak, 2005), and word sense induction (Biemann, 2006a). The algorithm, as defined in Biemann (2006a), is shown in Algorithm 5.4.

Algorithm 5.4 *Chinese Whispers*

1: **for all** $v_i \in V$ **do**
2: $class(v_i) = i$
3: **end for**
4: **for** it=1 to number-of-iterations **do**
5: **for all** $v \in V$, randomised order **do**
6: $class(v)$ = predominant class in $neigh(v)$
7: **end for**
8: **end for**
9: **return** partition P induced by class labels

Thus, given a graph G, each vertex in G is first assigned a unique class label (lines 1 to 3 in Algorithm 5.4). Vertices are then processed, in random order, with each vertex v inheriting the predominant class label within its neighbourhood (lines 5 to 7). This process continues for a predefined number of iterations, with the algorithm's run time linear in the number of edges in G.

The predominant class label for a vertex v is the class in the neighbourhood of v that has the maximum sum edge weights to v, as illustrated in Figure 5.10.

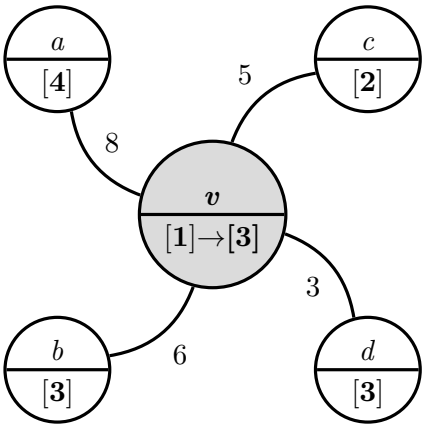


Figure 5.10: Assigning the predominant [class] label to vertex v.

In Figure 5.10, vertex v, initially assigned to class label [1], inherits the class label [3]. Class [3] is the predominant class within the neighbourhood of v as the sum of the edge weights $(6 + 3 = 9)$ between class [3] and vertex v is greater than either of the other two classes: class [2] = 5 and class [4] = 8. If more than one class is found to be the predominant class of a vertex then one is assigned at random. A vertex's class label is updated in each iteration of the algorithm (lines 5 to 7). Thus, if, in a further iteration of the algorithm vertices a and c were to inherit a new class, for example class [5], and vertices b and d remained as class [3] then v would inherit class [5].

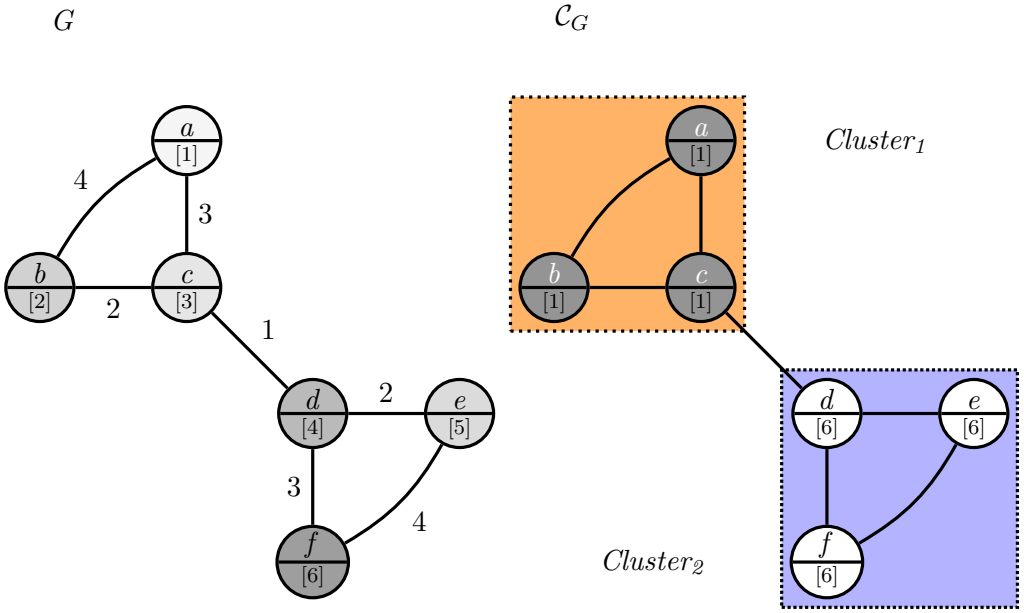


Figure 5.11: *Chinese Whispers* clustering example.

Through this iterative process, subspaces of the graph eventually stabilise to the predom-

inant class. Note that strongly connected components of the graph in which vertices have homogeneous class labels are 'self preserving' as they cannot be infected by class labels from other subspaces of the graph. As Figure 5.11 shows, vertices in G form two strongly connected subgraphs of G. Each vertex in G is initially assigned a unique class label: (a,[1]),...,(f[6]). Iterative assignment of predominant class labels stabilises to two predominant class labels [1] and [6] returning $\mathcal{C}_G$, a clustering of the two vertex sets: $\{a, b, c\}$ and $\{d, e, f\}$.

5.4.3.2 *Chinese Whispers* and Non-Determinism

Chinese Whispers is not guaranteed to converge. As illustrated in Figure 5.12, vertex v can inherit the class label [1] or [2], thus in each iteration of the algorithm v is assigned class [1] or class [2], with the algorithm settling on a preference for neither class.

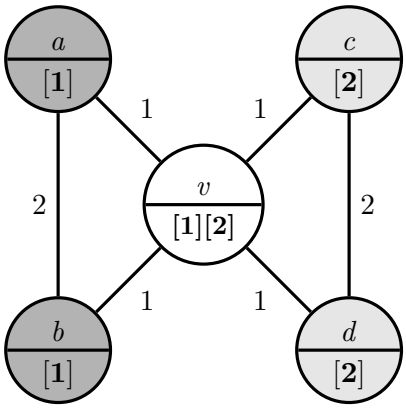


Figure 5.12: *Chinese Whispers* and non determinism.

This non-deterministic behaviour results in different outcomes for different runs of the algorithm over the same graph. For example, given the graph in Figure 5.13 (and assuming uniform edge weights), *Chinese Whispers* may return either one, two, three, or four partitions, whereas a natural partition for this graph would be four 3-cliques: $\{a, b, c\}, \{d, e, f\}, \{g, h, i\}, \{j, k, l\}$. Biemann's solution to this issue, *Deterministic Chinese Whispers*, involves running *Chinese Whispers* over the graph several times and then selecting the run which returns the highest number of clusters. This is not a particularly satisfactory solution, as the run with the highest number of clusters may not necessarily be the optimal solution. Biemann's defence is to downplay the issue of determinism, noting that edge weight ties (such as those as illustrated in Figure 5.12) are rare in weighted graphs. This may be true for graphs with fine-grained, real-valued weights, but for coarse-grained, integer weighted graphs this may be an issue, particularly as *Chinese Whispers* can only process graphs in which edge weights are integers. A preferable solution would be to allow the algorithm to soft cluster vertices. *Chinese Whispers* is a hard clustering algorithm, thus must assign each vertex in a graph to a single cluster. In Figure 5.12 vertex v is randomly assigned to class [1] or class [2], whereas a soft clustering algorithm would assign v to both classes, indicating that v has equal affinity to two subspaces of the graph.

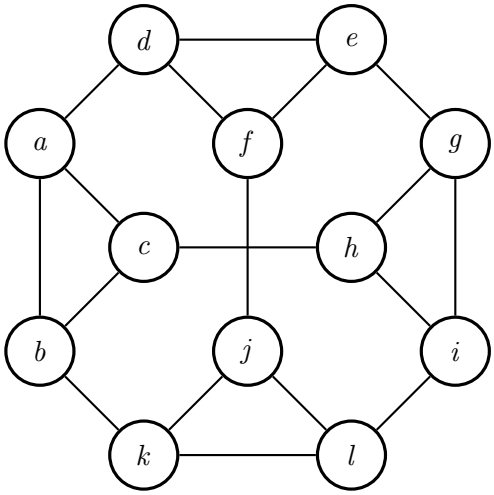


Figure 5.13: Tetrahedron graph.

5.4.3.3 Analysis of *Chinese Whispers*

The non-deterministic nature of *Chinese Whispers* makes it difficult to analyse. However, Biemann attempts an analysis by using artificially generated graphs (Biemann, 2007).

In the first analysis, n-bipartite-clique graphs are generated. As Figure 5.14 shows, an n-bipartite-clique graph consists of two n-cliques with each vertex in the one clique having just one edge connecting it to a vertex in the other clique. The aim is to apply *Chinese Whispers* in order to partition an n-bipartite-clique graph into two clusters, with each cluster containing exactly one clique. Biemann finds that only a third of *3*-bipartite-cliques were separated. This increased to just under two thirds for *4*-bipartite-cliques. The poor separation of clusters here is a direct result of the algorithm's non-deterministic nature where the random processing of vertices in early iterations of the algorithm allow a particular class label to dominate, and so spread across the entire graph. However, separation improves as n increases and is close to 100% separation for *10*-bipartite-cliques.

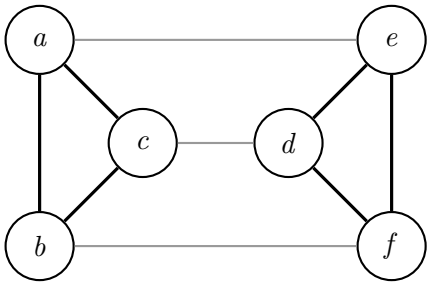


Figure 5.14: A 3-bipartite-clique graph.

In the second analysis, Biemann looks at scale-free (Barabási, 2003) small world graphs (Watts and Strogatz, 1998). Biemann uses Steyvers and Tenenbaum's model (Steyvers and Tenenbaum, 2005) to generate graphs of this type. The model starts with a graph G consisting of a small number of fully connected vertices. A new vertex v_{new} is added to G by selecting an existing vertex v_{exists} (using probability based on vertex degree) and

then linking v_{new} to M number of v_{exists} neighbours. Two graphs of this type are merged, using a variable merge rate parameter, to form what Biemann calls a *small-world-mixture* graph. The merge rate decides how many vertices in the first graph should be linked to vertices in the second. *Chinese Whispers* is then applied to the merged graph in order to separate the graph into two clusters, each, ideally, forming one of the original two graphs. Unsurprisingly, Biemann found that separation was less pronounced as the merge rate increased, again, indicating that a particular class dominates the graph. The best results were obtained for non equi-sized graph merges (300 vertices in the first graph, 30,000 in the second) where, even at a merge rate of 30%, merged graphs could be separated 50% of the time.

Biemann's analysis shows that *Chinese Whispers* is poorly suited to small graphs; indeed, the algorithm is poorly suited to graphs of any size that contain small ($\leq n = 10$) n-cliques. In NLP, data used to classify lexical items is often sparse, resulting in clusters of a small order, consequently *Chinese Whispers* would not be best suited here.

5.4.3.4 *Chinese Whispers* and an 'Ideal' Clustering Algorithm

Table 5.1 lists a number of aspects that might be considered when implementing a clustering algorithm. The table shows aspects that *Chinese Whispers* implements, setting them against those that an 'ideal' clustering algorithm might implement.

Aspect	***Chinese Whispers***	**'Ideal'**
Non-parametric	yes	yes
Deterministic	**no**	**yes**
Run-time is linear in the size of the graph	yes	yes
Guaranteed to terminate	yes	yes
Applies convergence	**yes**	**no**
Applies randomisation	**yes**	**no**
Vertex processing is order independent	yes	yes
Soft clusters vertices	**no**	**yes**
Applicable to integer edge weighted graphs	yes	yes
Applicable to real-valued edge weighted graphs	**no**	**yes**
Applicable to undirected weighted graphs	yes	yes
Applicable to directed weighted graphs	yes	yes
All vertices clustered	yes	yes
Finds clusters of varying shape	yes	yes
Finds clusters of varying size	yes	yes
Finds clusters of varying density	yes	yes
Automatically determines the number of clusters to return	yes	yes

Table 5.1: *Chinese Whispers* and an 'ideal' clustering algorithm.

The following section introduces a novel clustering algorithm that implements the 'ideal' aspects listed in Table 5.1.

5.5 The *MaxMax* Clustering Algorithm

This section introduces *MaxMax*, a novel graph-based soft clustering algorithm that is applicable to both undirected and directed edge-weighted graphs. As a deterministic algorithm, *MaxMax* will, for a particular input graph, always return the same set of clusters. Run-time is linear in the size of the input graph. As a non-parametric algorithm, *MaxMax* allows vertices to self organise, automatically finding the optimal number of clusters in a graph. As shown in Hope and Keller (2013a,b), this feature of the algorithm makes it well-suited to Word Sense Induction (WSI) where the number of senses words take may be unknown in advance.

5.5.1 *MaxMax*

MaxMax is based on a simple idea: each vertex in a graph should be assigned to the cluster containing the vertex to which it has *maximal affinity*. Given a weighted graph $G = (V, E)$, *affinity* between vertex pairs $u, v \in G$ is quantified by edge weights $w(u, v)$. A vertex u is said to have maximal affinity to a vertex v if the edge weight $w(u, v)$ is maximal amongst the weights of all edges incident on u. In this case, v is said to be a *maximal vertex* of u (v need not be unique). *MaxMax* applies two principles:

1. Vertex pairs u, v are assigned to the same cluster if either vertex is a maximal vertex of the other.

2. Maximal affinity implies a directed relationship: if v is a maximal vertex of u then there is a directed relationship from v to u.

The algorithm is defined in Algorithm 5.5 below. The two discrete stages of *MaxMax* are described in the following two sections.

Algorithm 5.5 *MaxMax*

Input: a weighted graph $G = (V, E)$

1: transform G to an unweighted directed graph $G' = (V, E')$ where:
$e(v, u) \in E'$ iff $e(u, v) \in E$ and v is a maximal vertex for u
2: label all vertices of G' as $ROOT$
3: **for** each vertex v of G' **do**
4: **if** v is labelled $ROOT$ **then**
5: relabel any vertex u reachable from v ($u \neq v$) as $\neg ROOT$
6: **end if**
7: **end for**

Stage One of *MaxMax*: Graph Transformation

In stage one (lines 1 and 2 of Algorithm 5.5), *MaxMax* takes a weighted graph G and transforms it to an unweighted directed graph G'. As Figure 5.15 shows, maximal affinity relationships between vertices in G are used to determine the direction of edges in G'.

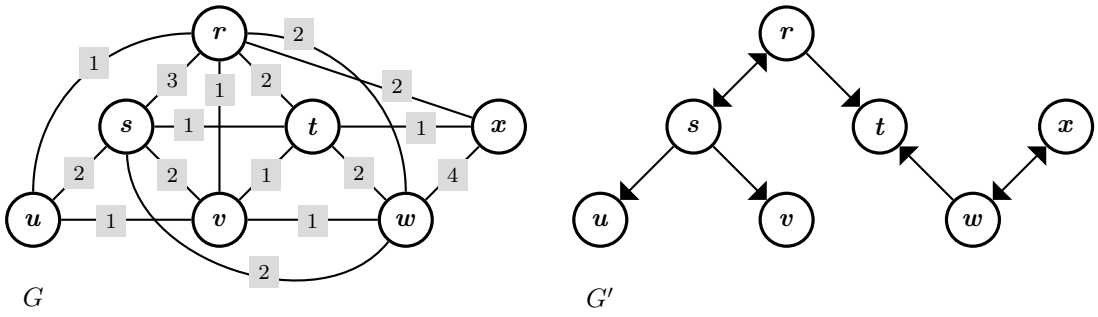


Figure 5.15: A weighted graph G and its transformation to an unweighted directed graph G'.

Vertices in G' are then labelled as $ROOT$ vertices. Thus, at the end of stage one each vertex in G' is a candidate root vertex of a possible cluster in G'.

Stage Two of *MaxMax*: Identifying Clusters

Clusters are identified in stage two (lines 3 to 7 of Algorithm 5.5) by tracing directed paths in G' to find rooted subgraphs of a particular type, namely *maximal quasi-strongly connected components*.

Note first that in a directed graph (*digraph*) a vertex v is said to be *reachable* from a vertex u if there is a directed path from u to v; secondly, that a vertex v that is reachable from a vertex u is said to be a *descendant* of u. For example, vertex v in graph G' of Figure 5.15 is reachable from vertices s and r, thus v is a descendant of s and r.

A directed graph is said to be *quasi-strongly connected* (QSC) if for any vertex pair v_i, v_j there is a vertex v_k (not necessarily distinct from v_i or v_j) such that v_i is reachable from v_k and v_j is reachable from v_k (Dasgupta et al., 2006). A QSC digraph contains at least one vertex v_r which is a *root* vertex, in the sense that every other vertex can be reached by following a directed path from v_r. Given a digraph G', a subgraph of G' is a *maximal* QSC subgraph if it is a QSC digraph and it is not possible to add any further vertices in G' without rendering the subgraph non-QSC.

Clusters are identified in the *MaxMax* algorithm by finding the root vertices of maximal QSC subgraphs of G'. As Algorithm 5.5 shows, this is achieved by labelling all vertices reachable from a given vertex as $\neg ROOT$. At the end of stage two, vertices that are still labelled $ROOT$ uniquely identify clusters, since they correspond to the roots of maximal QSC subgraphs of G'. As Figure 5.16 shows, this process allows vertices to be soft clustered: vertex t is assigned to two clusters $\{r, s, t, u, v\}$ and $\{t, w, x\}$, where r or s is the $ROOT$ vertex of $\{r, s, t, u, v\}$ and w or x is the $ROOT$ vertex of $\{t, w, x\}$. Note that which vertex becomes the root of a cluster depends on the order in which vertices are enumerated in the for loop. Crucially, however, this has no impact on the maximal QSC subgraphs (clusters) that are identified by the algorithm. This point is clarified in the following section.

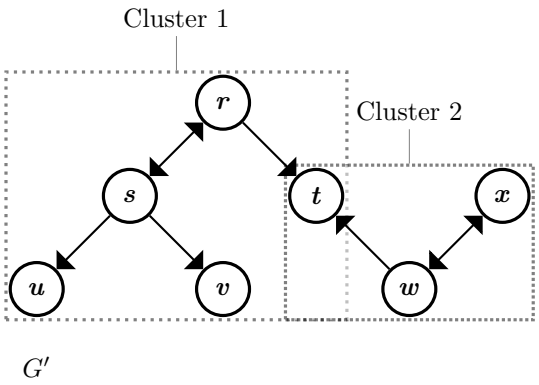


Figure 5.16: Two clusters in G': $\{r, s, t, u, v\}$ and $\{t, w, x\}$.

5.5.2 Correctness of *MaxMax*

The formulation of *MaxMax* presented above, and in Hope and Keller (2013a), in terms of QSC subgraphs makes it clear that the algorithm is guaranteed to terminate. This section presents a formal justification for the claim, made in the introduction to Section 5.5, that *MaxMax* is deterministic - in the sense that it will always return the same set of clusters for a given weighted graph as input. Note first that the digraph constructed in stage one of the algorithm must be unique given the input graph. This follows since the maximal vertices of each vertex of the input graph are determined independently of all other vertices solely from consideration of the weights of the incident edges.

For stage two it is necessary to demonstrate that given the digraph constructed in stage one, the algorithm will identify its unique set of maximal QSC subgraphs. Note that on termination of the algorithm each vertex will be labelled either $ROOT$ or $\neg ROOT$. The following two propositions argue that the set of vertices labelled $ROOT$ picks out exactly the set of maximal QSC subgraphs. The first proposition shows that each $ROOT$ vertex corresponds to a distinct maximal QSC subgraph. The second proposition shows that each maximal QSC subgraph has a root vertex amongst those labelled $ROOT$. Note that it is not claimed that the set of vertices labelled $ROOT$ on termination of *MaxMax* must be unique. This set may depend on the order of enumeration of vertices in the for loop at stage two of the algorithm. However, for a given input graph the set of clusters (i.e. maximal QSC subgraphs) determined by the set of $ROOT$ vertices will be unique.

Proposition 1: Each *ROOT* vertex is the root of a distinct maximal QSC subgraph.

Assume that a vertex v remains labelled $ROOT$ on termination of *MaxMax*. Consider the subgraph consisting of all vertices reachable from v and all edges with endpoints reachable from v. It is clear that this subgraph is QSC and has vertex v as a root. Now suppose that the subgraph is not *maximal* QSC. In this case there must be some vertex u such that v is reachable from u but not vice-versa. But if u is labelled $ROOT$, then v would have been relabelled $\neg ROOT$ at some point during stage two; alternatively, if u is labelled $\neg ROOT$ then it and any vertices reachable from u (and in particular v) must have been relabelled

$\neg ROOT$ at some point during stage two. Either possibility leads to a contradiction. The conclusion, therefore, is that the subgraph is maximal QSC.

Finally, note that for any vertex v labelled $ROOT$ on termination of *MaxMax*, all vertices reachable from v are labelled $\neg ROOT$. But then for any pair of $ROOT$ vertices, neither is reachable from the other and so the QSC subgraphs rooted at these vertices must be distinct.

Proposition 2: Each maximal QSC subgraph has a root vertex labelled *ROOT*.

Consider a maximal QSC subgraph $\hat{G}$ and suppose that $\hat{G}$ *does not* have a root vertex that is labelled $ROOT$ on termination of *MaxMax*. It is claimed that any vertex labelled $\neg ROOT$ is reachable from a vertex labelled $ROOT$. This can be seen from consideration of the way in which vertices are relabelled during the for loop in stage two of the algorithm. In particular, at each pass through the for loop a vertex is only labelled $\neg ROOT$ if there is some distinct $ROOT$ vertex from which it is reachable. Consider now some root vertex r of $\hat{G}$. By supposition, r is labelled $\neg ROOT$ and it follows that there must be a vertex v labelled $ROOT$ from which r is reachable. Clearly v cannot be a root for $\hat{G}$. But in that case $\hat{G}$ cannot be maximal QSC since there is a larger QSC subgraph rooted at v: a contradiction. The conclusion, therefore, is that each QSC subgraph has a root vertex that is labelled $ROOT$.

5.5.3 Time Complexity of *MaxMax*

Given a connected graph $G = (V, E)$, it can be shown that *MaxMax*'s run-time complexity is $O(|E|)$, that is, linear in the number of edges of G.

Note first that the transformation of an weighted graph G to an unweighted directed graph G' in stage one of the algorithm can be computed in $O(|E|)$ time. In constructing G' it is necessary to find maximal vertices of each vertex in G. For a given vertex u, the set of maximal vertices can be identified by scanning each of the edges from u to a vertex adjacent to u in order to determine those of maximal weight. This is done for each vertex of G, with each edge in G inspected once - noting that edges from u to v and from v to u are considered as separate edges in undirected graphs (Dasgupta et al., 2006). Consequently, G' can be constructed in time linear in the number of edges of G. Vertices are then marked as *root*, taking $O(|V|)$ time.

Turning now to the second stage of the algorithm, note that the for loop in lines 3 to 7 of the algorithm iterates over vertices to identify descendant vertices that should be marked $\neg root$. Naively tracing all of the descendants of each vertex in turn could in the worst case entail visiting $O(|V|)$ vertices on each pass through the loop, thus result in an overall time complexity of $O(|V|^2)$. However, once a vertex has been marked $\neg root$ none of its descendants needs to be visited again. Equivalently, no directed edge needs to be traversed more than once, thus overall complexity of the for loop is linear in the number of edges of G', hence linear in the number of edges of G. This yields an overall time complexity of $O(|E|)$.

5.5.4 Testing *MaxMax*: Graph Partitioning

Similar to Biemann's tests, described in Section 5.4.3.3, *MaxMax* is applied to partition artificially generated graphs. However, whereas Biemann evaluates *Chinese Whispers*'s ability to partition unweighted graphs, *MaxMax* is evaluated here using weighted n-bipartite-clique graphs. A weighted n-bipartite-clique consists of two n-cliques with each vertex in one clique having just one edge connecting it to a vertex in the other clique. Intra-edge weights within cliques are set higher than inter-edge weights between cliques. A simple example of a weighted n-bipartite-clique graph is shown in Figure 5.17.

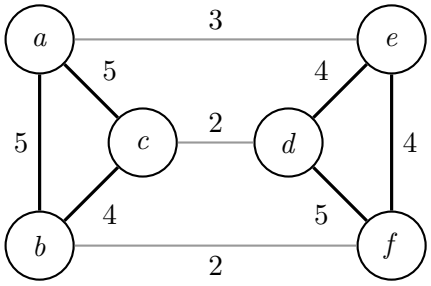


Figure 5.17: A weighted 3-bipartite-clique graph.

Weighted n-bipartite-clique graphs of the order $n = [3, 4, ..., 100]$ are generated, with *MaxMax* then applied to the graphs. Results, as expected, show that *MaxMax* separates the cliques perfectly. *MaxMax* is also applied in a second evaluation to scale-free, small world graphs, using the Steyvers and Tenenbaum model (Steyvers and Tenenbaum, 2005). Paired graph mixtures of the order V = [300, 3,000, 30,000] are generated, with intra-weights within graphs, again, set higher than inter-weights between graphs. Results show that *MaxMax* separates each merged graph, over all paired mixtures (e.g. 300, 30,00), into the two pre-merged graphs. Results for the two tests are to be expected as the intra-inter edge weighting in the merged graphs give a natural partition to two clusters. These tests, however, show that *MaxMax* is, in practice, doing what, in theory, it is expected to do.

5.5.5 Testing *MaxMax*: Graph Clustering

5.5.5.1 Introduction

This section presents an empirical analysis of *MaxMax*, comparing its performance with that of three other clustering algorithms, namely: *Affinity Propagation* (Frey and Dueck, 2007), *Chinese Whispers* (Biemann, 2006a), and *DBSCAN* (Ester et al., 1996). The *Chinese Whispers* and *DBSCAN* clustering algorithms are described in Sections 5.4.3 and 5.3.2.1; *Affinity Propagation* is described in Section 5.5.5.4. These particular algorithms were chosen as they are good representatives of a class of graph-clustering algorithms, including *MaxMax*, that are capable of finding the number of clusters in a graph automatically.

The four clustering algorithms are evaluated on a set of eight weighted graphs, derived from synthetic, two-dimensional datasets. The eight graphs have varying properties, con-

taining clusters that differ in density, size, and shape; therefore, should be a good test of an algorithm's ability to cluster data correctly. The question as to whether an evaluation using synthetic datasets can identify the key attributes of a clustering algorithm that makes it suited to WSI is discussed in Section 5.5.5.8.

5.5.5.2 The Test Set: Shapes

The Shapes test set, provided by the University of East Finland[6], consists of eight sets of two-dimensional data points. As Figure 5.18 shows, subsets of data points in each of the eight sets make up clusters. For example, the Spiral set in Figure 5.18 has three distinct sequences of data points, each of which, according to the gold standard key that is provided with the Shapes test set, represents a gold standard cluster.

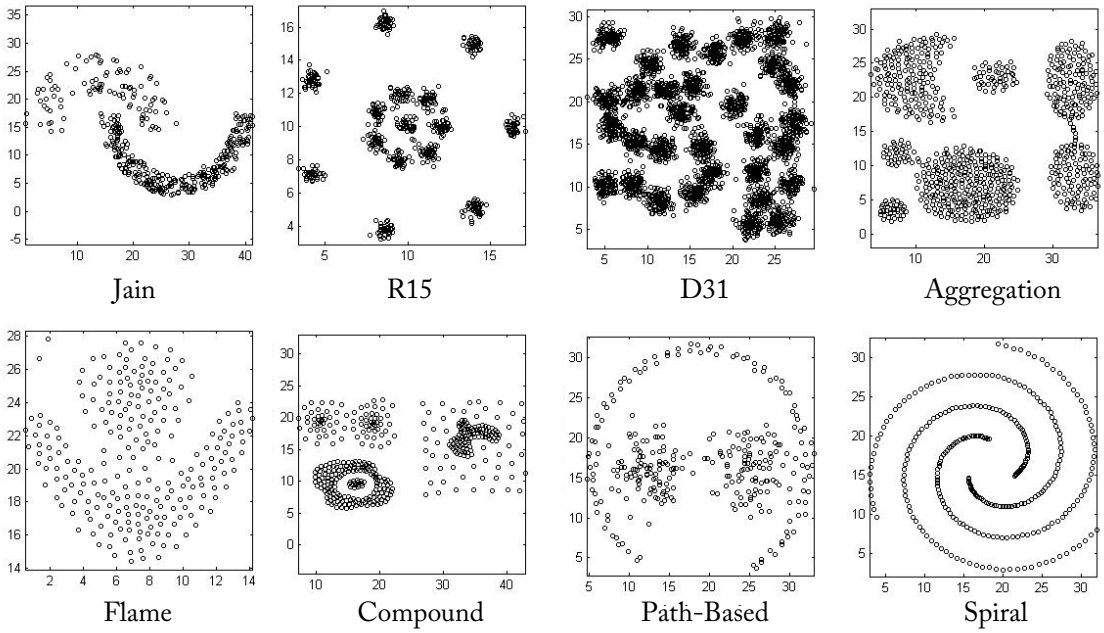


Figure 5.18: The Shapes test set.

Table 5.2 lists the number of points and gold standard clusters contained in a dataset along with its source.

[6] `http://cs.joensuu.fi/sipu/datasets/`

Dataset	|Points|	|Clusters|	Source
Jain	373	2	Jain and Law (2005)
R15	600	15	Veenman et al. (2002)
D31	3100	31	Veenman et al. (2002)
Aggregation	788	7	Gionis et al. (2007)
Flame	240	2	Fu and Medico (2007)
Compound	399	6	Zahn (1971)
Path-Based	300	3	Chang and Yeung (2008)
Spiral	312	3	Chang and Yeung (2008)

Table 5.2: The Shapes test set.

5.5.5.3 Applying the Clustering Algorithms to the Shapes Test Set

All four clustering algorithms process a graph $G = (V, E)$ representing S, one of the eight test set spaces in Shapes. Each vertex $v \in V$ represents a point on the two dimensional Euclidean plane of S. Edges $e(v_i, v_j) \in E$ connect vertex pairs v_i, v_j. Edge weights $w(v_i, v_j)$ quantify the *similarity* between vertex pairs v_i, v_j, where similarity $s(v_i, v_j)$ is the Euclidean distance between v_i and v_j. *Affinity Propagation* and *MaxMax* use negative Euclidean distance as edge weight; *Chinese Whispers* uses negative Euclidean distance scaled to positive integers (as required by the algorithm), and *DBSCAN* uses positive Euclidean distance.

Input Parameters: *Chinese Whispers* and *MaxMax* are parameter-free clustering algorithms. *Affinity Propagation* requires a shared real-valued number $s(v_i, v_i)$ to be set for all vertices $v \in V$. This value is the similarity between each vertex $v \in V$ and itself. The default value specified and applied in Frey and Dueck (2007), and that is also used in this evaluation, is the median of the edge weights in G. *DBSCAN* requires two input parameters to be set by the user: *MinPts* and *Eps*. In Ester et al. (1996), *MinPts* is set to 4 (the same value recommended in Tan et al. (2006)). If $MinPts$ is set to 4, the value of *Eps* can be found by plotting the sorted distances of the fourth nearest neighbour of each point (vertex) in the graph and then taking the value on the y axis of the plot at which there is a sharp increase in distance. Figure 5.19 shows how the *Eps* value is selected for the Spiral dataset. In this example, *Eps* is set to 2.0 as this corresponds to the knee of the curve in the plot.

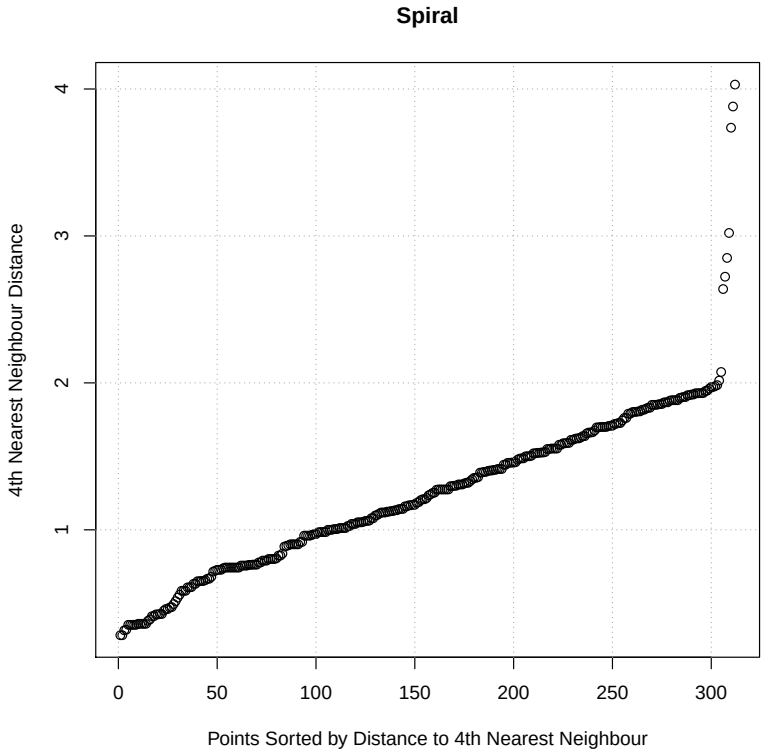


Figure 5.19: *DBSCAN*: 4th nearest nearest neighbour distance plot for the Spiral dataset.

Implementations: The particular implementations of the algorithms applied to the Shapes test set are as follows:

- *Affinity Propagation* - APCluster[7]. APCluster is a direct R port of the MATLAB code that is applied in Frey and Dueck (2007)[8].
- *Chinese Whispers* - Biemann's original implementation[9].
- *DBSCAN* - ELKI (Environment for Developing KDD-Applications Supported by Index-Structures)[10].
- *MaxMax* - the implementation presented in Section 5.5.1.

The *Chinese Whispers* and *DBSCAN* clustering algorithms have been previously described in Sections 5.4.3 and 5.3.2.1. The *Affinity Propagation* clustering algorithm is described in the following section.

5.5.5.4 The *Affinity Propagation* Clustering Algorithm

Affinity Propagation (*AP*) is a hard clustering algorithm, applicable to undirected and directed edge weighted graphs. *AP* works by recursively passing *messages* between a set

[7] `http://www.bioinf.jku.at/software/apcluster`
[8] `http://www.psi.toronto.edu/affinitypropagation/software/apcluster.m`
[9] `http://wortschatz.informatik.uni-leipzig.de/~cbiemann/software/CW.html`
[10] `http://elki.dbs.ifi.lmu.de`

of points in some given input space (e.g. vertices in a graph) until a high quality subset of points is found: *cluster exemplars* (analogous to centroids in k-means clustering). Points are then assigned to the cluster exemplar to which they have highest affinity. The clustering solution emerges through an iterative process of updating message values between points. Time complexity is $O(n^2\,t)$, where n is the number of points to be clustered and t is the number of iterations required to find the clustering solution. AP is described in the following section, as it is in Frey and Dueck (2007), in terms of graph-based clustering.

Message Passing: AP finds cluster exemplars by recursively passing *messages* between connected vertex pairs $(v_i, v_j) \in V$. Two types of real-valued messages are updated in each iteration of the algorithm:

1. Responsibility $r(v_i, v_j)$, sent from vertex v_i to candidate exemplar v_j, indicates how suitable vertex v_j is as an exemplar for vertex v_i, taking into account all other potential exemplars $v_{j'}$ for v_i. Responsibility is defined in Frey and Dueck (2007) as:

$$r(v_i, v_j) \;\leftarrow\; s(v_i, v_j) \;\; - \max_{v_{j'} \text{ s.t. } v_{j'} \neq v_j} \{a(v_i, v_{j'}) + s(v_i, v_{j'})\} \tag{5.1}$$

2. Availability $a(v_i, v_j)$, sent from candidate exemplar v_j to vertex v_i, indicates the degree to which v_i selects v_j as its exemplar, taking into account support from other vertices $v_{i'}$ in selecting v_j as an exemplar. Availability is defined in Frey and Dueck (2007) as:

$$a(v_i, v_j) \;\leftarrow\; \min\left\{0, r(v_j, v_j) \;+ \sum_{v_{i'} \text{ s.t. } v_{i'} \notin \{v_i, v_j\}} \max\{0, r(v_{i'}, v_j)\}\right\} \tag{5.2}$$

AP terminates if message values remain constant over ten iterations. Upon termination, responsibility and availability values for each (v_i, v_j) pair are combined, with each vertex $v_i \in V$ assigned to the cluster exemplar vertex v_j that maximises $r(v_i, v_j) + a(v_i, v_j)$. Assigning vertices to cluster exemplars therefore returns the clustering solution C_G for the input graph G.

5.5.5.5 Evaluation Measures

As there is some debate as to which measures best evaluate clustering solutions (Amigó et al., 2009; Navigli and Crisafulli, 2010; Klapaftis and Manandhar, 2013), the commonly applied measures of Precision, Recall, and F-Score are used here. Noting that the number of clusters returned by a clustering algorithm may not necessarily match the number of gold standard clusters, Precision, Recall and F-Score are calculated using pair counting (Tan et al., 2006), as follows:

$$Precision = \frac{TP}{TP + FP}, \tag{5.3}$$

$$Recall = \frac{TP}{TP + FN}, \tag{5.4}$$

$$F\text{–}Score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{5.5}$$

TP (True Positives) is the number of (v_i, v_j) vertex pairs in which both vertices are found in the same gold standard (GS) cluster and same clustering solution (C_G) cluster.

FP (False Positives) is the number of vertex pairs in which both vertices are found in the same GS cluster but different C_G clusters.

FN (False Negatives) is the number of vertex pairs in which the vertices are found in different GS clusters but found in the same C_G cluster.

Precision measures the *homogeneity* of a clustering solution C_G: whether all pairs in each cluster of C_G are of the same GS class. The higher Precision is the lower the number of false positive errors in C_G. Recall measures the *completeness* of a clustering solution C_G: whether all pairs in each cluster in GS are found in a one cluster of C_G. A clustering solution with high Recall has few positive pairs that are misclassified as negative pairs. In (5.5), Precision and Recall are combined to produce the F-Score: the harmonic mean between Precision and Recall, a value The harmonic mean between two values is typically closer to the smaller of the two values, thus a high F-Score value indicates that both Precision and Recall (homogeneity and completeness) are high.

5.5.5.6 Results

Results for the eight test sets in Shapes are shown in Tables 5.3 to 5.10, where $|C_G|$ is the number of clusters returned by a clustering algorithm.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	21	**1.000**	0.081	0.150
CW	1	0.614	**1.000**	0.761
DBSCAN ($Eps = 1.5$)	9	**1.000**	0.555	0.713
MaxMax	4	0.887	0.682	**0.771**

Table 5.3: Jain results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	16	0.993	0.959	**0.976**
CW	1	0.065	**1.000**	0.122
DBSCAN ($Eps = 0.3$)	15	**1.000**	0.063	0.119
MaxMax	11	0.253	0.923	0.397

Table 5.4: R15 results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	82	0.947	0.377	**0.539**
CW	1	0.032	**1.000**	0.062
DBSCAN ($Eps = 0.6$)	31	**1.000**	0.031	0.061
MaxMax	204	0.246	0.184	0.211

Table 5.5: D31 results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	35	0.994	0.131	0.231
CW	1	0.217	**1.000**	0.356
DBSCAN ($Eps = 1.1$)	7	**1.000**	0.214	0.353
MaxMax	7	0.384	0.921	**0.543**

Table 5.6: Aggregation results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	21	0.980	0.087	0.160
CW	1	0.536	**1.000**	**0.698**
DBSCAN ($Eps = 1.25$)	2	**1.000**	0.532	0.694
MaxMax	1	0.536	**1.000**	**0.698**

Table 5.7: Flame results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	23	0.934	0.177	0.297
CW	1	0.247	**1.000**	0.396
DBSCAN ($Eps = 2.0$)	2	**1.000**	0.305	**0.468**
MaxMax	16	0.362	0.595	0.450

Table 5.8: Compound results.

Algorithm	$\|C_G\|$	**Precision**	**Recall**	**F-Score**
AP	27	0.961	0.113	0.202
CW	1	0.333	**1.000**	**0.499**
DBSCAN ($Eps = 2.0$)	3	**1.000**	0.327	0.493
MaxMax	3	0.332	0.908	0.486

Table 5.9: Path-Based results.

Algorithm	$\|C_G\|$	Precision	Recall	F-Score
AP	35	0.995	0.085	0.157
CW	1	0.331	**1.000**	0.498
DBSCAN ($Eps = 2.0$)	3	**1.000**	0.333	**0.500**
MaxMax	2	0.337	0.889	0.489

Table 5.10: Spiral results.

5.5.5.7 Results Discussion

Table 5.11 ranks algorithm performance by average Precision, Recall and F-Score over the eight datasets in the Shapes test set. *Chinese Whispers*' (*CW*) scores are greyed out as this algorithm returns just one cluster for each of the eight datasets, thus is not partitioning the datasets - an issue that is discussed below.

Rank	Precision	Recall	F-Score
1	*DBSCAN* 1.000	*CW* 1.000	*MaxMax* 0.513
2	*AP* 0.976	*MaxMax* 0.699	*DBSCAN* 0.425
3	*MaxMax* 0.500	*DBSCAN* 0.295	*CW* 0.424
4	*CW* 0.297	*AP* 0.251	*AP* 0.339

Table 5.11: The four algorithms ranked by average Precision, Recall, and F-Score over the eight datasets in the Shapes test set.

Taking the F-Score as the overall measure of a clustering algorithm's performance (homogeneity and completeness), Table 5.11 shows that *MaxMax* returns better results than *Affinity Propagation* (*AP*), *Chinese Whispers* (*CW*), and *DBSCAN*. The following sections discuss a number of points related to the performance of each of the four clustering algorithms on the Shapes test set.

Chinese Whispers: *Chinese Whispers* (*CW*) returns a single cluster for each of the eight test graphs, thus is not partitioning graphs to clusters. Results in Table 5.11 should therefore be dismissed as valid clustering solution scores.

CW's results were unexpected, particularly as the datasets in Figure 5.18 often have distinct clusters, clusters that *CW* might well be expected to identify. On consideration of the properties of *CW*, however, the reasons for this become clear. *CW* applies an iterative, convergence process to find clusters[11]. This process allows one particular cluster label to dominate a graph, with this label spreading across the graph in each iteration of the algorithm. Note that unlike many of the graphs on which *CW* is tested in Biemann (2007), the graphs in the Shapes test set are fully connected (each vertex connects to every other vertex), thus dense regions of graphs, consisting of points (vertices) that are close

[11] 20 iterations per graph, as specified in Biemann (2007).

together, allow a particular cluster label to dominate the region; this label then spreads further to other regions of the graph in each iteration of the algorithm.

Given the above, *CW*'s results are not considered further in the following discussions relating to the other three algorithms.

DBSCAN: *DBSCAN* returns maximal Precision values of 1.0 for all eight datasets in the Shapes test set, thus finds maximally homogeneous clusters, each containing vertex pairs of just one *GS* class (cluster). *DBSCAN* also finds the correct number of *GS* clusters for six of the eight datasets, compared to *MaxMax* (two out of eight) and *AP* (none). However, Recall values in Tables 5.3 to 5.10 show that *DBSCAN* returns clustering solutions that are far from complete. Precision is maximal, thus clusters are pure; however, average Recall, at 0.295, indicates that over two thirds of vertex pairs (on average) are incorrectly assigned to these clusters. Notably, *DBSCAN* performs very poorly on the R15 and D31 datasets, both of which contain dense, globular shaped clusters: graphs that the *AP* algorithm performs well on. Conversely, *DBSCAN* performs well on the datasets that *AP* performs poorly on (Jain, Flame, Spiral). This suggests - given this particular test set - that *DBSCAN* has a preference for graphs in which clusters are less dense.

Affinity Propagation: As with *DBSCAN*, the *Affinity Propagation* (*AP*) algorithm returns high Precision clusters, thus is finding highly homogeneous clusters in the test set graphs. Recall, however, is very low for six of the eight datasets, notably where the density of clusters is comparatively low (Flame, Spiral, Jain, Path-Based). *AP* also over-generates clusters for all eight datasets, whereas *MaxMax* over-generates clusters for three datasets (Jain, D31, and Compound) and *DBSCAN* over-generates for just one dataset (Jain). That said, *AP* returns the best results, by some margin, for two particular datasets: R15 and D31. Note, however, that these two datasets contain clusters that are quite different from those found in the six other datasets, being both dense and globular in shape. Taking into account *AP*'s results for the six other datasets (where *MaxMax* clearly outperforms *AP*), results for R15 and D31 indicate that *AP* is acting in a similar manner to the k-means clustering algorithm. That is, cluster exemplar vertices in *AP* act as centroids, corralling surrounding vertices into globular shaped clusters centred on one specific vertex. Results for the Shapes test set indicate that *AP* is poorly suited to finding clusters that are not of this type. However, if word sense clusters are both dense and globular in shape (something that has yet to be shown in the WSI literature) then *AP* should, based on the evaluation results reported here, be highly suited to WSI.

MaxMax: Taking the average F-Score as the overall evaluation measure to consider, Table 5.11 shows that *MaxMax* returns the best results for the Shapes test set. Furthermore, as Tables 5.3 to 5.10 show, *MaxMax* is ranked as the first or second best performing algorithm on each of the eight datasets, thus appears to handle a wider variety of graph types than either *AP* or *DBSCAN*: graphs in which cluster density, size and shape varies. Though average Precision, at 0.5, is relatively low, Recall, at 0.699, is far higher, being over twice that of *AP* and *DBSCAN*. The clusters produced by *MaxMax* therefore tend

to be less homogeneous than those returned by *AP* and *DBSCAN*; the solutions it finds, however, are generally far more complete.

It is worth noting that the results in Table 5.11 are a product of the Shapes test set itself, where what constitutes a gold standard (*GS*) cluster can be a highly subjective matter. For example, the *GS* for the Flame dataset assigns points to two clusters, as shown in Figure 5.20 below.

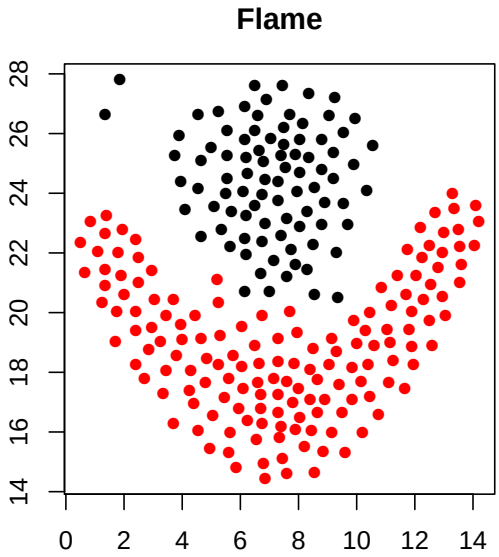


Figure 5.20: The two *GS* clusters in the Flame dataset.

However, it could be argued that the Flame dataset contains just one cluster (ignoring the two outliers, top-left). This is in fact what *MaxMax* finds. A converse point can be made about the granularity of clusters. For example, according to the *GS*, the Spiral dataset contains three clusters: the three concentric sequences of points that make up the spiral. This is a natural interpretation for humans to make, as patterns such as this are found in nature, e.g. in seashells, sunflowers, clouds etc. Arguably though, the three *GS* clusters in Spiral could be could be viewed as being made up of a number of sequences of smaller clusters, which is what *MaxMax* finds. *MaxMax* returns maximum Precision scores (1) for the Spiral dataset, thus is returning maximally homogeneous clusters. However, what *MaxMax* is doing is breaking each of these three sequences up into finer-grained clusters. As Figure 5.21 illustrates, *MaxMax* finds sequences of vertices (clusters) in a path up to the point at which no maximal connection between vertex pairs can be found.

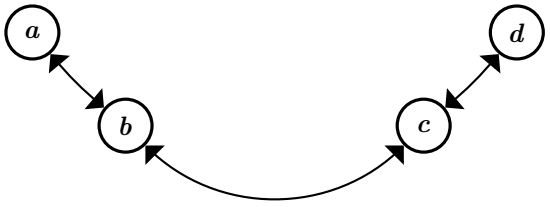


Figure 5.21: *MaxMax*: clusters in the vertex sequences of the Spiral dataset.

Thus, in Figure 5.21, if the edge weight $w(b,a) > w(b,c)$ and $w(c,d) > w(c,b)$ there is no maximal connection between b and c to allow vertices a, b, c, d to form a cluster. This could be viewed as a drawback of *MaxMax*; alternatively, it could be taken to show that *MaxMax* returns the natural clusters that exist within the Spiral dataset.

5.5.5.8 Conclusions

An evaluation using synthetic datasets can be useful for identifying key properties of clustering algorithms. This may in turn help to shed light on those attributes that are important for clustering algorithms suited to WSI. However, caution should be exercised in drawing firm conclusions from this evaluation as further work would be required to establish whether word sense clusters have known, recognisable signatures (i.e. particular densities, shapes, and sizes).

In graph terms, good word sense clusters are strongly connected components, where greater numbers of connections between cluster words reinforce the senses that clusters represent. However, the density and size of a word sense cluster varies according to the predominance of the sense it represents, with predominant senses of a target word represented by clusters that are both larger and denser than those of rarer senses (Widdows, 2004; Dorow, 2007). Thus, knowing, for example, that a clustering algorithm can find dense clusters of a particular size and shape, as *Affinity Propagation* does for datasets D31 and R 15, does not mean that the algorithm is suited to finding all senses of a target word.

Given the points above, what can be said is that the empirical analysis presented in this section shows that *MaxMax*, over eight graphs with varying properties, returns results that compare favourably with three representative graph clustering algorithms, and in particular returns the best results for average F-Score.

5.5.6 *MaxMax*: Discussion

The aim in devising *MaxMax* was to incorporate the 'ideal' aspects listed in Table 5.1 of Section 5.4.3. Table 5.12 recalls these aspects.

Possibly the key aspect here is that *MaxMax* is parameter-free. As noted in Tan et al. (2006), parameter-free clustering algorithms are preferable to those parameterised: "the fewer the parameters, the better" (Tan et al., 2006, p.576). In *MaxMax*, vertices are not prescribed to fit a fixed number of clusters as they are in other soft clustering algorithms (e.g. fuzzy c-means (Bezdek, 1981) and Expectation Maximization (Dempster et al., 1977)). Vertices are left to self organise to clusters, thus the number of clusters in a graph is found automatically.

MaxMax bears some resemblance to single-link Hierarchical Agglomerative Clustering (HAC) (Dasgupta et al., 2006) in that leaf vertices (clusters consisting of one vertex) in the first iteration of HAC are clustered using maximal affinity between vertex pairs. However, from thereon the algorithms operate very differently to one another, with *MaxMax* computing a flat, partitional clustering solution and HAC iteratively merging clusters to

form a hierarchical clustering solution. Furthermore, given an input graph $G = (V, E)$, HAC finds clusters in $O(|V|^3)$ time whereas *MaxMax* does so in $O(|E|)$ time.

Aspect	***MaxMax***	**'Ideal'**
Non-parametric	yes	yes
Deterministic	yes	yes
Run-time is linear in the size of the graph	yes	yes
Guaranteed to terminate	yes	yes
Applies convergence	no	no
Applies randomisation	no	no
Vertex processing is order independent	yes	yes
Soft clusters vertices	yes	yes
Applicable to integer edge weighted graphs	yes	yes
Applicable to real-valued edge weighted graphs	yes	yes
Applicable to undirected weighted graphs	yes	yes
Applicable to directed weighted graphs	yes	yes
All vertices clustered	yes	yes
Finds clusters of varying shape	yes	yes
Finds clusters of varying size	yes	yes
Finds clusters of varying density	yes	yes
Automatically determines the number of clusters to return	yes	yes

Table 5.12: Aspects of the *MaxMax* clustering algorithm.

MaxMax also shares a number of properties with the *Chinese Whispers* clustering algorithm described in Section 5.4.3. Both algorithms are: parameter-free; use affinity within the local neighbourhood of vertices to generate clusters, and have run times that are linear in the size of the input graph. However, there are two key differences between these algorithms. Firstly, *MaxMax* is deterministic whereas *Chinese Whisper*s can return different clustering solutions for the same input graph (as discussed in Section 5.4.3.2). Secondly, *MaxMax* can soft cluster vertices: given the input graph G in Figure 5.22, *Chinese Whispers* randomly assigns vertex c to either Cluster 1 or Cluster 2, whereas *MaxMax* returns the clustering solution C_G.

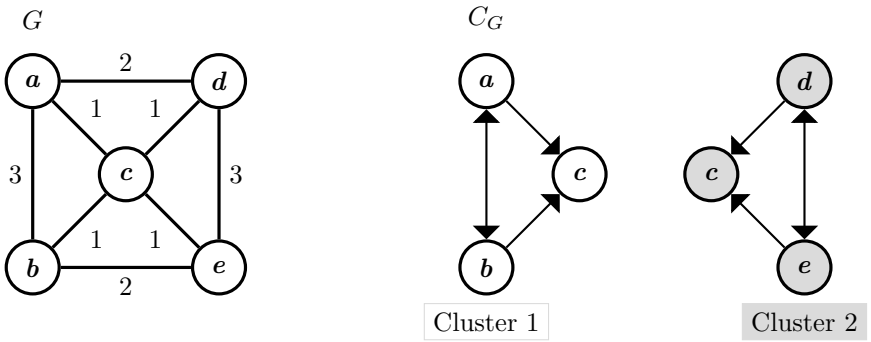


Figure 5.22: *MaxMax*: soft clustering example.

As with any clustering algorithm, there are particular behaviours of *MaxMax* that might be considered limitations. For example, given the graph G in Figure 5.23, where vertex b has maximal affinity with vertex a and vertex c has maximal affinity with vertex

b, *MaxMax* will cluster all three vertices together even though vertices a and c have no affinity to one another. This type of behaviour may be unsuited to particular tasks the algorithm is applied to.

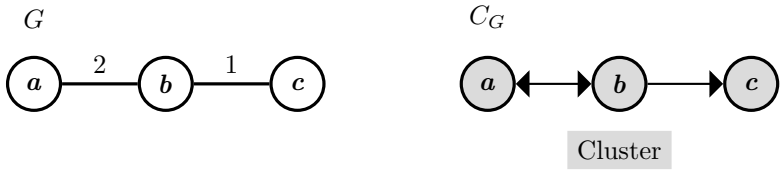


Figure 5.23: *MaxMax*: maximal affinity clustering example.

Also, given a graph in which edge weights are of uniform value *MaxMax* will cluster all vertices into one cluster, as illustrated in Figure 5.24.

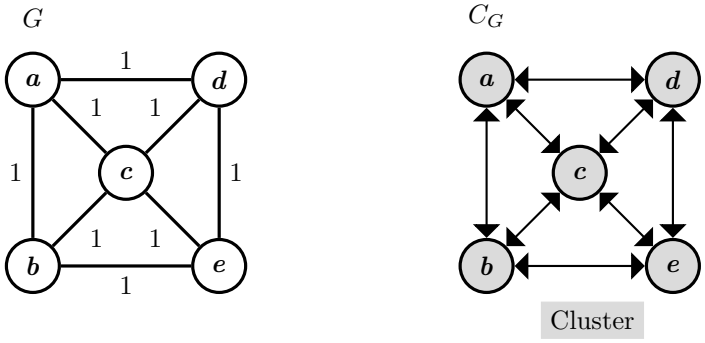


Figure 5.24: *MaxMax*: uniform edge weights example.

However, a weighted graph in which edge weights take the same value is a peculiar type of weighted graph. Indeed, *MaxMax*'s solution of returning one cluster could arguably be said to be the correct clustering solution in this case.

Part II

Word Sense Induction

Chapter 6

Word Sense Induction

This chapter begins with a discussion regarding word sense in which a number of arguments are set out that attempt to define what a word sense means. The following section introduces the Natural Language Processing (NLP) task of Word Sense Induction (WSI), the task of automatically discovering word senses from text. This section gives an overview of the main approaches to WSI that have been proposed in the literature, and contrasts a WSI approach to identifying word senses with the related task of Word Sense Disambiguation (WSD). The chapter concludes by noting the advantages that a graph-based approach to WSI has over others proposed in the literature.

6.1 Word Sense

In Natural Language Processing (NLP), word sense is often defined by Firth's maxim:

> "You shall know a word by the company it keeps"
> (Firth, 1957, p.11),

though an earlier, analogous, definition is given in Wittgenstein (1953):

> "The meaning of a word is its use in the language."
> (Wittgenstein, 1953, p.60, §43)[1].

This notion of word sense defined by use has been adopted by many linguists (Hornby (1954); Harris (1988); Sinclair (2004), amongst others), a notion that views a word's senses as a function of the word's distribution in varying contexts. Thus, the sense a word takes is defined by the words which surrounds it: its context. For example, the word orange within the context [A sweet, juicy orange.] is far likelier to be orange in its fruit sense than, for example, orange in its colour sense. However, inferring the meaning of a word though context alone can fail. For example, given the following lines:

[1] Wittgenstein's definition is more subtle; an ouroboros, where meaning is defined by the use of meaning and so on in infinite regress.

> "There, Leonato, take her back again:
> Give not this rotten orange to your friend;
> She's but the sign and semblance of her honor."
> (*Much Ado About Nothing,* Shakespeare, 1600, Act IV, Scene 1, Lines: 31-33),

orange could be viewed, through local context (the use of the word rotten), as taking its fruit sense. It is only by applying analogical reasoning that a connection is made between a rotten orange and an unvirtuous woman. Furthermore, the semantics of orange are so opaque in the contexts "Oranges and lemons say the bells of St.Clement's" (Tommy Thumb's Pretty Song Book, 1744) and "A Clockwork Orange" (Burgess, 1962) that it is questionable whether there are word senses that cover the uses of orange here. Indeed, some researchers doubt the veracity of the concept of word sense altogether. Kilgarriff (1997) makes a case for word sense not being a "workable basic unit of meaning", stating that "word senses are only ever defined relative to a set of interests" (Kilgarriff, 1997, p.25). However, this does not seem to resolve the issue, as a definition of the meaning of the set of interests would have to be made in order to define the senses of the words used. Whether or not one believes in word senses (and whatever cognitive vade mecum is used to align orthography with meaning), without recourse to ostensive or dictionary definitions the meanings of polysemous words can only be defined through use, as Kilgarriff concludes:

> "Where 'word senses' have a role to play in scientific[2] vocabulary, they are to be construed as abstractions over clusters of word usages."
> (Kilgarriff, 1997, p.25).

Word Sense Induction does exactly this, using "clusters of word usages" to identify word senses. This is in stark contrast to the related NLP task of Word Sense Disambiguation (WSD) which relies on a fixed list of senses. In WSD, the sense a word takes in a particular context can only be disambiguated to one that is contained in a lexicographic resource: a dictionary, thesaurus, or ontology; typically, WordNet (Miller et al., 1990) is used. Effectively, if a sense is not listed in the resource, the sense does not exist. Rare senses or those specific to a particular domain are often left undefined in these resources and even very common senses may be missing. For example, apple in its corporation sense is not listed in WordNet; therefore, apple in the context "Apple holding more cash than USA"[3] could not be disambiguated by a WSD system that is reliant on WordNet's sense inventory. Sense distinctions in inventories can also be too fine-grained, with senses that should be merged given separate entries. For example, two senses of orange defined in WordNet (3.0) are:

1. An orange colour or pigment.
2. Any pigment producing the orange colour.

A WSD system using WordNet as its sense inventory and classifying orange in the test instance [Kate paints the walls orange.] as sense 1 when an evaluation set tags it as sense 2

[2] By which Kilgarriff means the scientific study of all language not solely the lexis used in science writing.
[3] http://www.bbc.co.uk/news/technology-14340470 accessed on: 31/07/2011

would therefore 'fail' to disambiguate orange. Furthermore, sense inventories cannot keep pace with the extent to which new senses and new words are introduced into language. For example, new senses of the words *epic* (*particularly impressive or remarkable; excellent, outstanding, 'awesome'*) and *tweet* (*to post a message, item of information, etc. on Twitter*) were only added to the Oxford English Dictionary (OED) in June, 2013, though these senses have been in common use for some time[4]. Similarly, 'new' words, such as *zhoosh* (*make more exciting, lively, or attractive*) and *folktronica* (*a style of popular music incorporating elements of folk and electronic music*), though in use for some time, are relatively new entries in the OED[5]. Véronis (2004) also shows that annotators have difficulties in disambiguating words using a fixed list of senses (the *Petit Larousse* dictionary[6]). Véronis found that the surface clues/cues in the dictionary glosses were unclear, leaving annotators vague as to the correct sense they should assign to test examples. Tuggy (1993) suggests that vagueness is the natural condition of sense, where word senses "lie on a continuum between i.) clear cut cases of ambiguity and ii.) vagueness where clear cut boundaries do not hold" (Tuggy, 1993; Erk and McCarthy, 2009), though whether knowing this is more of a hindrance than a help is debatable. If, as Erk and McCarthy suggest: "it seems that a more complex representation of word sense is needed with a softer, graded representation of meaning rather than a fixed listing of senses"(Erk and McCarthy, 2009; Cruse, 2000), then a soft clustering WSI approach (as outlined in Chapter 5) may be one possible representation of this complexity. An induced set of senses may not perfectly align with those defined in sense inventories; however, non-alignment with a fixed list of senses does not necessarily invalidate these senses. As Kilgarriff states: "word senses only exist relative to a task" (Kilgarriff, 1997, p.1). It might even be suggested that each use of word has a unique sense (unique task) for every contextually disjoint unit it appears in. For example, the use of the word orange in a previously unseen context is always a 'new sense' of orange and it is only by finding some part analogue with previously encountered uses of orange that a degree of meaning can be assigned; naturally though, a question arises as to "whether you can make words mean so many different things" (Carroll, 1865). Even 'nonsensical' sentences such as Chomky's infamous "Colorless green ideas sleep furiously" and "Furiously sleep ideas green colorless" (Chomsky, 1965) can be made sense of. Chomsky's view is that both sentences are complete nonsense[7]. However, the words do make sense, *relative to the task* (their task being to be compositionally nonsensical). Further, the words make sense regardless of the task. Indeed, the semantic composition of the second, ungrammatical, sentence makes as much sense as the first if a figurative reading is applied. As a counter argument to Chomsky's 'nonsensical' examples take:

> "the hunched, courters'-and-rabbits' wood limping invisible down to the sloeblack, slow, black, crowblack, fishingboatbobbing sea"
> *Under Milk Wood* (Thomas, 1954, p.1).

[4]http://public.oed.com/the-oed-today/recent-updates-to-the-oed/june-2013-update
[5]Accessed on 26/06/2013.
[6]`http://www.editions-larousse.fr`
[7]Chomsky's point is primarily concerned with grammar, however, he clearly stresses the nonsensical nature of these sentences.

Though a wood cannot hunch, nor limp, and a sea cannot of itself be fishingboatbobbing, this is not nonsense; these lines make perfect sense.

Many current attempts at defining meaning remain influenced by the ideas set out in Frege (1892) and Russell (1905), having a tendency to view language as if it were a set of absolute and universal truths. Meaning thus becomes centred on denotation (reference), with application of some type of logic applied in an attempt to neatly box-up language into tractable units of meaning (van Benthem and Meulen, 2010). This has resulted in large parts of meaning in language being passed over: its connotations, tropes, and prosody[8]. Certainly, disambiguation of the referents in, and semantic composition of, sentences such as: [The cat sat on the mat.] is required, but equally so is an understanding of non-literal sense, as in: [The cat that got the cream.]. Accordingly, a computational solution to the problem of inducing word senses would, ideally, require a model of both prosaic, literal sense and the more enigmatic, figurative readings of sense derived through analogical reasoning.

6.2 Word Sense Induction

Word Sense Induction (WSI) is the task of automatically discovering word senses (uses[9]) from text. WSI makes no recourse to sense annotated corpora nor to training exemplars. In contrast to Word Sense Disambiguation (WSD) – a supervised classification task – WSI is an unsupervised clustering task. WSD systems classify target word senses using some external source of knowledge; for example, by matching target word contexts to those in a sense tagged corpus, whereas WSI systems cluster contexts of a target word. Each cluster is taken to represent a particular use – a sense – of the target word, with clusters then used to tag contexts with senses of the target word.

A WSI system has utility for any NLP application requiring word sense distinctions, notably in applications where word senses cannot be defined by a WSD approach (cf. Chapter 1). Indeed, without recourse to a 'look-up' sense inventory, or to ostensive or training examples of word sense, induction may be the only option available. In principle, WSI would allow language to be monitored: new meanings that existing words take or shifts in their existing meanings could be identified and *neologisms* (new words introduced into language) could be automatically assigned senses. Thus, WSI has the potential to assist the lexicographer's current task of having to manually identify, collate, and exemplify word senses: an enormous undertaking, given both the number of existing senses and the rate at which new senses are introduced into language.

The differences between a WSD and a WSI approach to sense discrimination are further outlined in the following sections.

[8] Hillis (1988) and Dyson (1998) present interesting theories, related to prosody, on the origins of word sense. Both theories propose that 'music made mind'; that is, meaning has its origin in song patterns.

[9] In WSI, use is preferred over sense.

6.2.1 Word Sense Disambiguation Systems

A Word Sense Disambiguation (WSD) system typically requires hand tagged examples of word sense (Navigli, 2009). Existing corpora of this type, such as *SemCor* (Landes et al., 1998), are extremely small compared to the vast amount of untagged text available to Word Sense Induction systems. Hand tagging text is time consuming and laborious, and is generally reliant upon a small number of annotators whose judgement of word sense may not necessarily match those of others. Further, word senses in WSD are drawn from sense inventories such as WordNet (Miller et al., 1990; Fellbaum, 1998) and OntoNotes (Hovy et al., 2006). These inventories are often incomplete, containing only a fraction of the senses words take. Thus, the number of senses a WSD system can assign to a word is limited to those defined in the inventory the system uses. Furthermore, sense inventories date: once defined, a sense of a word is set in 'semantic aspic', thus any semantic shift the sense accrues through its use in language is unavailable to a WSD system.

6.2.2 Word Sense Induction Systems

A Word Sense Induction (WSI) system makes no recourse to sense-tagged training data nor to a fixed list of sense definitions[10], word senses are induced directly from text. Various approaches to WSI exist, with surveys of these approaches found in Navigli (2009), Apidianaki and Van de Cruys (2011), and Navigli (2012). This section summarises the main approaches to WSI that have been proposed in the literature; a graph-based approach to WSI, as applied in this thesis, is introduced separately in Section 6.2.3.

Vector Clustering: This approach uses a vector space model (VSM) to induce word senses (Salton and McGill, 1983). Contexts of a given target word (the senses of which are to be induced) are represented as vectors. Contexts may be represented as either first order vectors, modelling contexts directly (Pedersen and Bruce, 1997), or second order vectors, representing word co-occurrence in contexts (Schütze, 1998). For example, in Schütze (1998) each word w occurring in a target word context is represented as a vector $\vec{w}$ in n dimensional Euclidean space, where n in this case is the number of nouns occurring in the target word's contexts. Each $[1, ..., n]$ element e_i in $\vec{w}$ stores a value v_i which quantifies the significance of w co-occurring with word w_i in target word contexts; here, v_i is the co-occurrence count of w and w_i. Each target word context is then represented as the *centroid* (normalised average) of the vectors of the words in the context. Context vectors are then clustered, with those most similar to each other (defined by measuring the cosine between vector pairs) assigned to the same cluster. Each cluster is then taken to represent a sense of the target word[11].

Word Clustering: In Pantel and Lin (2002) words are clustered using the *Clustering By Committee* (*CBC*) algorithm. *CBC* applies a notion of 'committees' (small, tight clusters

[10] In practice, a sense inventory is required for evaluation purposes.

[11] The outline given here is a basic overview of a VSM approach to WSI, see: Manning and Schütze (1999) and Widdows (2004) for further detail.

of words with high pair-wise similarity) to induce word senses. The intuition here is that each committee will clearly demarcate a unique word sense. Non committee words are then assigned to committees, with the committee a word is assigned to taken to represent a sense of the word.

Probabilistic Clustering: Brody and Lapata (2009) places sense induction in a probabilistic setting, modelling context words around a target word as samples from a multinomial sense distribution. Context words are then generated according to this distribution, with word senses identified by their different word distributions.

Latent Semantic Word Spaces: In Van de Cruys and Apidianaki (2011) words and contexts are mapped to a limited number of topical dimensions in latent semantic word space (Landauer and Dumais, 1997; Van de Cruys, 2008). Word senses are then identified by their associations with particular topical dimensions.

6.2.3 Graph-Based Word Sense Induction Systems

An alternative approach to those outlined above uses a graph model of word co-occurrence to induce word senses. WSI systems applying this approach include: Dorow and Widdows (2003b); Véronis (2004); Agirre et al. (2006b); Biemann (2007); Navigli and Crisafulli (2010); Di Marco and Navigli (2013). These systems use vertices to represent words found in target word contexts; WSI systems using collocations (pairs or triplets of words) as vertices are presented in Dorow et al. (2005); Bordag (2006); Klapaftis and Manandhar (2008, 2010a).

An Example of a Graph-Based WSI System: A graph is constructed for each target word of interest. A graph $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq V \times V$. Each vertex $v \in V$ represents a word that occurs in target word contexts. Target word contexts may be sentences, paragraphs, or some pre-defined context window in which the target word occurs; alternatively, 'contexts' may be words that have a grammatical relationship with the target word. Each edge $e\{u, v\} \in E$ is a pair of vertices. An edge $e\{u, v\}$ represents a symmetrical relationship between words u and v; here, that u and v co-occur in the contexts of a target word. A weight $w\{u, v\}$ is assigned to edge $e\{u, v\}$ quantifying the significance of u, v word co-occurrence. For example, $w\{u, v\}$ might simply be the number of times u and v co-occur in target word contexts. In this thesis $w\{u, v\}$ is an association measure score indicating the significance of u, v co-occurrence. Figure 6.1 shows an example of an edge-weighted graph for the target word *orange*.

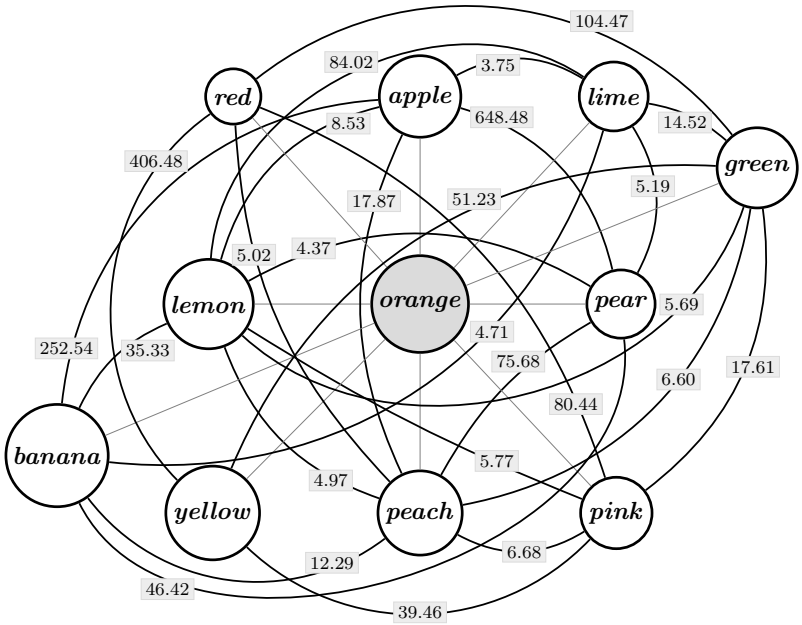


Figure 6.1: G_{orange}.

In Figure 6.1 vertices adjacent to *orange* represent words that occur in *orange*'s contexts. Edges drawn between vertex pairs indicate that the word pair co-occurs in *orange*'s contexts. Edge weights in G_{orange} are values returned by the Log Likelihood Ratio (LLR) (Dunning, 1993), used here as a measure of the significance of word pair co-occurrence.

Senses of the target word *orange* are induced by applying a clustering algorithm to partition G_{orange} to subgraphs (clusters). For example, applying the *MaxMax* clustering algorithm (proposed in Chapter 5) to G_{orange} returns the clustering solution $C_{G_{orange}}$ shown in Fig. 6.2.

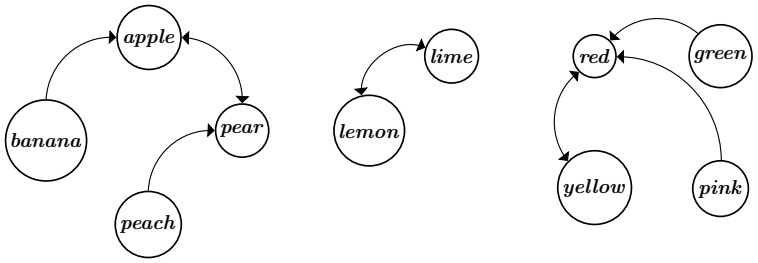


Figure 6.2: $C_{G_{orange}}$, a clustering solution for the graph G_{orange} shown in Fig. 6.1. Arrow direction indicates the highest edge weight for each each vertex $v \in G_{orange}$.

A cluster is then assigned a sense of *orange*, either by mapping the cluster to a sense listed in an inventory such as WordNet (Pantel and Lin, 2002; Widdows, 2004), or to a gold standard class (Biemann, 2007; Agirre and Soroa, 2007; Manandhar et al., 2010). For example, Figure 6.3 shows the clusters of Figure 6.2 labelled with WordNet senses.

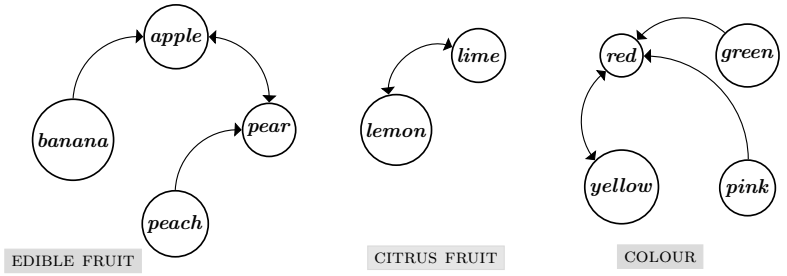


Figure 6.3: Labelling clusters with WordNet senses. Sense labels are obtained using the cluster to sense mapping algorithm proposed in Widdows (2004).

The sense labels shown in Figure 6.3 are obtained using the algorithm presented in Widdows (2004). This algorithm is described in detail in Chapter 8, though can be summarised as follows: label each cluster $c \in C_{G_{orange}}$ with the hypernym in WordNet that subsumes as many of the words in c as possible, as closely as possible. Figure 6.4 gives a simple, reduced, illustration of how the 'fruit' clusters in $C_{G_{orange}}$ acquire their sense labels.

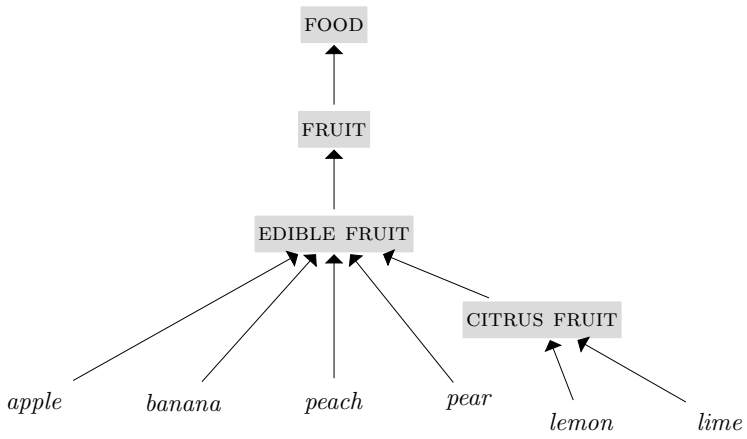


Figure 6.4: Labelling the 'fruit' clusters in $C_{G_{orange}}$.

Cluster labels are then used to tag contexts of the target word. One way in which to do this is to count the overlap (intersection) between the words in the context and the words in each cluster of the clustering solution. For example, given the clustering solution shown in Figure 6.3, orange in the context [orange, apple, and pear] would be assigned its EDIBLE FRUIT sense as this cluster shares two words with the EDIBLE FRUIT cluster whereas the CITRUS FRUIT and COLOUR clusters have no words in common with the context.

6.2.3.1 Discussion

Graph-based approaches to WSI have some affinity with *Existential Graphs* (Peirce, 1903) and *Semantic/Conceptual Networks* (Collins and Quillian, 1969; Collins and Loftus, 1975; Sowa, 1976, 1992). In particular, the application of $n^{th.}$ order neighbours to identify senses (as outlined in Chapter 2) is similar in theme to the notion of *spreading activation* (Collins and Loftus, 1975). However, the graph-based methods advanced in these works require either some form of logic to ground meaning or a notion of prototype to categorise meaning. In this thesis, graphs are applied in their original form as "The solution of a

problem relating to the geometry of position" (Euler, 1741)[12]. Each vertex (word) in a graph has a position that is relative to other vertices in the graph, with this topology used to demarcate word senses. This approach to WSI has advantages over others. For example, Véronis finds, in replicating Schütze's evaluation (Schütze, 1998), that a vector space model (VSM) approach to WSI fails to discover the less predominant senses of target words, whereas the graph-based approach Véronis applies (*HyperLex*) is able to do so. Véronis finds that:

> "Vector-based techniques come up against a major and highly crippling problem: large frequency differences between the uses of the same word cause most of the useful [sense] distinctions below the model's noise threshold to be thrown out."
> (Véronis, 2004, p.2)

In other words, the vector space model fails to isolate all senses of target words as it conflates (merges) senses within vectors, a finding also noted in Klapaftis (2008) and in Navigli (2009). Navigli also notes that this issue of sense conflation is evident in Pantel and Lin's *Clustering By Committee* approach to WSI where "committees conflate senses as each word belongs to a single committee" (Navigli, 2009, p.28). A *Latent Semantic Word Spaces* (Van de Cruys and Apidianaki, 2011) approach to WSI can also fail to find senses of target words as the dimensionality reduction technique applied, which aims to remove information that is irrelevant to the problem space, can lead to information that is pertinent to finding rarer senses being discarded. In a graph-based model of word sense all information can be retained, with measures of vertex cohesion used to find cohesive subgraphs of variable sizes; thus, both predominant (large cohesive clusters) and rare (small cohesive clusters) senses of a target word can be identified. A graph-based model using a soft clustering algorithm can assign words to more than one cluster, thus polysemous context words are able to contribute to the definition of more than one sense of a target word. Modelling WSI as a graph also allows the problem space to be viewed at various levels of granularity: as single vertices, first order neighbours, second order neighbours, and so on, right up to a global view of the entire graph. Thus, the problem space of WSI can be visualised in concrete terms whereas abstraction of the problem space to a vector or matrix representation is far less transparent.

[12] http://eulerarchive.maa.org/docs/originals/E053.pdf accessed on 1/8/2011

Chapter 7

A Review of Existing Approaches to Word Sense Induction

7.1 Introduction

This chapter reviews a number of approaches to Word Sense Induction (WSI) that have been proposed in the literature. The approaches described here are those considered in this thesis to be key for WSI, with several of the methods applied in these approaches extended or adapted in the approaches that are introduced in this thesis: all apply a graph-based approach to WSI; all apply lexical co-occurrence graphs of some type, accordingly, graphs of this type are first defined.

7.1.1 Lexical Co-occurrence Graphs

A lexical co-occurrence graph represents co-occurrence relationships between lexical entities. Typically, lexical entities are words, though other entities, for example collocations, are used. Lexical co-occurrence graphs can be global or local. A global lexical co-occurrence graph, such as the WordNet graph used in Dorow (2007), contains all co-occurrence relationships for all target words. A local co-occurrence graph, as is typically used in this thesis, contains one target word and some predefined neighbouring space of the target word. Given a target word, a text containing instances of the target word, and some predefined unit of context (e.g. sentences), each word co-occurring with the target word within a context is represented in the lexical co-occurrence graph as a vertex. Two vertices are linked in the graph by an edge if the two words represented by the vertices appear within the same context. For example, given an initially empty lexical co-occurrence graph G and the context:

[oranges are not the only fruit],

if nouns are of interest, then the words orange(s) and fruit are represented as vertices in G: v_{orange} and v_{fruit} with an edge set between the two vertices. Each edge is weighted, with an edge weight quantifying the association strength between two vertices. For example, co-occurrence counts may be used to denote association strength, thus if orange and fruit

co-occur n times within contexts, the edge weight $w(v_{\text{orange}}, v_{\text{fruit}})$ is set to n. A lexical co-occurrence graph is the input to a WSI system, with the graph partitioned using a graph-based clustering algorithm and each cluster taken to represent a word sense.

7.2 Dorow and Widdows

7.2.1 Dorow and Widdows (2003a)

In Dorow and Widdows (2003a), a graph model of word co-occurrence is applied to induce word senses from part of speech tagged text. Induced senses are then mapped onto WordNet's hypernym-hyponym structure in order to label each induced sense with a WordNet sense (though any taxonomy with a hypernym-hyponym structure could be used). The model is applied directly to the text to be disambiguated, thus may be applied to domain specific text to induce specialised senses that are undefined in standard dictionaries.

Given a target word *tw*, the senses of which are to be induced, noun co-ordination patterns in which *tw* appears are extracted from the British National Corpus (BNC). For example, if the target word is *orange* then contexts such as [pink, orange, red], [orange, lemon, and lime] and [apple or orange] are extracted; the rationale being that words in such pattern are semantically related. The context words: pink, red, lemon, lime, apple are the first order neighbours of *orange*. The same process is applied to the first order neighbours of *orange*, finding, for example, that pink has the first order neighbours: red, violet, fuchsia and that apple has the first order neighbours: banana, pear, lemon. The first order neighbours of the neighbours of *orange* form the set of second order neighbours of *orange*. This data is used to construct a lexical co-occurrence graph in which each vertex is connected to its top n neighbours (based on co-occurrence frequency in noun co-ordination patterns).

Dorow and Widdows construct a lexical co-occurrence graph G_{tw} for each target word *tw* considered. G_{tw} consists of *tw*, the top twenty first order neighbours of *tw*, and the top ten second order neighbours of *tw*. G_{tw} is then pruned, removing *tw* and any neighbouring vertices with weak connectivity[1]. A clustering algorithm is then applied to partition G_{tw}, with each partition (cluster) taken to represent a sense of *tw*. Dorow and Widdows hypothesise that a target word, if ambiguous, acts as a hub in the graph, connecting unrelated senses in G_{tw}. By removing the target word from G_{tw}, the semantic space around *tw* separates out into distinct contextual uses, taken to represent distinct senses of *tw*.

Dorow and Widdows use a Markov model based algorithm, *MCL* (Markov Cluster Algorithm) (van Dongen, 2000), to cluster G_{tw}. The premise of *MCL* is that random walks originating from each vertex in a graph will tend to stay within the remit of highly connected regions of the graph. *MCL* simulates flow (Bollobás, 1998; Heineman et al., 2008) by iteratively recomputing transition probabilities between graph vertices using two parameters: expansion and inflation. Expansion allows vertices to see new neighbours;

[1] It is not entirely clear what weak means here. Presumably, some threshold is applied with respect to the number of neighbours a vertex has, with vertices having few neighbours considered weak.

inflation promotes 'popular' (strongly connected) neighbours. In dense regions of a graph, flow becomes concentrated through expansion and inflation. As flow dissipates the transition probabilities converge, with the limiting matrix containing the transition probabilities taken as the clustering of the graph.

Dorow and Widdows find that *MCL* is highly affected by the skew in sense predominance and by the granularity of the senses of target words. In particular, the size of the target word graph affects *MCL*'s induction process: if the target word graph is too small, senses become conflated; if the graph is too large, noise is introduced. Dorow and Widdows conclude that tuning the expansion and inflation parameters of *MCL* in order to find all senses of all target words is infeasible as target word senses may be of varying granularity and target word graphs will vary in size. Notably, Dorow and Widdows show that *MCL* cannot find rare senses of target words. The solution to these issues is to apply the method proposed in Pantel and Lin (2002), using an initially fixed sized lexical co-occurrence graph G_{tw} (as outlined above) and in each iteration of the method apply *MCL* to G_{tw} and select the single most distinctive cluster in G_{tw}: the cluster with the strongest connection to *tw*. This cluster is removed from G_{tw} with *MCL* then reapplied to the transformed graph. This process iterates until the similarity between *tw* and its closest neighbour falls below a set threshold.

Dorow and Widdows evaluate this induction system on a set of target nouns with varying degrees of ambiguity, from homonyms such as *arms* to words having systemic polysemy (Pustejovsky, 1991) such as *cherry*. The authors consider just the top two sense clusters in their evaluation, applying a cluster labelling algorithm (Widdows, 2003) to assign WordNet senses to clusters. The sense assigned to a cluster is the sense (synset) in WordNet that subsumes as many of the words in the cluster as possible as closely as possible (Widdows, 2004, 2003). For example, given the target word *cherry*, the system finds the senses shown in Table 7.1.

Clusters	**Senses** (WordNet 1.7.1)
cedar, larch, mahogany, water, sycamore, lime, teak, ash, hornbeam, oak, walnut, hazel, pine, beech, alder, thorn, poplar, birch, chestnut, blackthorn, spruce, holly, yew, laurel, maple, elm, fir, hawthorn, willow	WOOD
bacon, cream, honey, pie, grape, blackcurrant, cake, banana	FOODSTUFF

Table 7.1: Two senses of *cherry* as defined in Dorow and Widdows (2003a).

No formal evaluation against a baseline or gold standard is given by Dorow and Widdows. In this evaluation context, WordNet is used to label clusters with senses and the judgement as to whether a sense label is a valid sense of a target word is left to the reader to decide.

7.2.2 Dorow et al. (2005)

Dorow et al. (2005) presents two methods for clustering semantically related words. The first method results in a hard clustering of words; the second, a soft clustering. Hard clustering places each word into a single cluster; soft clustering allows ambiguous words, which may connect semantically disjoint subspaces of the graph, to belong to more than one cluster.

A lexical co-occurrence graph is constructed using the noun co-ordination patterns described in the previous section. Thus, given the pattern [apple, banana, and pear], vertices apple, banana, pear, along with the undirected edges $e_{\{\text{apple, banana}\}}$, $e_{\{\text{apple, pear}\}}$, and $e_{\{\text{banana, pear}\}}$ are added to the graph. The graph is then trimmed to remove any edge that is not in a triangle. Triangles, as outlined in Chapters 2 and 3, are indicative of semantic cohesion between words.

The first method applies the concept of *curvature* (Eckmann and Moses, 2002) to find vertices in the graph with weak connectivity. Curvature, as a measure, is equivalent to the unweighted local clustering coefficient defined in Watts and Strogatz (1998) and described in Chapter 3 of this thesis. Given a vertex v and its neighbourhood $\mathcal{N}(v)$, the curvature of v is the number of triangles in $\mathcal{N}(v)$ that v is a member of divided by the number of triangles v could be a member of. A vertex with low curvature is embedded within a weakly connected subspace of the graph. From a semantic perspective, such a vertex is said to have high ambiguity. A vertex with high curvature is at the centre of a strongly connected neighbourhood, implying that the vertex is the hub of a subspace of the graph with strong semantic cohesion. Vertices with a curvature less than 0.5 are deleted from the graph[2]. This splits the graph into a hard clustering of semantically related words. Each cluster is then expanded to include the semantically fuzzy, low curvature, vertices that link to the cluster. An example cluster, taken from Dorow et al. (2005), is:

{**pomelo**, **papaya**, *banana*, *potato*, *pineapple*, *mango*, *peach*, *palm*, *pear*, *parsnip*}.

The words {**pomelo**, **papaya**} are the core cluster members, with {*banana*, *potato*, *pineapple*, *mango*, *peach*, *palm*, *pear*, *parsnip*} added in the expansion step. As Dorow et al. note, the core words denote a specific word sense, with the added words often of a more ambiguous nature. Thus, **papaya** is most certainly a fruit whereas *palm* can be: a food/fruit (as in hearts of palm), a plant, or a body part.

The second method clusters graph edges rather than graph nodes. The rationale given is that a clustering of vertex pairs (edges) will contain a greater degree of semantic relatedness than a clustering of single vertices: a strategy based on Yarowsky's "one sense per collocation" tenet (Yarowsky, 1995, p.189)[3]. Given a lexical co-occurrence graph G, a transformation of G to an edge graph G_e is made by introducing an edge vertex $v_{e_{\{i,j\}}}$ for each edge $e_{\{i,j\}}$ in G. An edge vertex is linked to another edge vertex in G_e if the two edges in the original graph G appear in a triangle. For example, given the two edge vertices $v_{e_{\{i,j\}}}$ and $v_{e_{\{j,k\}}}$, if edge $e_{\{i,k\}}$ exists in G then the two edge vertices in G_e are

[2]It is not clear why the authors set the threshold at 0.5; indeed, Dorow (2007) goes to some length to establish a value of 0.35 as the most suitable value to apply.

[3]A highly similar method is described in Klapaftis (2008).

linked. This graph is then clustered using the *MCL* algorithm. By using edges as vertices, words can be members of more than one cluster. As a result, the hard clustering output of *MCL* actually produces a soft clustering of the vertices in G_e. At this stage, the clusters are found to be too fine-grained, thus Dorow et al. merge any two clusters whose overlap exceeds a shared information content value $\geq 50\%$.

The two methods are evaluated within a lexical acquisition/word sense induction context. Evaluation is based on the idea that a target word can be mapped to a hypernym in WordNet using other, semantically related, words (the cluster words), with the hypernym representing the target word's sense. For example, given the target word *orange*, the most similar cluster to *orange* is selected, using cosine similarity. The target word is removed from the cluster along with any morphological variants such as *oranges* and *orangey*. The class labelling algorithm in Widdows (2003) is then applied to find a sense label for the target word. The top five hypernyms are selected, with the best match between *orange* as a hyponym and each candidate hypernym set as the sense label. The best matching hypernym is that 'nearest' to *orange*; that is, the hypernym with the least number of WordNet noun hierarchy levels between itself and the target word. The baseline used for comparison is derived by taking the pair: (*orange*, *orange's most similar neighbour*) and finding the least common subsumer of the pair, this is the hypernym that subsumes both *orange* and *orange's most similar neighbour* as closely as possible; effectively, this hypernym is taken to be the correct sense of *orange*.

Evaluation result show that neither the curvature method nor the edge graph method pass the baseline. This is not surprising, given that the target word itself is used to find the baseline hypernym sense label. The edge graph method was found to outperform the curvature method, returning 50% better sense labelling, achieving a highest accuracy score of 85% at ≤ 6 WordNet noun hierarchy levels, which gives some credence to Dorow et al.'s view that edge graphs are less sense conflating than those of vertex graphs.

7.3 *HyperLex* - Véronis (2004)

Véronis (2004) introduces a hard clustering algorithm, *HyperLex*, which induces word uses[4] in a completely unsupervised manner.

Given a target word *tw* and a corpus of part-of-speech (POS) tagged paragraphs in which *tw* occurs, Véronis constructs a lexical co-occurrence graph G_{tw} consisting of the context words of *tw* (nouns and adjectives only with a frequency ≥ 10). An edge is set between two context words i, j if they co-occur in paragraphs ≥ 5 times. Edges are weighted using: $1 - \max(p(i|j), p(j|i))$, where $p(i|j)$ is the conditional probability of finding word i in a paragraph given that word j is present. Note here, that *lower* edge weight implies a *stronger* association between two words. Edges with weights ≥ 0.9 are discarded.

In preliminary experiments, Véronis applies a generalisation of the clustering coefficient (Watts and Strogatz, 1998) to the weighted case[5] in order to verify his intuition that lexical

[4] Véronis, along with many WSI researchers, prefers to apply the term word use rather than word sense.

[5] Summing the edge weights from *tw* to *tw*'s neighbours; disregarding neighbour to neighbour edge weights

co-occurrence graphs have scale-free and small world properties (Barabási, 2003; Strogatz, 2004). The graphs are found to be scale-free, containing a small number of vertices of high degree (high connectivity) and many of low degree (weak connectivity). The graphs are also found to be small world networks, as defined in Watts and Strogatz (1998), with path lengths in each lexical co-occurrence graph $G_{tw} \approx G_{random}$, a random graph (Erdős and Rényi, 1959) and clustering coefficients $G_{tw} \gg G_{random}$. Véronis also notes that the degree of a vertex is near linear in its correlation with frequency. These findings form the basis of the *HyperLex* algorithm.

HyperLex has two stages. Given a lexical co-occurrence graph G_{tw} for a target word *tw*, the algorithm proceeds as follows. In the first stage, high density components of the graph are detected by finding vertices of high degree[6]. These vertices act as the *hubs* of the senses of *tw*. The algorithm finds the vertex with the (current) highest degree in G_{tw}. This vertex, along with all vertices connected to it, is removed from the graph. This set of words is taken to represent one sense of *tw*. The algorithm continues in this fashion until no more hubs are detected (using a threshold value of 6 most frequent neighbour vertices with an average edge weight ≤ 0.8).

The second stage of the algorithm computes a minimum spanning tree (MST) over the original graph G_{tw}, with *tw* set as the root of the MST and each *hub* node attached under *tw* with maximum edge weight = 0. The MST is used to perform word sense disambiguation. Given a context (paragraph) of *tw*, each context word cw_i is assigned a hub score vector $\vec{hubs}_{cw_i}$. For each hub h_j, a score is computed and stored in $\vec{hubs}_{cw_i}$, the score being the distance between cw_i and h_j in the MST. The hub score vectors for all context words are then summed, with the highest scoring hub assigned as the sense of *tw*.

The algorithm is evaluated using ten polysemous words that are found to be particularly hard for human annotators to disambiguate. Véronis reports Precision at 97% (compared to 73% for the most frequent sense baseline) and Recall at 82%. These are exceptionally good results for a completely unsupervised system. However, only ten words are evaluated, with the evaluation done manually by Véronis. A far more comprehensive and objective evaluation would therefore be required to give credence to the reported performance of *HyperLex*.

7.4 Agirre et al.

Following on from the preliminary investigations of Agirre et al. (2006a) and the evaluation results in Agirre et al. (2006b), where a parameter optimised version of *HyperLex* was found to be competitive against supervised systems[7], Agirre et al. propose a graph-based system for word sense induction and word classification (Agirre et al., 2007).

Given a target word and a corpus consisting of contexts (paragraphs) in which the target word appears, the system applies a two stage clustering process to induce the

[6] Véronis, noting the linear correlation with frequency, actually uses frequency for computational efficiency reasons.

[7] Agirre et al. (2007) also find that a weighted generalisation (Mihalcea et al., 2004b; Mihalcea and Radev, 2011) of the *PageRank* algorithm (Brin and Page, 1998) returns results comparable to those of *HyperLex*.

senses of the target word. In the first stage, a lexical co-occurrence graph is constructed from the target word corpus. This graph is then clustered, using *HyperLex* for the noun target words and the *HITS* (Hypertext Induced Topic Selection) algorithm (Kleinberg, 1999) for the verb target words[8]. *HITS* determines two values for a vertex v: $a(v)$, the authority of v and $h(v)$, the hub value of v. Both values are defined through mutual recursion where $a(v) = \sum_{(i,v)\in E} h(i)$ and $h(v) = \sum_{(v,i)\in E} a(i)$. A vertex is said to be a good hub if it points to many good authorities, and a good authority if it is pointed to by many good hubs.

The method in Véronis (2004), outlined in the previous section, is applied to the graph, selecting vertices with high degree as representatives of sense clusters and using a minimum spanning tree (MST) to score each context of a target word. Thus, each word cw_i in a context paragraph p_j is assigned a hub score vector $\vec{hubs}_{cw_i}$. Assuming, for expository purposes, that five hubs (five senses) are found and that a hub score vector for a context word cw_i is: $\vec{hubs}_{cw_i} = \{0, 0, 0.5, 0, 0\}$, where 0.5 is the distance in the MST between cw_i and hub_3. As only one element of the hub score vector contains a value greater than zero, cw_i is thus assigned to this single hub (hub_3); a single sense of the target word. As performed in Véronis (2004), the hub score vectors for all words in a context are summed, with the highest scoring hub assigned as the sense of the target word in that context. Agirre et al. find that this method produces many clusters (said to be 'micro-senses' by Agirre and Soroa). This level of sense granularity is too fine to be matched against the evaluation task's fixed list of senses (*OntoNotes*, Hovy et al. (2006)).

In the second stage, the clusters from stage one are merged to produce a coarser-grained set of clusters. A context by context matrix is constructed, with each element of the matrix storing the relatedness score between two contexts. The relatedness score for two contexts p_i and p_j is determined by computing cosine similarity between the normalised hub score vectors of p_i and p_j. This matrix is then pruned, retaining relatedness scores ≥ 0.6. The final, pruned, matrix M may be viewed as a weighted graph G_M in which each context of M is represented as a context vertex with edge weights set as the relatedness score between two context vertices[9]. Finally, G_M is clustered using the *MCL* algorithm (van Dongen, 2000), with each cluster taken to represent a sense of the target word.

The system is evaluated within the framework of the SemEval-2007 WSI task (Agirre and Soroa, 2007)[10]. 100 target words were evaluated: 65 verbs and 35 nouns. Two evaluations were carried out: 1.) a 'supervised' evaluation[11] using the standard clustering measures of F-Score, Purity, and Entropy (defined in Section 7.1.7) and 2.) an unsupervised evaluation in which the organisers of the task (Agirre and Soroa) map the induced senses (clusters) onto *OntoNote* (Hovy et al., 2006) senses.

For the unsupervised evaluation, Agirre et al. claim the highest scoring result, with

[8] For verbs, χ^2 is used to weight edges as the authors found, in preliminary experiments, that conditional probability weighting, as used in *HyperLex*, gave too high a weight to common verbs such as *be* and *use*.

[9] Agirre et al. state that G_M is a directed graph. This is not true: G_M is an undirected (symmetric) weighted graph where $w_{\{i,j\}} = w_{\{j,i\}}$.

[10] The system was also evaluated in the *SemEval-2007 WEPS* (WEb People Search) task (Artiles et al., 2007), returning the worst results see: (Agirre et al., 2007, p. 4).

[11] In the clustering literature this evaluation would be classed as unsupervised.

an F-Score of 78.7 (the average F-Score, all participating systems, is 65.4). However, this result should be tempered against Agirre and Soroa's later report which shows that the five other participating systems return higher Purity and lower Entropy values (Agirre and Soroa, 2007, p.10). Klapaftis (2008) also notes that a high F-Score indicates a bias in favour of clustering solutions which return all instances (contexts) of a target word within a single cluster (Klapaftis, 2008, p.109).

For the supervised evaluation, Agirre et al. report results in terms of Recall. The system was ranked fourth out of the six participating systems with a recall of 78.5, a value lower than both the average recall of 79.1 and that of the most frequent sense (MFS) baseline at 78.7.

7.4.1 Summary and Discussion

Agirre et al.'s system is heavily parametrised, with parameters tuned over Senseval-3 data[12] to: find hubs, prune graphs, merge clusters, and set particular values for the inflation and expansion rates in the *MCL* algorithm.

The system also applies two types of graph clustering algorithm in the first stage: *HyperLex* for nouns and *HITS* for verbs. Different edge weights are set for each algorithm: conditional probabilities in *HyperLex* and χ^2 in *HITS*. The system also uses different features for nouns and verbs, with lemmatised nouns used for noun target words and lemmas of nouns, verbs and adjectives used for verb target words – all of which makes it difficult to assess the capabilities of the system. Agirre and Soroa also have a priori knowledge, as stated in (Agirre et al., 2007), that the gold standard test data contains an average cluster size of 3.6 clusters, thus they manually tune the inflation parameter of the *MCL* algorithm to return between one and four clusters.

7.5 Klapaftis et al.

7.5.1 Klapaftis (2008)

Klapaftis (2008) presents a word sense induction system that combines the edge graph transform method in Dorow et al. (2005) with the hard clustering approach of Véronis (2004). This combination of methods allows a soft clustering of lexical items, in which each lexical item may belong to more than one cluster, thus take more than one sense.

Klapaftis clusters vertex pairs (edges) rather than single vertices. The intuition here, as in Dorow et al. (2005), is that edge clustering is less sense conflating, with connectivity between word pairs less ambiguous than connectivity between single words. As in Véronis (2004), given a target word *tw* and a base corpus *bc* of paragraphs containing *tw*, the aim is to induce the senses of *tw* from *bc*. Klapaftis begins by constructing a lexical co-occurrence graph G_{tw} for *tw* using the contextual data in *bc*. Each edge weight $w_{\{i,j\}} \in G_{tw}$ is set to

[12] Agirre et al. do not state which dataset was used for parameter tuning. Presumably, the data was drawn from the English all words task (Snyder and Palmer, 2004) and/or the English lexical sample task (Mihalcea et al., 2004a).

the average conditional probability[13] between vertices i, j:

$$w_{\{i,j\}} = \frac{p(i|j) + p(j|i)}{2}$$

Vertex pairs with a co-occurrence frequency ≤ 9 and an edge weight ≤ 0.4 are discarded. G_{tw} is then transformed to an edge graph G_e. G_e is clustered using Véronis' method of finding high density components in the graph, each of which is taken to be a representation of a sense of the target word. Clusters are further populated by taking each edge in G_{tw}, not observed in G_e, and assigning it to the cluster to which it has highest similarity. The similarity score between an unclustered edge ue_i and a cluster is the average of the distributional similarity (Jaccard) scores between ue_i and each edge ce_j in the cluster:

$$Jaccard(ue_i, ce_j) = \frac{ue_i \cap ce_j}{ue_i \cup ce_j}$$

where intersection is the number of times the words in the edges ue_i and ce_j co-occur in paragraphs of *bc* and union is the combined occurrence of the edge words in *bc*.

The final, extended clusters are used to sense tag each instance of *tw* (in either *bc* or a test corpus *tc*) with a sense. Given an instance p_{tw} of *tw* (a paragraph in which *tw* occurs), a score is assigned to each induced cluster c_i. This score is the overlap between word pairs in p_{tw} and c_i, with the highest scoring cluster assigned as the sense tag for p_{tw}.

The algorithm is evaluated using Véronis' *HyperLex* model as a baseline. This allows Klapaftis to assess whether edge clustering returns a better induction of word senses than that of a single vertex model. Using the evaluation methodology set out in Agirre and Soroa (2007), Klapaftis found that the algorithm achieved higher supervised Recall values, lower Entropy, and higher Purity than the *HyperLex* model; effectively, producing less sense conflating clusters than *HyperLex*. However, the algorithm returns a much higher number of clusters than the single vertex model, which results in a lower F-Score than that of *HyperLex*. The issue here is that *homogeneity* (maximal homogeneity is where a cluster contains only examples of one sense) increases in line with cluster numbers. This, however, is offset by a reduction in *completeness* (maximal completeness is where all examples of a sense are contained in just one cluster). The problem here, which shall be addressed in Chapter 9 of this thesis, relates to the types of measures used in evaluating WSI systems; as Klapaftis notes, evaluating WSI systems is a difficult task, with the measures typically applied to evaluate clustering algorithms (Purity, Entropy, F-Score) all having particular drawbacks.

7.5.2 Klapaftis and Manandhar (2008)

In Klapaftis and Manandhar (2008), a graph model is applied to word sense induction in which each vertex of the graph is a collocation. Here, a collocation consists of any two words co-occurring within a context of the target word. Edge weights are set to the co-occurrence frequency of the collocations that define the end points of an edge. The

[13] Following Cimiano et al. (2005), where conditional probability was shown to be the best indicator of collocational association.

method is evaluated using the nouns in the SemEval-2007 WSI task (Agirre and Soroa, 2007).

Klapaftis and Manandhar's approach is to use two types of corpora: a base corpus *bc*, consisting of paragraphs in which a target word *tw* appears, and a reference corpus *rc*, the British National Corpus (BNC). Initially, each paragraph in *bc* and *rc* is part-of-speech (POS) tagged using the *GENIA* tagger (Tsuruoka et al., 2005)[14]. Nouns are then extracted from the paragraphs in *bc* and *rc* and lemmatised. Thus, at this stage, each paragraph in *bc* and *rc* is represented as a list of lemmatised nouns. A contingency table is then constructed for each word in *bc*, as illustrated in Table 7.2.

Observed counts	*bc* for *tw*=**network**	*rc*
cable	213	2439
all words	23279	24038639

Table 7.2: Contingency table for target word **network** and context word ***cable***

Dunning's Log-Likelihood Ratio (LLR) measure (Dunning, 1993) is applied to the counts in the contingency table in order to asses whether the context word ***cable*** is a significant word for the target word **network**. In simple terms, if the distribution of ***cable*** is greater in *bc* than *rc* then cable is considered to be a significant word for **network**. Context words with LLR significance values ≤ 4 are discarded. Collocations consisting of two significant words, found within the same paragraphs of *bc*, are extracted. Collocations with a frequency count < 8 and a conditional probability value < 0.2 are discarded. A graph G_c is then constructed in which each vertex is a collocation, with an edge linking two vertices if they appear together within a paragraph of *bc*. At this stage, G_c is sparse due to the low inter-connectivity between collocations. Klapaftis and Manandhar apply a smoothing technique[15] to add new edges between unconnected vertices, using the overlap between word neighbourhoods. Each vertex v is associated with a vertex vector $\vec{v}$ which lists the set of vertices connected to v. The Jaccard measure is applied to find a similarity score between two vertex vectors:

$$Jaccard(\vec{v_i}, \vec{v_j}) = \frac{\vec{v_i} \cap \vec{v_j}}{\vec{v_i} \cup \vec{v_j}}.$$

Two vertices are linked by an edge if they are found to have the highest mutual similarity scores. Additionally, any vertices attached to the unclustered vertex are also connected. Edge weights between vertices i, j are then set using conditional probability: $max(p(i|j), p(j|i))$. The final, extended, graph is then clustered using Biemann's *Chinese Whispers* algorithm (Biemann, 2006a). Finally, each paragraph in *bc* is scored against each cluster returned by *Chinese Whispers*, with the highest scoring cluster assigned as the sense of the target word in the paragraph: the score being the number of collocations in a paragraph matching those in a cluster.

[14]The BNC is already POS-tagged; here, *GENIA* is used to retag the BNC for compatibility with the base corpus.

[15]Based on the method in Cimiano et al. (2005)

The method is evaluated using all 35 nouns in the SemEval-2007 WSI task (Agirre and Soroa, 2007). Two versions of the algorithm were evaluated: one with the Jaccard smoothing; one without. For the unsupervised task, Klapaftis and Manandhar found that the smoothed version performed better than the non-smoothed version, however both algorithms returned lower F-Scores than a baseline single vertex model. Both algorithms performed far better in the supervised task, outperforming the other participate systems, a result Klapaftis and Manandhar believe is due to the algorithm's ability to produce less sense conflating clusters.

7.5.3 Summary and Discussion

As in Agirre et al. (2007), the approaches outlined in this section require parametrisation. The clustering algorithm applied, *HyperLex*, is a parametrised model, requiring the user to set the hub (cluster) size. Thus, the user is required to tune a system that incorporates *HyperLex* to the data it is applied to. This appears to rather defeat the purpose of inducing word senses, as words have varying and unknown numbers of senses, with different levels of sense granularity. In particular, the second approach in Klapaftis and Manandhar (2008) requires a variety of different methods in order to construct a single system: a smoothing process to reintroduce lexis, various edge weight definitions, and parameter tuning to find optimal thresholds. That said, this system was shown to be the best performing system in the supervised task set in SemEval-2007. Concerning collocation vertices, a question might be put: if word pairs are used, why not triplets? The point being, which n-connectivity linking of words works best: two, three, four etc.? There is, however, an issue here as to where an association between words ends and the clustering of whole context examples begins, along with the issue of data sparseness. Furthermore, given 'the right' clustering algorithm, it could be argued that collocation vertices are not required. That is, if a clustering algorithm uses the edge weights between single vertices in the right manner then edge weights will account for the collocational aspects of word pairings.

7.6 *SquaT++* and *B-MST* - Di Marco and Navigli (2013)

Two novel graph-based WSI algorithms are presented in Di Marco and Navigli (2013): *SquaT++* and *Balanced Maximum Spanning Tree* (*B-MST*). The algorithms are evaluated in a Web search result task where the aim is to: 1.) automatically induce the senses of each search query q in a test set using a search query word co-occurrence graph G_q; 2.) return a ranked, diversified set of senses for each search query q.

7.6.1 *SquaT++*

SquaT++ is an extension of the *SquaT* algorithm that was proposed in Navigli and Crisafulli (2010). *SquaT++* exploits three graph patterns, namely: *Triangles* (cycles of length 3, equivalent to the clustering coefficient measure described in Chapter 3); *Squares* (cycles of length 4); *Diamonds* (subgraphs with 4 vertices and 5 edges that form a square with a diagonal). The strength of each pattern is measured as follows, where v is a vertex that

represents a word in a search query word co-occurrence graph G_q:

$$Triangles(v) = \frac{\text{triangles } v \text{ participates in}}{\text{triangles } v \text{ could participate in}},$$

$$Squares(v) = \frac{\text{squares } v \text{ participates in}}{\text{squares } v \text{ could participate in}},$$

$$Diamonds(v) = \frac{\text{diamonds } v \text{ participates in}}{\text{diamonds } v \text{ could participate in}}.$$

The three measures are then combined to produce the *SquaT++* score for v:

$$SquaT{+}{+}(v) = \alpha \cdot Triangles(v) + \beta \cdot Squares(v) + \gamma \cdot Diamonds(v)$$

where $\alpha+\beta+\gamma = 1$. The clustering solution is obtained by removing vertices in G_q whose *SquaT++* value is below a threshold σ and returning the remaining connected components in G_q.

7.6.2 *B-MST*

B-MST is a variant of the *Maximum Spanning Tree* (*MST*) algorithm that was introduced in Di Marco and Navigli (2011). The aim of *B-MST* is to balance the number of word co-occurrences in each sense cluster. The algorithm first computes the *maximal spanning tree* of a word co-occurrence graph G_q. A *spanning tree* of a graph with n vertices is a subset of $n-1$ edges of the graph that form a tree. A maximal spanning tree is a spanning tree of a weighted graph that has maximal edge weight. Clusters are obtained by iteratively removing the edges in G_q which represent structurally weak relations: edges with low weight. The *B-MST* algorithm can be summarised as follows:

1. Remove vertices from G_q whose degree is 1.
2. Compute the maximum spanning tree TG_q of G_q.
3. Calculate cluster mean cardinality by dividing the number of vertices in G_q by the number N, where N is the number of clusters one wants to return (note that N is a user defined parameter).
4. Remove each edge $e \in TG_q$ if the removal of e does not lead to clusters with cardinality less than half of the cluster mean cardinality.

The condition in step 4 prevents the algorithm returning: 1.) very small cluster; 2.) clusters that are all of equal size.

7.6.3 A Web Search Result Clustering Task

SquaT++ and *B-MST* are evaluated in a Web search result task (Di Marco and Navigli, 2013), with results compared to those of three representative clustering algorithms, namely: *Chinese Whispers* (described in Chapter 5); *HyperLex* (described in Section 7.3, and *Curvature* clustering (described in Section 7.2.2). Given a test set of search queries,

this task requires a participant to induce the senses of each search query and then cluster search results for the query according to their semantic similarity to the induced word senses. Two test sets are used: AMBIENT (AMBIguous ENTries)[16], containing single word queries, and MORESQUE (MORE Sense-tagged QUEry results)[17], containing queries of 2, 3, and 4 words. Each query q is associated with the top 100 results returned for q by the *Yahoo!* search engine, with the web snippet text of these results manually annotated with a WordNet sense of q. A word co-occurrence graph $G_q = (V, E)$ is constructed for each test set search query q, using data extracted from either the Google Web 1T corpus[18] or the ukWaC corpus[19] (depending upon the particular evaluation applied). V is the set of words co-occurring with q along with the first and/or second order neighbours of these words (depending upon the evaluation applied). E is the set of undirected edges, denoting co-occurrence of word pairs in V. Edge weights are Dice coefficient scores (Tan et al., 2006), thresholded to keep only the highest ranking word co-occurrence pairs. A clustering algorithm, e.g. $SquaT++$, is then applied to G_q, returning a set of sense clusters C_{G_q}. Similarity between queries and clusters is measured using three different overlap measure, with clusters ranked by their similarity scores. This process (which is highly summarised here) returns a ranked and diversified clustering of search results: a list of heterogeneous results in which search results that are similar to those at the top of the ranking are prevented from ranking too highly in the list.

Three evaluations of the clustering algorithms are performed: two extrinsic (unsupervised) evaluations; the third, a qualitative (manual) evaluation. The extrinsic evaluations use three evaluation metrics: Adjusted Rand Index, Jaccard Index (pairwise evaluation metrics), and F-Score (Tan et al., 2006). The first extrinsic evaluation assesses the quality of search result clusters. For pairwise measures, *SquaT++* is shown to outperform all other algorithms on both corpora (Google Web 1T and ukWaC), with *Chinese Whispers* ranked second, *Curvature* third, and *B-MST* and *HyperLex* obtaining lower results. For F-Score, an inverse trend is observed: *HyperLex*, *B-MST* and, to a lesser extent, *Chinese Whispers* achieve the best results, whereas *Curvature* and *SquaT++* obtain lower F-Scores. Di Marco and Navigli state that this is a direct result of *HyperLex*, *B-MST* and *Chinese Whispers* generating greater numbers of clusters than *SquaT++* and *Curvature*, allowing these three algorithms to have a greater chance of obtaining higher recall. Di Marco and Navigli (2013) suggest that this result implies that *HyperLex*, *B-MST* and *Chinese Whispers* better diversify among the topics of the retrieved search results. The second extrinsic evaluation assesses the degree of diversification: how many different meanings of a query are covered in the top ranking results. Results for this evaluation show that exploiting local graph patterns, as *Curvature* and *SquaT++* do, typically leads to worse results than the three algorithms that apply a global (whole graph) approach to clustering. Interestingly, *Curvature* and *SquaT++* return similar results, implying that the *Triangle* measure is the key graph pattern in the *SquaT++* algorithm. Di Marco and Navigli (2013)

[16] `http://credo.fub.it/ambient`
[17] `http://lcl.uniroma1.it/moresque`
[18] `http://catalog.ldc.upenn.edu/LDC2006T13`
[19] (`http://wacky.sslmit.unibo.it/doku.php?id=corpora`

hypothesise that the lack of significant difference in the diversification performance of the pattern-based WSI algorithms is due to the relatively low number of clusters they produce. The best performance on both corpora and over all evaluation measures is obtained by *HyperLex*, with *B-MST* ranked second and *Chinese Whispers* third. Di Marco and Navigli conclude that the diversification performance of *HyperLex* makes it the most promising candidate for Web search result clustering; however, they also note that *HyperLex* is a highly parameterised algorithm whereas *B-MST* is a simpler algorithm, requiring just one parameter set by the user (the number of clusters to return). The third evaluation is a qualitative (manual) evaluation, carried out by human annotators. This evaluation corroborates the findings in the extrinsic evaluations, showing that *Curvature* is the worst ranking system (possibly due to the low numbers of senses it returns) and that *HyperLex* and *B-MST* better discriminate between the meanings of an input query. Finally, it is worth noting that all five clustering algorithms outperform four non-semantic clustering methods (methods that apply no semantic analysis to search results). This finding implies that induction of the senses of an ambiguous search query is of benefit.

Part III

Systems and Evaluations

Chapter 8

Preliminary Evaluations

This chapter presents two preliminary evaluations of the *MaxMax* clustering algorithm that was introduced in Section 5.5. The evaluation contexts are limited in scope, though do exemplify the viability of a graph-based approach to Natural Language Processing (NLP). The first evaluation, reported in Section 8.1, presents a comparative evaluation between *MaxMax* and the *Chinese Whispers* clustering algorithm (described in Section 5.4.3). This evaluation is set within the context of a language separation task. Section 8.2 presents a comparative evaluation between *MaxMax*, *Chinese Whispers*, and a Markov model based algorithm, *MCL* (van Dongen, 2000), set within the context of a Word Sense Induction (WSI) task.

8.1 Evaluation 1: A Language Separation Task

This section reports on a comparative evaluation between *MaxMax* and *Chinese Whispers*. The aim of the evaluation is to assess whether *MaxMax*'s clustering performance is comparable to that of *Chinese Whispers*. To allow for a fair comparison, a task is selected from those evaluated in Biemann (2007) where *Chinese Whispers* is applied to part of speech tagging, language separation, and WSI. The pertinent choice, germane to this thesis, would be WSI. However, Biemann applies the WSI evaluation method proposed in Bordag (2006): purportedly a method to evaluate WSI; in practice one that evaluates word sense disambiguation. Bordag uses pseudowords (Gale et al., 1992a; Schütze, 1992) to generate ambiguous contexts for target word pairs. For example, given a corpus containing the two target words *banana* and *door*, all contexts of *banana* and all contexts of *door* are merged, resulting in *bananadoor* (the pseudoword) contexts. The aim then is to partition *bananadoor* contexts into the pre-merged *banana* and *door* contexts. This is not a WSI task. Partitioning the merged contexts of two target words is not comparable with inducing the senses of two target words; as Biemann states: "Pseudoword evaluation and the restriction to only a few sample words make it difficult to draw conclusions about WSD[1] performance from these experiments" (Biemann, 2007, p.158). Dismissing

[1]Bordag (2006) clearly states: "In this paper a novel solution to automatic and unsupervised word sense induction (WSI) is introduced." (Bordag, 2006, p.1).

the disambiguation task therefore leaves two choices. As a lexicographic classification of the notional classes of parts of speech is of little interest, language separation is selected as the task for the first preliminary evaluation, noting that the task itself is simply a conduit through which *Chinese Whispers* and *MaxMax* can be compared.

8.1.1 Language Separation

Biemann and Teresniak (2005) and Biemann (2007) show that a polylingual corpus can be divided into its monolingual parts by means of a graph-based clustering approach: *Chinese Whispers*. The aim of this evaluation is see if *MaxMax* can do so similarly.

Before outlining the evaluation, a caveat is in order. Language separation is, as Biemann notes, a "solved problem" (Biemann, 2007, p.114). Shallow techniques, such as those presented in McNamee (2005), are all that are required to separate a polylingual corpus into its monolingual parts. That said, the graph-based approaches that are applied in this evaluation make no recourse to training data, nor to any inherent clues that might identify a language such as ASCII/Unicode encodings. Furthermore, *Chinese Whispers* and *MaxMax* are non-parametrised algorithms, thus prior knowledge of the number of languages to be separated or of their distribution in the test set is not a prerequisite.

8.1.2 Evaluation Data

The most direct evaluation approach would take the datasets used in Biemann and Teresniak (2005)[2] and Biemann (2007)[3], apply *MaxMax* to the data, then present results for *MaxMax* and *Chinese Whispers*. Unfortunately, this approach is not possible for the following reasons:

1. The *Chinese Whispers* package[4] includes a seven language test set, presumed, initially, to be the seven language test set used in Biemann and Teresniak (2005). This set consists of a vertex input file `7lang_ nodes.txt` and an edge input file `7lang_edges.txt` that simply allow *Chinese Whispers* to be run. No class labels are provided for the vertices, nor any gold standard reference set that would allow an evaluation of *Chinese Whispers* or, indeed, of any other clustering algorithm.

2. Biemann's *langSepP* package[5] contains a test set `mix1K.txt` consisting of 10,000 sentences in 10 languages. This set is just 1% of the size of the test set used in Biemann (2007); again, no gold standard reference set is provided.

3. The measures used to quantify the strength of association between words in sentences and the thresholds applied differ across implementations. In Biemann and Teresniak (2005) an association measure based on the Poisson distribution is applied, with the threshold set at 0.4; in Biemann (2007) this is set at 1.64. Delving into the *langSepP*

[2] 7 languages: Dutch, Estonian, English, French, German, Icelandic, and Italian. 100,000 sentences per language.

[3] 10 languages: French, English, Icelandic, Japanese, Italian, Norwegian, German, Finnish, Dutch, and Sorbian [sic]. 100,000 sentences per language.

[4] `http://wortschatz.informatik.uni-leipzig.de/~cbiemann/software/CW.html`

[5] `http://wortschatz.uni-leipzig.de/~cbiemann/software/langSepP.html`

code, in an attempt to find a reference set for `mix1K.txt`, finds that Biemann applies Dunning's Log Likelihood Ratio (Dunning, 1993), with the threshold set at 6.63.

8.1.3 The Test Set

Noting the issues above, I devised a test set, *AW*, which consists of Lewis Carroll's novel *Alice's Adventures in Wonderland* (Carroll, 1865) in five European languages: English[6], French[7], Italian[8], Danish[9], and German[10].

English	French	Italian	Danish	German	Total(AW)	Average
1596	1734	1547	2074	1637	8588	1718

Table 8.1: The number of sentences in the test set *AW*.

The test set is constructed as follows. Sentences are extracted from the five monolingual plain text versions of *Alice's Adventures in Wonderland*[11]. Punctuation is removed, except for possessive apostrophes (for example, as in Alice's cat), clitics (Je t'aime), and hyphens (arci-profundo). Case is retained. The processed sentences are then merged to form the polylingual test text *AW*.

As Table 8.1 illustrates, the number of sentences in AW is not comparable with Biemann's test set of 100,000 sentences. However, as Biemann and Teresniak (2005) and Biemann (2007) show, a large amount of text is not required to separate languages successfully. Biemann and Teresniak (2005) show that *Chinese Whispers* is able to separate a polylingual corpus into its seven monolingual parts using just 100 sentences per language, returning Precision and Recall values > 0.96. These findings are further validated in Biemann (2007) for a polylingual corpus of ten languages, finding that Precision and Recall values $\simeq 1$ are returned using 1000 sentences per language. Therefore. the number of sentences in AW should be quite sufficient for the evaluation.

8.1.4 The Test Set Graph

The data in AW is transformed to a test set graph G_{AW} as follows. Similarity between each sentence pair $(s_i, s_{j\neq i})$ in AW is calculated as the number of shared word types, defined as:

$$similarity(s_i, s_{j\neq i}) = |\{\text{words in } s_i\} \cap \{\text{words in } s_j\}| \,. \tag{8.1}$$

The premise here is that sentence pairs of the same language should return higher similarity values than those of different languages. For example, given the three sentences:

[6] `http://www.gutenberg.org`
[7] `http://www.livres-et-ebooks.fr`
[8] `http://www.liberliber.it`
[9] `http://www.ebbemunk.dk`
[10] `http://projekt.gutenberg.de`
[11] Sentences are demarcated by a full stop, exclamation mark, or question mark.

s_1 Up above the world you fly like a tea-tray in the sky

s_2 But if I'm not the same the next question is who in the world am I

s_3 Loin au-dessus du monde tu voles comme un plateau de thé dans le ciel

similarity $(s_1, s_2) = 3$ as s_1 and s_2 share word types {the, in, world}, whereas the similarity between s_1 and its French equivalent $s_3 = 0$.

The similarity values between sentences are used as edge weights in a graph G_{AW}. Each sentence s_i in AW is represented as a vertex v_i in G_{AW}, with an edge $e(v_i, v_j)$ connecting vertices v_i, v_j if the similarity between sentences $s_i, s_j > 0$. G_{AW} is then clustered using *Chinese Whispers/MaxMax*. The optimal clustering solution would therefore partition G_{AW} into a set of clusters that align with the monolingual constituent parts of AW: five languages; five clusters. However, as *Chinese Whispers* and *MaxMax* are non-parametrised algorithms this is highly unlikely to occur. Therefore, a mapping between each language and its most representative cluster is applied, as described in the following section.

8.1.5 Mapping Languages to Clusters

Biemann (2007) proposes a method for mapping each class c in a set of gold standard classes C to a cluster k in a clustering solution K. In this evaluation C is a set of five classes: *Alice's Adventures in Wonderland* in five languages and K is the clustering solution for C that is returned by *Chinese Whispers/MaxMax*. An illustrative example of the method is shown in Table 8.2.

	$\boldsymbol{K}$	k_1	k_2	k_3	k_4	k_5	**Map**
$\boldsymbol{C}/\lvert\boldsymbol{C}\rvert$							
$c_1/100$		**75**	0	0	0	0	$c_1 \Rightarrow k_1$
$c_2/200$		5	**110**	40	1	20	$c_2 \Rightarrow k_2$
$c_3/300$		5	120	60	**30**	10	$c_3 \Rightarrow k_4$
$c_4/400$		15	0	**280**	25	0	$c_4 \Rightarrow k_3$

Table 8.2: An illustration of the mapping method proposed in Biemann (2007). $|C|$ denotes the number of objects in a class. Map lists the class to cluster mappings.

Classes are mapped to clusters as follows. For each $c \in C$:

$$c \Rightarrow \max_{k \in K} \frac{|k \cap c|}{|c|}. \tag{8.2}$$

If more than one class maps to the same cluster k_i then k_i is assigned to:

$$k_i \Rightarrow \max_{c \in C} \frac{|k_i \cap c|}{|c|}. \tag{8.3}$$

For example, in Table 8.2 c_1 is mapped to k_1 as k_1 maximises Equation (8.2). c_2 and c_3

both map to k_2, however as c_2 maximises Equation (8.3) c_2 is mapped to k_2:

$$\left(\frac{|k_2 \cap c_2|}{|c_2|} = \frac{110}{200} = 0.55\right) > \left(\frac{|k_2 \cap c_3|}{|c_3|} = \frac{120}{300} = 0.4\right).$$

Once each class is mapped to a cluster any remaining clusters are left unmapped. For example, k_5 in Table 8.2 is left unmapped. Unmapped clusters thus affect the Recall results that are presented in Section 8.1.7.

8.1.6 Evaluation Measures

Five evaluation measures assess the performance of *MaxMax* and *Chinese Whispers*: Precision, Recall, F-Score, Entropy, and Purity. These measures are commonly applied in the evaluation of clustering solutions. All five are said to be external measures as they assess the degree to which objects in clusters align with objects in 'external' gold standard classes. The measures are defined as follows:

Precision

Precision between a cluster k and a class c measures the fraction of k that consists of objects of c:

$$Precision(k, c) = \frac{|k \cap c|}{|k|}. \tag{8.4}$$

Precision for a clustering solution K is the average Precision of mapped pairs:

$$Precision(K) = \frac{1}{|C|} \sum_{i=1}^{|C|} Precision(k_*, c_i), \tag{8.5}$$

where $|C|$ is the number of classes in the test set and k_* is the cluster that class c_i maps to (as defined in Section 8.1.5). $Precision(K)$ returns a value in the range $[0, ..., 1]$ where 0 indicates a complete misclassification of objects and 1 indicates that each cluster contains objects of just one particular class.

Recall

Recall between a cluster k and a class c measures the extent to which k contains all objects of c:

$$Recall(k, c) = \frac{|k \cap c|}{|c|}. \tag{8.6}$$

Recall for clustering solution K is defined as:

$$Recall(K) = \frac{1}{|C|} \sum_{i=1}^{|C|} Recall(k_*, c_i), \tag{8.7}$$

where, as in (8.5), k_* is the cluster that c_i maps to. $Recall(K)$ returns a value in the range $[0, ..., 1]$, with 1 indicating that for each class $c \in C$ there is a cluster $k \in K$ that contains all objects of c.

F-Score

The F-Score between a cluster k and a class c measures the extent to which k contains 1.) only objects of c and 2.) all objects c:

$$F\text{–}Score(k,c) = \frac{2 \times Precision(k,c) \times Recall(k,c)}{Precision(k,c) + Recall(k,c)}. \tag{8.8}$$

The F-Score for a clustering solution K is the average F-Score of mapped class-cluster pairs:

$$F\text{–}Score(K) = \frac{1}{|C|} \sum_{i=1}^{|C|} F\text{–}Score(k_*, c_i). \tag{8.9}$$

The F-Score measures both the *homogeneity* and *completeness* of a clustering solution. If all objects in a cluster are of one class, the cluster is said to be maximally homogeneous; if all objects of one class are found in one cluster, the cluster is said to be maximally complete. The F-Score is the harmonic mean of Precision and Recall and returns a value closer to the lowest of the two values returned by Precision and Recall, a high F-Score thus indicates that both Precision and Recall are high.

Entropy

Entropy for a cluster k measures the degree to which k consists of objects of a single class:

$$Entropy(k) = -\sum_{i=1}^{|C|} \frac{|k \cap c_i|}{|k|} \log_2 \frac{|k \cap c_i|}{|k|}. \tag{8.10}$$

If k contains objects of a single class then $Entropy(k) = 0$; in information theoretic terms k is said to have zero uncertainty. Entropy for a set of clusters K is the sum of the entropies of each cluster weighted by the size of each cluster:

$$Entropy(K) = \sum_{i=1}^{|K|} \frac{|k_i|}{N} Entropy(k_i), \tag{8.11}$$

where N is the number of objects in the dataset and $|K|$ is the number of clusters.

Purity

Purity for a cluster k measures the extent to which k contains objects of a single class:

$$Purity(k) = \max_{c \in C} \frac{|k \cap c|}{|k|}. \tag{8.12}$$

If all objects of some class $c \in C$ are found in k then $Purity(k) = 1$, the highest score for Purity. Purity for a set of clusters K is the sum of the purities of each cluster weighted

by the size of each cluster:

$$Purity(K) = \sum_{i=1}^{|K|} \frac{|k_i|}{N} Purity(k_i). \tag{8.13}$$

8.1.7 Evaluation Results

Evaluation results are reported in Table 8.3. Results show that *Chinese Whispers* (*CW*) returns 127 clusters, with the average number of objects (vertices representing class sentences) in clusters 67 and the average number of objects in mapped clusters 687.

	Precision	**Recall**	**F-Score**	**Entropy**	**Purity**	$\|K\|$	**Avg.**	**Map Avg.**
CW	**0.95**	0.39	0.54	**0.40**	**0.91**	127	67	687
MaxMax	0.90	**0.77**	**0.80**	0.54	0.89	55	156	1469

Table 8.3: Evaluation results.

MaxMax returns 55 clusters. The average size of clusters is 156 and the average size of mapped clusters is 1469. Though both algorithms clearly over-generate with respect to the number of gold standard classes, *MaxMax*'s mapped clusters are far closer in size to those of the gold standard classes where average class size is 1718.

Entropy and Purity Results

As Table 8.3 shows, *Chinese Whispers* returns lower Entropy and higher Purity than *MaxMax*. These results therefore indicate that there is greater homogeneity in *Chinese Whispers*' clusters. However, *Chinese Whispers* generates over twice the number of clusters as *MaxMax*. Given two clustering solutions K_1 and K_2 where $|K_1| > |K_2|$, Entropy and Purity are biased to favour K_1 as a partition of class objects into greater numbers of clusters typically results in lower Entropy and higher Purity (Tan et al., 2006). Indeed, if K_1 were the degenerate solution consisting of a partition of class objects to singleton clusters then maximum scores would be obtained: $Entropy(K_1)$ would be 0 and $Purity(K_1)$ would be 1. Moreover, Entropy and Purity do not account for the completeness of a clustering solution, thus cannot give an overall assessment of clustering performance.

Precision, Recall and F-Score Results

Table 8.3 shows that *Chinese Whispers* returns higher Precision than *MaxMax*. However, Precision is a measure of homogeneity, thus can be maximised by a degenerate clustering solution. For example, Precision would be maximised by assigning a single sentence from each of the five different language classes to five separate clusters. *Chinese Whispers* may classify sentences more precisely than *MaxMax* but does so for far fewer of them. Indeed, the comparatively high Recall of *MaxMax* (0.77 compared to 0.39) indicates that *MaxMax* correctly clusters nearly twice the number of sentences as *Chinese Whispers*; MaxMax thus returns a more complete clustering solution.

MaxMax returns a higher F-Score than *Chinese Whispers* (0.80 compared to 0.54). F-Score is biased to prefer clustering solutions that contain a high number of *true positives*. A true positive is an object that is identified as being a positive example of a particular class that is actually of that class. In this evaluation, true positives are sentences of a particular language that are assigned to the mapped cluster representing that language. F-Score, thus, has a preference for "finding things even at the cost of also returning some junk" (Manning and Schütze, 1999, p.269). *MaxMax*'s trade-off in returning lower Precision than *Chinese Whispers* ("returning some junk") is higher Recall ("finding things"). If, as is standard practice in evaluations that apply Precision and Recall, the F-Score is taken as the single measure of overall performance then *MaxMax* is shown, in this particular evaluation context, to outperform *Chinese Whispers*.

8.2 Evaluation 2: A Word Sense Induction Task

As discussed in Chapter 2, graph representations of the interconnections between words illustrate how polysemous words can connect words of unrelated semantics.

G_{cherry}

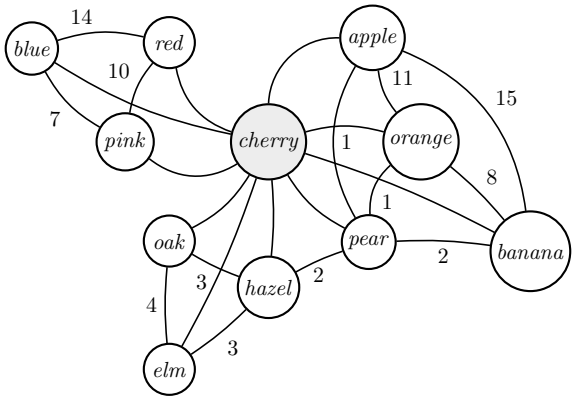


Figure 8.1: G_{cherry}, a word co-occurrence graph for the polysemous word *cherry*.

For example, the graph G_{cherry} in Figure 8.1 shows that vertices representing the semantically unrelated words *blue* and *banana* are connected by *cherry*. As the hub vertex in the graph, *cherry* draws vertices that represent its various senses into one connected component, with the traversal of *cherry* leading from one 'sense' of *cherry* to another. However, as shown in Figure 8.2, if *cherry* is deleted, G_{cherry} partitions to a set of clusters,

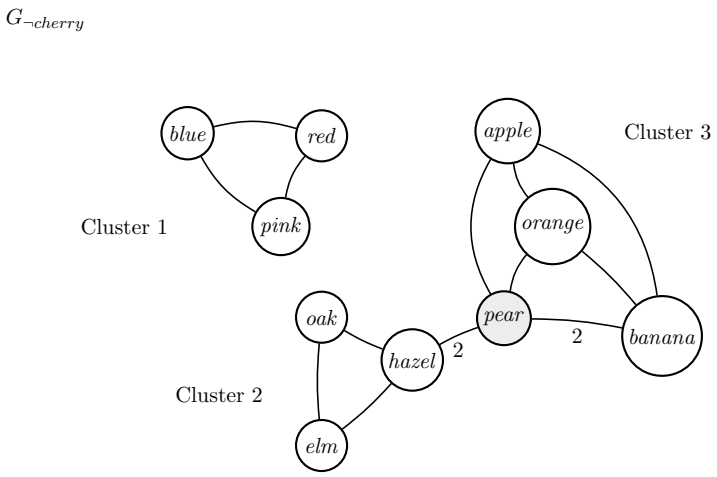


Figure 8.2: $G_{\neg cherry}$: G_{cherry} with the *cherry* vertex deleted.

with the words in each cluster taken to represent a particular sense of *cherry*.

Clusters	**Senses**
{*blue, pink, red*}	COLOUR
{*elm, hazel, oak, pear*}	WOOD
{*apple, banana, orange, pear*}	FRUIT

Table 8.4: A possible mapping between clusters in $G_{\neg cherry}$ and senses of *cherry*.

As Table 8.4 illustrates, *blue*, being a member of the cluster $\{blue, pink, red\}$, may now be interpreted as taking its COLOUR sense; similarly, *banana* its FRUIT sense, which in turn define two senses of the word *cherry*.

Note that demarcation of a word to one sense is not guaranteed as cluster words may themselves be polysemous. For example, in G_{cherry} *pear* has a strongest edge weight to both *hazel* and *banana*, thus is admitted to two clusters. Consequently, *pear* is interpreted as taking both its FRUIT and WOOD sense.

8.2.1 Two Approaches to Word Sense Induction

The two approaches to WSI that are applied in this evaluation are based on the strategy presented above. A word co-occurrence graph G_{tw} is constructed for each target word *tw* evaluated. *tw* is removed from G_{tw}, partitioning the graph to a set of clusters C_{tw}; said to be sense clusters of *tw* as words in clusters are taken to represent the senses of *tw*.

The evaluation compares results obtained by the approach proposed in Dorow (2007) with those obtained using the *MaxMax* and *Chinese Whispers* clustering algorithms. The evaluation is somewhat limited as Dorow reports results for just six polysemous words[12], nevertheless, it should provide some indication of *MaxMax*'s ability to induce word senses. Dorow's approach is described in Section 8.2.2, with the approach using *MaxMax*/*Chinese*

[12]Dorow reports that results for "several other words were similarly encouraging"(Dorow, 2007, p.54). A request was sent to Dorow for the results of these words though no reply was forthcoming.

Whispers described in Section 8.2.3. Results for the two approaches are reported in Section 8.2.4, with a discussion of the results presented in Section 8.2.5.

8.2.2 Dorow's Approach

Dorow applies a Markov model-based clustering algorithm, *MCL* (van Dongen, 2000), to partition a target word graph G_{tw} to a set of sense clusters C_{tw}. As previously noted in Chapter 7, the clustering solution returned by *MCL* is dependent upon the values set for the inflation and expansion input parameters. Finding the optimal parameters to apply therefore requires trialling all possible input parameter combinations. This process could be automated, though would be computationally intensive for large word co-occurrence graphs. The size of G_{tw} also affects *MCL*'s clustering solution: if G_{tw} is small, rarer senses of the target word may become conflated with predominant senses; if G_{tw} is large, noise may be be introduced. Dorow's solution is to apply *MCL* to fixed sized graphs in a manner similar to that presented in Pantel and Lin (2002). Given a fixed sized graph G_{tw} for a target word *tw*, an iterative clustering process is applied, selecting at each iteration the single most significant cluster[13], removing it from G_{tw} and setting it aside in C_{tw}, the clustering solution. In each iteration, the graph is recomputed, minus all vertices and adjacent edges contained in C_{tw}. This process continues until the similarity between *tw* and its closest neighbour falls below a pre-defined threshold[14]. Upon completion, each cluster in C_{tw} is labelled with a WordNet sense (synset) using the class labelling algorithm proposed in Widdows (2004). The complete process, as defined in Dorow (2007), is as follows:

Dorow's *MCL*-Based Algorithm

For each target word *tw* evaluated:

1. Compute G_{tw}, a graph containing vertices not found in nor linked to clusters in C_{tw}. If the similarity between *tw* and its closest neighbour falls below a pre-defined threshold go to step 6.
2. Recursively remove all vertices with degree = 1.
3. Remove *tw* from G_{tw}.
4. Apply *MCL* to G_{tw}.
5. Select the most significant cluster returned by *MCL*, add it to C_{tw} then return to step 1.
6. Assign a WordNet sense to each cluster $c \in C_{tw}$ using a class labelling algorithm.

G_{tw} initially consists of a set number of vertices: the target word *tw*; the top twenty neighbours of *tw*, and the top ten neighbours of the neighbours of *tw*. The top n neighbours of a word w are the n vertices with the highest edge weights to w. Dorow uses co-occurrence counts as edges weights in G_{tw}. Step 2 of the algorithm recursively removes vertices with

[13] Dorow does not define how the 'most significant cluster' is selected.
[14] Dorow does not define the value set for this threshold.

a degree of one as these vertices make no contribution to the semantic unity of a cluster. In step 6, the class labelling algorithm proposed in Widdows (2004) assigns a WordNet sense to each cluster $c \in C_{tw}$, as described below.

Widdows' Class Labelling Algorithm

Widdows (2004) presents a method for labelling each cluster $c \in C_{tw}$ with the WordNet synset that represents the majority sense in c. The method first associates each word $w \in c$ with a set of WordNet synsets h_w, the hypernyms of the synsets containing w. The set of hypernyms for all words in c is thus:

$$h_c = \bigcup_{w \in c} h_w. \tag{8.14}$$

The hypernym in h_c selected as the label for c is that which subsumes as many of the words in c as possible, as closely as possible. Figure 8.3 provides an example of how hypernym scores are calculated for:

$c = \{apple, pear, banana, orange, lemon\}$.

$h_c = \{$FOOD, FRUIT, EDIBLE FRUIT, CITRUS FRUIT$\}$

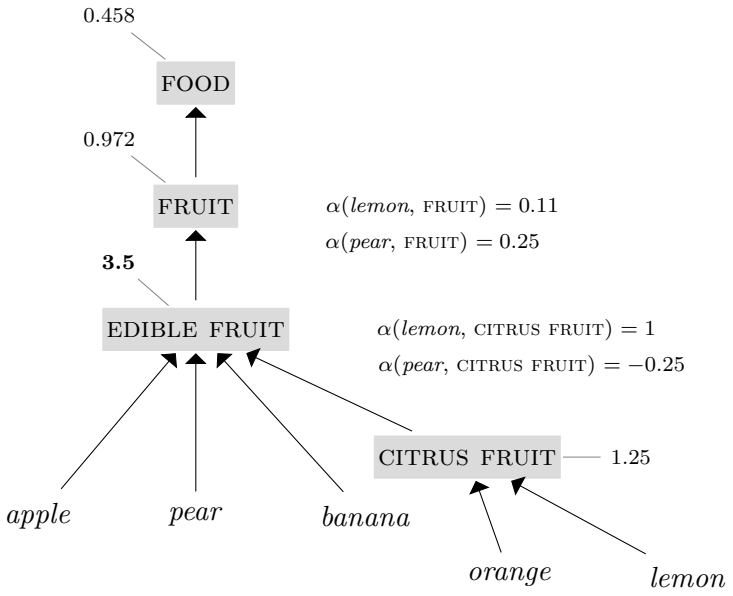


Figure 8.3: Widdows' class labelling algorithm.

An affinity score α is calculated for each word $w \in c$ and each hypernym $h \in h_c$, defined as:

$$\alpha(w,h) = \begin{cases} \frac{1}{distance(w,h)^2} & \text{if } w \subset h \\ -0.25 & \text{if } w \not\subset h, \end{cases} \tag{8.15}$$

where $distance(w,h)$ is the shortest path in WordNet between w and h, and $w \subset h$ denotes that w is subsumed by h (that is, w is a hyponym of h). For example, if w is *orange* and h is FRUIT, there is only one shortest path in Figure 8.3 between w and h:

orange ⊂ CITRUS FRUIT ⊂ EDIBLE FRUIT ⊂ FRUIT.

Thus, $distance(orange, \text{FRUIT}) = 3$ and $\alpha(orange, \text{FRUIT}) = \frac{1}{3^2} = 0.11$. The reciprocal of $distance(w, h)$ squared and the penalty value of -0.25 for hypernyms that do not subsume cluster words are selected, according to Widdows, "after a variety of heuristic tests" (Widdows, 2004, p.90). The hypernym chosen as the class label for the cluster c is that which maximises the affinity score in (8.15):

$$Class\ Label(c) = \max_{h \in h_c} \sum_{w \in c} \alpha(w, h). \tag{8.16}$$

The scores for hypernyms in h_c are shown in Figure 8.3 where EDIBLE FRUIT is found to be the hypernym that maximises the affinity score:

$$\underbrace{3 \times 1}_{apple,\ pear,\ banana} + \underbrace{2 \times 0.25}_{orange,\ lemon} = 3.5.$$

Thus, EDIBLE FRUIT is the label assigned to cluster c.

8.2.3 The *MaxMax/Chinese Whispers* Approach

The *MaxMax/Chinese Whispers* approach is defined as follows:

MaxMax/Chinese Whispers-Based Algorithm

For each target word *tw* evaluated:

1. Compute G_{tw}.
2. Recursively remove all vertices with degree = 1.
3. Remove *tw* from G_{tw}.
4. Apply *MaxMax/Chinese Whispers* to G_{tw} to return a set of sense clusters C_{tw}.
5. Assign a WordNet sense to each cluster in $c \in C_{tw}$ using Widdows' class labelling algorithm.

Edge weights in G_{tw} are Log-Likelihood Ratio (LLR) scores, this being a measure that Dorow recommends though does not apply. Co-occurrence counts were also applied (separately) in G_{tw}, though found to return clustering results identical to those of G_{tw} using LLR as edge weight.

Unlike Dorow's approach, this approach requires no iterative reconstruction of G_{tw}, nor has it to account for the connectivity relationships held between G_{tw} and C_{tw}. Furthermore, there is neither a necessity to find the current most similar cluster, nor to define a stopping condition. However, *Chinese Whispers* is an iterative algorithm, with Biemann recommending six iterations for weighted graphs, whereas *MaxMax* is not, and will return the sense clusters in a single run. Furthermore, given the graph in Figure 8.1, *Chinese Whispers* must assign the *pear* vertex to just one sense cluster whereas the soft clustering approach of *MaxMax* allows *pear* to belong to both the FRUIT and the WOOD sense clusters.

8.2.4 Evaluation Results

Results for the six polysemous words evaluated in Dorow (2007) are shown in Tables 8.5 to 8.16. Dorow's results are presented first, showing the sense labels obtained from both WordNet version 1.7.1 (Dorow's labelling source) and WordNet 3.0. Results for *MaxMax* and *Chinese Whispers* follow, where sense labels are obtained from WordNet 3.0. Dorow lists just the top two scoring sense clusters; for *MaxMax* and *Chinese Whispers* all sense clusters with ≥ 3 words are reported and ranked using the scores returned by Widdows' class labelling algorithm. A 'true' sense of a target word is taken to be one that is listed in WordNet (3.0), these senses are indicated in the tables by a double tick mark. A single tick mark indicates a sense that is undefined in WordNet, though deemed, arguably so, to be a sense of the target word. These sense assignments are, as they are in Dorow (2007), completely subjective, thus it is left to the reader to decide whether the labels are plausible senses of target words.

Chinese Whispers and *MaxMax* return the same number and same type of senses, with the overlap of words in clusters found to be 0.941, a value calculated using a straightforward measure of cluster word overlap, defined below.

Cluster Word Overlap Measure: Given the clusters M, returned by *MaxMax*, and C, returned by *Chinese Whispers*, where the number of clusters is identical, $|M| = |C|$, a measure of word overlap is used to define how similar M is to C:

$$Overlap(M, C) = \frac{\sum_{m \in M} \max_{c \in C} \frac{m \cap c}{m \cup c}}{|M|}, \tag{8.17}$$

$Overlap(M, C)$ returns the average Jaccard similarity score for (m,c) pairs where c is the cluster in C that returns the highest Jaccard score for m.

As the overlap measure score suggests, *MaxMax* and *Chinese Whispers* return clustering solutions that are highly similar, thus rather than duplicate clusters, only the minor differences observed for *Chinese Whispers* are noted in Tables 8.5 to 8.16 in red typeface, with all other clusters identical for both algorithms.

Cherry

Cherry has four senses defined in WordNet 3.0: WOOD, TREE, EDIBLE FRUIT, and COLOUR.

Clusters	**Sense** 1.7.1	**Senses** 3.0
cedar, larch, mahogany, water, sycamore, lime, teak, ash, hornbeam, oak, walnut, hazel, pine, beech, alder, thorn, poplar, birch, chestnut, blackthorn, spruce, holly, yew, laurel, maple, elm, fir, hawthorn, willow	WOOD ✔✔	WOOD ✔✔
bacon, cream, honey, pie, grape, blackcurrant, cake, banana	FOOD ✔	EDIBLE FRUIT ✔✔

Table 8.5: Cherry *Dorow-MCL*.

Clusters	**Senses** 3.0
apple, apricot, banana, fig, grape, melon, nectarine, orange, peach, pear, pineapple, plum	EDIBLE FRUIT ✔✔
apple, apricot, banana, fig, onion, orange, peach, pear, pineapple, plum	
oak, maple, alder, ash, beech, birch, elm, pine, sycamore, tree, amelanchier, tulip tree, tupelo, **rosewood**	WOOD ✔✔
white, black, asian, blue, brown, green, grey, pink, red, yellow	COLOUR ✔✔
walnut, chestnut, mahogany, **rosewood**	WOOD ✔✔
chestnut, dogwood, mahogany, walnut	
yew, cherry laurel, holly, laurel	FLOWERING TREE ✔✔
aroma, scent, flavour, appearance, bouquet, colour, fragrance, quality, texture	SCENT ✔
fruit, cereal, wine, nut, berry, vegetable, leave, raisin, seed, almond, currant, flour, sultana	FRUIT ✔✔

Table 8.6: Cherry *MaxMax/Chinese Whispers*.

Jersey

Jersey has five senses defined in WordNet 3.0: AMERICAN STATE, ISLAND, GARMENT, FABRIC, and DAIRY COW.

Clusters	**Senses** 1.7.1	**Senses** 3.0
israel, colombo, guernsey, denmark, malta, greece, belgium, sweden, turkey, gibraltar, portugal, ireland, mauritius, britain, cyprus, netherlands, norway, australia, italy, japan, canada, kingdom, spain, austria, zealand, england, france, germany, switzerland, finland, poland, america, usa, iceland, holland, scotland, luxembourg, uk	COUNTRY	COUNTRY
bow, apron, sweater, tie, anorak, hose, bracelet, helmet, waistcoat, jacket, pullover, equipment, cap, collar, suit, fleece, tunic, shirt, scarf, belt, crucifix	GARMENT	GARMENT ✔✔

Table 8.7: Jersey *Dorow-MCL*.

Clusters	**Senses** 3.0
new jersey, pennsylvania, illinois, connecticut, louisiana, maryland, new york, ohio, virginia, west virginia, california, carolina, indiana, iowa, missouri	AMERICAN STATE ✔✔
manchester, birmingham, bristol, leeds, liverpool, newcastle, sheffield	METROPOLIS
bomber jacket, cap, coat, hat, jacket, jean, jumper, polo shirt, pullover, sandal, shirt, shorts, singlet, slacks, sneaker, sock, suit, sweater, sweatshirt, t-shirt, tee shirt, tie, top, trouser, vest, waistcoat	GARMENT ✔✔
bomber jacket, boot, breeches, cap, coat, hat, jacket, jean, jumper, legging, polo shirt, pullover, sandal, shirt, shoes, shorts, singlet, ski, slacks, sneaker, sock, suit, sweater, sweatshirt, t-shirt, tee shirt, tie, top, trouser, vest, waistcoat	
germany, france, belgium, britain, england, italy, japan, spain, uk, west germany	COUNTRY
bermuda, anguilla, bahamas, cayman islands, falkland islands, jamaica, montserrat, territory	ISLAND ✔✔
ayrshire, friesian, holstein	DAIRY COW ✔✔
isle, channel island, guernsey, argyll, hampshire, highlands, northern ireland, scotland, shetland, car, devon, israel, man, philippines	ISLAND ✔✔

Table 8.8: Jersey *MaxMax/Chinese Whispers*.

Oil

WordNet 3.0 defines four senses for oil: LIQUID, ART/PAINT, PETROCHEMICAL, EDIBLE PLANT OIL.

Clusters	**Senses** 1.7.1	**Senses** 3.0
heat, coal, power, water, gas, food, wood, fuel, steam, tax, heating, kerosene, fire, petroleum, dust, sand, light, steel, telephone, timber, supply, drainage, diesel, electricity, acid, air, insurance, petrol	OBJECT	ENTITY
tempera, gouache, watercolour, poster, pastel, collage, acrylic	PAINT ✔✔	ART/PAINT ✔✔

Table 8.9: Oil *Dorow-MCL*.

Clusters	**Senses** 3.0
garlic, onion, tomato, carrot, celery, sage, cabbage, leek, parsley, thyme,	HERBACEOUS PLANT
fuel, diesel fuel, food, lubricant, material, petrol, power	SUBSTANCE
grease, barnacle, dirt, filth, grime, insulator, soot, spill, turpentine	DIRT ✔
herb, garlic, seasoning, wine vinegar	SEASONING ✔
assets, business, income, investment, land, profits, property, share	POSSESSION
pharmaceutical, chemical, engineering, fertilizer, metal, polymer, product, substance, waste	CHEMICAL ✔
gas, water, coal, liquid, electricity, grease	FLUID ✔
energy, resource, power, skill, money, assets	ABILITY POWER ✔
industry, steel industry, affairs, agriculture, education, environment, government, manufacturing, profession, service industry, shipbuilding	INDUSTRY ✔
wine vinegar, castor sugar, mustard, seed, tomato ketchup, vegetable oil, walnut oil, wine	CONDIMENT ✔
nature, essence, effectivity, finitude, form, idea, juice, opposite, origin, soul	QUALITY
watercolour, tempera, fresco, chalk, drawing, pastel, ink, etching, gouache, painting, pencil, print, paper, pen	ART/PAINT ✔✔

Table 8.10: Oil *MaxMax*.

Head

Head has thirty three senses defined in WordNet 3.0 including: BODY PART, PERSON, MIND-EMOTION, and LEADER.

Clusters	**Senses** 1.7.1	**Senses** 3.0
voice, torso, back, chest, face, abdomen, side, belly, groin, spine, breast, bill, rump, midhair, hat, collar, waist, tail, stomach, skin, throat, neck, speculum	BODY PART	BODY PART ✔✔
ceo, treasurer, justice, chancellor, principal, founder, president, commander, deputy, administrator, constable, librarian, chief, governor, captain, premier, executive, curator, secretary, assistant, committee, patron, ruler	PERSON	PERSON ✔✔

Table 8.11: Head *Dorow-MCL*.

Clusters	**Senses** 3.0
legs, arms, back, tail, breast, torso, shoulder, thigh, chest, neck, elbow, stomach, wrist	BODY PART ✔✔
death, injury, accident, damage, disease, illness, loss, sickness, suspension	ILL HEALTH ✔
government, state, condition, country, economy, employer, party, power, institution, society	ORGANISATION
body, mind, soul, person, heart	SOUL
hair, eyes, cheek, ear, face, lip, nose, skin, teeth, throat	BODY PART ✔✔
teacher, child, learner, lecturer, librarian, nurse, pupil, school, therapist, worker	PERSON ✔✔
president, senate, senator, deputy, constable, executive	LEGISLATOR ✔

Table 8.12: Head *MaxMax*.

Squash

Squash has three senses defined in WordNet 3.0: PLANT, EDIBLE FRUIT, and COURT GAME.

Clusters	**Senses** 1.7.1	**Senses** 3.0
football, terrace, court, gardening, rounders, running, theatre, snooker, bar, baseball, cup, gymnastics, bowling, mini, wrestling, chess, netball, area, fishing, music, golf, horse, gymnasium, lounge, table, basketball, city, ball, league, volleyball, bath, room, fencing, cricket, weather, hockey, polo, walking, dancing, boxing, swimming, solarium, croquet, tennis, rugby, handball, development, athletics, archery, badminton, pool, playground, whirlpool, garden, sauna, soccer, union,	SPORT ✔	ENTITY
apple, rum, mint, lime, cola, honey, coconut, lemonade, chocolate, lemon, bergamot, pepper, eucalyptus, acid, rind	FOOD ✔	PHYSICAL ENTITY

Table 8.13: Squash *Dorow-MCL*.

Clusters	**Senses** 3.0
jug, bowls, bucket, dish, jar, mug, plate, table tennis	CONTAINERFUL
basketball, baseball, court, handball, volleyball	BALL ✔
coffee, **cola**, drink, food, juice, meal, oil, salt, water coffee, drink, food, juice, meal, oil, salt, water	FOOD NUTRIENT ✔
cricket, cicada, empire, football, hockey, rounders, rugby, soccer	FIELD GAME
sock, shoes, boot, clothes, coat, dress, hat, shirt, stocking, suit, trouser	CLOTHING
hair, skin, blood, body, bone, bones, eyes, face, feature, flesh, muscle	BODY PART
badminton, bowling, fencing, golf, netball, tennis	COURT GAME ✔✔
pea, bean, bacon, carrot, chips, maize, rice, sausage, vegetable	PLANT ✔✔
chocolate, **cola**, crisp	DRINK ✔
cheese, chocolate, crisp, day, ice cream, nut, peanuts	FOOD ✔

Table 8.14: Squash *MaxMax/Chinese Whispers*.

Lemon

Lemon has five senses defined in WordNet 3.0: EDIBLE FRUIT, COLOUR, TREE, FLAVOUR, DEFECTIVE ARTIFACT.

Clusters	**Senses** 1.7.1	**Senses** 3.0
bread, cheese, mint, butter, jam, cream, pudding, yogurt, sprinkling, honey, jelly, toast, ham, chocolate, pie, syrup, milk, meat, beef, cake, yoghurt, grain	FOODSTUFF	FOOD ✔
hazel, elder, holly, family, virgin, hawthorn	SHRUB	SHRUB ✔

Table 8.15: Lemon *Dorow-MCL*.

Clusters	**Senses** 3.0
peach, apricot, cherry, fig, grape, melon, nectarine, pear, plum, strawberry, guava, horne, lychee, mango, pawpaw, pomegranate	EDIBLE FRUIT ✔✔
onion, garlic, cabbage, carrot, celery, leek, pepper, sage, basil, parsley, thyme, tomato, vinegar	HERBACEOUS PLANT ✔
herb, spice, peel	FLAVORING ✔✔
whisky, soda, ginger, gin, lime, lime juice, lager	ALCOHOLIC DRINK

Table 8.16: Lemon *MaxMax*.

8.2.5 Evaluation Results: Discussion

The results in Tables 8.5 to 8.16 show that *MaxMax*'s performance is comparable to *Chinese Whispers*. This is an encouraging result given that *Chinese Whispers* has been shown to be of utility in various NLP tasks (Korkontzelos and Manandhar, 2009; Zhang and Sun, 2011; Jurgens, 2012; Fountain and Lapata, 2012; Di Marco and Navigli, 2013). That said, if just the top two scoring clusters are considered, as they are in Dorow (2007), the *MCL* approach finds six out of a possible twelve WordNet senses whereas *MaxMax* finds just five. However, looking further down the rankings, *MaxMax* finds many of the senses of the target words. For example, *MaxMax* finds all WordNet senses of *cherry* as well as the WordNet-undefined SCENT sense of *cherry*. As Dorow reports on just the top two scoring clusters, a fair comparison between Dorow's approach and the *MaxMax/Chinese Whispers* approach cannot be made. *MCL* may, for all one knows, be finding just two predominant senses of target words with other senses left undefined. Indeed, the clusters returned by *MCL* are noticeably larger than those returned by *MaxMax*/*Chinese Whispers* and have a high degree of semantic cohesion: many words, semantically related to each other. However, a manual check of the co-ordination patterns used to construct the graphs indicates that clusters of this size and type could not occur, though there may very well be some aspect of Dorow's *MCL* method, not stated in Dorow (2007), that would allow these clusters to be returned.

Widdows' class labelling algorithm is heuristic, reliant on a hierarchical structure (WordNet's hypernym/hyponym architecture) that was not designed for the calculation

of similarity between words and their senses, thus sense labelling can go askew. For example, Widdows' algorithm assigns the *MaxMax* cluster {*basketball*, *baseball*, *court*, *handball*, *volleyball*}, for the target word *squash*, to WordNet's BALL sense, though COURT GAME would be more appropriate. These mislabelled clusters result in *MaxMax* being penalised even though the algorithm has found a valid representation of a sense of *squash*. *MaxMax/Chinese Whispers* also return senses that are deemed to be incorrect but that are often semantically related to the target word, for example, *oil* as POSSESSION and *head* related to SOUL. However, odd clusters do occur, though even here there is a degree of semantic relatedness with the target word. For example, the highest scoring sense for *squash* is CONTAINERFUL. This sense appears to be unrelated to *squash*. However, upon reflection, a relationship does exist between *squash* as a drink and CONTAINERFUL as JUGS of *squash* can fill MUGS with *squash*.

8.3 A Summary of the Preliminary Evaluations

The evaluations presented in this chapter, though limited in scope, exemplified the potential of a graph-based approach to NLP. Notably, *MaxMax*, the novel clustering algorithm introduced in Chapter 5, was shown to return results that are comparable to those of two state of the art clustering algorithms: *Chinese Whispers* and *MCL*.

The first evaluation, set within the context of a language separation task, showed that *Chinese Whispers* returns a more homogeneous clustering solution than *MaxMax*, but that *MaxMax* returns a more complete one. Taking the F-Score to be the measure of overall performance (homogeneity and completeness), *MaxMax* (F-Score = 0.80) was shown to outperform *Chinese Whispers* (0.54). The second evaluation, set within the context of a WSI task, showed that *MaxMax* returns results comparable to those of *Chinese Whispers*. This evaluation also showed that *MaxMax* was able to induce word senses in a far more straightforward manner than *MCL*.

Results, though encouraging, are not conclusive. Further evaluations are required in which a greater number of target word senses are induced and evaluated against objective, gold standard, reference sets. The following two chapters present two evaluations that are considerably more testing than those presented in this chapter. The first requires the induction of the senses of 27,071 words; the second evaluates *MaxMax*'s clustering performance within the formal framework of the SemEval-2010 Word Sense Induction and Disambiguation task.

Chapter 9

Clustering Coefficients and Word Sense Induction

9.1 Introduction

9.1.1 Evaluations

The previous chapter introduced a graph-based approach to Natural Language Processing (NLP), exemplified in two limited evaluations. This chapter presents a far more comprehensive evaluation of this approach, the aim of which is to induce the senses of British National Corpus (BNC) nouns whose senses are defined in WordNet 3.0 (27,071 nouns in total). The evaluation methodology follows that given in Dorow (Dorow, 2007, pp.72–79), which itself adopts methods proposed in Pantel and Lin (2002). This reliance on the work of Dorow is explained by the rarity of tests designed to evaluate Word Sense Induction (WSI) systems. To date, just three tests have been devised: Dorow's/Pantel and Lin's; Bordag's (Bordag, 2006), and the SemEval task (Agirre and Soroa, 2007; Manandhar et al., 2010). As discussed in the previous chapter, Bordag's evaluation methodology is applicable to Word Sense Disambiguation (WSD) not WSI. Therefore, the only valid evaluation frameworks for comparing WSI systems are Dorow's/Pantel and Lin's (reported in this chapter) and the SemEval task (reported in the following chapter).

9.1.2 Approaches

The graph-based approaches to WSI presented in this chapter deviate from what one might call 'a standard approach' in which edge weights are used to partition a graph into a set of sense clusters. WSI is, instead, approached here by using the clustering coefficient (CC) measure that was defined in Chapter 3 to assess semantic cohesion in subgraphs of a graph. The basic approach is as follows: given a word co-occurrence graph, vertices with low clustering coefficient values are deleted from the graph, fragmenting it into a set of connected components, each of which is taken to represent a word sense. Dorow (2007) finds that this approach is able to induce senses with some success, returning results that are comparable to those reported in Pantel and Lin (2002): the best results reported to date for a completely unsupervised approach to WSI. However, Dorow's approach requires

the demarcation of a suitable CC threshold to apply. The approach can also conflate word senses. Therefore, two novel approaches are proposed in this chapter, both of which avoid these issues; both of which are shown to return better results, and to have far higher coverage of senses, than the approaches of Dorow and Pantel and Lin.

9.1.3 Aim

Clustering coefficient measures have been previously applied in WSI systems (see: Véronis (2004); Widdows (2004); Korkontzelos et al. (2009); Navigli and Lapata (2010), amongst others). However, to date, no evaluation has been carried out to compare systems that use the unweighted clustering coefficient (CC) with those that apply a weighted clustering coefficient (WCC). The main aim of this chapter is to assess whether WSI systems using a weighted clustering coefficient (specifically, the measure proposed in Section 4.3) are better at inducing word senses than those using the unweighted clustering coefficient; thus, to evaluate whether a measure of topology that is supplemented by a measure of 'topography' (WCC) can better induce word senses than a measure of topology alone (CC).

9.1.4 Graph Generation

The WSI systems presented in this chapter use a word co-occurrence graph $G_{WordNet}$[1]. This graph is based on information extracted from the British National Corpus (BNC)[2], a 100 million word collection of written (90%) and spoken (10%) examples of modern (late twentieth century) British English. Vertices in $G_{WordNet}$ represent BNC nouns whose senses are defined in WordNet and edges represent noun co-occurrence in particular types of BNC contexts.

Graph Vertices

Each vertex in $G_{WordNet}$ represent a noun found in a *noun coordination pattern* of the BNC, that is, nouns found in lists such as

> orange, lemon, or lime

or

> lions and tigers and bears.

The use of these particular types of patterns is based on the observation that nouns in lists are often semantically related (Widdows, 2004; Dorow, 2007). For example: orange, lemon, and lime are all types of citrus fruit and lions, tigers, and bears are all types of wild animals.

As sentences in the BNC are marked-up with part of speech (PoS) tags, noun coordination patterns can be identified through use of regular expressions. The particular regular expression used here is:

```
NP(, NP)*,?( CJC NP)+                    (9.1)
```

[1] In Dorow (2007), $G_{WordNet}$ is constructed using WordNet 1.7.1. In this evaluation, $G_{WordNet}$ is constructed using WordNet 3.0.

[2] http://www.natcorp.ox.ac.uk/

where

> CJC is a conjunction = $(and|or|nor)$

and

> NP is a noun phrase = AT?(CRD)*(ADJ)*(NOUN)+ in which AT is a determiner, CRD is a cardinal/ordinal, and ADJ is an adjective[3].

Each noun in an identified pattern is extracted and converted to its base form using the stemmer provided in the MIT Java WordNet Interface[4]. For example, lions in the pattern lions and tigers and bears is converted to its base form lion; similarly, tigers is converted to tiger and bears to bear. Each noun found to have a base form in WordNet is then represented as a vertex in $G_{WordNet}$. Therefore, the plural noun lions and the singular noun lion, for example, are conflated to the base form lion, with a single vertex v_{lion} in $G_{WordNet}$ used to represent the different morphological forms of lion.

Graph Edges

Edges in $G_{WordNet}$ represent co-occurrence of noun pairs in coordination patterns. For example, given the pattern

> lions and tigers and bears,

vertices $\{v_{lion}, v_{tiger}, v_{bear}\}$ are generated and all vertex pairs in this set are connected by an edge, as shown in Figure 9.1.

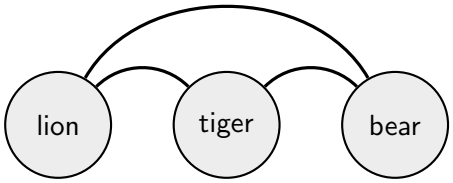


Figure 9.1: Generating $G_{WordNet}$: nouns that co-occur in a coordination pattern are connected.

Edge Weights

$G_{WordNet}$ in its basic form is an undirected, unweighted word co-occurrence graph, thus the order in which nouns occur in lists is ignored, and any two nouns in a coordination pattern are considered to be equally strongly related. This version of $G_{WordNet}$ is used by the WSI system presented in Section 9.3.1.

[3]The regular expression in (9.1) omits details of matching against BNC PoS tags and words (e.g. <w NN1>orange, <w NN2>tigers, <w CJC>and) as this would make the expression difficult to read. Thus NOUN, for example, in the regular expression above is shorthand, taken to match against words tagged in the BNC as: <w NN0> (common noun, neutral for number), <w NN1> (singular common noun), <w NN2> (plural common noun), or <w NP0> (proper noun). The complete list of BNC PoS tags can be found at http://ucrel.lancs.ac.uk/claws5tags.html.

[4]http://projects.csail.mit.edu/jwi/

The WSI systems presented in Section 9.3.2 use edge weighted versions of $G_{WordNet}$. The topology in these graphs is identical to that of the unweighted, undirected graph. Different systems, however, use different sets of edge weights, these being either: Frequency (co-occurrence counts), Conditional Probability, or Log Likelihood Ratio scores (the three vertex association measures defined in Section 3.2.3).

9.2 Clustering Coefficients and Ambiguity

As outlined in Chapter 3, the clustering coefficient (CC) can be used to quantify the ambiguity of words. Given a word w and its neighbourhood $\mathcal{N}(w)$, a high CC value for w indicates strong connectivity within $\mathcal{N}(w)$, implying a high level of semantic relatedness, thus low ambiguity for w. Ambiguity can also be quantified by vertex degree and word frequency. For example, in Véronis (2004) ambiguity is measured by vertex degree, with vertices (words) of low degree taken to be less ambiguous than those with high degree. In Pedersen et al. (2004), word frequency is used as an indicator of ambiguity. Here, the frequency of a word is taken as a measure of its information content (Shannon, 1948), with rarer, low frequency words said to contain more information than words of high frequency. For example, a relatively high frequency word such as *bird* imparts less information than a lower frequency word such as *finch*, as it is less 'surprising' for *bird* to be observed in language. In turn, *finch* contains less information content than lower frequency words such as *chaffinch*, *linnet* or *twite*. Thus, in this phylogeny of finches, specificity counters ambiguity, with specificity reflected in lower frequencies of use.

Dorow (2007) shows that out of the three measures, the clustering coefficient better quantifies ambiguity. Dorow considers all nouns in co-ordination patterns extracted from the BNC that are also defined in WordNet 1.7.1. Pearson's Correlation Coefficient (Stigler, 1989) is then applied to find a possible correlation between WordNet 1.7.1 polysemy counts (the number of senses defined for nouns in WordNet) and frequency, degree, and the CC measure. These correlations are shown in Table 9.1.

	WordNet	Frequency	Degree	CC
WordNet	1	**0.475**	**0.480**	**-0.538**
Frequency		1	0.963	-0.865
Degree			1	-0.884
CC				1

Table 9.1: Correlation between WordNet 1.7.1 polysemy counts and word frequency, vertex degree, and the clustering coefficient (CC).

Dorow finds that the clustering coefficient, with a negative correlation of -0.538, is a stronger indicator of ambiguity than either frequency (0.475) or degree (0.480). Note here that negative correlation is a positive indicator of the measure's ability to quantify ambiguity: CC values decrease as polysemy rises.

Evaluations in this thesis use WordNet 3.0. Accordingly, I applied the same analysis[5] Dorow applies to WordNet 1.7.1 to WordNet 3.0. The results, as shown in Table 9.2, reveal that correlations are far weaker here, and that frequency and degree are stronger indicators of ambiguity than *CC*.

	WordNet	Frequency	Degree	*CC*
WordNet	1	**0.262**	**0.358**	**-0.203**
Frequency		1	0.920	-0.210
Degree			1	-0.269
CC				1

Table 9.2: Correlation between WordNet 3.0 polysemy counts and word frequency, vertex degree, and the clustering coefficient (*CC*).

Figure 9.2 and Table 9.3 provide additional insight. Figure 9.2 plots the correlation between polysemy count and the unweighted clustering coefficient (*CC*), and polysemy count and the weighted clustering coefficient (*WCC*)[6] for all values of polysemy in WordNet 3.0.

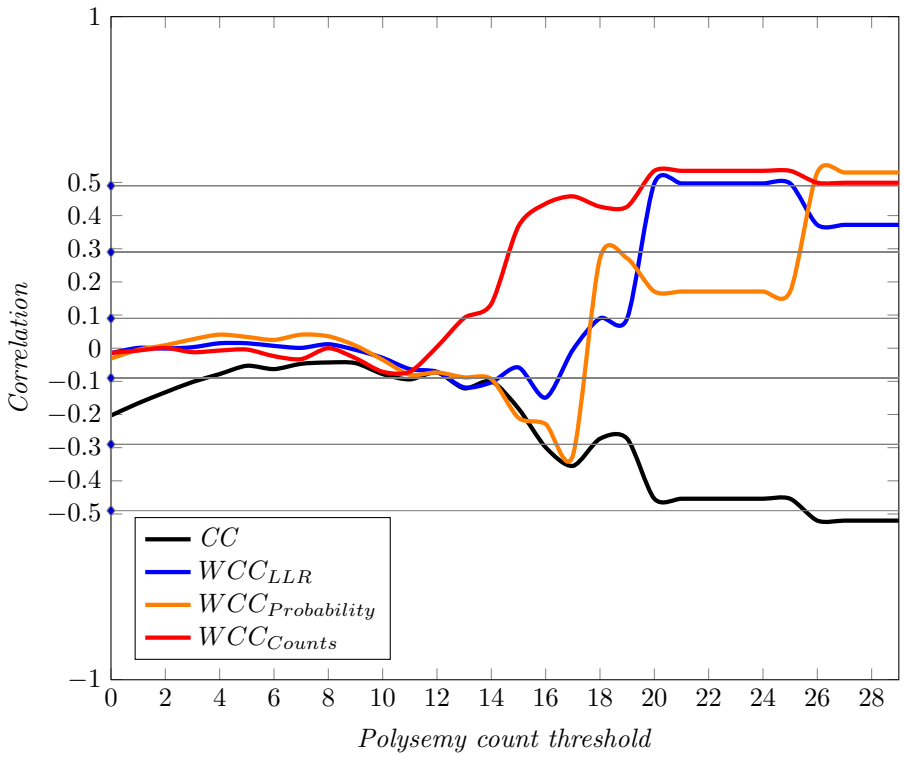


Figure 9.2: Correlation between WordNet 3.0 polysemy counts and the unweighted clustering coefficient (*CC*), and polysemy counts and the weighted clustering coefficient (*WCC*).

Table 9.3 qualifies the correlation values in Figure 9.2, listing the negative and positive

[5] Using the PearsonsCorrelation class in the Commons Math 2.2 package `http://commons.apache.org`

[6] WCC_{LLR} uses log-likelihood ratio values as edge weights, $WCC_{Probability}$ uses directed conditional probabilities, and WCC_{Counts} uses frequency counts.

correlation values for uncorrelated, weak, moderate, and strong correlation, as defined in Cohen (1988). Each *CC* / *WCC* measure is listed against its respective correlation strength.

Strength	**-Correlation**	**+Correlation**	**Measure**	**Corr.**
Uncorrelated	0 to -0.09	0 to 0.09	WCC_{Counts}	-0.014
			WCC_{LLR}	-0.015
			$WCC_{Probability}$	-0.031
Weak	-0.1 to -0.29	0.1 to 0.29	*CC*	-0.203
Moderate	-0.3 to -0.49	0.3 to 0.49		
Strong	-0.5 to -1	0.5 to 1		

Table 9.3: Correlation strength (Cohen, 1988).

Ideally, *CC* / *WCC* values would decrease linearly as polysemy rises, with a negative correlation of -1 indicating a perfect alignment between *CC* / *WCC* values and polysemy count. Evidently, this is not the case here. The majority of words (91.85%) have between one and four senses. For these words, the correlation between polysemy count and *CC* scores is weak; for *WCC* it is so low as to be uncorrelated. Even if the polysemy range is broadened to include all words that have between one and twelve senses (99.68 % of all words), correlation between polysemy and *CC* scores remains weak, with *WCC* still uncorrelated. It is only in the higher order of polysemy count (words having ≥ 12 senses) that a moderate to strong correlation becomes evident. However, as only 126 words (0.47% of all words) have ≥ 12 senses, this moderate to strong correlation for such a small number of words is all but irrelevant.

The correlation analysis shows that *WCC* scores have a moderate to strong *positive* correlation with words of high polysemy. However, looking at the way in which *WCC* is computed reveals that high weights contained in small cliques are amassed to such an extent that (relatively) large scores are returned for words of high polysemy. Highly polysemous words have higher vertex degree, thus even if the topology of the space is weakly connected (low *WCC* values), there remains a larger space in which small cliques with high edges weights may reside. Conversely, *CC* is, as Newman et al. point out, "heavily biased in favour of vertices with low degree because of the factor $k(k-1)$ in the denominator" (Newman et al., 2006, p.287). Given a word w and its neighbourhood $\mathcal{N}(w)$, each neighbour that does *not* connect to any other neighbour diminishes the numerator by $|\mathcal{N}(w)|-1$. The larger $\mathcal{N}(w)$ is, the greater the likelihood that such neighbours will occur, hence lower *CC* values. In simple terms: smaller neighbourhoods have a greater likelihood of forming cliques, therefore of returning higher *CC* scores. Aligning this to WSI, words with low polysemy typically have smaller neighbourhoods than words with high polysemy and so, by default of the way in which the *CC* measure is calculated, return higher scores. This finding raises a possible objection to the use of the unweighted clustering coefficient for WSI, as the measure is intrinsically biased towards less polysemous words. For the

weighted clustering coefficient, this situation is further complicated by the addition of edge weights, with possibly intractable relations holding between graph topology, edge weights, and the size of $\mathcal{N}(w)$. Attempting to give a formal, mathematical definition of the weighted clustering coefficient is a far from simple prospect. Indeed, neither mathematicians (e.g. Kalna and Higham (2006)) nor physicists (e.g. Saramäki et al. (2007)) attempt such an analysis.

Clustering Coefficients and Ambiguity Summary

The correlation analysis for WordNet 3.0 presented in this section indicates that the clustering coefficient and its weighted generalisation are poor indicators of word sense. However, Dorow's analysis for WordNet 1.7.1 finds only a moderate strength correlation between the clustering coefficient and polysemy, yet in evaluations the measure is found to successfully induce word senses. Véronis also finds that a weighted generalisation of the clustering coefficient can be successfully applied to the induction of word senses (Véronis, 2004). As evaluation results in Section 9.5 will show, approaches based on the unweighted and weighted clustering coefficient perform far better than the correlation analysis would suggest.

9.3 Clustering Coefficient Approaches to WSI

This section first considers Dorow's unweighted clustering coefficient (CC) approach to WSI. Noting a number of problems with Dorow's approach, two novel approaches are proposed in Section 9.3.2: the first uses strong versus weak clustering coefficient values to induce senses; the second applies a threshold on graph edge weights.

9.3.1 Dorow's Approach

9.3.1.1 Using Graph Percolation to Find a Clustering Coefficient Threshold

Dorow applies the clustering coefficient (CC) to partition a graph $G_{WordNet}$ into a set of sense clusters. Vertices in $G_{WordNet}$ represent nouns in BNC co-ordination patterns that have a definition listed in WordNet 1.7.1, edges connect co-occurring nouns, and edge weights are word co-occurrence counts for noun pairs in patterns. The clustering coefficient is computed for all vertices in $G_{WordNet}$. Vertices with CC values less than a given threshold θ are deleted, causing the graph to fragment into a set of connected components. Each component is then taken to represent a sense cluster.

Dorow's first task is to find a suitable CC threshold to apply, one that will partition $G_{WordNet}$ into clusters that retain high levels of semantic relatedness between words, and so clearly demarcate word senses. However, finding a suitable threshold to apply is not straightforward. A trade-off exists between cohesion and coverage: setting the CC threshold too high will return semantically coherent clusters but give low coverage; setting the threshold too low will give high coverage but return semantically heterogeneous clusters.

Dorow's solution is to apply the graph theoretical concept of *percolation* (Erdős and Rényi, 1960; Bollobás and Riordan, 2006). Percolation is a *phase transition* in a graph in which an increase/decrease in vertex connectivity causes the structure of the graph to change (transition) from one type of topological state (phase) to another. For example, given an initially sparsely connected graph, if vertex connectivity is iteratively introduced into the graph, a phase transition is said to occur at the point at which a sharp rise in connectivity between the initially sparsely connected subgraphs is observed, with the transition leading to the formation of a giant connected component containing the majority of the graph's vertices. The point at which the phase transition begins is said to be the point at which the graph starts to percolate.

In Dorow's approach, a CC threshold θ acts as the control mechanism for the level of connectivity in $G_{WordNet}$. A CC threshold $\theta = 0$ would allow the highest possible level of connectivity, with only outlier, singleton, vertices left unconnected (high coverage; low cohesion). A CC threshold $\theta = 1$ would allow only those vertices which form cliques to connect (high cohesion; low coverage). Consequently, the aim is to find a suitable value for θ within the range $[> 0, ..., < 1]$.

Given $G_{WordNet}$, Dorow constructs a plot showing the relative size[7] of the largest connected component $Srel(\theta)$ as a function of θ, as illustrated in Figure 9.3.

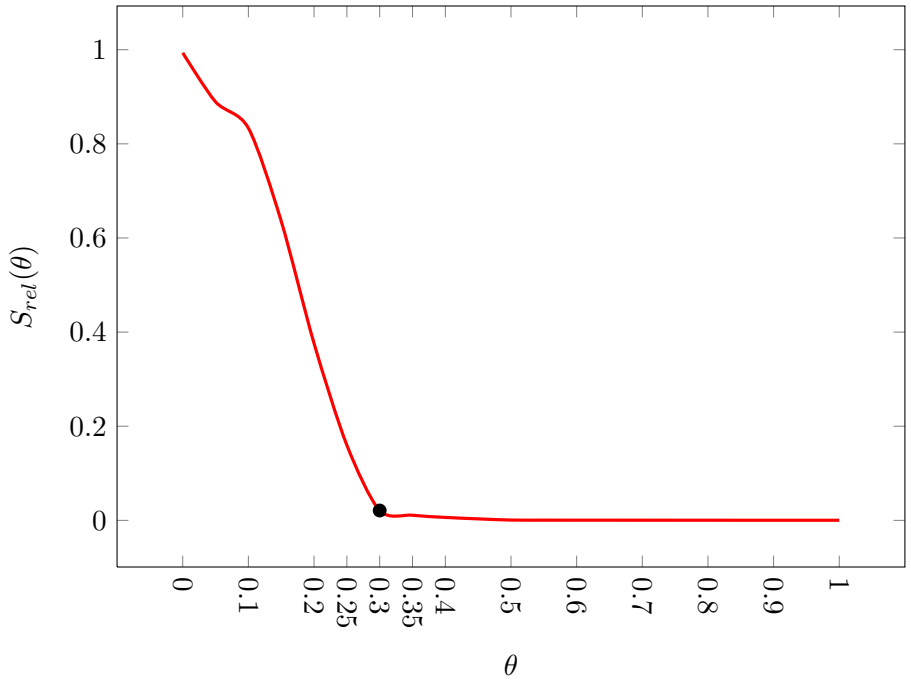


Figure 9.3: Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient (CC) threshold θ.

As Figure 9.3 illustrates, low values of θ allow large connected components to form; for $\theta \simeq 0$ a giant connected component dominates the graph. Conversely, high values of θ result in numerous, small clusters.

[7]The size of the largest connected component in $G_{WordNet}$ divided by the number of vertices in $G_{WordNet}$.

The plateau for $\theta = [0.3, ..., 1]$ indicates that $G_{WordNet}$ is highly fragmented. A phase transition begins at approximately $\theta = 0.3$, thus it is at this CC threshold that the graph starts to percolate[8], with connectivity between clusters in $G_{WordNet}$ sharply increasing for $\theta < 0.3$.

Dorow theorises that a relatively high level of semantic relatedness will hold between cluster words, whilst still preserving reasonable coverage of word senses, if a CC threshold is selected that aligns with the start of percolation. Placing the threshold before the start of percolation may give higher coverage of word senses, though risks distributing particular word senses across many clusters. Placing the threshold value further into the phase transition returns larger, possibly highly homogeneous clusters of semantically related words, in which all words that define a particular sense may be found; however, coverage of word senses will be low.

Applying $\theta = 0.3$ to $G_{WordNet}$ returns 9,249 connected components (clusters), as shown in Figure 9.4.

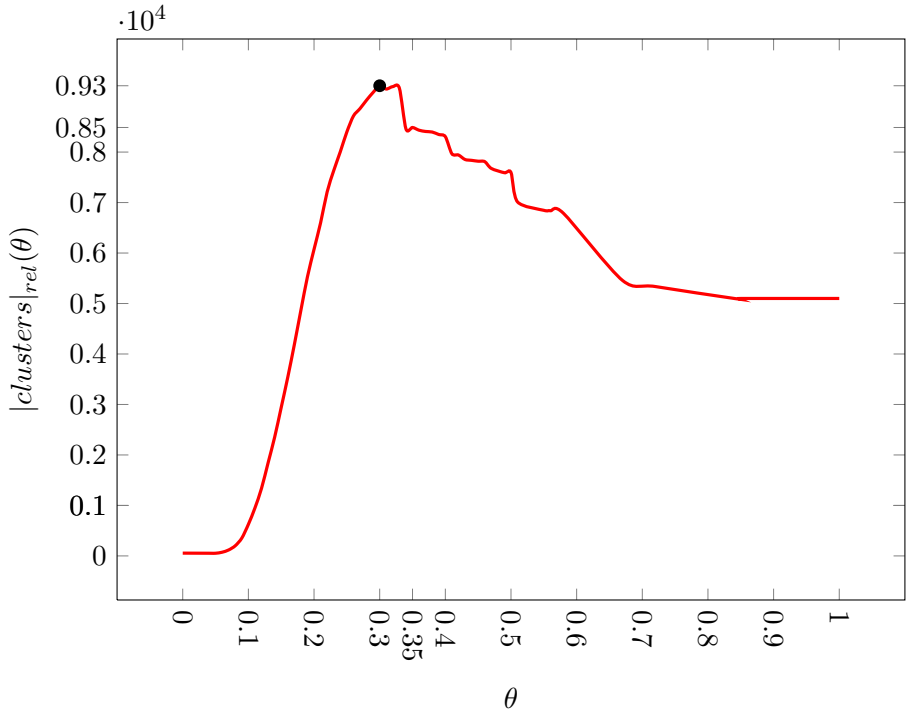


Figure 9.4: The number of clusters $|clusters|_{rel}(\theta)$ returned as a function of the clustering coefficient (CC) threshold θ.

Each of the 9,249 clusters is taken to represent a particular word sense. Thus, the trade-off for returning clusters of (possibly) high semantic cohesion is relatively low coverage; indeed, out of the 27,071 words in $G_{WordNet}$, only the 3,906 words in the 9,249 clusters can have senses induced. These values also suggest that on average each of the 3,906 words has 2.37 senses, as will be clarified in the following section.

[8]The plot shown in Figure 9.3 is for $G_{WordNet}$ (WordNet 3.0) with 0.30 selected as the value for θ. In Dorow, 0.35 is selected as the value for θ. The plots for WordNet 1.7.1 and 3.0 are, however, highly similar see: Dorow (2007) p.67.

9.3.1.2 Dorow's Method for Inducing Word Senses

Having found a clustering coefficient (CC) threshold to apply, Dorow begins by computing CC scores for all vertices in $G_{WordNet}$. A reduced, expository example is given in Figure 9.5.

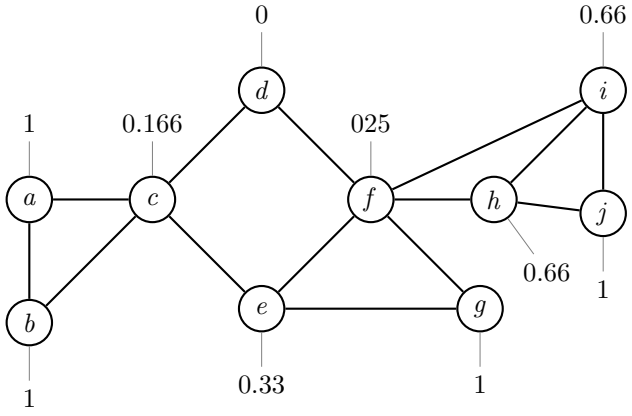


Figure 9.5: Compute clustering coefficient (CC) scores for all vertices in the graph.

where, for example, vertex c is shown to have a CC score of 0.166.

Vertices with CC values less than the CC threshold θ are deleted from the graph along with their adjacent edges, as shown in Figure 9.6.

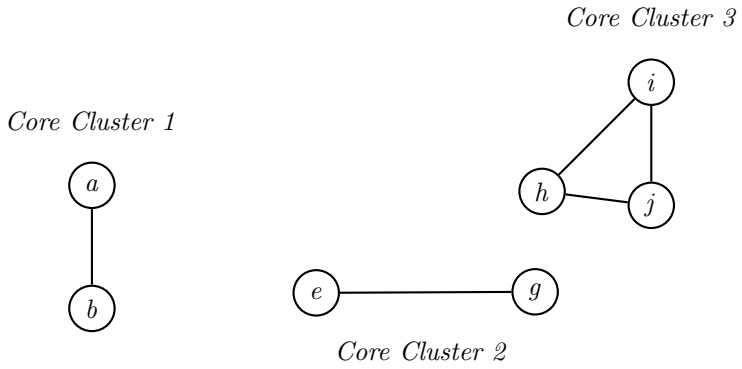


Figure 9.6: Delete vertices with CC scores $< \theta = 0.30$.

The remaining connected components form a hard clustering, with each vertex a member of just one connected component. Each connected component is said to be a core cluster. Core clusters are then expanded by allowing previously deleted vertices with ≥ 2 links to a core cluster admission into that cluster, as illustrated in Figure 9.7.

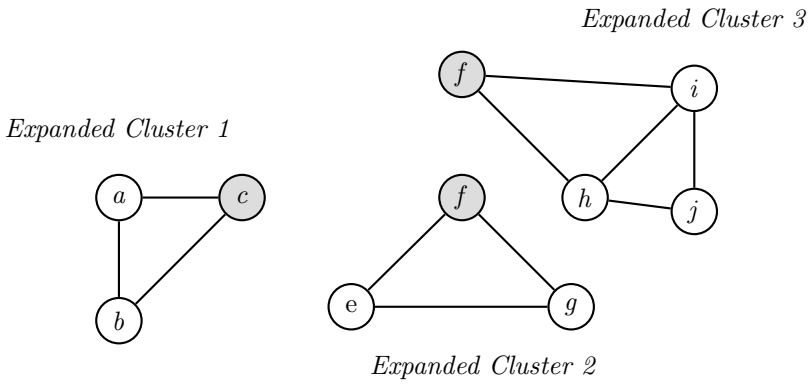


Figure 9.7: Expansion of core clusters.

The expansion step returns a soft clustering, allowing previously deleted vertices to belong to more than one core cluster. Effectively, the words in a core cluster are taken to represent a strong definition of sense. Polysemy of a word, added in the expansion step, is thus indicated by its assignment to ≥ 2 core clusters. For example, vertex f in Figure 9.7 has two links to *Core Cluster 2* and two links to *Core Cluster 3*, thus is admitted to both clusters; the implication being that f has two senses. Dorow's method therefore considers only those words added in the expansion step to be polysemous, with core clusters words taken to be monosemous. Table 9.4 lists a number of clusters returned using Dorow's method.

Core Cluster Words	Expansion Step Words
applewood, fruitwood	*cherry, ivory, pine, oak*
cerise, umber, ultramarine	*red green, ochre, violet, blue*
alstroemeria, gladiolus, freesia, gypsophila	*carnation, rose, dahlia, lily, chrysanthemum, daisy*
tempera, gouache	*pen, paint, oil*
glycerine, lanolin, hexane	*fragrance, oil*
kerosene, gasoline	*fuel, oil*
adenine, uracil, guanine, thymine	

Table 9.4: Examples of clusters returned by Dorow's method.

Dorow's hypothesis therefore is that core clusters, formed of words with relatively high *CC* scores, define a particular sense, with the senses of words added in the expansion step effectively 'induced' by the less ambiguous core cluster words. For example, in Table 9.4 **cerise** is an arguably less ambiguous word than *blue* for defining the sense COLOUR; similarly, **freesia** is less ambiguous than *daisy* for defining the sense FLOWER.

The method used to assign cluster words to senses is described in Section 9.4. However, a number of issues relating to Dorow's approach are discussed first, leading to the introduction of two novel approaches, introduced in Section 9.3.2.

9.3.1.3 Problems with Dorow's Approach

Dorow selects a CC threshold that aligns with the start of a phase transition in $G_{WordNet}$, returning clusters that are neither too small nor too large; of a size, Dorow theorises, that will retain strong semantic coherence. However, attempting to port this method to find a suitable threshold for the weighted clustering coefficient (WCC) proved to be extremely difficult.

Dorow finds a threshold by sight, noting a clearly defined curve (knee) in Figure 9.3. This method translates poorly to WCC. Values returned by WCC are so fine-grained that finding a curve from which to select a suitable threshold becomes too intricate an exercise. Extremely small adjustments to the threshold result in very different clustering solutions: at a particular value of θ, large connected components are returned (thus massive conflation of word senses), however, a slight adjustment to θ can return many, small clusters (thus fragmentation of word senses). For example, given the plot in Figure 9.8, it is not possible to accurately define the point at which the graph starts to percolate.

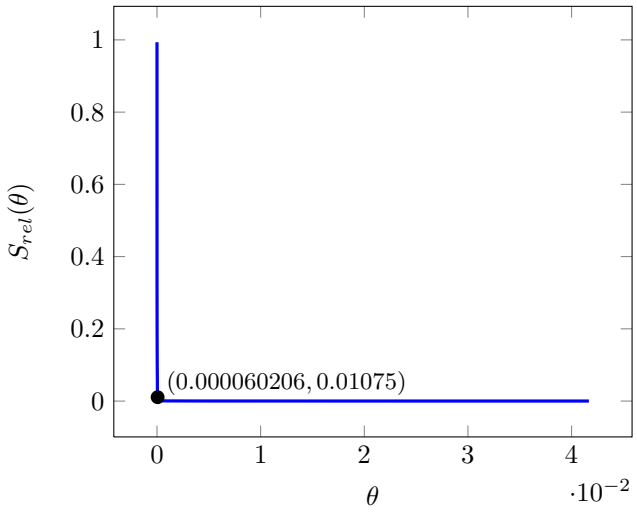


Figure 9.8: Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient threshold θ (WCC).

Even by zooming in on the knee in Figure 9.8, as shown in Figure 9.9, it still remains unclear as to exactly where percolation begins.

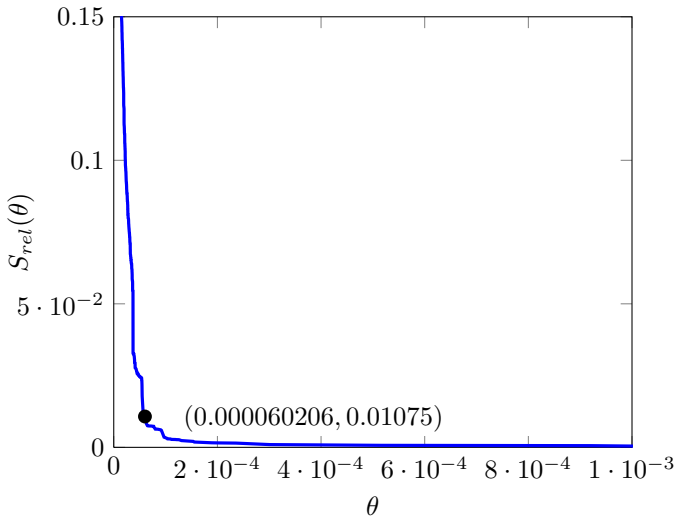


Figure 9.9: Relative size $S_{rel}(\theta)$ of the largest connected component in $G_{WordNet}$ as a function of the clustering coefficient threshold θ (WCC).

If a threshold is selected at what appears, vaguely, by sight, to be a suitable threshold of 0.000060206 then only 2,834 clusters (compared to the 9,249 returned for the unweighted clustering coefficient) are returned, as shown in Figure 9.10.

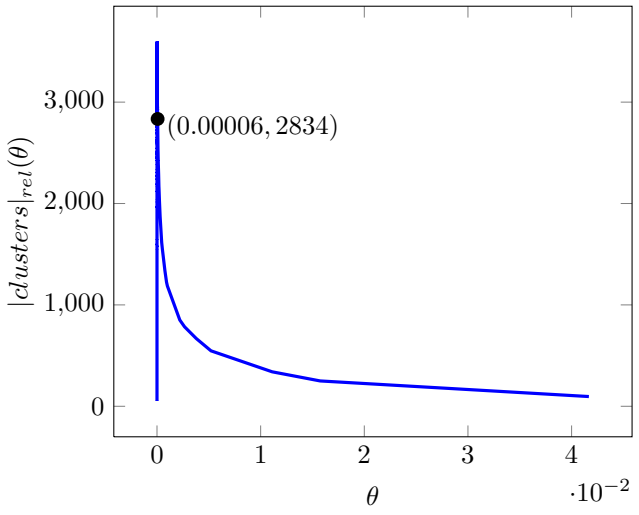


Figure 9.10: The number of clusters $|clusters|_{rel}(\theta)$ returned as a function of the clustering coefficient threshold θ (WCC).

Thus, even if this 'suitable' threshold is selected, the number of clusters returned is so low that word senses would be highly conflated.

An analogy can be made that should, in part, illustrate why finding a threshold for the unweighted clustering coefficient is a far simpler process than it is for the weighted generalisation of the measure. If a graph is viewed as a map, then the clustering coefficient, in its standard, unweighted form, is a measure of the flat, two-dimensional topology of the graph. The unweighted clustering coefficient considers the triangular topology within the neighbourhoods of words, thus, typically, returns coarse-grained results: 1 (all neighbours

are connected), 0.66 (two thirds connected), 0.33 (a third), and so on. Thus, a *CC* threshold of 0.30 in this evaluation is easily interpreted as: 'retain all vertices which have just under a third of their neighbours connected'. The weighted clustering coefficient, however, attempts to combine topology with a measure of 'topography': how 'high' (strong) or 'low' (weak) the topology is. A *WCC* threshold of 0.000060206, as applied above, has no simple interpretation. With hindsight, it is obvious that attempting to combine topology with 'topography' results in a measure that is difficult to analyse; indeed, none of the papers that present weighted generalisations of the clustering coefficient attempt to do this[9]. In the evaluations presented here, the weighted clustering coefficient is simply applied, with the supposition that a measure of topology supplemented by weight may return better results than the unweighted measure. If the results prove this conjecture to be true then analysis of the *WCC* measure can be set as a future research aim.

A further issue arises with Dorow's approach, namely, $n^{\text{th.}}$ order *infections*. The term infections is taken from contagion network theory (Newman et al., 2006), a field of graph-theory which aims to model epidemics by finding *agents* (people) in networks who are particularly susceptible to infection and/or particularly positioned to be able to pass infection on. As Figure 9.11 illustrates, Dorow's approach allows infections to occur, resulting in the conflation of word senses.

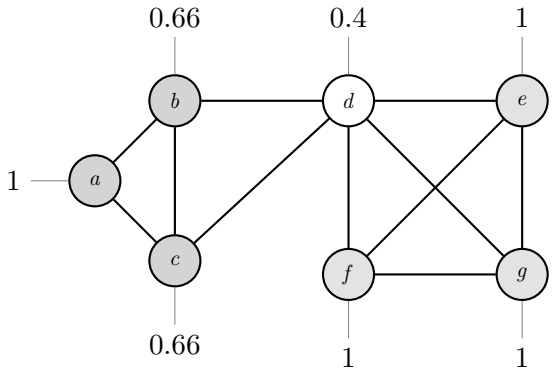


Figure 9.11: $n^{\text{th.}}$ order infections.

Consider two cliques in Figure 9.11: {*a,b,c*} and {*e,f,g*}, noting that vertex *d*, with a *CC* value of 0.4 (higher than the *CC* threshold of 0.3 established in Section 9.3.1), connects the cliques. Vertex *d*, a second-order neighbour of vertex *a*, allows *a* to connect to its third-order neighbours: *e*, *f*, and *g*. However, the third-order neighbours of vertex *a* are both distant and cliquish enough to plausibly represent a very different semantic space from that of the first-order space around *a*. As this example shows, Dorow's approach allows semantically heterogeneous clusters to form via second (or greater) order connections.

[9] Grindrod (2002) provides a probabilistic interpretation, however the implications for the present work are not clear.

9.3.2 Two Novel Approaches

Finding Dorow's approach to be susceptible to $n^{\text{th.}}$ order infections, and noting the difficulty of ascertaining a suitable threshold for the weighted clustering coefficient, two novel approaches are devised. The first approach uses a strong-weak clustering coefficient hierarchy to induce senses; the second applies edge weight thresholding. These approaches alleviate the necessity of finding a clustering coefficient threshold. Both approaches are also less susceptible to infections as they are based on measures that are applied to clearly defined subgraphs of $G_{WordNet}$.

9.3.2.1 A Strong-Weak Clustering Coefficient (SWCC) Approach

The strong-weak clustering coefficient (SWCC) approach presented in this section hypothesises that words with high clustering coefficient scores will better induce senses than those with low clustering coefficient scores. Figure 9.12 illustrates this approach for the graph $G_{\mathcal{N}(w)}$ where: vertex w represents the target word; vertices a, b, c, d, e represent first-order neighbours of w, and θ is the clustering coefficient threshold. Clustering coefficient scores are pre-computed for all vertices in $G_{WordNet}$, thus the score for each vertex v_i in $G_{\mathcal{N}(w)}$ is a measure over its particular first-order space $\mathcal{N}(v_i)$.

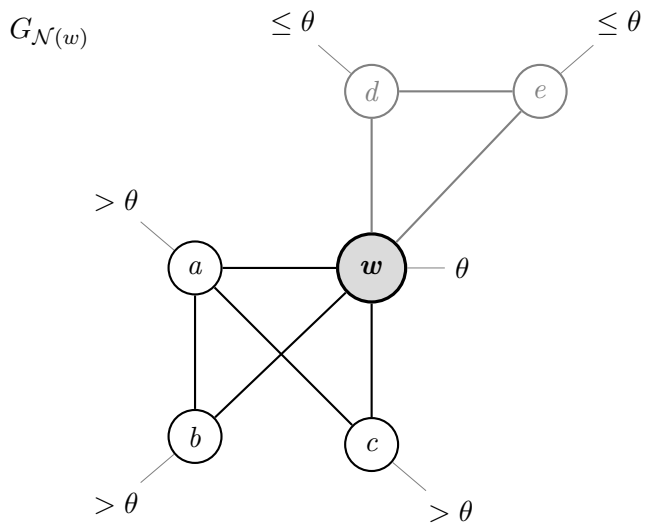


Figure 9.12: A strong-weak clustering coefficient approach.

As Figure 9.12 shows, vertices a, b, and c, have clustering coefficient scores $> \theta$. These vertices are selected to induce the senses of w. The rationale here is that relatively high clustering coefficient scores imply a relatively strong degree of semantic relatedness, thus a, b, and c should better define the senses of w than vertices d and e. The approach is defined algorithmically as follows.

For each target word w:

1. Compute $G_{\mathcal{N}(w)}$, the graph consisting of w and its first-order neighbours.
2. Delete vertices in $G_{\mathcal{N}(w)}$ with a clustering coefficient score $\leq \theta$.

3. Apply the Bron-Kerbosch algorithm to find maximal cliques in $G_{\mathcal{N}(w)}$.

A maximal clique of a graph is a clique (a fully connected subgraph) whose vertices are not contained within a larger clique: Figure 9.13 shows an example.

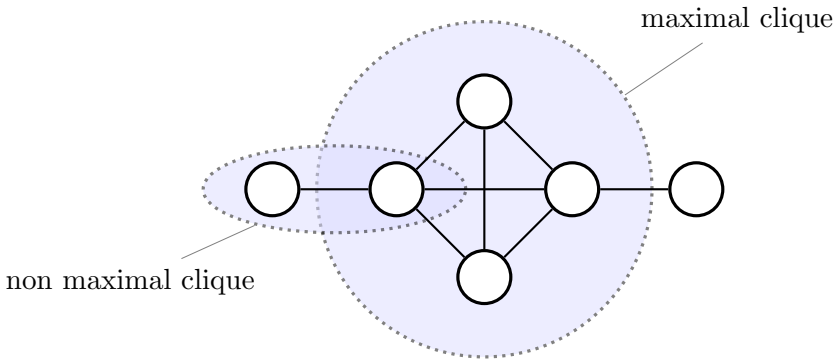


Figure 9.13: A maximal clique.

The Bron-Kerbosch algorithm (Bron and Kerbosch, 1973) is a recursive backtracking algorithm that finds maximal cliques in undirected graphs. Worst-case time complexity is $O(3^{n/3})$ for an n-vertex graph (Tomita et al., 2006). Bron-Kerbosch is considered to be one of the fastest maximal clique finding algorithms (Stix, 2004), found to be more efficient in practice than others (Cazals and Karande, 2008).

The outcome of the SWCC approach is a preference for specific senses of the target word over those more general in nature. All word (vertex) pairs of a maximal clique co-occur in contexts. The hypothesis therefore is that word pairs in a maximal clique will be highly semantically related and so clearly define a sense of the target word. For example, Figure 9.14 shows a small excerpt of the neighbourhood around the target word *mouse*, illustrating a preference for *mouse*'s rodent sense (step 3 in the algorithm returns the maximal cliques {*vole*, *shrew*} and {*vole*, *mole*}) over that of the more general animal/mammal sense (vertices *cat* and *dog* are deleted in step 2).

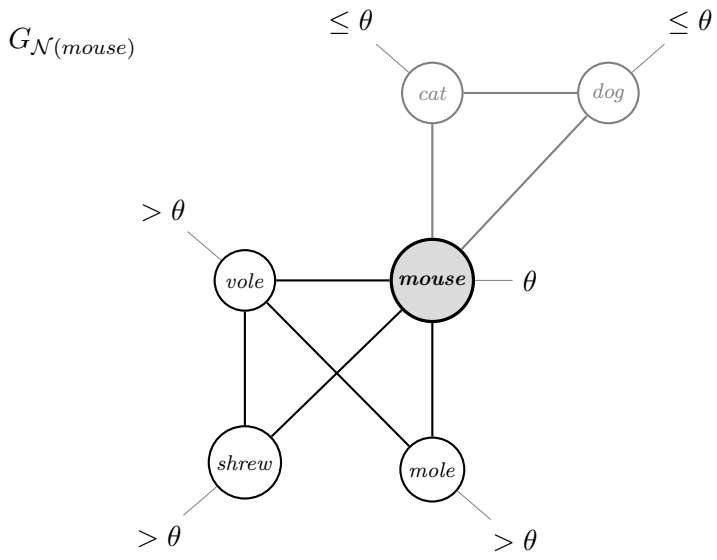


Figure 9.14: A strong-weak clustering coefficient approach for the target word *mouse*.

9.3.2.2 An Edge Weight Thresholding (EWT) Approach

Compared to the approach presented in the previous section, the Edge Weight Thresholding (EWT) approach described in this section is relatively straightforward. This approach allows a comparison to be made (reported in Section 9.5) between a clustering coefficient based approach to WSI and one that applies a 'standard' clustering approach in which edge weights are used to assign vertices to clusters.

In this approach a weighted graph G_w is transformed to a weighted graph $G_{\mathrm{T_w}}$. G_w is a subgraph of $G_{WordNet}$ consisting of a target word and its first order neighbours. Edge weights $w(v_i, v_j)$ between vertices v_i, v_i in G_w are values returned by an association measure $AM(i, j)$ for two nouns i and j found in BNC noun co-ordination patterns. G_w is transformed to $G_{\mathrm{T_w}}$ by deleting edges in G_w with edge weight $< \mathrm{T}_w$, where T_w is some predefined threshold. The approach is defined algorithmically as follows.

For each target word w:

1. Compute G_w, the weighted graph consisting of w and its first-order neighbours, with edge weights defined by an association measure AM.
2. Delete edges in G_w with edge weight $\leq \mathrm{T}_w$ returning the graph G_{T_w}
3. Apply *MaxMax* to return a set of clusters.

Figure 9.15 illustrates the approach. In this particular example edge weights are co-occurrence counts (the number of times word pairs co-occur in noun co-ordination patterns) and the threshold T_w is set to 3.

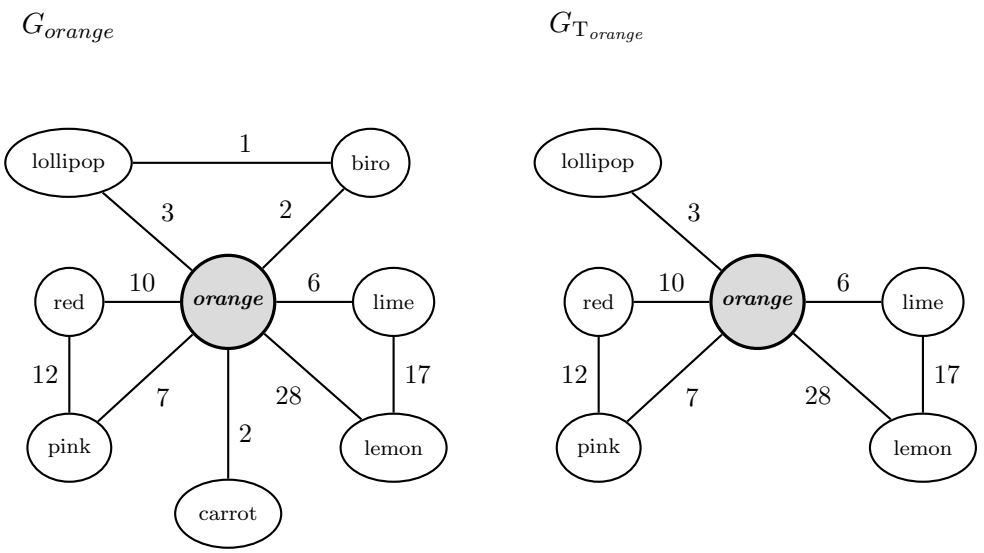


Figure 9.15: The edge weight thresholding approach applied to the target word *orange*. The threshold T_w is set to 3.

Given the graph $G_{\mathrm{T}_{orange}}$ in Figure 9.15, *MaxMax* returns two clusters of ≥ 2 vertices: {red, pink} and {orange, lime, lemon}. These clusters may be interpreted, respectively, as representing *orange*'s COLOUR sense and FRUIT sense. Note that the target word is clustered in this example in order to illustrate how this approach could be used to find predominant senses of target words. The implementation of the EWT approach applied in the evaluation removes the target word in order to avoid bias in the cluster-to-sense mapping procedure described in Section 9.4.1.

9.4 Evaluation Methodology

This section presents the methodology used to evaluate the two novel approaches. The section consists of two parts. The first part describes a method for mapping clusters to senses; the second part defines the evaluation measures used to validate performance. Selecting appropriate evaluation measures for WSI is not straightforward, thus some discussion regarding these measures is given.

9.4.1 Mapping Clusters to Senses

The evaluation of a WSI system requires a method to map clusters to senses. Senses in this evaluation are defined by WordNet 3.0 synsets. Clusters are those returned by either the strong-weak clustering coefficient (SWCC) approach or the edge weight thresholding (EWT) approach. Clusters are said to represent *the hypothesis* and WordNet synsets *the reference*: the 'gold standard' against which the hypothesis is tested. This evaluation uses the mapping method proposed in Pantel and Lin (2002), described as follows.

9.4.1.1 A Method to Map Clusters to Senses

Pantel and Lin (2002) propose a method to map clusters to senses. Given a target word w, its hypothesis C_w, and reference S_w, a similarity score is computed between each cluster $c \in C_w$ and S_w. If the similarity score exceeds a predefined threshold σ then c is said to represent a true sense of w.

Each cluster $c \in C_w$ is mapped to its most similar sense $s \in S_w$, where similarity between c and s is taken to be the average of the similarities between cluster words $\tilde{w}$ and s, defined as:

$$sim(c, s) = \frac{\sum_{\tilde{w} \in c, \tilde{w} \neq w} sim(\tilde{w}, s)}{|\tilde{w} \in c, \tilde{w} \neq w|}. \tag{9.2}$$

Similarity between a cluster word $\tilde{w}$ and a sense s is defined as the maximum similarity between s and a sense $\tilde{s}$ of $\tilde{w}$:

$$sim(\tilde{w}, s) = max_{\tilde{s} \in S(\tilde{w})} sim(\tilde{s}, s), \tag{9.3}$$

with the similarity between two senses computed using the Lin similarity measure (Lin, 1998)[10]:

$$sim(\tilde{s}, s) = \frac{2 \times \log P(lcs)}{\log P(\tilde{s}) + \log P(s)}. \tag{9.4}$$

In (9.4), $P(s)$ is the probability of s. Probabilities are calculated using counts taken from the British National Corpus[11]. *lcs* denotes the lowest common subsuming synset in WordNet for $\tilde{s}$ and s. For example, CITRUS FRUIT is the *lcs* out of all senses (synsets) for the words *lemon* and *orange*.

An Example: Given a target word, for example *orange*, and a cluster $c \in C_{orange} = \{tangerine, lemon, lime\}$, if the senses of the words in c are, on average, maximised by the CITRUS FRUIT sense of *orange* then c is mapped to this sense. If the similarity score $sim(\{tangerine, lemon, lime\}, \text{CITRUS FRUIT}) \geq$ the similarity threshold σ, then c is said to represent a true sense of *orange*.

9.4.2 Evaluating Clusters

The method described in the previous section maps each target word cluster to a sense of the target word, returning a similarity score for each mapped pair. However, an evaluation of the performance of an induction system requires a measure of the validity of these mapped pairs. In this evaluation, the standard external evaluation measures of Precision and Recall are used to validate performance, with Precision for a target word w defined as:

$$Precision_{Strict}(w) = \frac{|\{s_i \in S_w | \exists c_j \in C_w : sim(c_j, s_i) \geq \sigma\}|}{|C_w|}, \tag{9.5}$$

[10]The Lin similarity measure is computed using my Java implementation of WordNet::Similarity (Pedersen et al., 2004). Java WordNet::Similarity can be downloaded from: `http://wordnet.princeton.edu/wordnet/related-projects`.

[11]The target words evaluated are nouns in BNC co-ordination patterns, thus probabilities in Lin's measure are calculated using counts taken from this corpus rather than the default SemCor corpus in WordNet::Similarity.

and Recall for a target word w defined as:

$$Recall(w) = \frac{|\{s_i \in S_w | \exists c_j \in C_w : sim(c_j, s_i) \geq \sigma\}|}{|S_w|}. \quad (9.6)$$

C_w is the set of clusters returned for w by a WSI system. S_w is the set of WordNet senses for w. The numerator in both measures is the number of *one to one* mappings between a cluster c_j and a WordNet sense s_i where $sim(c_j, s_i) \geq \sigma$. Therefore, if more than one cluster maps to a sense, only one of of these clusters is counted in the numerator. Defining Precision here as $Precision_{Strict}$ will be duly explained.

Precision for a WSI system is the average Precision for all words evaluated, defined as:

$$Precision(system) = \frac{1}{|words|} \sum_{w \in words} Precision(w), \quad (9.7)$$

with Recall for a WSI system defined as:

$$Recall(system) = \frac{1}{|words|} \sum_{w \in words} Recall(w). \quad (9.8)$$

As is standard practice, the F-Score is used to give an overall measure of system performance, defined as:

$$F\text{-}Score(system) = \frac{2 \times Precision(system) \times Recall(system)}{Precision(system) + Recall(system)}. \quad (9.9)$$

The measure in (9.5) is Precision in its standard form (Tan et al., 2006, p.297), where each sense in S_w has just one cluster in C_w assigned to it. This is the measure Dorow applies. However, in this evaluation a second measure of Precision is also applied, defined as:

$$Precision_{Relaxed}(w) = \frac{|\{c_i \in C_w | \exists s_j \in S_w : sim(c_i, s_j) \geq \sigma\}|}{|C_w|}. \quad (9.10)$$

This measure allows a *many to one* mapping: many clusters mapped to a single sense, so accounts for situations in which a sense of a target word is distributed across a number of clusters. Note that every cluster mapped to a sense must still pass the similarity threshold σ, thus every cluster mapped to a sense is a true sense according to Pantel and Lin's definition of a true sense. This is arguably a fairer measure of Precision than (9.5) as it allows systems with an ability to return fine-grained sense distinctions to receive scores that reflect this ability. Strictly speaking, (9.10) is a measure of accuracy (Tan et al., 2006, p.149). However, for clarity, this measure will be denoted $Precision_{Relaxed}$, with the measure in (9.5) denoted as $Precision_{Strict}$.

Two Examples

The computation of the Precision and Recall measures is perhaps best illustrated using actual examples (returned by the edge weight thresholding approach). These examples should also demonstrate why $Precision_{Relaxed}$ is arguably a fairer measure for WSI systems

than $Precision_{Strict}$.

Example 1: Table 9.5 lists the four clusters returned for the target word *orange*.

Cluster	⇒ **Sense**$_{orange}$	**Similarity Score**
1. lime, guava, lemon, grapefruit, grape	FRUIT	0.785
2. red, purple, pink, yellow, ultramarine	COLOUR	0.662
3. apple, kiwi, pear, avocado, banana, nut, carrot, melon, pineapple, sugar cane, fig, mango, plantain, blackberry	FRUIT	0.648
4. lollipop, biro	FRUIT	0.187

Table 9.5: Clusters mapped to WordNet senses for the target word *orange*.

WordNet 3.0 defines five senses of *orange*[12], thus the denominator for Recall = 5. Four candidate sense clusters[13] are returned, thus the denominator for both strict and relaxed Precision = 4. Two senses of *orange* are found, FRUIT and COLOUR, resulting in $Recall(orange) = \frac{2}{5} = 0.4$ and $Precision_{Strict}(orange) = \frac{2}{4} = 0.5$.

$\lvert C_{orange}\rvert$	$\lvert S_{orange}\rvert$	1 : 1	M : 1
4	5	2	3

Table 9.6: Values used in the calculation of $Precision_{Strict}$, $Precision_{Relaxed}$ and *Recall* for the target word *orange* where 1:1 is shorthand for the numerator in (9.5) and (9.6), and M:1 for the numerator in (9.10).

However, the FRUIT sense is actually distributed over two clusters, with cluster 3 arguably representing the more general *food*-FRUIT sense of *orange*, and cluster 1 defining a more precise sense of *orange* as *citrus*-FRUIT. Neither cluster is wrong as both map to the WordNet FRUIT sense of *orange*, returning scores which far exceed the similarity threshold $\sigma = 0.25$. Consequently, $Precision_{Relaxed}$ counts both clusters as correct senses of *orange*, returning a score of $\frac{3}{4} = 0.75$.

Example 2: $Precision_{Strict}$ penalises systems that make fine sense distinctions within the remit of the relatively broad sense definitions in WordNet 3.0. This issue is further clarified in Table 9.7 which lists the ten clusters returned for the target word *mouse*. Here,

[12]The five senses defined for *orange* (noun) in WordNet 3.0 are: FRUIT, COLOUR, TREE, PIGMENT, and SOUTH AFRICAN RIVER. As previously noted, WordNet's definitions can be baffling. Indeed, for *orange*, WordNet 3.0 defines sense 2 as *orange colour or pigment* and sense 4 as *any pigment producing the orange colour*. Naturally, such 'double' definitions of the same sense affect Precision and Recall scores.

[13]The cluster $\{lollipop, biro\}$ is mapped to the sense of *orange* it is closest to (FRUIT in this example). That is, sense assignments for clusters are only relative to the senses of the target word, *not* the senses of the words in clusters; as $\{lollipop, biro\}$ does not pass the sense threshold of 0.25 it is considered to bear no relation to any of *orange*'s senses.

two relatively broad senses of *mouse* are found: *mouse* in its ANIMAL sense and *mouse* in its COMPUTING sense.

Cluster	$\Rightarrow$ **Sense**$_{mouse}$	**Similarity Score**
1. hamster, gerbil	ANIMAL	0.796
2. mole, shrew, vole, lemming	ANIMAL	0.637
3. chipmunk, quail	ANIMAL	0.601
4. cat, hyaena, humans, dog, rat, monkey, man, human, porcupines	ANIMAL	0.540
5. rabbit, cow, sheep, deer, pig	ANIMAL	0.519
6. lizard, frog, locust, cockroach, insect, weevil, mammal, beetle, bird, earthworm, vermin, owl, bat, slug, spider	ANIMAL	0.469
7. trackball, joystick, keyboard	COMPUTING	0.460
8.modem, keyboard, monitor, adapter, memory	COMPUTING	0.415
9. mutant, type	ANIMAL	0.323
10. icon, cursor, menu, pointer, key	COMPUTING	0.309

Table 9.7: Clusters mapped to WordNet 3.0 senses for the target word *mouse*.

WordNet 3.0 defines four senses of *mouse*: ANIMAL (rodent), BRUISE TO THE EYE, TIMID PERSON, and COMPUTING (peripheral device), thus $Recall(mouse) = \frac{2}{4} = 0.5$ and $Precision_{Strict}(mouse) = \frac{2}{10} = 0.2$.

$\lvert C_{mouse}\rvert$	$\lvert S_{mouse}\rvert$	$1:1$	$M:1$
10	4	2	10

Table 9.8: Values used in the calculation of $Precision_{Strict}$, $Precision_{Relaxed}$ and *Recall* for the target word *mouse*.

$Precision_{Strict}$ dismisses eight of the ten clusters and so penalises the arguably valid distinctions made here, where: cluster 1 arguably represents the *pet*-ANIMAL sense of *mouse*; cluster 2 its *wild/woodland*-ANIMAL sense, and clusters 4 and 5 the more general *mammal*-ANIMAL sense. Cluster 6 could be interpreted as representing *mouse* in its *pest*-ANIMAL sense, with cluster 9 its *laboratory*-ANIMAL sense. Similarly, for the COMPUTING sense of *mouse*: cluster 8 represents the *computer-peripheral*-COMPUTING sense of *mouse*, with cluster 7 its *input-device*-COMPUTING sense and cluster 10 its *human-computer-interaction*-COMPUTING sense. Importantly, every cluster here passes the similarity threshold $\sigma = 0.25$, thus $Precision_{Relaxed}$ counts each cluster as a contributor in its

numerator, resulting in $Precision_{Relaxed}(mouse) = \frac{10}{10} = 1.0$.

9.5 Evaluation Results

This section reports results for the two novel WSI approaches that were introduced in Section 9.3.2. Results are reported for both $Precision_{Strict}$, as in Dorow (Dorow, 2007), and $Precision_{Relaxed}$, with all evaluations applying a cluster to sense similarity threshold of $\sigma = 0.25$, the same threshold set in Pantel and Lin (2002) and Dorow (2007).

Table 9.9 lists the results returned by Pantel and Lin and Dorow. These results are the best results reported to date for a completely unsupervised approach to WSI.

Measure	**Precision**	**Recall**	**F-Score**
Pantel and Lin (WordNet 1.5)	60.8	50.8	55.4
Dorow (WordNet 1.7.1)	61.1	48.4	54.0

Table 9.9: Results reported in Pantel and Lin (2002) and Dorow (2007).

However, a direct comparison between the results reported in Table 9.9 and those reported in this section cannot be made as WordNet 3.0 is used in this evaluation. Additionally, WSI using WordNet 3.0 is arguably a harder task. WordNet 3.0 contains a greater number of words than WordNet 1.5 and 1.7.1, thus a greater number of senses must be induced.

To allow for a fair comparison, I applied Dorow's approach to WordNet 3.0. The results are shown in Table 9.10.

Measure	**Precision**	**Recall**	**F-Score**
Dorow $Precision_{Strict}$	54.5	40.7	46.6
Dorow $Precision_{Relaxed}$	60.9	40.7	48.8

Table 9.10: Results for Dorow's approach using WordNet 3.0.

The results reported in Table 9.10 for Dorow's approach applied using WordNet 3.0 are lower than those reported in Dorow (2007) using WordNet 1.7.1, thus support the conjecture that inducing word senses using WordNet 3.0 is a harder task.

9.5.1 Strong-Weak Clustering Coefficient Approach Results

Results for the strong-weak clustering coefficient (SWCC) approach are shown in Tables 9.11 and 9.12.

Measure	Precision	Recall	F-Score
CC	15.1	67.7	24.7
WCC_{Counts}	9.1	66.1	15.9
WCC_{LLR}	12.5	66.3	21.1
$WCC_{Probability}$	15.3	67.2	24.9
$Baseline_{Cliques}$	7.0	**73.5**	12.8
$Baseline_{ConnectedComponents}$	48.2	37.6	42.3
$Dorow$	**54.5**	40.7	**46.6**

Table 9.11: Strong-weak clustering coefficient approach, $Precision_{Strict}$.

Measure	Precision	Recall	F-Score
CC	51.9	67.7	58.8
WCC_{Counts}	58.8	66.1	62.2
WCC_{LLR}	55.9	66.3	60.6
$WCC_{Probability}$	52.0	67.2	58.6
$Baseline_{Cliques}$	**76.7**	**73.5**	**75.1**
$Baseline_{ConnectedComponents}$	61.7	37.6	46.7
$Dorow$	60.9	40.7	48.8

Table 9.12: Strong-weak clustering coefficient approach, $Precision_{Relaxed}$.

In Tables 9.11 and 9.12, CC is the unweighted clustering coefficient and WCC is the novel weighted clustering coefficient that was introduced in Section 4.3. WCC_{Counts} uses word pair frequency counts as edge weights, with WCC_{LLR} using the log likelihood ratio and $WCC_{Probability}$, conditional probability in directed graphs. All measures are taken over subgraphs $G_w \subset G_{WordNet}$, where G_w consists of a target word w (one of the 27,071 target words evaluated), and its first-order neighbourhood $\mathcal{N}(w)$. Results for two baselines are also shown: $Baseline_{Cliques}$ is maximal cliques found in $G_w \subset G_{WordNet}$, and $Baseline_{ConnectedComponents}$ is the set of connected components found in $G_w \subset G_{WordNet}$.

Results show that the strong-weak clustering coefficient approach returns higher Recall (highest Recall, 67.7 for CC) than Dorow's approach (40.7), indicating a greater coverage of word senses.

Results for $Precision_{Strict}$ (one cluster mapped to one sense) are very poor, yet looking at the scores for $Precision_{Relaxed}$ (many clusters mapped to one sense) finds the results much improved; just short of Dorow: 58.8 (WCC_{Counts}), 60.9 (Dorow). These results indicate that the strong-weak clustering coefficient approach finds true senses of words (in accordance with Pantel and Lin's definition of a true sense) distributed over many clusters. This is to be expected given the input data. Target words are taken from noun

co-ordination patterns extracted from a relatively small corpus wherein words which define a particular sense could not be expected to neatly coalesce into one cluster.

If $Precision_{Relaxed}$ is accepted as a fair measure for the evaluation of WSI systems (as it is here), and the F-Score is taken to measure overall system performance (as is standard practice), then the strong-weak clustering coefficient approach is shown to outperform Dorow's approach by some margin, returning a highest F-Score of 62.2 (WCC_{Counts}) compared to Dorow's approach (48.8).

There appears to be little difference between applying unweighted and weighted clustering coefficients, with WCC approaches returning only marginally higher F-Scores: 24.7 (CC) compared to 24.9 ($WCC_{Probability}$) using $Precision_{Strict}$; 58.8 (CC) compared to 62.2 (WCC_{Counts}) using $Precision_{Relaxed}$. The application of three different edge weightings in the WCC approaches draws no firm conclusion as to which is best to apply.

A surprising outcome of the evaluation (accepting $Precision_{Relaxed}$ as a valid evaluation metric) is that $Baseline_{Cliques}$ returns by far the best results. As Figure 9.16 shows, $Baseline_{Cliques}$ returns F-Score results right across the similarity threshold range $\sigma = [0, .., 1]$ that are better than or equal to those of the best performing WSI approaches.

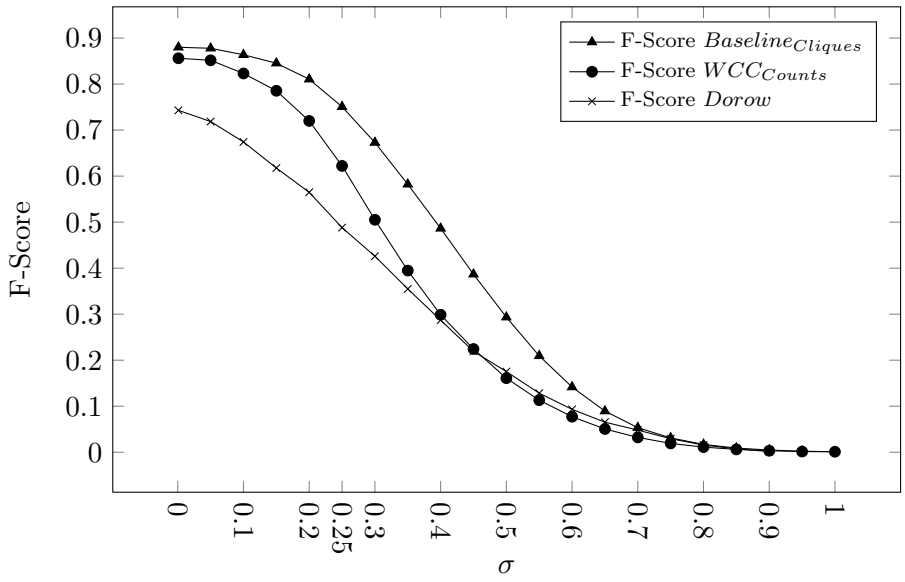


Figure 9.16: F-Scores for $Baseline_{Cliques}$, WCC_{Counts}, and *Dorow* for all values of the similarity threshold σ.

Computing maximal cliques is a straightforward approach: vertices that form a clique are simply taken to represent a word sense. However, computing maximal cliques is expensive, with a worst case run-time of $O(3^{V/3})$ using the Bron-Kerbosch algorithm (Bron and Kerbosch, 1973). This result also leaves two rather unsatisfying conclusions: 1.) that the relatively complex process of devising and computing clustering coefficient measures amounted to very little; 2.) that the best performing system, $Baseline_{Cliques}$, which uses no measure of word association, provides no insight into how word senses are formed.

9.5.1.1 Evaluation of Six Weighted Clustering Coefficient Measures in the Strong-Weak Clustering Coefficient Approach

The previous section reported results for *WCC*, the novel weighted clustering coefficient measure that was introduced in Section 4.3. This section compares the performance of this measure with the performance of five other weighted clustering coefficient measures, namely: *Barrat* (Barrat et al., 2004), *Lopez-Fernandez* (Lopez-Fernandez et al., 2004), *Onnela* (Onnela et al., 2005), *Opsahl & Panzarasa* (Opsahl and Panzarasa, 2009), and *Zhang & Horvath* (Zhang and Horvath, 2005) - the five measures that were described in Chapter 4. All six measures, along with the unweighted clustering coefficient *CC*, are evaluated in the strong-weak clustering coefficient approach to WSI, with results reported in Tables 9.13 and 9.14.

Measure	Precision	Recall	F-Score
CC	15.1	67.7	24.7
$Zhang\,\&\,Horvath_{Counts}$	10.3	66.7	17.8
$Zhang\,\&\,Horvath_{LLR}$	13.7	66.9	22.7
$Zhang\,\&\,Horvath_{Probability}$	**16.4**	**68.2**	**26.4**
WCC_{Counts}	9.1	66.1	15.9
WCC_{LLR}	12.5	66.3	21.1
$WCC_{Probability}$	15.3	67.2	24.9
$Onnela_{Counts}$	8.9	65.2	15.7
$Onnela_{LLR}$	11.6	65.1	19.7
$Onnela_{Probability}$	14.6	65.9	23.9
$Barrat_{Counts}$	6.5	62.1	11.7
$Barrat_{LLR}$	9.8	61.7	16.9
$Barrat_{Probability}$	11.7	62.5	23.9
$Opsahl\,\&\,Panzarasa_{Counts}$	6.0	62.4	11.0
$Opsahl\,\&\,Panzarasa_{LLR}$	9.5	62.3	16.5
$Opsahl\,\&\,Panzarasa_{Probability}$	11.6	62.2	19.7
$Lopez\text{-}Fernandez_{Counts}$	4.7	54.3	8.7
$Lopez\text{-}Fernandez_{LLR}$	5.2	54.1	9.5
$Lopez\text{-}Fernandez_{Probability}$	6.8	55.3	12.1

Table 9.13: $Precision_{Strict}$ results for the clustering coefficient measures.

Results in Tables 9.13 and 9.14 show that *Zhang & Horvath* is the best performing measure, thus lending support to Kalna and Higham's view that "out of the possible ways that have been proposed to generalise the clustering coefficient to the case of a weighted network, there is one very promising candidate; namely the Zhang-Horvath version" (Kalna and Higham, 2006, p.8). That said, Tables 9.13 and 9.14 also show that there are three distinct levels of performance, with the three best performing measures (*Zhang & Horvath*, *WCC*, *Onnela*) the only measures shown to outperform the unweighted *CC* measure and whose results are clearly separated from those of the fourth (*Barrat*) and fifth (*Opsahl &*

Panzarasa) best performing measures, which, in turn, are clearly separated from those of *Lopez-Fernandez* - by far the worst performing measure, a likely result of this measure ignoring potentially useful information. Indeed, results in Tables 9.13 and 9.14 indicate that the less information a measure uses the worse its performance is. For example, *Lopez-Fernandez*, which uses edge weights between neighbours of a target word vertex (thus ignoring two of the three edge weights in each triangle around a target word vertex), performs worse than *Barrat* and *Opsahl & Panzarasa*, which use edge weights between the target word vertex and its neighbours (thus ignoring just one of the three edge weights in each triangle around a target word vertex).

Measure	**Precision**	**Recall**	**F-Score**
CC	51.9	67.7	58.8
$Zhang\,\&\,Horvath_{Counts}$	**60.9**	**66.9**	**63.7**
$Zhang\,\&\,Horvath_{LLR}$	59.4	66.7	63.0
$Zhang\,\&\,Horvath_{Probability}$	57.3	68.2	62.3
WCC_{Counts}	58.8	66.1	62.2
WCC_{LLR}	55.9	66.3	60.6
$WCC_{Probability}$	52.0	67.2	58.6
$Onnela_{Counts}$	58.4	65.2	61.6
$Onnela_{LLR}$	54.8	65.1	59.5
$Onnela_{Probability}$	50.2	65.9	58.6
$Barrat_{Counts}$	48.7	62.1	54.6
$Barrat_{LLR}$	46.9	61.7	53.3
$Barrat_{Probability}$	45.4	62.5	52.6
$Opsahl\,\&\,Panzarasa_{Counts}$	48.8	62.4	54.8
$Opsahl\,\&\,Panzarasa_{LLR}$	46.7	62.3	54.4
$Opsahl\,\&\,Panzarasa_{Probability}$	46.1	62.2	52.9
$Lopez\text{-}Fernandez_{Counts}$	37.3	54.3	44.2
$Lopez\text{-}Fernandez_{LLR}$	36.6	54.1	43.6
$Lopez\text{-}Fernandez_{Probability}$	34.7	55.3	42.6

Table 9.14: $Precision_{Relaxed}$ results for the clustering coefficient measures.

It is also notable that the second and third best performing measures, *WCC* and *Onnela*, have the same 'particular advantages' that *Zhang & Horvath* has (Kalna and Higham, 2006) as they collapse to the binary (unweighted CC) case if edge weights in the graph are 0 or 1 and use each potentially useful item of information in the target vertex subgraph at some point in their calculation: all edges; all edge weights.

Two further observations can be made about the evaluation results: firstly, all six measures return higher Recall than Precision (whereas the opposite is true for the baseline measures and *Dorow* in Tables 9.11 and 9.12); secondly, the use of word co-occurrence counts as edge weights returns the best results in the $Precision_{Relaxed}$ evaluation (Table 9.14), yet the use of word pair conditional probability as edge weights returns the

best results in the $Precision_{Strict}$ evaluation (Table 9.13). However, it is unclear as to why this should be the case. Indeed, it is unclear how one could begin to analyse, and so compare, weighted clustering coefficient measures 'in vitro', that is, separate from an evaluation of their performance as utilities of some system (as is done here), as analysis of measures that attempt to quantify both the connectivity structure (topology) and the weights ('topography') of a graph is, I believe, a mathematically intractable problem. Indeed, even the more mathematically inclined papers on the subject (Grindrod, 2002; Onnela et al., 2005; Saramäki et al., 2007) side-step this issue. Furthermore, given that the unweighted clustering coefficient can be analysed from various mathematical, graph-theoretic, and probabilistic perspectives (Watts and Strogatz, 1998; Dorow et al., 2005; Newman et al., 2006), this also raises doubts as to whether a true generalisation of the unweighted clustering coefficient to the weighted case can be given.

9.5.2 Edge Weight Thresholding Approach Results

Results for the edge weight thresholding (EWT) approach are shown in Tables 9.15 and 9.16 where: |Words| is the number of words that can be evaluated by a particular measure; *Baseline* is the set of connected components found in $G_{WordNet}$[14]; *Counts* uses word co-occurrence counts as edge weights thresholded at a word co-occurrence count of 3, and LLR use log likelihood ratio scores thresholded at various levels (see Table 3.1 in Section 3.2.3 for details regarding the significance of these values).

Measure	**Precision**	**Recall**	**F-Score**	**|Words|**
$LLR = 15.13$	**60.2**	53.2	56.5	11138
$LLR = 10.83$	51.9	53.1	52.5	16850
$LLR = 6.63$	43.2	55.8	48.7	21716
$LLR = 3.84$	38.7	**59.1**	46.8	22899
Counts	55.7	58.6	**57.1**	9101
Baseline	47.7	41.6	44.4	27071
Dorow	54.5	40.7	46.6	3906

Table 9.15: Edge Weight Thresholding approach, $Precision_{Strict}$.

[14]Finding all maximal cliques in $G_{WordNet}$ proved to be prohibitively expensive to compute.

Measure	Precision	Recall	F-Score	|Words|
$LLR = 15.13$	72.1	53.2	61.2	11138
$LLR = 10.83$	65.5	53.1	58.7	16850
$LLR = 6.63$	59.6	55.8	57.6	21716
$LLR = 3.84$	56.6	**59.1**	57.8	22899
Counts	**74.2**	58.6	**65.5**	9101
Baseline	49.9	41.6	45.4	27071
Dorow	60.9	40.7	48.8	3906

Table 9.16: Edge Weight Thresholding approach, $Precision_{Relaxed}$.

The results in Tables 9.15 and 9.16 show that the edge weight thresholding approach outperforms the strong-weak clustering coefficient approach. Using $Precision_{Strict}$, the highest F-Score for the edge weight thresholding approach is 57.1 ($Counts$); for the strong-weak clustering coefficient approach, 24.9 ($WCC_{Probability}$). Using $Precision_{Relaxed}$, the highest F-Score for the edge weight thresholding approach is 65.5 ($Counts$); for the strong-weak clustering coefficient approach, 62.2 (WCC_{Counts}).

Results are also better than those reported in Dorow (2007) and Pantel and Lin (2002), with coverage of word senses far higher. For example, at a $LLR = 3.84$, nearly six times as many words are evaluated than in Dorow (22,899 compared to 3,906), with over twenty four times the number of words evaluated than in Pantel and Lin (941 word sense clusters). Even at this level of coverage, the edge weight thresholding approach still outperforms Dorow's approach, with F-Scores of 46.8 for $Precision_{Strict}$ and 57.8 for $Precision_{Relaxed}$ compared to Dorow's approach at 46.6/48.8.

Scores for $Precision_{Strict}$ and $Precision_{Relaxed}$ are both higher and closer in value than those obtained for the soft-weak clustering coefficient approach. This suggests that a greater number of words which define a particular sense are drawn together in one cluster.

Considering the Log Likelihood Ratio (LLR) measure, the highest LLR threshold, set at 15.13, returns the best Precision results, with the lowest threshold, set at 3.84, returning the best Recall results. This is to be expected. High LLR thresholds allow only strong word associations to pass into the transformed graph, resulting in more precise definitions of word sense. Weaker LLR thresholds allow a greater number of word associations to pass into the transformed graph, thus noise may be introduced, resulting in lower Precision; however, the greater number of word associations here provides broader coverage of word senses, thus higher Recall.

Out of the five measures applied, the *Counts* measure is shown to return the best score. Coverage here, at 9101 words, is the lowest out of all edge weight thresholding measures, yet is still over twice the number of words evaluated in Dorow and nearly ten times the number of words considered in Pantel and Lin. Importantly, F-Scores for this measure, at 57.1 ($Precision_{Strict}$) and 65.5 ($Precision_{Relaxed}$), are higher than those reported for Dorow's approach (46.6/48.8) and in Pantel and Lin (2002) (55.4). Thus,

the relatively straightforward approach of counting the number of times words co-occur in contexts outperforms all other approaches. This result echoes the findings of the previous evaluation where the relatively straightforward approach of $Baseline_{Cliques}$ was found to return the best results. The results for both evaluations therefore suggest that complex approaches to WSI may not be required.

As Figure 9.17 shows, if $Precision_{Strict}$ is applied in the F-Score calculations, the edge weight thresholding (WCC_{Counts}) approach returns F-Score results right across the similarity threshold range $\sigma = [0, .., 1]$ that are better than or equal to those of the best performing WSI approaches[15].

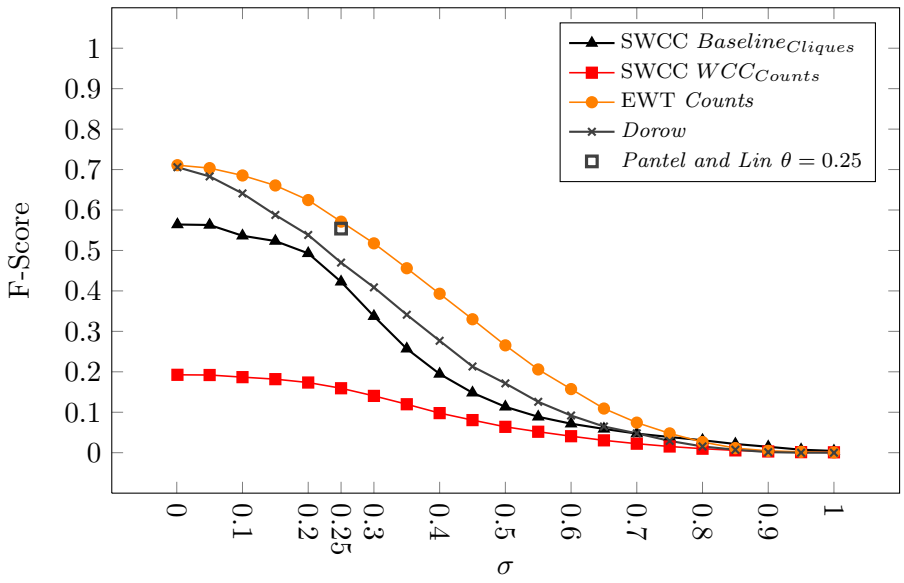


Figure 9.17: F-Scores ($Precision_{Strict}$) for the highest scoring measures in the strong-weak clustering coefficient (SWCC) approach and the edge weight thresholding (EWT) approach, all values of the cluster-sense similarity threshold σ.

As Figure 9.18 shows, if $Precision_{Relaxed}$ is applied in the F-Score calculations, the baseline approach, $Baseline_{Cliques}$, returns F-Score results for $\sigma = [0, .., 0.5]$ that are better than or equal to those of the best performing approaches, with only the edge weight thresholding (WCC_{Counts}) approach returning better results for $\sigma > 0.5$.

[15]Pantel and Lin only report results for $\theta = 0.25$, as indicated in Figure 9.17.

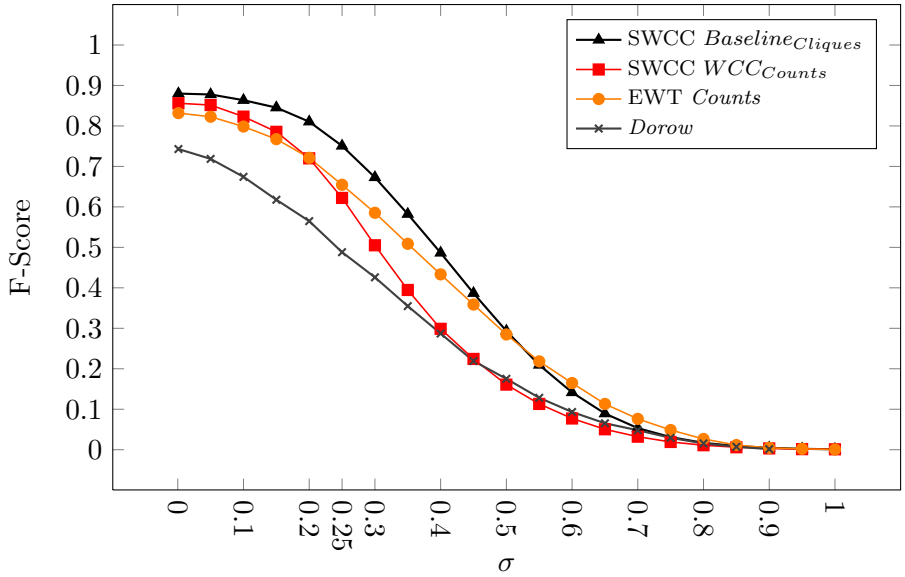


Figure 9.18: F-Scores ($Precision_{Relaxed}$) for the highest scoring measures in the strong-weak clustering coefficient (SWCC) approach and the edge weight thresholding (EWT) approach, all values of the cluster-sense similarity threshold σ.

9.6 Conclusions

The previous chapter exemplified a graph-based approach to WSI, though evaluated systems on just six polysemous words. This chapter presented a far more comprehensive evaluation, requiring WSI systems to induce the senses of 27,071 words.

A key aim of the evaluation was to assess the induction performance of weighted generalisations of the clustering coefficient; the hypothesis being that a weighted generalisation would better delineate word senses than the unweighted measure. As Dorow reports results for the unweighted clustering coefficient, the initial proposal was to apply Dorow's methodology, so allowing a direct comparison to be made between the two measures. However, this proved to be far from straightforward. Firstly, the moderately strong correlation Dorow found between polysemy and clustering coefficient values for WordNet 1.7.1 was not evident for WordNet 3.0, where correlation was at best weak (the unweighted clustering coefficient) and at worst non existent (the weighted clustering coefficient). Secondly, Dorow's method for finding a suitable clustering coefficient threshold to apply proved to be illusory for the weighted clustering coefficient. Thirdly, analysis of the connected components returned by Dorow's method showed that $n^{\text{th.}}$ order infections occur, resulting in the conflation of word senses. These findings led to the introduction of two novel approaches, with the first applying a strong-weak clustering coefficient approach and the second applying an edge weight thresholding approach.

A correlation analysis of the clustering coefficient and its weighted generalisation showed that both measures are poor predictors of word sense. However, this was not borne out in the evaluations where the clustering coefficient approaches applied here were shown to outperform both Dorow's and Pantel and Lin's approaches, thus exceed the best

results reported to date. Coverage was also far higher than either of these approaches, thus counter Dorow's claim that coverage is traded for cohesion.

Results for the weighted clustering coefficient exceed those of the unweighted clustering coefficient. However, the expectation that the weighted clustering coefficient would convincingly outperform its unweighted counterpart was shown to be unfounded. Furthermore, the weighted clustering coefficient is simply applied in this evaluation; a limitation of this evaluation, therefore, is that no formal analysis equivalent to that presented in Watts and Strogatz (1998) and Eckmann and Moses (2002) for the unweighted clustering coefficient is given. However, neither mathematicians (Kalna and Higham, 2006) nor physicists (Saramäki et al., 2007) attempt such an analysis, with graph theoreticians tending to pass on analysis of weighted graphs altogether (as evident in Newman et al.'s comprehensive anthology of graph theory papers (Newman et al., 2006)).

A surprising outcome of the evaluations was that the best results were obtained by two straightforward approaches, neither of which applies clustering coefficients: if a measure of accuracy ($Precision_{Relaxed}$) is accepted as a valid measure, then computing maximal cliques within the neighbourhoods of target words outperforms all other approaches; if accuracy is rejected, with the measure of Precision replacing it ($Precision_{Strict}$), then the best approach is to use co-occurrence counts between words. These results suggest possibilities for future research, where simple WSI methods would be applied to corpora far larger than the corpus used here (BNC), the expectation being that greater volumes of co-occurrence data will better delineate word senses[16].

[16]I originally intended to supply the WSI systems presented in this chapter with word co-occurrence counts taken from the Google Web 1T corpus, a corpus consisting of approximately one trillion words (10,000 times the size of the BNC). I first indexed Google Web 1T using Lucene, then implemented a search program to query the index and return a count of the number of times a query appears in the corpus, e.g. the query "serve as the inspiration" appears 1390 times in the corpus. However, the use of Google Web 1T was abandoned as: 1.) the largest context (n-gram) size is just five words, and 2.) the counts are not guaranteed to be correct. For example, the count for a context of size n can be higher than the counts for contexts of size *n-1*: $count(w_1, w_2, w_3) > (count(w_1, w_2), count(w_2, w_3))$.
Google Web 1T: `http://www.ldc.upenn.edu`
Lucene: `http://lucene.apache.org`

Chapter 10

The SemEval Word Sense Induction Task

This chapter introduces three novel Word Sense Induction (WSI) systems. The systems are evaluated within the framework of the SemEval-2010 Word Sense Induction & Disambiguation task (Manandhar et al., 2010), with results compared to those of the twenty six participant systems. For the two WSI evaluations, results show that the three novel systems return the best results in one evaluation yet return the worst results in the other. These findings lead to an analysis of the metrics applied in the evaluations which finds them biased to prefer degenerate clustering solutions..

10.1 The SemEval-2010 Word Sense Induction and Disambiguation Task

The SemEval-2010 Word Sense Induction and Disambiguation task (Manandhar et al., 2010) follows on from the task set by Agirre and Soroa in SemEval-2007 (Agirre and Soroa, 2007). The task is to devise a system that will induce the senses of a given set of target words, with the system then evaluated using software provided by the task's co-ordinators. Participants are required to induce the senses of 100 target words: 50 verbs and 50 nouns. Systems are then assessed using: 1.) two unsupervised WSI evaluations; 2.) a supervised Word Sense Disambiguation (WSD) evaluation.

Manandhar et al. provide a training set and a test set. These sets consist of target word instances: sentences and/or paragraphs containing a target word. The training set is generated using a web-based method which first finds all WordNet senses of a target word *tw* and then creates search queries[1] of the type:

$$tw_{\text{SENSE}_i} \text{ AND } tw_{\text{SENSE}_i}\{\text{hypernyms, hyponyms, synonyms, meronyms, holonyms}\}.$$

For example, a query for the target word *gas* in its ENERGY sense might be *gas* AND fuel, as fuel is a hypernym of the target word *gas*. For each page returned, all HTML paragraphs are extracted, with each paragraph containing the target word taken to represent an

[1] Queries are issued to the *Yahoo!* search API: `http://developer.yahoo.com`

instance of a target word sense. The test set is extracted from OntoNotes (Hovy et al., 2006), where the target words are the same as those in the training set. Test set text is extracted from North American news websites (e.g. CNN and ABC, amongst others), with each target word tagged with an OntoNotes sense. Participants are required to tag each instance in the test set with a sense of the target word; the sense being derived by the participant's induction system. Systems are allowed to incorporate morphological and syntactic components. For example, lemmatisation of instance words and part of speech tagging of instances is allowed. No other external resources are permitted, thus participants can only use co-occurrence and/or distributional similarity data from the test/training instances.

The original premise of the task implied that participants should induce word senses in the test set using clusters obtained from the training set. However, it became apparent through discussions on the task website[2] that the training set is not a prerequisite for sense induction. Consequently, systems that are able to induce senses directly from the test set were allowed to do so. Indeed, it might be suggested that provision of a training set rather defeats the point of devising induction systems, where the task is to find word senses without recourse to training data. However, the task's co-ordinators supply a training set in order to give participants who wish to apply k-type clustering algorithms the ability to tune k and so return an 'optimal' clustering solution, though as no gold standard is provided, the quality of clustering solutions cannot be verified. Parameter tuning over the training set is therefore an exercise in finding a value of k that best fits a participant's assumption. For example, that $k = 3$ best fits an assumption that the majority of target words will have three senses; an exercise that can be avoided by participants whose systems induce senses directly from the test set instances.

10.2 Evaluation Measures

The performance of a WSI system is assessed using two external evaluation measures: the V-Measure (Rosenberg and Hirschberg, 2007) and Paired F-Score (Artiles et al., 2009). These measures are introduced in this section. Each measure is first defined, followed by an analysis of how accurately it reflects the quality of a clustering solution. Both measures purport to accurately reflect alignment between a *hypothesis* (the clusters returned by a system) and a *reference* (the set of gold standard classes); in practice, the measures are found to behave somewhat differently.

10.2.1 The V-Measure

The V-Measure (Rosenberg and Hirschberg, 2007) measures the *homogeneity* and *completeness* of a clustering solution. Homogeneity is the degree to which clusters consist of objects belonging to a single gold standard class. Completeness is the degree to which clusters consists of all objects of a single gold standard class. The V-Measure is defined as the harmonic mean of homogeneity and completeness: $V = \frac{2 \times homogeneity \times completeness}{homogeneity + completeness}$. The

[2] `https://groups.google.com/group/semeval2010-senseinduction`

measure returns scores within the range $[0, ..., 1]$. A clustering solution returning $V = 1$ indicates a perfect alignment between objects in clusters and objects in gold standard classes.

The V-measure computes homogeneity and completeness using entropy H. The measure is defined as follows: Let N be the set of instances in the test set, with two separate partitions of N into: 1.) gold standard classes $C = \{c_i | i = 1, ..., n\}$, and 2.) clusters $K = \{k_i | i = 1, ..., m\}$. a_{ck} represents the number of objects of class c found to be objects of cluster k. Homogeneity and completeness can then defined as:

Homogeneity

$$homogeneity = \begin{cases} 1 & \text{if } H(C,K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{otherwise} \end{cases} \tag{10.1}$$

where

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{c=1}^{|C|}a_{ck}}$$

$$H(C) = -\sum_{c=1}^{|C|}\frac{\sum_{k=1}^{|K|}a_{ck}}{n}\log\frac{\sum_{k=1}^{|K|}a_{ck}}{n}.$$

Completeness

$$completeness = \begin{cases} 1 & \text{if } H(K,C) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{otherwise} \end{cases} \tag{10.2}$$

where

$$H(K|C) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{k=1}^{|K|}a_{ck}}$$

$$H(K) = -\sum_{k=1}^{|K|}\frac{\sum_{c=1}^{|C|}a_{ck}}{n}\log\frac{\sum_{c=1}^{|C|}a_{ck}}{n}.$$

The measure appears to be well-suited to the task of evaluating clustering solutions. For example, whereas the often applied evaluation measures of Purity and Entropy (Tan et al., 2006) only account for homogeneity, the V-Measure accounts for homogeneity and completeness. However, in practice the measure is found to be biased to favour clustering solutions in which there are a greater number of clusters than classes (Reichart and Rappoport, 2009; Amigó et al., 2009; Pedersen, 2010).

10.2.2 The Paired F-Score

The standard F-Score measure requires that each gold standard class $c \in C$ is associated with just one cluster in K (Tan et al., 2006). Thus, if a word sense induction system returns a greater number of clusters than there are classes, use of standard Precision, Recall, and F-Score measures would require a one-to-one paired mapping between each class in the gold standard set C and just one cluster in the clustering solution K.

The Paired F-Score (Artiles et al., 2009) circumvents this issue by pairing instances in gold standard classes C and instances in clusters K. The extent to which pairs in C and K overlap then defines the following measures:

$$Paired\,Precision = \frac{|pairs(C) \cap pairs(K)|}{|pairs(K)|}, \tag{10.3}$$

$$Paired\,Recall \quad = \frac{|pairs(C) \cap pairs(K)|}{|pairs(C)|}, \tag{10.4}$$

$$Paired\,F\text{-}Score = \frac{2 \times Paired\,Precision \times Paired\,Recall}{Paired\,Precision + Paired\,Recall}. \tag{10.5}$$

For example, given the classes C and clusters K in Table 10.1, $Paired\,Precision = 5/11 = 0.45$, $Paired\,Recall = 5/5 = 1.0$, and $Paired\,F\text{-}Score = 0.62$.

		Pairs	**|Pairs|**
$C =$	$[a, b, d], [c, f], [e, g].$	$(a, b), (a, d), (b, d), (c, f), (e, g)$	5
$K =$	$[a, b, d, e, g], [c, f].$	$(a, b), (a, d), (a, e), (a, g), (b, d),$ $(b, e), (b, g), (d, e), (d, g), (e, g),$ (c, f)	11
$C \cap K$		$(a, b), (a, d), (b, d), (e, g), (c, f)$	5

Table 10.1: Paired F-Score example.

Despite surface similarities with the standard F-Score measure, the Paired F-Score is a different measure. Differences between the two measures are listed below, illustrating how Paired F-Score is biased to favour large clusters.

- Precision and Recall measures in Paired F-Score return a score of zero for clusters containing a single item; this is not necessarily the case if standard Precision and Recall are applied.
- Clusters containing two items, one of which is correctly classified, the other misclassified receive a score of zero. Again, if standard Precision and Recall are applied scores will not necessarily be zero.
- In the standard F-Score there is a linear relationship between the scores and the percentage of misclassified items; in the Paired F-Score this relationship is non-linear, as shown in Figure 10.1.

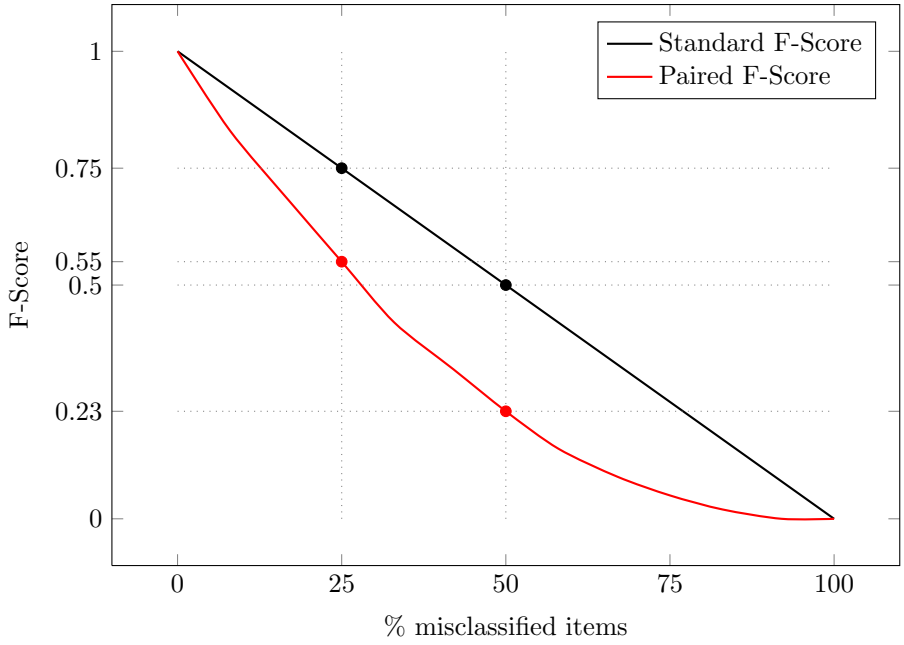


Figure 10.1: The standard F-Score and Paired F-Score for % misclassified items.

For example, Figure 10.1 shows that at 50% misclassification, the standard F-Score = 0.5, with the Paired F-Score = 0.23. At 25% misclassification, the standard F-Score = 0.75, with the Paired F-Score = 0.55.

- The Paired F-Score is biased to favour large clusters, therefore penalises systems that return relatively small clusters; clusters that may have high standard Precision and Recall.

  To illustrate this point, I wrote a computer program that automatically generates partitions of classes and clusters, where each partition initially represents items from a gold standard class c. Misclassification is then introduced into c. For example, given $c = [a, b, c]$, partitions: $k_1 = [a, b, 1]$ and $k_2 = [a, 1, 2]$ are generated, where letters a, b, c represent correctly classified items in k_1 and k_2 and numbers 1 and 2 represent misclassified items.

  Figure 10.2 and Table 10.2 show a number of generated partitions, illustrating the effect of misclassification relative to cluster size for both standard and Paired F-Score. Note that the patterns shown for these relatively small partitions are reflected in those far larger in size[3].

[3]Partitions were generated containing 2, 3, ..., 100 objects with the same pattern shown across the range. The size of a partition (cluster) and the size of the gold standard class it is evaluated against are identical, thus Precision and Recall for generated partitions return the same scores.

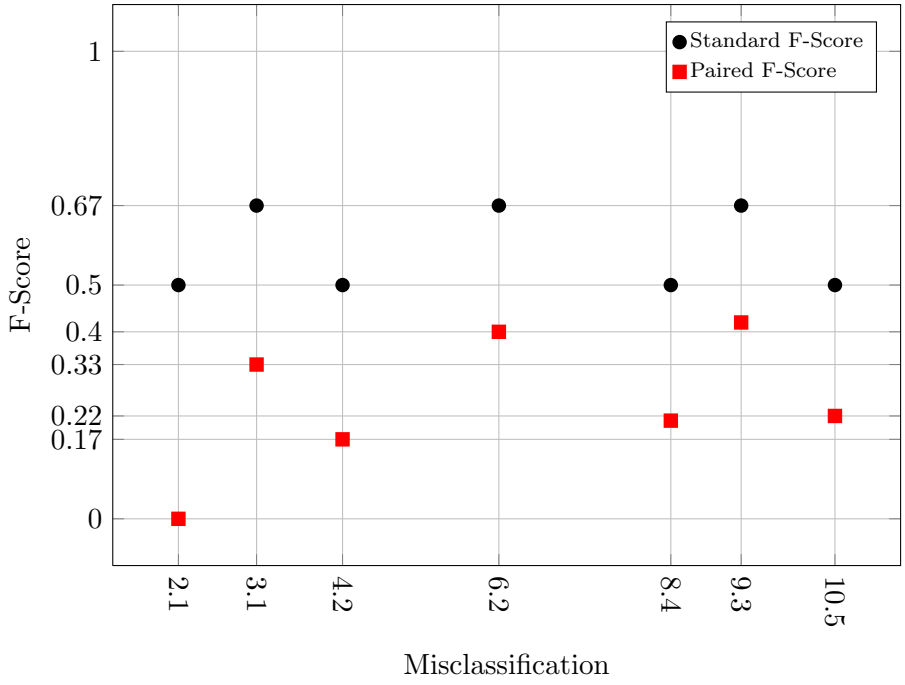


Figure 10.2: Misclassification relative to cluster size. The X axis label 3.1, for example, denotes a partition k containing three items, one of which is misclassified: $c = [a, b, c]$, $k = [a, b, 1]$.

Cluster size	Misclassified	Standard F-Score	Paired F-Score
2	1	0.50	0.00
3	1	0.67	0.33
4	2	0.50	0.17
6	2	0.67	0.40
8	4	0.50	0.21
9	3	0.67	0.42
10	5	0.50	0.22

Table 10.2: Misclassification relative to cluster size.

As Table 10.2 shows, the Paired F-Score returns lower scores than the standard F-Score measure. This table also shows that if the proportion of misclassified instances relative to cluster size is taken into account, Paired F-Score returns higher scores for larger clusters, whereas standard F-Score returns, as expected, the same score. For example, a cluster with 4 items and 50% misclassification (example 4.2 in Figure 10.2) and a cluster with 8 items and 50% misclassification (example 8.4 in Figure 10.2) will receive the same score using standard F-Score (0.5), whereas the Paired F-Score returns a higher score for the larger cluster (0.21 compared to 0.17). This bias in the Paired F-Score measure to favour larger clusters is due to its use of pair matching: each instance in a cluster of size n pairs with $n - 1$ other instances, therefore disproportionately penalises misclassification in small clusters.

10.3 Three Novel WSI Systems Applied to the SemEval-2010 Word Sense Induction & Disambiguation Task

Three novel WSI systems are applied to the SemEval-2010 Word Sense Induction & Disambiguation task, namely: SNN, SNN_{GR}, and *Word Overlap*. The SNN and *Word Overlap* systems induce word senses directly from the test set, thus are classified here as unsupervised, or *knowledge poor*, systems. The third system, SNN_{GR}, uses an external parser to extract grammatical relations from the test set; thus, in comparison with the SNN and *Word Overlap* systems, can be classified here as a *knowledge rich* system. The two SNN systems are described in Section 10.3.1, with the *Word Overlap* system described in Section 10.3.2. All three systems are evaluated in Section 10.4.

10.3.1 The Shared Nearest Neighbours Systems

The SNN and SNN_{GR} systems induce word senses using the Shared Nearest Neighbours (SNN) algorithm described in Chapter 5. These systems are based on the notion that similarity between two data items can be measured by the number of nearest neighbours the items share. In this evaluation, data items are test set instances: sentences and/or paragraphs containing a target word. Nearest neighbours are the *significant word features* of an instance, where *word features* for the *SNN* system are bag of words and ordered word pairs, with grammatical relations between words replacing ordered word pairs in the SNN_{GR} system[4]. *Significant* word features are word features that pass some predetermined threshold.

Extracting Word Features from Test Set Instances

The initial step in the *SNN* approach is to extract word features from test set instances. For example, given the expository instance:

[girls sip strawberry daiquiris],

the following information would be extracted:

1. **Bag of words pairs** :
   {girls, sip}, {girls, strawberry}, {girls, daiquiris}, {sip, strawberry}, {sip, daiquiris}, {strawberry, daiquiris}.

2. **Ordered pairs** :
   (girls, sip), (girls, strawberry), (girls, daiquiris), (sip, strawberry), (sip, daiquiris), (strawberry, daiquiris).

3. **Grammatical relations** :
   nsubj(sip, girls), *amod*(daiquiris, strawberry), *dobj*(sip, daiquiris).

[4]Grammatical relations are extracted from test set instances using the Stanford Parser: `http://nlp.stanford.edu/software/lex-parser.shtml`.

Each word in a target word instance is then associated with a set of word features. For example, given the instance above the word strawberry would be associated with:

1. bag of words features: girls, sip, daiquiris,
2. relative order features: girls_{Left2}, sip_{Left1}, $\text{daiquiris}_{Right1}$ (extracted from ordered pairs),
3. the grammatical relation $amod$(daiquiris).

The relative order features extracted from ordered pairs are used in the SNN system to act as a rudimentary analogue of the grammatical relations used in the SNN_{GR} system.

Computing Association Measures Between Words and Word Features

Counts are updated as test instances are processed. For example, each time strawberry is seen with girls, the bag of words count for the word pair strawberry, girls is updated, along with the separate counts for the individual words strawberry and girls and the overall count for all bag of words pairs seen. In so doing, an accurate association measure score can be computed between strawberry and girls.

The *SNN* systems use the Log Likelihood Ratio (LLR) association measure (Dunning, 1993) to quantify the significance between words and/or dependency relations. Thus, if the LLR score between strawberry and girls passes some pre-defined LLR threshold, then girls is deemed to be a significant word feature for strawberry. In this evaluation, LLR significance thresholds are set at a number of levels, from 0 (no LLR thresholding) through: $3.84, 6.63, 7.9, 10.83$, noting that LLR significance thresholds higher than 10.83 (1 in 1,000 odds of seeing a pair) would be meaningless given the amount of data contained in the test set.

Varying the Context Size

The size of the context around a target word is also varied in order to assess how much context is required to induce word senses. For example, given the example instance for the target word sip:

[valley girls sip sea breezes],

a context window of size 1 would return [girls sip sea], one word to the left of the target word and one word to its right.

Computing Similarity Between Test Instances

The *SNN* systems compute similarity between test instance pairs by using the significant word features of each word in each instance; an approach that allows both first order (direct) and second order (indirect) associations to be used in similarity calculations. For example, in the expository instances:

i = [girls sip strawberry daiquiris], j = [girls sip banana daiquiris]

there is a direct, first order relationship between i and j. To take just two word comparisons: girls$_i$ and girls$_j$ share the word features: {sip, daiquiris} and strawberry and banana share the word features {girls, sip, daiquiris}. If these word features are found to be significant word features for words in both instances, then similarity between i and j can be computed. However, for instances:

i = [girls sip strawberry daiquiris], k = [boys drink orange juice]

there is no direct association. There may, though, exist a set of features shared by the words in i and k; that is, second order words that can be used to calculate similarity between the two instances. For example, in instance i girls *sip* and in instance k boys *drink* but both words may be found in other test instances to: *play*, *kiss*, *dance*. Thus, these shared word features, if found to be significant word features for both boys and girls, can be used to compute similarity between instances i and k.

Clustering Test Instances

Similarity values between test instances are used as edge weights in a target word graph. For example, if the similarity between two target word test instances i and $j = 3$ (i and j share three significant word features), then the edge weight between vertices i and j (representing the instances) in the target word graph is set to 3. The *MaxMax* clustering algorithm (introduced in Chapter 5) is then applied to the target word graph, with each cluster returned by *MaxMax* taken to represent a sense of the target word. A perfect clustering solution would therefore place the test instances of each sense of the target word into a single cluster.

10.3.2 The Word Overlap System

The *Word Overlap* system uses a straightforward model of similarity between test instances to induce word senses; its role is to act as a comparative to the relatively complex *SNN* systems.

Senses of a target word are induced by counting the number of times context words co-occur in target word instances. For example, given the example instance for the target word sip:

[city girls sipped sea breezes][5],

the following information is extracted -

Unordered Pairs: {city, girls}, {city, sea}, {city, breezes}, {girls, sea}, {girls, breezes}, {sea, breezes}.

[5]The lemmatiser in the MIT WordNet Interface is used to find inflected forms of target words: `http://projects.csail.mit.edu/jwi/`

The count for a pair is incremented each time the pair is observed, with the final counts used as edge weights in a target word graph. Therefore, if the final count for the pair $\{\mathsf{city}, \mathsf{sea}\} = 3$, then the edge weight between vertices city and sea in the target word graph is set to 3. The *MaxMax* algorithm is then applied to the graph, with each cluster returned taken to represent a sense of the target word. A perfect clustering solution would therefore place the test instance words of one sense of the target word into one cluster. Each cluster is therefore a *lexical* representation of a sense of the target word: whereas *SNN* systems cluster target word instances, *Word Overlap* clusters *words* in target word instances.

10.3.3 Assigning Clusters to Target Word Instances

Sections 10.3.1 and 10.3.2 describe how the *SNN* and *Word Overlap* systems derive candidate sense clusters, though do not state how clusters are associated with target word senses. This section describes the assignment process.

***SNN* Systems:** The *SNN* and *Word Overlap* systems attach unique identification labels to the clusters they generate, for example, given a clustering solution for a target word the first cluster returned by a system might be tagged as $cluster_1$, the second as $cluster_2$, and so on.

SNN systems assign one cluster to each test instance. As *SNN* systems cluster test instances, each target word test instance is assigned the cluster containing this test instance. For example, given the (example) test instance for the target word sip:

$$[\text{valley girls } \underline{\text{sip}} \text{ sea breezes}],$$

if $cluster_7$ contains this instance, the instance is tagged as $cluster_7$:

$$[\text{valley girls } \underline{\text{sip}} \text{ sea breezes}]^{cluster_7}.$$

***Word Overlap* System:** The *Word Overlap* system returns a lexical clustering, thus can assign several clusters to one instance. For example, given the example instance above, if three of the context words are in $cluster_7$ and one is in $cluster_{15}$ then both clusters are assigned to the test instance:

$$[\text{valley girls } \underline{\text{sip}} \text{ sea breezes}]^{cluster_7(0.75),\, cluster_{15}(0.25)},$$

with the values 0.75 (3 of the 4 instance words are in $cluster_7$) and 0.25 (1 word out of the 4 instance words is in $cluster_{15}$) used as weights by the evaluation software. Presumably, this software applies some form of quasi-probabilistic interpretation of the number of words in clusters that are correctly assigned to a test instance that represents a particular sense of the target word; a distributed scoring mechanism that is, in the view of the task's organisers, the preferred method of returning clusters, stating that: "participants return all induced senses per instance with associated weights, as this will enable a more objective supervised evaluation" (Manandhar et al., 2010).

10.4 Evaluation Results

This section reports results for the *SNN* and *Word Overlap* systems introduced in Sections 10.3.1 and 10.3.2. The systems are evaluated using the software provided by the co-ordinators of the task[6], with results compared to those of the twenty six participant systems.

10.4.1 V-Measure Results

Figures 10.3 and 10.4 report V-Measure results for the *SNN* and *Word Overlap* systems. Results are also reported for the best and worst performing system out of the twenty six participant systems, along with the average score returned by these systems. Systems against which *SNN* and *Word Overlap* systems are compared are said here to be participant systems as these systems were evaluated at the time of SemEval-2010; *SNN* and *Word Overlap* systems were evaluated post SemEval-2010.

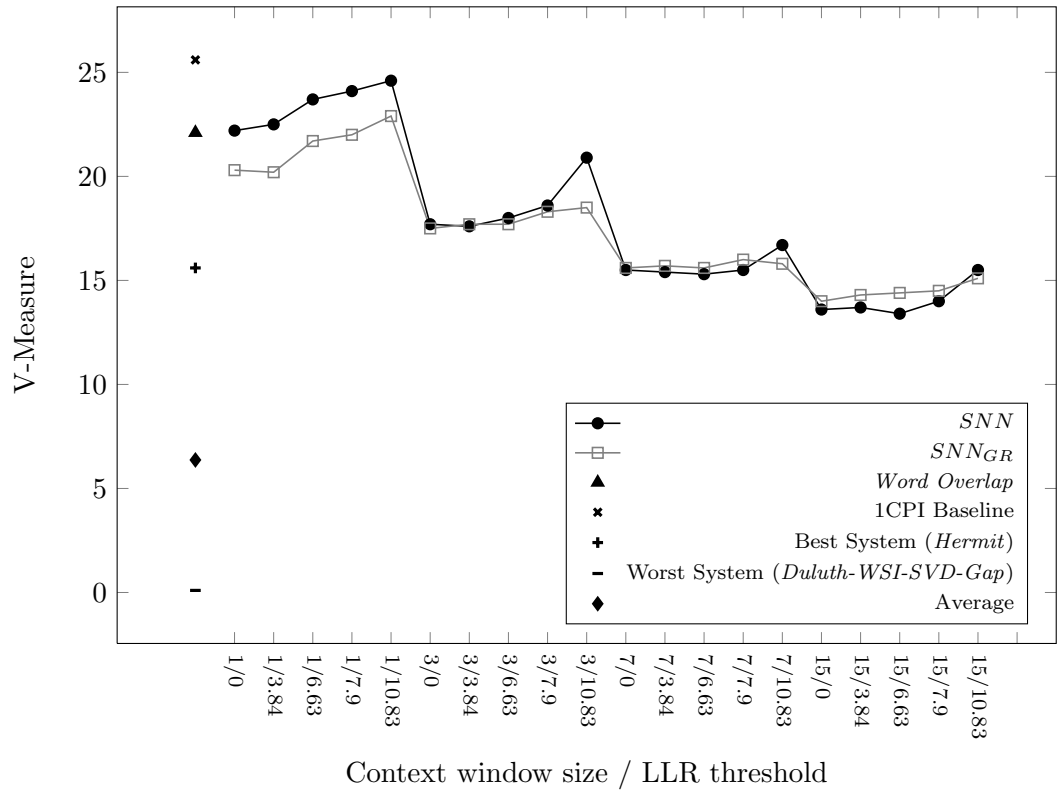


Figure 10.3: V-Measure results: verbs.

[6] `http://www.cs.york.ac.uk/semeval2010_WSI/datasets.html`

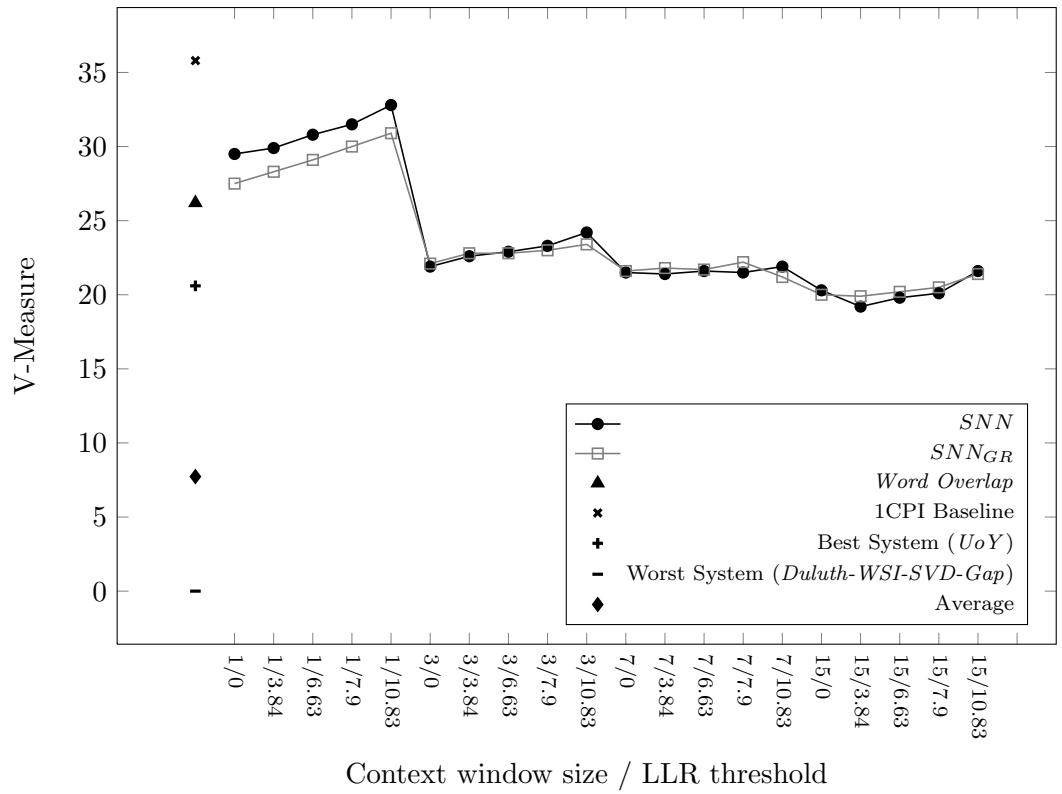


Figure 10.4: V-Measure results: nouns.

10.4.1.1 V-Measure Results Analysis

Figures 10.3 and 10.4 show that the *SNN* and *Word Overlap* systems return higher V-Measure scores than any of the twenty six participant systems.

Verbs: For verbs, SNN returns the best results, with the highest score, 24.6, attained using a context window of size 1 and applying the strongest LLR threshold (10.83.). This score is 9 points above that of the best performing participant system (*Hermit*, 15.6)[7], and just over 18 points higher than the average score of 6.37. Even the comparatively simple *Word Overlap* system at 22.1 outperforms *Hermit* by 6.5 points. The SNN_{GR} system is found to make marginal gains over SNN as context size increases (7,15); however, performance drops for both systems here, with the SNN system still able to outperform SNN_{GR} at a LLR threshold of 10.83.

[7] *Hermit* (Jurgens and Stevens, 2010) uses Sparse Distributed Memory (SDM) vectors (Kanerva, 1988) to represent words and instances. Each word w_i in an instance is represented as a SDM word vector $\vec{w_i} = \{0_1, 0_2, 0_3..., 0_{5000}\}$ with a small number n of the 5000 zero-valued elements randomly selected and set to $+1$ and n set to -1. Each instance is the sum of its word vectors, thus the instance [girls sip strawberry daiquiris] would be represented as the sum of the vectors: $\vec{\text{girls}} + \vec{\text{sip}} + \vec{\text{strawberry}} + \vec{\text{daiquiris}}$; a first-order representation. SDM vectors act as *holographic memories* of a word and/or instance (Plate, 2003) where the size of the vectors, coupled with the randomly generated keys of $+1s$ and $-1s$, results in each vector having a high likelihood of being orthogonal to all other vectors; thus, each vector is a unique representation of an item within the target word's set of instances. The SDM instance vectors are clustered using K-Means, with Hierarchical Agglomerative Clustering applied to merge clusters into a set of sense clusters. I applied SDM vectors in a fourth system. This system is not reported here as SDM vectors were used to store the same information that was used in the *SNN*-based systems; thus, unsurprisingly, the SDM and *SNN* systems returned the same results.

Nouns: For nouns, SNN is again the highest scoring system, with the highest score of 32.8 attained, as with verbs, using a context window of size 1 and applying the strongest LLR threshold of 10.83. This score is 12.2 points higher than that of the best performing system (*UoY*, 20.6)[8] and over 25 points higher than the average score of 7.73. Again, the simple *Word Overlap* system at 26.2 outperforms the best participant system. The SNN_{GR} system gains, marginally, over SNN as context size increases (7,15) for LLR thresholds < 10.83, though less so than for verbs, implying (in this evaluation context) that grammatical relations are more important for verbs than nouns.

10.4.1.2 V-Measure Results Discussion

Given the test set instances particular to this task, Figures 10.3 and 10.4 indicate that small contexts best induce word senses. For contexts of a particular size, the higher the LLR threshold the better the results. Thus, applying the highest LLR threshold (10.83) to the smallest context (1) returns the best results. This suggests that context words within the immediate vicinity of a target word, that have a strong association to the target word, best predict target word senses. Senses therefore appear to be best induced using key, adjacent context words: words that act as strong cues to the various senses of a target word.

The usefulness of syntactic information is marginally evident for larger contexts $(7, 15)$ where SNN_{GR} is shown to outperform SNN for LLR thresholds < 10.83, however performance over larger contexts is relatively poor for both systems. Results clearly show the SNN ordered word pairs approach to return the best overall results, outperforming the SNN_{GR} grammatical relations approach over smaller contexts $(1, 3)$.

A Degenerate Clustering Solution

Ranked results for the *SNN*, *Word Overlap*, and twenty six participant systems are shown in Table 10.3. The baselines, provided by the organisers of the task and shown in italics, are: *1CPI*, one cluster per instance; *MFS*, most frequent sense (all instances in one cluster), and *Random*, which randomly assigns instances to one of four clusters. The best and worst performing participant systems are shown in bold typeface.

[8] *UoY* (Korkontzelos and Manandhar, 2010), the system submitted by the task's co-ordinators, is a graph-based WSI system in which significant (greater than some LLR threshold) words and collocations are represented as vertices in a graph. The *Chinese Whispers* (Biemann, 2006a) algorithm is used to cluster vertices, returning a set of sense clusters.

System	Verbs	Nouns
1CPI	25.6	35.8
SNN	**24.6**	**32.8**
SNN$_{GR}$	**22.9**	**30.9**
Word Overlap	**22.1**	**26.9**
Hermit	**15.6**	16.7
KSU KDD	12.4	18.0
UoY	8.5	**20.6**
Duluth-WSI	5.7	11.4
Duluth-WSI-SVD	5.7	11.4
Duluth-R-110	8.5	8.6
KCDC-PCGD	8.4	7.3
Duluth-WSI-Co	6.0	9.2
KCDC-PC	7.3	7.7
KCDC-GD	8.5	5.9
KCDC-GD-2	8.0	6.1
KCDC-GDC	7.8	6.2
KCDC-PC-2	6.1	7.7
Duluth-Mix-Narrow-PK2	5.5	7.8
Duluth-Mix-Narrow-Gap	5.1	8.0
Duluth-MIX-PK2	5.2	5.8
Duluth-R-15	5.1	5.4
Duluth-WSI-Co-Gap	3.6	5.6
Random	4.6	4.2
Duluth-R-13	3.7	3.5
Duluth-Mix-Gap	3.0	2.9
Duluth-WSI-Gap	1.5	4.2
Duluth-Mix-Uni-PK2	4.7	0.8
Duluth-R-12	2.5	2.2
KCDC-PT	3.10	1.0
Duluth-Mix-Uni-Gap	3.0	0.2
Duluth-WSI-SVD-Gap	**0.1**	**0.0**
MFS	0.0	0.0

Table 10.3: Systems ranked by V-Measure.

As Table 10.3 shows, the *SNN* and *Word Overlap* systems are the best performing systems, with *SNN* best overall. *SNN* outperforms the participant system *Hermit* by 9 points (verbs) and *UoY* by 12.2 points (nouns). Notably, system scores are typically higher for nouns than verbs. One explanation for this, put forward by the organisers of the task, is that nouns have a larger and therefore more distinctive contextual vocabulary than verbs, thus noun senses are better delineated.

It is important to note that the *1CPI* baseline, which returns better V-Measure scores than any of the twenty nine systems, is not an upper bound in this evaluation. *1CPI*'s scores are a result of a degenerate clustering solution in which each test instance is assigned to a single cluster. The comparatively high scores returned by *1CPI* are therefore a direct result of V-Measure's bias to favour numerous small clusters. Scores returned by V-Measure may therefore be misleading, for example, a system returning a score that is comparable with *1CPI* indicates that it is returning a near degenerate solution; however, scores close to *1CPI* (such as the scores returned by *SNN*) may actually indicate that the system is finding word senses in a perfectly precise manner. That is, a system gener-

ating many poorly defined sense clusters inclines towards *1CPI*, yet a system generating many clusters which align with many senses raises the V-Measure score through both its alignment with the gold standard sense classes *and* its generation of numerous clusters.

10.4.2 Paired F-Score Results

V-Measure evaluation results showed that the *SNN* systems are the best performing WSI systems. However, these results are countered by those in Figures 10.5 and 10.6 for the Paired F-Score evaluation, showing that the *SNN* systems are the worst performing systems.

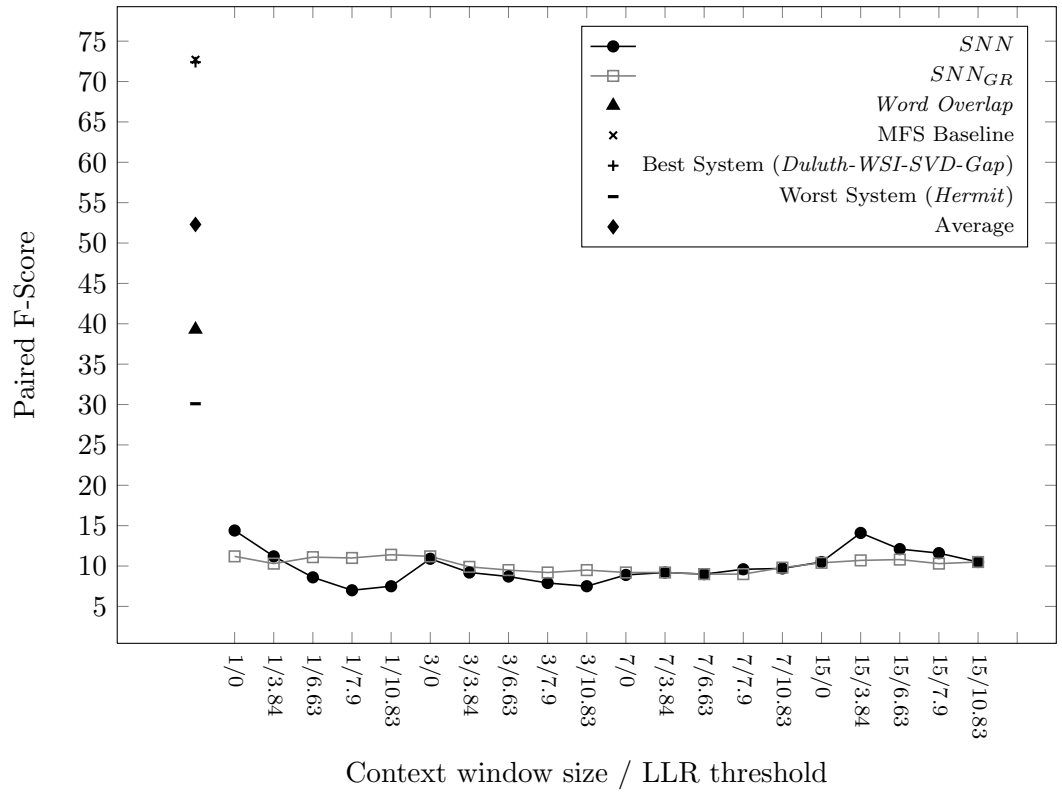


Figure 10.5: Paired F-Score results: verbs.

10.4.2.1 Paired F-Score Analysis

Verbs: For verbs, the highest *SNN* system score is 14.4 (*SNN*). This is 58 points below the best scoring system (*Duluth-WSI-SVD-Gap* at 72.4), 37.9 points below the average score of 52.3, and 15.7 points below the worst performing system (*Hermit* at 30.1). The *Word Overlap* system at 39.3 performs better than the worst performing system by 9.2 points, though is still 13 points below the average score. The best scoring system, *Duluth-WSI-SVD-Gap*[9] at 72.4 is close to the *MFS* baseline at 72.7. However, the *MFS* baseline

[9] *Duluth-WSI-SVD-Gap* is one of a total of sixteen systems submitted by The University of Minnesota, Duluth (Pedersen, 2010). The Duluth-WSI systems are a composite of techniques and measures in which, broadly speaking, features of target word instances are represented as vectors in a second-order vector space model. The clustering solution is provided by the *SenseClusters* package, with the *PK2* and/or

is not an upper bound in this evaluation as it is a degenerate clustering solution in which every instance is placed in just one cluster, thus the best performing system, *Duluth-WSI-SVD-Gap*, may actually be returning a near degenerate solution.

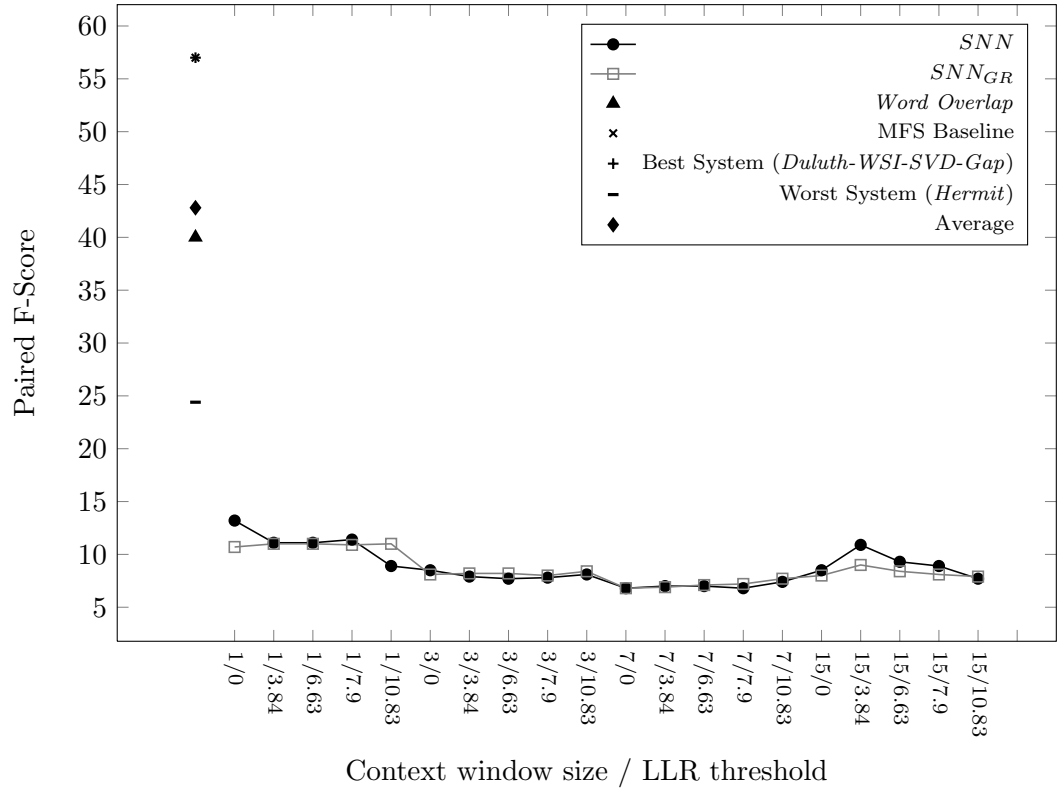


Figure 10.6: Paired F-Score results: nouns.

Nouns: For nouns, the highest *SNN* system score is 13.2 (*SNN*). This is 43.9 points below the best scoring system (*Duluth-WSI-SVD-Gap* at 57.0); 29.6 points below the average score of 42.8, and 11.2 points below the worst performing system (*Hermit* at 24.4). The *Word Overlap* system at 40.0 performs better than the worst performing system by 15.6 points and is not far short of the average score, being just 2.8 points below this value. *Duluth-WSI-SVD-Gap* is, as for verbs, the best performing system, returning the same score as the *MFS* baseline (57.0), implying that this system is returning a degenerate clustering solution.

10.4.2.2 Paired F-Score Results Discussion

As Table 10.4 verifies, *SNN* systems are the worst performing in the Paired F-Score evaluation. However, a comparison between the system rankings in Table 10.3 (systems ranked by V-Measure) with those in Table 10.4 (systems ranked by Paired F-Score) reveals

Adapted Gap Statistic used to find the optimal number of clusters to return (Pedersen and Kulkarni, 2006). A number of the methods used by the Duluth systems are, compared with those applied by other participant systems, computationally expensive; for example, *Duluth-WSI-SVD-Gap* uses Singular Value Decomposition (SVD) to reduce the dimensions of co-occurrences matrices, with Fisher's exact test used to denote the statistical significance of co-occurrence pairs.

that the best performing participant system for the V-Measure, *Hermit*, is the worst performing system for the Paired F-Score[10]. Similarly, the best performing participant system for the Paired F-Score, *Duluth-WSI-SVD-Gap*, is the worst performing system in the V-Measure evaluation. This finding is also reflected in scores returned by the *SNN* systems, where the gain in obtaining the best V-Measure results is offset by obtaining the worst Paired F-Score results.

System	Verbs	Nouns
MFS	72.7	57.0
Duluth-WSI-SVD-Gap	**72.4**	**57.0**
KCDC-PT	69.7	56.4
KCDC-GD	70.0	51.6
Duluth-Mix-Gap	65.8	54.5
KCDC-GD-2	69.3	50.4
KCDC-GDC	70.0	48.5
Duluth-Mix-Uni-Gap	61.2	57.0
KCDC-PC	62.9	50.4
Duluth-Mix-Uni-PK2	55.9	57.1
KCDC-PC-2	61.7	49.7
KCDC-PCGD	65.6	44.8
Duluth-WSI-Gap	53.9	53.4
Duluth-WSI-Co-Gap	51.5	53.3
UoY	66.6	38.2
Duluth-MIX-PK2	48.3	51.7
Duluth-Mix-Narrow-Gap	51.3	47.4
Duluth-WSI-Co	48.2	50.2
Duluth-R-12	52.6	44.3
Duluth-Mix-Narrow-PK2	48.2	37.1
Duluth-WSI	46.7	37.1
Duluth-WSI-SVD	46.7	37.1
KSU KDD	54.7	24.6
Word Overlap	**39.3**	**40.0**
Duluth-R-13	41.5	36.2
Random	34.1	30.4
Duluth-R-15	28.9	26.7
Hermit	**30.1**	**24.4**
Duluth-R-110	16.4	15.8
SNN	**14.4**	**13.2**
$\boldsymbol{SNN_{GR}}$	**11.4**	**11.0**
1CPI	0.08	0.11

Table 10.4: Systems ranked by Paired F-Score.

A Degenerate Clustering Solution

As the analysis in Section 10.2.2 showed, the Paired F-Score is biased to favour clustering solutions that return large clusters. The best performing system for the Paired F-Score is *Duluth-WSI-SVD-Gap*. This system returns an average of 1.02 clusters, thus is near identical to the *MFS* baseline which places all test instances of a target word into one cluster: a degenerate clustering solution. Thus, by default of the bias inherent in the

[10]Duluth-R-110 is a random system.

Paired F-Score measure, a near degenerate solution which places the majority of the test instances, for each target word, into one cluster gains the highest Paired F-Score.

10.4.3 Using Different Clustering Algorithms in the SNN and Word Overlap Systems

Tables 10.3 and 10.4 in the previous sections reported results for *SNN* and *Word Overlap* systems that used *MaxMax* as the clustering algorithm. This section reports results for *SNN* and *Word Overlap* systems that use three other representative graph clustering algorithms, namely: *Affinity Propagation* (Frey and Dueck, 2007), *Chinese Whispers* (Biemann, 2006a), and *DBSCAN* (Ester et al., 1996) - the same three algorithms that *MaxMax* was compared with in Chapter 5. The aim of the evaluation presented this section is therefore to see if any of these clustering algorithms can improve upon *MaxMax*'s performance. Results are presented in Tables 10.5 and 10.6 and discussed in the following sections.

10.4.3.1 V-Measure Results

V-Measure results, reported in Table 10.5, show that whichever clustering algorithm is applied, *SNN* and *Word Overlap* system performance is noticeably better for nouns than verbs.

System	Verbs	Nouns
Affinity Propagation SNN	**24.9**	**33.1**
Affinity Propagation SNN_{GR}	23.7	31.2
Affinity Propagation Word Overlap	23.2	28.5
MaxMax SNN	24.6	32.8
MaxMax SNN_{GR}	22.9	30.9
MaxMax Word Overlap	22.1	26.9
Chinese Whispers SNN	24.4	32.5
Chinese Whispers SNN_{GR}	22.7	30.7
Chinese Whispers Word Overlap	22.1	26.8
DBSCAN SNN	19.8	22.3
DBSCAN SNN_{GR}	18.3	21.9
DBSCAN Word Overlap	18.1	20.8
1CPI	25.6	35.8
Random	4.6	4.2
MFS	0.0	0.0

Table 10.5: Ranked V-Measure results for the *SNN* and *Word Overlap* systems that use *Affinity Propagation*, *Chinese Whispers*, *DBSCAN*, or *MaxMax* as the clustering algorithm.

As previously reported in Table 10.3, the highest ranked WSI systems also tend to perform much better on nouns. Therefore, finding that *SNN* and *Word Overlap* systems (using any one of the four clustering algorithms) do so is taken as a positive sign of system

performance. Results in Table 10.5 also show that the three systems outperform the *MFS* and *Random* baselines, indicating that all four clustering algorithms are distributing target word senses to clusters in a non-random fashion.

With regard to individual clustering algorithms, the use of *Affinity Propagation* is shown to return the best results for the *SNN* systems and the *Word Overlap* system, though differences between *Affinity Propagation*'s results and those of the second (*MaxMax*) and third (*Chinese Whispers*) best performing clustering algorithms are not marked. *DBSCAN* is shown to be the worst performing clustering algorithm by some margin. However, the comparatively good results of *Affinity Propagation*, *MaxMax*, and *Chinese Whispers* are close to the *1CPI* baseline, indicating that these algorithms are returning clustering solutions made up of 'micro senses' (Agirre et al., 2006b); that is, many small clusters which have high homogeneity but low completeness. Two observations regarding the results in Table 10.5 are notable: firstly, system rankings are the same for all clustering algorithms, which suggests that it is the methodology applied in the systems that affects clustering algorithm performance (of which the clustering algorithm is but one facet); secondly, marginal differences between the results of *MaxMax* and *Chinese Whispers* suggest that these algorithms are returning highly similar clustering solutions.

10.4.3.2 Paired F-Score Results

Paired F-Score results, reported in Table 10.6, show that regardless of which clustering algorithm is applied, *SNN* system performance is marginally better for verbs than nouns, with the opposite true for the *Word Overlap* system. As previously reported in Table 10.4, the highest ranked WSI systems perform better on verbs. Given that *Word Overlap* systems are shown in Table 10.6 to return far better results than *SNN* systems and that these results are better for nouns, it is therefore difficult to draw any positive correspondence between system performance in Tables 10.4 and 10.6. Table 10.6 also shows that *SNN* systems fall well short of the *Random* baseline, results which are at odds with those reported in the V-Measure evaluation. However, *Word Overlap* systems (bar *Affinity Propagation* for verbs) do surpass the *Random* baseline. Furthermore, they surpass the *1CPI* baseline and fall short of the *MFS* baseline (noting that the *MFS* baseline is not an upper bound in the Paired F-Score evaluation), thus indicating that *Chinese Whispers*, *DBSCAN*, and *MaxMax* are attempting to distribute target word senses to clusters, unlike the best performing system in Table 10.4 (*Duluth-WSI-SVD-Gap*) which assigns the majority of test set instances to just one cluster.

With regard to individual clustering algorithms, *Affinity Propagation* is shown to be the worst performing clustering algorithm, thus countering its performance in the V-Measure evaluation, and *DBSCAN* is shown to be the best performing clustering algorithm by some margin. These results therefore indicate that *DBSCAN* is returning a more complete clustering solution; results which echo those previously presented in Chapter 5 where *DBSCAN* was shown to have far higher Recall than either *Affinity Propagation*, *Chinese Whispers*, or *MaxMax*. The same two observations that were made in the V-Measure evaluation can also be made here: firstly, clustering algorithm rankings are the same

for all three systems; secondly, marginal differences between the results of *MaxMax* and *Chinese Whispers* suggest that these algorithms are returning similar clustering solutions.

System	Verbs	Nouns
DBSCAN Word Overlap	**49.3**	**47.2**
DBSCAN SNN	17.5	16.8
DBSCAN SNN_{GR}	16.4	16.2
Chinese Whispers Word Overlap	39.5	40.2
Chinese Whispers SNN	14.3	13.4
Chinese Whispers SNN_{GR}	11.6	10.8
MaxMax Word Overlap	39.3	40.0
MaxMax SNN	14.4	13.2
MaxMax SNN_{GR}	11.4	11.0
Affinity Propagation Word Overlap	32.7	33.5
Affinity Propagation SNN	10.2	9.8
Affinity Propagation SNN_{GR}	10.1	9.7
MFS	72.7	57.0
Random	34.1	30.4
1CPI	0.08	0.11

Table 10.6: Ranked Paired F-Score results for the *SNN* and *Word Overlap* systems that use *Affinity Propagation*, *Chinese Whispers*, *DBSCAN*, or *MaxMax* as the clustering algorithm.

10.4.3.3 Key Points and Conclusions

A number of conclusions can be drawn from this evaluation, as follows:

- Results in Tables 10.5 and 10.6 repeat the pattern that was previously observed in Tables 10.3 and 10.4: the best performing clustering algorithm in the V-Measure evaluation (*Affinity Propagation*) is found to be the worst performing clustering algorithm in the Paired F-Score evaluation, and the best performing clustering algorithm in the Paired F-Score evaluation (*DBSCAN*) is found to be the worst performing clustering algorithm in the V-Measure evaluation. Given the known biases of the V-Measure and Paired F-Score, it could therefore be argued that *MaxMax* and *Chinese Whispers*, by ranking second/third and third/second respectively in Tables 10.5 and 10.6, outperform *Affinity Propagation* and *DBSCAN* in this evaluation.

- Tables 10.5 and 10.6 report system performance, therefore do not directly indicate which of the four clustering algorithms best induces word senses. The clustering algorithm applied in a system uses information supplied by the system – information about: the vertices representing test instances or words in test instances; connections between vertices; and weights assigned to these connections. Therefore, if the information passed to the clustering algorithm is of poor quality with respect to inducing word senses, which may be a result of pre-clustering processes applied to the test set or even directly attributable to the quality of the test set itself, the clustering

algorithm will perform poorly - a point rarely made in the graph clustering literature (Newman et al., 2006; Tan et al., 2006; Aggarwal and Reddy, 2013).

- The clustering algorithm applied in a WSI system may be unsuited to the task. That is, the algorithm may have a preference for finding certain types of clusters, clusters of a particular shape, density, or size that are distinctly different to those of word sense clusters. For example, *Affinity Propagation* was shown in Chapter 5 to perform better on graphs that contained dense, globular shaped clusters. Thus, if word sense clusters are not of this type, it is reasonable to expect that *Affinity Propagation* will, compared with clustering algorithms that have no particular preference for specific cluster types, perform poorly. Indeed, the question as to whether word sense clusters are of a particular type (or types) may be a fruitful line of enquiry for future research to consider.

10.4.4 Supervised Evaluation Results

Results for the unsupervised evaluation, reported in Sections 10.4.1 and 10.4.2, are, arguably, the key results for WSI researchers to consider. Results for the supervised Word Sense Disambiguation (WSD) evaluation are reported in this section for completeness. *SNN* and *Word Overlap* systems use the *MaxMax* clustering algorithm.

The supervised evaluation follows the methodology applied in the SemEval-2007 Word Sense Induction and Discrimination task (Agirre and Soroa, 2007) by splitting the test set into two parts: 1.) a mapping corpus, used to map induced senses to gold standard senses; 2.) an evaluation corpus, used to evaluate a WSI system in a supervised WSD setting. However, in order to avoid the problem found in the SemEval-2007 task of different splits of the test set returning different system rankings, the SemEval-2010 evaluation reports results as an average of five random splits of the test set. Results for the *SNN* and *Word Overlap* systems and the twenty six participant systems are shown in Tables 10.7 and 10.8. Results are reported, as they are in the evaluation write up (Manandhar et al., 2010) and on the task website[11], using Recall (named Supervised Recall in this evaluation).

As Tables 10.7 and 10.8 show, the supervised evaluation applies two different splits of the test set: an 80/20 split and a 60/40 split. Two of the three baselines that were used in the unsupervised evaluation are used here: *Random* and *MFS* (Most Frequent Sense). The evaluation measure used, Supervised Recall (SR), is not defined in the task, though presumably is standard Recall for clustering; that is, the average Recall of clusters returned by a WSI system: a measure of the completeness of a clustering solution.

[11] `http://www.cs.york.ac.uk/semeval2010_WSI/task_14_ranking.html`

System	SR % All	SR % Nouns	SR % Verbs
UoY	62.44	59.43	66.82
Duluth-WSI	60.46	54.66	68.92
Duluth-WSI-SVD	60.46	54.66	68.92
Duluth-WSI-Co-Gap	60.34	54.09	68.65
Duluth-WSI-Co	60.27	54.68	67.60
Word Overlap	59.86	54.42	67.72
Duluth-WSI-Gap	59.81	54.36	67.76
KCDC-PC-2	59.76	54.09	68.04
KCDC-PC	59.73	54.55	67.29
KCDC-PCGD	59.53	53.33	68.56
KCDC-GDC	59.08	53.39	67.38
KCDC-GD	59.03	52.97	67.87
KCDC-PT	58.88	53.07	67.35
SNN	58.76	52.83	67.37
KCDC-GD-2	58.72	52.78	67.38
Duluth-WSI-SVD-Gap	58.69	53.22	66.66
MFS	58.67	53.22	66.63
Duluth-R-12	58.46	53.05	66.44
Hermit	58.34	53.56	65.30
$\boldsymbol{SNN_{GR}}$	58.27	53.38	65.14
Duluth-R-13	58.01	52.27	66.38
Random	57.25	51.45	65.69
Duluth-R-15	56.76	50.91	65.30
Duluth-Mix-Narrow-Gap	56.63	48.11	69.06
Duluth-Mix-Narrow-PK2	56.15	47.54	68.70
Duluth-R-110	54.75	48.28	64.20
KSU KDD	52.18	46.63	60.28
Duluth-MIX-PK2	51.62	41.10	66.96
Duluth-Mix-Gap	50.61	40.04	66.02
Duluth-Mix-Uni-PK2	19.29	1.82	44.78
Duluth-Mix-Uni-Gap	18.72	1.55	43.76

Table 10.7: Supervised Recall (SR) for the test split: 80% mapping, 20% evaluation.

Table 10.7 reports results for the 80/20 split (80% mapping corpus, 20% evaluation corpus). These results show that the *SNN* and *Word Overlap* systems surpass the *Random* baseline, with the *SNN* and *Word Overlap* systems shown to surpass the *MFS* baseline. The key findings in Table 10.7 are, however, that the comparatively simple *Word Overlap* system outperforms both *SNN* systems, and that *SNN* and *Word Overlap* outperform *Hermit* and *Duluth-WSI-SVD-Gap*: the two best performing participant systems in the unsupervised evaluation (*Hermit* for the V-Measure evaluation; *Duluth-WSI-SVD-Gap* for the Paired F-Score).

System	SR % All	SR % Nouns	SR % Verbs
UoY	61.96	58.62	66.82
Duluth-WSI-Co	60.07	54.59	68.05
Duluth-WSI-Co-Gap	59.51	53.45	68.33
Duluth-WSI-SVD	59.48	53.45	68.27
Duluth-WSI	59.48	53.45	68.27
Word Overlap	59.37	53.28	68.31
Duluth-WSI-Gap	59.32	53.19	68.23
KCDC-PCGD	59.10	52.60	68.56
KCDC-PC-2	58.90	53.35	66.99
KCDC-PC	58.89	53.58	66.64
KCDC-GDC	58.29	52.14	67.26
KCDC-GD	58.27	51.88	67.59
MFS	58.25	52.45	66.70
KCDC-PT	58.25	52.18	67.11
Duluth-WSI-SVD-Gap	58.24	52.45	66.66
SNN	58.14	52.33	66.82
KCDC-GD-2	57.90	51.67	66.99
Duluth-R-12	57.72	51.74	66.42
Duluth-R-13	57.59	51.13	67.00
$\boldsymbol{SNN_{GR}}$	57.35	52.62	64.25
Hermit	57.27	52.53	64.16
Duluth-R-15	56.53	49.95	66.11
Random	56.52	50.21	65.73
Duluth-Mix-Narrow-Gap	56.19	47.67	68.59
Duluth-Mix-Narrow-PK2	55.65	46.86	68.45
Duluth-R-110	53.60	46.70	63.63
Duluth-MIX-PK2	50.46	39.70	66.13
KSU KDD	50.42	44.25	59.41
Duluth-Mix-Gap	49.77	38.86	65.64
Duluth-Mix-Uni-PK2	19.12	1.77	44.40
Duluth-Mix-Uni-Gap	18.91	1.52	44.23

Table 10.8: Supervised Recall (SR) for the test split: 60% mapping, 40% evaluation.

Table 10.8 reports results for the 60/40 test set split (60% mapping corpus, 40% evaluation corpus). These results again show that the *SNN* and *Word Overlap* systems surpass the *Random* baseline, however only *Word Overlap* surpasses the *MFS* baseline. System rankings in Table 10.8 are also noticeably similar to those in Table 10.7, though results are generally lower for all systems than those reported in Table 10.7 - a consequence of the reduced size of the mapping corpus in the 60/40 split. Manandhar et al. (2010) observe that this reduction in the size of the mapping corpus has a significant negative impact on the performance of systems that generate greater number of clusters, e.g. differences, with respect to the *MFS* baseline, between the results of the *KSU KDD* system and the *UoY* system (where *KSU KDD* over-generates to a greater extent) are significantly greater for *KSU KDD* in the 60/40 split than the 80/20 split. However, differences, with respect to the *MFS* baseline, between the results of the best performing system in the Paired

F-Score (*Duluth-WSI-SVD-Gap*, generating an average of 1.02 clusters) and the best performing system system in the V-Measure (*Hermit*, generating an average of 10.78 clusters) are, considering the range within which systems scores fall, slight: *Duluth-WSI-SVD-Gap* +0.02, *Hermit* −0.35 for the 80/20 split; *Duluth-WSI-SVD-Ga*p −0.01, *Hermit* −0.98 for the 60/40 split. Furthermore, as Pedersen (2010) notes, results in Tables 10.7 and 10.8 fall within a very narrow range, with little separation between the results of systems that have been purposefully designed to induce word senses and those of (over-generating) random systems (Duluth-R12/13/15/110).

System	**Average Number of Clusters**
1CPI	89.15
SNN$_{GR}$	21.72
SNN	19.56
KSU KDD	17.50
Word Overlap	15.85
UoY	11.54
Hermit	10.78
Duluth-R-110	9.71
Duluth-R-15	4.97
Duluth-WSI	4.15
Duluth-WSI-SVD	4.15
Random	4.00
Gold Standard	3.79
Duluth-R-13	3.00
KCDC-PC-2	2.93
KCDC-PC	2.92
KCDC-PCGD	2.90
KCDC-GDC	2.83
KCDC-GD-2	2.82
KCDC-GD	2.78
Duluth-Mix-Narrow-PK2	2.68
Duluth-MIX-PK2	2.66
Duluth-WSI-Co	2.49
Duluth-Mix-Narrow-Gap	2.42
Duluth-Mix-Uni-PK2	2.04
Duluth-R-12	2.00
Duluth-Mix-Gap	1.61
Duluth-WSI-Co-Gap	1.60
KCDC-PT	1.50
Duluth-WSI-Gap	1.40
Duluth-Mix-Uni-Gap	1.39
Duluth-WSI-SVD-Gap	1.02
MFS	1.00

Table 10.9: The average number of clusters/senses generated by WSI systems.

With respect to the three novel WSI systems, the key findings in Table 10.8 are that the *Word Overlap* system outperforms the *SNN* systems and, again, is shown to outperform

the two best performing participant systems in the unsupervised evaluations. It is also notable that the *Word Overlap* system generates a greater number of clusters on average (15.85) than 25 of the 26 participant systems, yet still outperforms systems whose average number of clusters is far closer to that of the gold standard (3.79). This would imply that the larger clusters *Word Overlap* returns are relatively good word sense clusters, with comparatively high homogeneity (as the V-Measure indicates) and sufficient completeness (as the supervised evaluation indicates) to enable *Word Overlap* to return better scores. Taking the two sets of results in Tables 10.7 and 10.8 into consideration therefore suggests that the comparatively simple *Word Overlap* system is better at inducing word senses than the relatively complex *SNN* systems. In particular, it appears that 'knowledge enriching' the *SNN* system with grammatical relations (SNN_{GR}) has a deleterious effect.

The overall conclusion, taking all three evaluation measures into consideration (V-Measure, Paired F-Score, and SR), is that as no system does well in all three evaluations it is difficult to say which system best induces word senses. This, combined with the bias of the V-Measure to prefer systems that return many small, homogeneous clusters, and the bias of Paired F-Score and Supervised Recall to prefer systems that return larger, more complete clusters, suggests that valid measures, which give a fair assessment of WSI systems, remain to be found - a point noted throughout this chapter, in Meila (2007); Amigó et al. (2009); Pedersen (2010); Manandhar et al. (2010), and further considered by the task's co-ordinators in Klapaftis and Manandhar (2013).

10.4.5 Using Clustering Coefficient Measures to Induce Word Senses in the Test Set

This section evaluates the performance of the unweighted clustering coefficient (described in Chapter 3) and six weighted clustering coefficient measures (described in Chapter 4) within the framework of the SemEval-2010 Word Sense Induction & Disambiguation task. These are the same seven clustering coefficient measures that were previously evaluated in Section 9.5.1.1. The measures are used in a variation of the strong-weak clustering coefficient (*SWCC*) approach to WSI that was introduced in Section 9.3.2, with this approach defined in Algorithm 10.1.

Algorithm 10.1 Strong-Weak Clustering Coefficient WSI (*SWCC*)

1: **for** each test set target word tw **do**
2: Compute $G_{\mathcal{N}(tw)}$, the graph consisting of tw and its first-order neighbours.
3: Delete vertices in $G_{\mathcal{N}(tw)}$ with a clustering coefficient score $\leq \theta$.
4: Apply the Bron-Kerbosch algorithm to find maximal cliques in $G_{\mathcal{N}(tw)}$.
5: Assign each word not found in a maximal clique to the clique containing the word to which it has highest edge weight.
6: **end for**

In Algorithm 10.1, first-order neighbours are words in the test instances of a target word, θ is the clustering coefficient score of the target word, and edge weights in $G_{\mathcal{N}(tw)}$ are

log likelihood ratio (LLR) scores for test instance word pairs: the same word association measure that was used in the *SNN* WSI systems.

SWCC initially constructs 'core clusters' (steps 2 to 4 in Algorithm 10.1), clusters consisting of words with higher clustering coefficient scores than the target word. The hypothesis here is that neighbours of the target word with higher clustering coefficient scores than the target word will have greater semantic relatedness amongst their neighbouring vertices, thus will be good candidates for defining senses of the target word. However, using core clusters to tag test instances with target word senses would mean that many of the test instances remain unclassified. Therefore, step 5 in Algorithm 10.1 expands core clusters by assigning each unassigned test instance word to the clique containing the word to which it has highest LLR edge weight (for the unweighted *CC*, each unassigned word is assigned to cliques containing a word to which it was originally connected to in $G_{\mathcal{N}(tw)}$). Test instances are then tagged using the process that was applied in the *Word Overlap* system (described in Section 10.3.3). Results for the seven clustering coefficient measures are shown in Table 10.10 for all words (nouns and verbs), where *WCC* is the novel weighted generalisation of the clustering coefficient that was introduced in Section 4.3.

System	**V-Measure**	**Paired F-Score**	**SR 80:20**	**SR 60:40**	**|Clusters|**
CC	15.8	**52.4**	**61.3**	**59.7**	18.9
Zhang & Horvath	16.8	44.2	56.7	54.4	16.7
Onnela	16.7	44.3	56.1	54.7	16.4
WCC	16.4	44.8	56.2	54.2	16.9
Opsahl & Panzarasa	18.5	26.2	57.6	55.7	21.7
Barrat	18.1	26.9	57.5	56.1	21.2
Lopez-Fernandez	**23.6**	15.7	50.4	48.3	24.4

Table 10.10: Results for the seven clustering coefficient measures used in the strong-weak clustering coefficient approach to WSI.

Results in Table 10.10 show that the unweighted clustering coefficient *CC* is the overall best performing measure, ranked first in both the Paired F-Score evaluation and in the 80/20 and 60/40 test set splits of the supervised evaluation. However, *CC* is outperformed by all weighted clustering coefficient measures in the V-Measure evaluation. Notably, all seven *SWCC* systems are shown to over-generate clusters, a result of the maximal cliques method that is applied here. However, these systems may be finding finer-grained senses of target words than are defined in the gold standard.

Considering the weighted clustering coefficient measures with respect to the Paired F-Score and Supervised Recall (SR) results, Table 10.10 has three distinct bands of performance (as found in the previous evaluation of the weighted clustering coefficient measures), where the three best performing measures (*Zhang & Horvath*, *Onnela*, *WCC*) are shown to return scores that are highly similar and that are clearly separated from those of the fourth (*Opsahl & Panzarasa*) and fifth (*Barrat*) best performing measures, which, in turn, are clearly separated from those of the worst performing measure, *Lopez-Fernandez*. This indicates (as it did in the previous evaluation of these measures) that the less information

a measure uses the worse its performance is. However, the score for *Lopez-Fernandez* in the V-Measure evaluation is at odds with this observation, though could be a result of the V-Measure's bias for WSI systems that generate a greater number of clusters than other systems. A further observation is that marginal differences between the results of *Zhang & Horvath*, *Onnela*, and *WCC*, which use all three edge weights in each triangle around a target word vertex, indicate that these measures are returning highly similar clustering solutions; an observation that also applies to the *Opsahl & Panzarasa* and *Barrat* measures, which use two edge weights in each triangle around a target word vertex.

The conclusion drawn in this evaluation is the same as that drawn in the evaluation presented in Section 9.5.1.1. That is, it remains unclear how one could apply a combinatorial analysis to a set of measures that aim to quantify both the topology and weighted vertex connectivity of a graph into a single value (score). As noted in the previous evaluation, even the more mathematically inclined papers on weighted clustering coefficients (Onnela et al., 2005; Kalna and Higham, 2006; Saramäki et al., 2007) skirt this issue, with Saramäki et al. summarising the problem: "our conclusion is that there is no single general-purpose measure for characterizing clustering in weighted complex networks" (Saramäki et al., 2007, p.4). This issue, allied to the known biases of the three evaluation measures applied in the task, plus the fact that it is the WSI system not the clustering method per se that generates word sense clusters, makes it difficult to say which of the six weighted clustering coefficient measures best induces word senses.

10.5 Conclusions

The SemEval-2010 Word Sense Induction & Disambiguation task attempted to provide a fair evaluation of word sense induction systems, evident in adjustments made to the arguably imprecise evaluation methods that were applied in the SemEval-2007 task (Agirre and Soroa, 2007) and in Manandhar et al.'s post-evaluation assessment of systems, using a number of measures other than V-Measure and Paired F-Score[12]. The evaluation also provided a formal framework in which word sense induction researchers could compare systems, rather than have to resort to some possibly ad hoc evaluation methodology of their own devising. Participants were also given a space in which to: discuss word sense induction; critique the methodology applied, and, for those participants who analysed the evaluation measures (Pedersen, 2010; Manandhar et al., 2010), note the importance of finding better measures for future evaluations.

The evaluation measures used to assess the word sense induction performance of systems were shown to be unsuited to the task, both through analysis and by results. Systems returning high V-Measure results were found to return low Paired F-Score results; systems returning high Paired F-Score results were found to return low V-Measure results. No system was found to return good results for both measures. The novel *SNN* and *Word Overlap* systems introduced in this chapter were shown to return the highest V-Measure scores, implying that these systems are returning clustering solutions that have relatively

[12] `http://www.cs.york.ac.uk/semeval2010_WSI/task_14_ranking.html`

high levels of cluster homogeneity and completeness. However, these results were countered by those in the Paired F-Score evaluation which imply that the clusters returned by these systems actually have low values of completeness. These findings lead to an analysis of the evaluation measures, showing that both measures are biased to favour degenerate clustering solutions. The V-Measure was shown to favour a degenerate clustering solution in which each instance is placed in a separate cluster. The Paired F-Score was shown to favour a degenerate clustering solution in which all instances are placed in just one cluster. As Pedersen (2010) states:

> "The results of the evaluation are in some sense confusing - a system that ranks near the top according to one measure may rank at the bottom or middle of another. There was not any single system that did well according to all of the different measures[13]. The situation is so extreme that in some cases a system would perform near the top in one measure, and then below random baselines in another. These stark differences suggest a real need for continued development of other methods for evaluating unsupervised sense induction."
> (Pedersen, 2010, p.2)

The results reported in this evaluation therefore provide no clear indication as to which approach best induces word senses; rather, they leave researchers with the quandary as to which methods, existing or 'other', may best be applied in the evaluation of word sense induction systems. A number of suggestions that future evaluations might consider are as follows:

Valid Evaluation Measures: Valid evaluation measures that accurately reflect the performance of a word sense induction system should be used. Two measures that might be considered are B-Cubed (Bagga and Baldwin, 1998) and BLANC (Recasens and Hovy, 2011)[14]. Indeed, Amigó et al., in their formal assessment of external evaluation measures (Amigó et al., 2009), found that B-Cubed was the only measure to give an unbiased account of clustering solutions, stating that:

> "A practical conclusion of our work is that the combination of B-Cubed Precision and Recall metrics is the only one that is able to satisfy all constraints (for non-overlapping clustering). We take this result as a recommendation to use B-Cubed metrics for generic clustering problems."[15]
> (Amigó et al., 2009, p.29)

Greater Consensus on Gold Standard Word Senses: Gold standard word senses require consensus. This requires that more than one or two people verify both the quality of the test set and the word senses that are assigned to target words in test instances.

[13] Pedersen (2010) includes the third set of results for the supervised word sense disambiguation evaluation.

[14] Manandhar et al. (2010) report post evaluation results for B-Cubed `http://www.cs.york.ac.uk/semeval2010_WSI/task_14_ranking.html`.

[15] Amigó et al. provide a generalisation of B-Cubed that is applicable to the evaluation of overlapping (soft) clustering solutions.

Manual annotation is time consuming; however, a web-based approach might be envisaged, similar to that applied for lexical substitution in Biemann (2013), in which some type of voting scheme is used to gain consensus on the word senses that should be assigned to test instances.

Less Abstract Senses: Test words should have few senses and these senses should be concrete. Target words in this evaluation, such as *cultivate*, *foundation*, *operate*, and *shape* (amongst others), have many abstract senses that humans would find difficult to define and separate from each other; to expect a machine program to find these senses, at this stage in the development of word sense induction systems, is too ambitious. Test words might be selected that have two or three clearly defined, predominant senses; thus, the task would be to induce these senses, for example, to induce the two predominant senses of *triangle* as an INSTRUMENT and a SHAPE rather than to attempt to induce the many, abstract senses of *instrument* and *shape*.

Chapter 11

Summary, Conclusions and Future Research

11.1 Summary and Conclusions

This thesis presented a set of novel graph-theoretic methods for extracting word meanings from graphs built from plain or part of speech tagged text. The hypothesis, stated in the introduction of the thesis, that meanings of words can be discovered directly from text was shown to be true by partitioning graph models of word co-occurrence to word sense clusters. The hypothesis that parameter-free methods are well-suited to discovering word meanings from text was validated in a variety of evaluation frameworks, where graph-based Word Sense Induction (WSI) systems that incorporate parameter-free methods were shown to return results comparable to or better than those of parameterised systems.

The graph models of word sense introduced in this thesis are simple to understand and straightforward to implement, and could be easily adapted to Natural Language Processing (NLP) tasks other than WSI as they require neither complex linguistic preprocessing of text nor recourse to external knowledge resources. Indeed, the models of word sense presented in this thesis and the methods that are applied to them were deliberately designed to be as transparent as a possible; the intention being to see if word senses could be induced without recourse to computational methods of a more complex nature. The conjecture made was that semantic similarity between words co-occurring in contexts can be represented using graph models of word co-occurrence; models that encapsulate both topological and weighted relations between words by representing words as vertices and similarity between word pairs as weighted edges connecting vertex pairs. Chapter 2 introduced this idea, showing how maximal affinity (similarity) between sets of word pairs defines semantic unity. This chapter also showed how the topology of graphs can be used to find subgraphs with high vertex (word) cohesion, with these subgraphs used to define word senses.

Chapter 3 considered measures of the strength of association between word pairs, measures used as edge weights in word co-occurrence graphs. The chapter's aim was to select, out of the many proposed, those measures that may be best-suited to graph-based WSI. A review of association measures found that two measures that are often applied in

NLP are unsuited to sparse data, thus are not well-suited to the low co-occurrence counts typically found in the corpora used in this thesis. A third measure, though exacting in its assessment of the association between word pairs, was found to be too computationally intensive for practical use. Three measures were selected: word co-occurrence frequency (used as a baseline measure); conditional probability (for use in directed graphs), and the log likelihood ratio. A further review of graph-based measures of vertex connectivity found that the majority of these measures are unsuited to finding vertex cohesion in graphs. In particular, the more complex measures were found to merge small vertex cliques with high cohesion into larger clusters: cliques that may best define less predominant senses of words. This second review led to the introduction of the clustering coefficient, a measure of vertex cohesion in unweighted graphs. Noting that many real world graphs contain weights that quantify some qualitative aspect between vertices, Chapter 4 introduced three novel generalisations of the clustering coefficient to the weighted case, with the first generalisation applied in the evaluation presented in Chapter 9. Results for this evaluation showed that the novel generalisation returns marginally better results than the unweighted clustering coefficient.

An overview of clustering methods was given in Chapter 5. This chapter began by illustrating why commonly-applied clustering methods, shown to return good results for many tasks, are not well-suited to the task of WSI. This led to a review of soft, fuzzy, and parameter-free clustering methods that have been applied in WSI; clustering methods that are arguably best-suited to WSI. This chapter also introduced a novel graph-based parameter-free soft clustering algorithm, *MaxMax*. This algorithm allows vertices to self organise to clusters by using the maximal affinity each vertex has to other vertices in the input graph, thus finds the number of clusters in the input graph automatically: no fixed k restriction on the number of clusters is prescribed, nor any restriction on the size, shape, and density of clusters. This algorithm was shown to be deterministic; guaranteed to terminate, and to run in time linear in the number of edges in the graph. Tests using weighted n-bipartite clique and small world mixture graphs showed that *MaxMax* separates merged graphs into their component parts perfectly. An evaluation using synthetic datasets also showed *MaxMax* to return results that are comparable with those of three other clustering algorithms; algorithms that find the number of clusters in a graph automatically, thus are representative of the same class as *MaxMax*. This algorithm was used in the WSI systems evaluated in Chapters 8, 9, and 10 where system performance was shown to be comparable to the current best performing WSI systems.

Chapter 6 began with a discussion regarding word sense in which a number of arguments were presented that attempt to either define or disprove the concept of word sense. The task of WSI was then formally introduced, with this approach to identifying word senses compared with the approach taken in the related task of Word Sense Disambiguation (WSD). This chapter also argued the case for the use of a graph-based approach to WSI over others that could be applied. This led to a survey, in Chapter 7, of graph-based approaches to WSI that have been proposed in the literature.

Chapter 8 presented two preliminary evaluations of the *MaxMax* clustering algorithm.

Though limited in scope, both evaluations exemplified the viability of a graph-based approach to NLP. Results for the first evaluation showed *MaxMax*'s clustering performance to be comparable to that of the *Chinese Whispers* clustering algorithm (a clustering algorithm that has been shown to be of use in various NLP tasks). Results for second evaluation showed that *MaxMax* induces a greater number of senses in a far more efficient manner than *MCL*, a Markov model clustering algorithm.

A far more comprehensive evaluation was presented in Chapter 9. The aim of this evaluation was to induce the senses of British National Corpus co-ordination pattern nouns that are defined in WordNet 3.0 (27,071 nouns in total) using the unweighted and weighted clustering coefficient measures that were introduced in Chapters 3 and 4. Two novel WSI systems were used to induce senses: the first used a strong-weak clustering coefficient approach; the second used an edge weight thresholding approach. The hypothesis in the strong-weak clustering approach is that neighbouring words of a target word with higher clustering coefficient scores than the target word should best define senses of the target word. A correlation analysis of the clustering coefficient and its weighted generalisation showed that both measures are poor predictors of word sense. However, this was not borne out in the evaluation where the clustering coefficient approaches were shown to outperform two state of the art WSI systems; furthermore, to have far higher coverage of word senses. Results for the weighted clustering coefficient measure were shown to exceed those of its unweighted counterpart. However, the expectation that a weighted generalisation of the clustering coefficient would convincingly outperform the unweighted clustering coefficient was shown to be unfounded. One outcome of using the weighted clustering coefficient in this evaluation, and in a further evaluation presented in Chapter 10, was that it became clear that a measure which attempts to combine topology and 'topography' (weights) is a difficult, if intractable, measure to analyse from a mathematical or probabilistic perspective. Thus, unlike the unweighted clustering coefficient (which can be analysed from these perspectives), there is no transparency in the process that results in certain sets of words being returned over others. A surprising outcome of this evaluation was that the best results were obtained by using two simple 'baseline' approaches: computing maximal cliques within the neighbourhoods of target words outperformed all other approaches (using an evaluation measure of Accuracy); a straightforward approach using co-occurrence counts between words outperformed all other approaches (using an evaluation measure of Precision). These results therefore suggest that comparatively simple methods may best induce word senses.

Chapter 10 introduced three novel WSI systems: two knowledge poor (unsupervised) systems, with the third system knowledge enriched by grammatical relations. All three systems used the *MaxMax* algorithm to generate sense clusters. The systems were evaluated within the framework of the SemEval-2010 Word Sense Induction and Disambiguation task, with results compared to those of the twenty six participant systems. Results for the WSI evaluation showed that the three novel systems returned the best results for one evaluation measure yet returned the worst results for the other. This outcome led to an analysis of the evaluation measures which found that each measure is biased to favour par-

ticular types of degenerate clustering solutions, solutions that return higher scores than any of the participant systems. The chapter concluded with a number of suggestions that future evaluations might consider.

11.2 Future Research

The methods presented in this thesis for inducing word senses use data extracted from the British National Corpus, a 100 million word corpus[1]. These methods are shown to return results that are comparable to the best results reported in the literature. A straightforward extension of the research reported in this thesis would therefore use larger corpora, as greater volumes of data would contain higher word co-occurrence counts, thus should better define word senses; furthermore, larger corpora should contain a greater number of word senses.

A simple model of word relatedness is applied in this thesis: co-occurrence in co-ordination patterns. A variation of this model might use dependency relations as graph edges, with the dependencies between words (vertices) automatically extracted from text using the lexical attraction model proposed in Yuret (1998). The graph models of word sense would remain unsupervised, yet, in theory, should be more precise. For example, dependency relations might demarcate the two senses of the word row in [boys row boats pasts rows of buoys]. The use of dependency relations may also allow one to disambiguate the meanings of multi-word expressions such as flavour of the month and raining cats and dogs, or the play on words in soixante huit hearts.

A further idea is to have graph edges represent the contextual use of words, for example by using Sparse Distributed Memory (SDM) vectors (Kanerva, 1988) or Holographic Reduced Representations (HRR) (Plate, 2003): 'memories' of word use, analogous to the notion of Memory-Based Learning proposed in Daelemans and van den Bosch (2005). A graph model of word co-occurrence using this approach would represent 'the memory' of a word's contextual use as a SDM or HRR vector vertex, with two words (vertices) having their shared contextual use (their 'shared memory') stored on the connecting edge: the sum of word pair vectors for SDM; the convolution of word pair vectors for HRR.

Two areas of particular interest are neologisms and synonymy. Sense inventories such as WordNet cannot keep pace with the extent to which new words and new senses are introduced into language. The conjecture here is that the induction methods that are used in this thesis to identify word meanings may also be applicable to the discovery of neologisms. Concerning synonymy, preliminary experiments have been carried out in which I attempt to find the synonyms of a given target word using graph-based methods that are similar to those introduced in this thesis. These experiments show that synonyms of a target word frequently occur in target word contexts. Thus, if (target word, context word) pairs are ranked by association measure scores, synonyms of the target word often appear high up in the ranking[2]. Unfortunately, this is also the case for antonyms of target

[1] `http://www.natcorp.ox.ac.uk/`

[2] Using the log-likelihood ratio as the association measure between the target word and words occurring within target word contexts (sentences). Co-occurrence counts are obtained from the ukWaC corpus:

words. Consequently, a graph-based clustering method that uses association measure scores as edge weights will often return clusters that contain synonyms and antonyms of target words. The implementation of a computational process that can separate words of the same or similar meaning from those of the opposite meaning will therefore require further consideration. This research is ongoing.

`http://wacky.sslmit.unibo.it`. Synonyms and antonyms of a target word are taken to be those listed in WordNet 3.0.

Bibliography

Aggarwal, C. and Reddy, C. (2013). *Data Clustering: Algorithms and Applications*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. Taylor & Francis.

Agirre, E. and Edmonds, P. (2007). *Word Sense Disambiguation: Algorithms and Applications*. Text, Speech and Language Technology. Springer.

Agirre, E., Martínez, D., de Lacalle, O., and Soroa, A. (2006a). Evaluating and Optimizing the Parameters of an Unsupervised Graph-Based WSD Algorithm. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pages 89–96. Association for Computational Linguistics.

Agirre, E., Martínez, D., de Lacalle, O., and Soroa, A. (2006b). Two Graph-Based Algorithms for State-of-the-Art WSD. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 585–593. Association for Computational Linguistics.

Agirre, E. and Soroa, A. (2007). SemEval-2007 Task 02: Evaluating Word Sense Induction and Discrimination Systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 7–12. Association for Computational Linguistics. Prague, Czech Republic.

Agirre, E., Soroa, A., and Donostia, B. (2007). UBC-AS: A Graph Based Unsupervised System for Induction and Classification. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 346–349.

Aleman-Meza, B., Nagarajan, M., Ramakrishnan, C., Ding, L., Kolari, P., Sheth, A. P., Arpinar, I. B., Joshi, A., and Finin, T. (2006). Semantic Analytics on Social Networks: Experiences in Addressing the Problem of Conflict of Interest Detection. In *Proceedings of the 15th International Conference on World Wide Web*, pages 407–416. ACM.

Amigó, E., Gonzalo, J., Artiles, J., and Verdejo, F. (2009). A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Information Retrieval*, 12(4):461–486.

Anand, P., Escuadro, H., Gera, R., and Martell, C. (2011). Triangular Line Graphs and Word Sense Disambiguation. *Discrete Applied Mathematics*.

Apidianaki, M. (2008a). *Automatic Sense Acquisition for Word Sense Disambiguation and Lexical Selection in Translation*. PhD thesis, Université Paris 7-Denis Diderot.

Apidianaki, M. (2008b). Translation-Oriented Word Sense Induction based on Parallel Corpora. *Actes de LREC*.

Apidianaki, M. (2009). Data-Driven Semantic Analysis for Multilingual WSD and Lexical Selection in Translation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 77–85. Association for Computational Linguistics.

Apidianaki, M. (2010). Discovering Semantic Relations by Means of Unsupervised Sense Clustering. *Semantic Relations, Theory and Applications*, 3.

Apidianaki, M., He, Y., and Way, A. (2009). Capturing Lexical Variation in MT Evaluation Using Automatically Built Sense-Cluster Inventories. In Kwong, O., editor, *PACLIC*, pages 53–62. City University of Hong Kong Press.

Apidianaki, M. and Van de Cruys, T. (2011). A Quantitative Evaluation of Global Word Sense Induction. *Computational Linguistics and Intelligent Text Processing*, pages 253–264.

Artiles, J., Amigó, E., and Gonzalo, J. (2009). The Role of Named Entities in Web People Search. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 534–542.

Artiles, J., Borthwick, A., Gonzalo, J., Sekine, S., and Amigó, E. (2010). WePS-3 Evaluation Campaign: Overview of the Web People Search Clustering and Attribute Extraction Tasks. In *CLEF (Notebook Papers/LABs/Workshops)*.

Artiles, J., Gonzalo, J., and Sekine, S. (2007). The SemEval-2007 WEPS Evaluation: Establishing a Benchmark for the Web People Search Task. *Proceedings of SemEval*, pages 64–69.

Baeza-Yates, R. and Ribeiro-Neto, B. (2011). *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison Wesley Professional, 2nd edition.

Bagga, A. and Baldwin, B. (1998). Algorithms for Scoring Coreference Chains. In *The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*, pages 563–566.

Barabási, A. (2003). *Linked*. A Plume Book. Plume.

Baroni, M. and Bisi, S. (2004). Using Cooccurrence Statistics and the Web to Discover Synonyms in a Technical Language. In *Proceedings of LREC 2004*, pages 1725–1728.

Barrat, A., Barthelemy, M., Pastor-Satorras, R., and Vespignani, A. (2004). The Architecture of Complex Weighted Networks. volume 101, page 3747. National Academy of Sciences.

Benhardus, J. and Kalita, J. (2013). Streaming Trend Detection in Twitter. *International Journal of Web Based Communities*, 9(1):122–139.

Bentley, J. L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communication*, 18:509–517.

Berendt, B. and Navigli, R. (2006). Finding Your Way Through Blogspace: Using Semantics for Cross-Domain Blog Analysis. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 1–8.

Berge, C. (1984). *Hypergraphs: Combinatorics of Finite Sets*. North-Holland Mathematical Library. Elsevier Science.

Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The Semantic Web. *Scientific American*, 284(5):28–37.

Bernhard, D. (2010). Morphonet: Exploring the Use of Community Structure for Unsupervised Morpheme Analysis. *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pages 598–608.

Bezdek, J. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers.

Biemann, C. (2006a). Chinese Whispers - an Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proceedings of the HLT-NAACL-06 Workshop on Textgraphs-06*.

Biemann, C. (2006b). Unsupervised Part-of-Speech Tagging Employing Efficient Graph Clustering. In *Proceedings of the COLING/ACL 2006 Student Research Workshop*, pages 7–12. Association for Computational Linguistics.

Biemann, C. (2007). *Unsupervised and Knowledge-Free Natural Language Processing in the Structure Discovery Paradigm*. PhD thesis, University of Leipzig.

Biemann, C. (2013). Creating a System for Lexical Substitutions from Scratch Using Crowdsourcing. *Language Resources and Evaluation*, 47(1):97–122.

Biemann, C. and Teresniak, S. (2005). Disentangling from Babylonian Confusion–Unsupervised Language Identification. *Computational Linguistics and Intelligent Text Processing*, pages 773–784.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Bogdanova, D. (2010). A Framework for Figurative Language Detection Based on Sense Differentiation. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 67–72. Association for Computational Linguistics.

Bollobás, B. (1998). *Modern Graph Theory*. Springer.

Bollobás, B. and Riordan, O. (2006). *Percolation*. Cambridge University Press.

Bondy, J. and Murty, U. (2011). *Graph Theory: An Advanced Course*. Springer.

Bontcheva, K. and Rout, D. (2012). Making Sense of Social Media Streams through Semantics: a Survey. In *Semantic Web Journal*.

Bordag, S. (2006). Word Sense Induction: Triplet-Based Clustering and Automatic Evaluation. *Proceedings of EACL-06. Trento*.

Bordag, S. (2008). A Comparison of Co-occurrence and Similarity Measures as Simulations of Context. In *Proceedings of the 9th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 52–63. Springer-Verlag.

Borgelt, C. (2006). *Prototype-Based Classification and Clustering*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, Universitätsbibliothek.

Borin, L. and Forsberg, M. (2010). From the People's Synonym Dictionary to Fuzzy Synsets - First Steps. In *LREC 2010 Workshop on Semantic Relations. Theory and Applications*, pages 18–25.

Borin, L., Forsberg, M., and Lönngren, L. (2008). The Hunting of the BLARK - SALDO, a Freely Available Lexical Database for Swedish Language Ttechnology. *Resourceful Language Technology. Festschrift in Honor of Anna Sågvall Hein*, (7):21–32.

Boyd-Graber, J. and Blei, D. (2007). PUTOP: Turning Predominant Senses into a Topic Model for Word Sense Disambiguation. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 277–281. Association for Computational Linguistics.

Brin, S. and Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117.

Brody, S. and Lapata, M. (2009). Bayesian Word Sense Induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–111. Association for Computational Linguistics.

Bron, C. and Kerbosch, J. (1973). Algorithm 457: Finding All Cliques of an Undirected Graph. *Communications of the ACM*, 16:575–577.

Burgess, A. (1962). *A Clockwork Orange*. Heinemann.

Carpuat, M. and Wu, D. (2007). Improving Statistical Machine Translation Using Word Sense Disambiguation. In *EMNLP-CoNLL*, pages 61–72.

Carroll, L. (1865). *Alice's Adventures in Wonderland*. Macmillan.

Cazals, F. and Karande, C. (2008). A note on the problem of reporting maximal cliques. *Theoretical Computer Science*, 407(1):564–568.

Chan, Y. S., Ng, H. T., and Chiang, D. (2007). Word Sense Disambiguation Improves Statistical Machine Translation. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 33.

Chang, C.-H., Kayed, M., Girgis, R., and Shaalan, K. F. (2006). A Survey of Web Information Extraction Systems. *Knowledge and Data Engineering, IEEE Transactions*, 18(10):1411–1428.

Chang, H. and Yeung, D.-Y. (2008). Robust Path-Based Spectral Clustering. *Pattern Recognition*, 41(1):191–203.

Chen, J., Zaïane, O., and Goebel, R. (2008). An Unsupervised Approach to Cluster Web Search Results Based on Word Sense Communities. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 1, pages 725–729.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. The MIT Press Paperback Series. M.I.T. Press.

Cimiano, P., Hotho, A., and Staab, S. (2005). Learning Concept Hierarchies from Text Corpora Using Formal Concept Analysis. *Journal of Artificial Intelligence Research*, 24(1):305–339.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. L. Erlbaum Associates.

Cohen, T. and Widdows, D. (2009). Empirical Distributional Semantics: Methods and Biomedical Applications. *Journal of Biomedical Informatics*, 42(2):390–405.

Collins, A. and Loftus, E. (1975). A Spreading-Activation Theory of Semantic Processing. *Psychological Review*, 82(6):407.

Collins, A. and Quillian, M. (1969). Retrieval Time from Semantic Memory. *Journal of Verbal Learning and Verbal Behavior*, 8(2):240–247.

Cruse, D. (2000). *Aspects of the Microstructure of Word Meanings*, volume Polysemy: Theoretical and Computational Approaches, pages 30–51. Oxford University Press.

Daelemans, W. and van den Bosch, A. (2005). *Memory-Based Language Processing*. Studies in Natural Language Processing. Cambridge University Press.

Dasgupta, S., Papadimitriou, C., and Vazirani, U. (2006). *Algorithms*. McGraw-Hill.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Di Marco, A. and Navigli, R. (2011). Clustering Web Search Results with Maximum Spanning Trees. In *Proceedings of the XIIth International Conference of the Italian Association for Artificial Intelligence (AI*IA)*, pages 201–212.

Di Marco, A. and Navigli, R. (2013). Clustering and Diversifying Web Search Results with Graph-Based Word Sense Induction. *Computational Linguistics*, 39(4).

Diestel, R. (2006). *Graph Theory.* Springer.

Dorow, B. (2007). *A Graph Model for Words and their Meanings.* PhD thesis, Institut für Maschinelle Sprachverarbeitung der Universität Stuttgart.

Dorow, B. and Widdows, D. (2003a). Discovering Corpus-Specific Word Senses. In *Proceedings of the Tenth Conference, European Chapter of the Association for Computational Linguistics-Volume 2*, pages 79–82. Association for Computational Linguistics.

Dorow, B. and Widdows, D. (2003b). Discovering Corpus-Specific Word Senses. In *Proceedings of the Tenth Conference, European Chapter of the Association for Computational Linguistics-Volume 2*, pages 79–82. Association for Computational Linguistics.

Dorow, B., Widdows, D., Ling, K., Eckmann, J., Sergi, D., and Moses, E. (2005). Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination. In *2nd Workshop organized by the MEANING Project.*

Douw, L., Schoonheim, M., Landi, D., van der Meer, M., Geurts, J., Reijneveld, J., Klein, M., and Stam, C. (2011). Cognition is Related to Resting-State Small-World Network Topology: an Magnetoencephalographic Study. *Neuroscience*, 175:169–177.

Dunn, J. C. (1973). *Fuzzy Mathematics in Pattern Classification.* PhD thesis, Cornell University, Ithaca, New York.

Dunning, T. (1993). Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74.

Dyson, G. (1998). *Darwin Among the Machines: the Evolution of Global Intelligence.* Helix Book. Perseus Books.

Easley, D. and Kleinberg, J. (2010). *Networks, Crowds, and Markets: Reasoning About a Highly Connected World.* Cambridge University Press.

Eckmann, J. and Moses, E. (2002). Curvature of Co-Links Uncovers Hidden Thematic Layers in the World Wide Web. *Proceedings of the National Academy of Sciences*, 99(9):5825.

Epstein, J. (2006). *Generative Social Science: Studies in Agent-Based Computational Modeling.* Princeton University Press.

Erdős, P. and Rényi, A. (1959). On Random Graphs. *Publicationes Mathematicae Debrecen*, 6:290–297.

Erdős, P. and Rényi, A. (1960). *On the Evolution of Random Graphs.* Akad. Kiadó.

Erk, K. and McCarthy, D. (2009). Graded Word Sense Assignment. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1*, pages 440–449, Singapore. Association for Computational Linguistics.

Erk, K., McCarthy, D., and Gaylord, N. (2012). Measuring Word Meaning in Context. 29(3):511–554.

Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, volume 1996, pages 226–231. Portland: AAAI Press.

Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. pages 128–140.

Evert, S. (2005). *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD thesis, Institut für Maschinelle Sprachverarbeitung der Universität Stuttgart.

Evert, S. and Krenn, B. (2004). Computational Approaches to Collocations. *Introductory Course at the European Summer School on Logic, Language, and Information (ESSLLI 2003), Vienna*.

Fano, R. and Hawkins, D. (1961). Transmission of Information: A Statistical Theory of Communications. *American Journal of Physics*, 29:793.

Feldman, R. and Sanger, J. (2007). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. The MIT Press.

Ferrer-i-Cancho, R. and Solé, R. (2001). The Small World of Human Language. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1482).

Ferret, O. (2004). Discovering Word Senses from a Network of Lexical Cooccurrences. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING 2004. Association for Computational Linguistics.

Firth, J. (1935). The Technique of Semantics. *Transactions of the Philological Society*, 34(1):36–73.

Firth, J. (1957). A Synopsis of Linguistic Theory 1930-55. *Studies in Linguistic Analysis*, pages 1–32.

Fisher, R. (1922). On the Interpretation of χ^2 from Contingency Tables, and the Calculation of P. *Journal of the Royal Statistical Society*, 85(1):87–94.

Flake, G. W., Tarjan, R. E., and Tsioutsiouliklis, K. (2004). Graph Clustering and Minimum Cut Trees. *Internet Mathematics*, 1(4):385–408.

Fortunato, S. and Barthélemy, M. (2007). Resolution Limit in Community Detection. *Proceedings of the National Academy of Sciences*, 104(1).

Fountain, T. and Lapata, M. (2012). Taxonomy Induction Using Hierarchical Random Graphs. *2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 446–476.

Freeman, L. (1979). Centrality in Social Networks: I. Conceptual Clarification. *Social Networks*, 1(3):215–239.

Frege, G. (1892). ber Sinn und Bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, 100(1):25–50.

Frey, B. J. and Dueck, D. (2007). Clustering by Passing Messages Between Data Points. *Science*, 315:972–976.

Fu, L. and Medico, E. (2007). FLAME, a Novel Fuzzy Clustering Method for the Analysis of DNA Microarray Data. *BMC Bioinformatics*, 8(1):3.

Furnas, G., Landauer, T., Gomez, L., and Dumais, S. (1984). Statistical Semantics: Analysis of the Potential Performance of Keyword Information Systems. In *Human Factors in Computer Systems*, pages 187–242. Ablex Publishing Corporation.

Gale, W., Church, K., and Yarowsky, D. (1992a). Work on Statistical Methods for Word Sense Disambiguation. In *Probabilistic Approaches to Natural Language: Papers from the 1992 AAAI Fall Symposium*, pages 23–25.

Gale, W. A., Church, K. W., and Yarowsky, D. (1992b). A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26(5-6):415–439.

Gionis, A., Mannila, H., and Tsaparas, P. (2007). Clustering Aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):4.

Girvan, M. and Newman, M. (2002). Community Structure in Social and Biological Networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821.

Grindrod, P. (2002). Range-Dependent Random Graphs and their Application to Modeling Large Small-World Proteome Datasets. *Physical Review E*, 66(6).

Guha, R., McCool, R., and Miller, E. (2003). Semantic Search. In *Proceedings of the 12th International Conference on World Wide Web*, pages 700–709. ACM.

Hänggi, J., Wotruba, D., and Jäncke, L. (2011). Globally Altered Structural Brain Network Topology in Grapheme-Color Synesthesia. *The Journal of Neuroscience*, 31(15).

Harris, Z. (1988). *Language and Information*. Columbia University Press.

Harris, Z. S. (1954). Distributional Structure. *Word*, 10(23):146–162.

Heineman, G., Pollice, G., and Selkow, S. (2008). *Algorithms in a Nutshell*. In a Nutshell. O'Reilly.

Hillis, W. (1988). Intelligence as an Emergent Behavior; or, the Songs of Eden. *Daedalus*, 117(1):175–189.

Hope, D. and Keller, B. (2013a). MaxMax: A Graph-Based Soft Clustering Agorithm Applied to Word Sense Induction. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing 14th International Conference, CICLing 2013*, volume 7816 of *Lecture Notes in Computer Science*, pages 368–381. Springer-Verlag. Samos, Greece, March 24-30.

Hope, D. and Keller, B. (2013b). UoS: A Graph-Based System for Graded Word Sense Induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 689–694. Association for Computational Linguistics. Altanta, Georgia, June 14-15.

Hornby, A. S. (1954). *A Guide to Patterns and Usage in English.* Oxford University Press.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). OntoNotes: the 90% Solution. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 57–60. Association for Computational Linguistics.

Ide, N. and Véronis, J. (1998). Introduction to the Special Issue on Word Sense Disambiguation: the State of the Art. *Computational Linguistics*, 24(1):2–40.

Jain, A. and Dubes, R. (1988). *Algorithms for Clustering Data.* Prentice Hall.

Jain, A. K. and Law, M. H. (2005). Data Clustering: A User's Dilemma. In *Pattern Recognition and Machine Intelligence*, pages 1–10. Springer.

Jurgens, D. (2012). An Evaluation of Graded Sense Disambiguation Using Word Sense Induction. *Proceedings of *SEM First Joint Conference on Lexical and Computational Semantics, 2012. Association for Computational Linguistics*, pages 189–198. Montreal,Canada.

Jurgens, D. and Klapaftis, I. (2013). SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval).*

Jurgens, D. and Stevens, K. (2010). HERMIT: Flexible Clustering for the SemEval-2 WSI Task. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 359–362. Association for Computational Linguistics.

Kadushin, C. (2012). *Understanding Social Networks: Theories, Concepts, and Findings.* Oxford University Press.

Kalna, G. and Higham, D. (2006). Clustering Coefficients for Weighted Networks. In *Symposium on Network Analysis in Natural Sciences and Engineering*, pages 45–51.

Kalna, G. and Higham, D. (2007). A Clustering Coefficient for Weighted Networks, with Application to Gene Expression Data. *AI Communications*, 20(4):263–271.

Kanerva, P. (1988). *Sparse Distributed Memory.* Bradford Books. MIT Press.

Kann, V. and Rosell, M. (2006). Free Construction of a Free Swedish Dictionary of Synonyms. In *Proceedings of the 15th NODALIDA Conference*, pages 105–110. Citeseer.

Karinthy, F. (1929). *Minden másképpen van*. Atheneum Irodai es Nyomdai R.-T. Kiadása.

Kaufman, L. and Rousseeuw, P. (1990). *Finding Groups in Data*. A Wiley-Interscience.

Kilgarriff, A. (1997). I Don't Believe in Word Senses. *Computers and the Humanities*, 31(2):91–113.

Klapaftis, I. (2008). *Unsupervised Concept Hierarchy Induction : Learning the Semantics of Words*. PhD thesis, University of York.

Klapaftis, I. and Manandhar, S. (2008). Word Sense Induction Using Graphs of Collocations. In *Proceeding of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence*, pages 298–302. IOS Press.

Klapaftis, I. and Manandhar, S. (2010a). Word Sense Induction and Disambiguation Using Hierarchical Random Graphs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 745–755. Association for Computational Linguistics.

Klapaftis, I. P. and Manandhar, S. (2006). Term Sense Disambiguation for Ontology Learning. *Intelligent Systems Design and Applications*, 2:844–849.

Klapaftis, I. P. and Manandhar, S. (2010b). Taxonomy Learning Using Word Sense Induction. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics(NAACL-HLT)*, pages 82–90, Los Angeles, California. Association for Computational Linguistics.

Klapaftis, I. P. and Manandhar, S. (2013). Evaluating Word Sense Induction and Disambiguation Methods. *Language Resources and Evaluation*, pages 1–27.

Klaua, D. (1965). Uber einen Ansatz zur mehrwertigen Mengenlehre. *Monatsberichte der Deutschen Akademie der Wissenschaften Berlin*, 7:859–867.

Kleinberg, J. (1999). Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press.

Korkontzelos, I., Klapaftis, I., and Manandhar, S. (2009). Graph Connectivity Measures for Unsupervised Parameter Tuning of Graph-Based Sense Induction Systems. In *Proceedings of the NAACL HLT Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, pages 36–44. Association for Computational Linguistics.

Korkontzelos, I. and Manandhar, S. (2009). Detecting Compositionality in Multi-Word Expressions. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 65–68. Association for Computational Linguistics.

Korkontzelos, I. and Manandhar, S. (2010). UoY: Graphs of Unambiguous Vertices for Word Sense Induction and Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 355–358. Association for Computational Linguistics.

Kriegel, H., Kroger, P., Sander, J., and Zimek, A. (2011). Density-Based Clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240.

Kumpula, J., Saramäki, J., Kaski, K., and Kertész, J. (2007). Limited Resolution in Complex Network Community Detection with Potts Model Approach. *The European Physical Journal B-Condensed Matter and Complex Systems*, 56(1):41–45.

Landauer, T. and Dumais, S. (1997). A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211.

Landes, S., Leacock, C., and Tengi, R. (1998). *WordNet: An Electronic Lexical Database*, chapter Building Semantic Concordances, pages 199–216. MIT Press, Cambridge, MA.

Lau, J. H., Cook, P., McCarthy, D., Newman, D., and Baldwin, T. (2012). Word Sense Induction for Novel Sense Detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 591–601. Association for Computational Linguistics.

Lesk, A. (2008). *Introduction to Bioinformatics*. OUP Oxford.

Lin, D. (1998). An Information-Theoretic Definition of Similarity. In *Proceedings of the 15th International Conference on Machine Learning*, volume 1, pages 296–304. San Francisco.

Lin, D. and Pantel, P. (2002). Concept Discovery from Text. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, pages 1–7. Association for Computational Linguistics.

Liu, B. and Zhang, L. (2012). A Survey of Opinion Mining and Sentiment Analysis. In *Mining Text Data*, pages 415–463. Springer.

Lloyd, S. (1982). Least Squares Quantization in PCM. *Information Theory, IEEE Transactions on*, 28(2):129–137.

Lopez-Fernandez, L., Robles, G., and Gonzalez-Barahona, J. (2004). Applying Social Network Analysis to the Information in CVS Repositories. In *IEE Seminar Digests*, volume 101.

Lubovac, Z., Gamalielsson, J., and Olsson, B. (2006). Combining Functional and Topological Properties to Identify Core Modules in Protein Interaction Networks. *Proteins*, 64(4):948–959.

Luce, R. and Perry, A. (1949). A Method of Matrix Analysis of Group Structure. *Psychometrika*, 14(2):95–116.

MacQueen, J. et al. (1967). Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, page 14. California, USA.

Mäkelä, E. (2005). Survey of Semantic Search Research. In *Proceedings of the Seminar on Knowledge Management on the Semantic Web*.

Manandhar, S., Klapaftis, I. P., Dligach, D., and Pradhan, S. S. (2010). SemEval-2010 Task 14: Word Sense Induction and Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68. Association for Computational Linguistics. Uppsala, Sweden.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Manning, C. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press.

Mazurie, A., Bonchev, D., Schwikowski, B., and Buck, G. (2010). Evolution of Metabolic Network Organization. *BMC Systems Biology*, 4(1).

McCarthy, D., Keller, B., and Navigli, R. (2010). Getting Synonym Candidates from Raw Data in the English Lexical Substitution Task. In *Proceedings of the 14th Euralex International Congress*.

McCarthy, D., Koeling, R., Weeds, J., and Carroll, J. (2007). Unsupervised Acquisition of Predominant Word Senses. *Computational Linguistics*, 33(4):553–590.

McCarthy, D. and Navigli, R. (2007). SemEval-2007 Task 10: English Lexical Substitution Task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 48–53. Association for Computational Linguistics.

McNamee, P. (2005). Language Identification: A Solved Problem Suitable for Undergraduate Instruction. *Journal of Computing Sciences in Colleges*, 20(3):94–101.

Medelyan, O. (2007). Computing Lexical Chains with Graph Clustering. In *Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop*, pages 85–90. Association for Computational Linguistics.

Meila, M. (2007). Comparing Clusterings – an Information Based Distance. *Journal of Multivariate Analysis*, 98(5):873–895.

Mihalcea, R., Chklovski, T., and Kilgarriff, A. (2004a). The Senseval-3 English Lexical Sample Task. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28. Barcelona, Spain, Association for Computational Linguistics.

Mihalcea, R. and Radev, D. (2011). *Graph-Based Natural Language Processing and Information Retrieval*. Cambridge University Press.

Mihalcea, R., Tarau, P., and Figa, E. (2004b). PageRank on Semantic Networks, with Application to Word Sense Disambiguation. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Milgram, S. (1967). The Small World Problem. *Psychology Today*, 2(1):60–67.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235.

Miner, G., Elder, J., Hill, T., Nisbet, R., Delen, D., and Fast, A. (2012). *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*. Elsevier Science.

Minoiu, C. and Reyes, J. (2011). A Network Analysis of Global Banking: 1978-2009. *IMF Working Paper; 238*, 11:74.

Mitkov, R., editor (2003). *The Oxford Handbook of Computational Linguistics*. Oxford Handbooks in Linguistics. Oxford Uuniversity Press.

Mitzlaff, F., Atzmueller, M., Stumme, G., and Hotho, A. (2013). Semantics of User Interaction in Social Media. In *Complex Networks IV*, pages 13–25. Springer.

Murtagh, F. and Contreras, P. (2011). Methods of Hierarchical Clustering. *CoRR*.

Nascimento, M. C. and de Carvalho, A. C. (2011). Spectral Methods for Graph Clustering - A Survey. *European Journal of Operational Research*, 211(2):221–23.

Nasiruddin, M. (2013). State of the Art of Word Sense Induction: A Way Towards Word Sense Disambiguation for Under-Resourced Languages. *arXiv preprint arXiv:1310.1425*.

Navigli, R. (2009). Word Sense Disambiguation: A Survey. *ACM Computing Surveys (CSUR)*, 41(2):10.

Navigli, R. (2012). A Quick Tour of Word Sense Disambiguation, Induction and Related Approaches. In *SOFSEM 2012: Theory and Practice of Computer Science*, volume 7147 of *Lecture Notes in Computer Science*, pages 115–129. Springer Berlin / Heidelberg.

Navigli, R. and Crisafulli, G. (2010). Inducing Word Senses to Improve Web Search Result Clustering. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 116–126. Association for Computational Linguistics.

Navigli, R. and Lapata, M. (2007). Graph Connectivity Measures for Unsupervised Word Sense Disambiguation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1683–1688.

Navigli, R. and Lapata, M. (2010). An Experimental Study of Graph Connectivity for Unsupervised Word Sense Disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):678–692.

Navigli, R. and Vannella, D. (2013). SemEval-2013 Task 11: Evaluating Word Sense Induction & Disambiguation within an End-User Application. In *Proceedings of the* 7^{th} *International Workshop on Semantic Evaluation (SemEval 2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)*, pages 193–201, Atlanta, USA.

Nešetřil, J. and de Mendez, P. (2012). *Sparsity: Graphs, Structures, and Algorithms*. Algorithms and Combinatorics. Springer.

Newman, M. (2006). Modularity and Community Structure in Networks. *Proceedings of the National Academy of Sciences*, 103(23).

Newman, M., Barabási, A.-L., and Watts, D. J. (2006). *The Structure and Dynamics of Networks*. Princeton University Press.

Oliveira, H. G. and Gomes, P. (2011). Automatic Discovery of Fuzzy Synsets from Dictionary Definitions. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Three*, pages 1801–1806. AAAI Press.

Onnela, J., Saramäki, J., Kertész, J., and Kaski, K. (2005). Intensity and Coherence of Motifs in Weighted Complex Networks. *Physical Review E*, 71(6).

Opsahl, T. (2009). *Structure and Evolution of Weighted Networks*. PhD thesis, Queen Mary College, University of London.

Opsahl, T. and Panzarasa, P. (2009). Clustering in Weighted Networks. *Social networks*, 31(2):155–163.

Pantel, P. and Lin, D. (2002). Discovering Word Senses from Text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 613–619. ACM.

Parimala, M., Lopez, D., and Senthilkumar, N. (2011). A Survey on Density-Based Clustering Algorithms for Mining Large Spatial Databases. *International Journal of Advanced Science and Technology*, 31:59–66.

Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302):157–175.

Pedersen, T. (2010). Duluth-WSI: SenseClusters Applied to the Sense Induction Task of SemEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 363–366. Association for Computational Linguistics.

Pedersen, T. and Bruce, R. (1997). Distinguishing Word Senses in Untagged Text. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, volume 2, pages 197–207.

Pedersen, T. and Kulkarni, A. (2006). Automatic Cluster Stopping with Criterion Functions and the Gap Statistic. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 276–279. Association for Computational Linguistics.

Pedersen, T., Patwardhan, S., and Michelizzi, J. (2004). WordNet:: Similarity: Measuring the Relatedness of Concepts. In *Demonstration Papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics.

Peirce, C. S. (1903). *A Syllabus of Certain Topics of Logic.* Alfred Mudge & Son.

Pivovarov, G. and Trunov, S. (2011). Clustering and Classification in Text Collections Using Graph Modularity. *Arxiv preprint arXiv:1105.5789, submitted to Journal of Machine Learning Research.*

Plate, T. (2003). *Holographic Reduced Representation: Distributed Representation for Cognitive Structures.* CSLI Lecture Notes. CSLI Publications.

Pustejovsky, J. (1991). The Generative Lexicon. *Computational Linguistics*, 17(4):409–441.

Read, J., Hope, D., and Carroll, J. (2007). Annotating Expressions of Appraisal in English. In *Proceedings of the Linguistic Annotation Workshop*, pages 93–100. Association for Computational Linguistics.

Recasens, M. and Hovy, E. (2011). BLANC: Implementing the Rand Index for Coreference Evaluation. *Natural Language Engineering*, 1(1):1–26.

Reichart, R. and Rappoport, A. (2009). The NVI Clustering Evaluation Measure. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 165–173. Association for Computational Linguistics.

Ritter, A., Cherry, C., and Dolan, B. (2010). Unsupervised Modeling of Twitter Conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 172–180. Association for Computational Linguistics.

Rosenberg, A. and Hirschberg, J. (2007). V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420.

Russell, B. (1905). On Denoting. *Mind*, 14(56):479–493.

Sahlgren, M. (2008). The Distributional Hypothesis. *Italian Journal of Linguistics*, 20(1):33–54.

Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill Computer Science Series. McGraw-Hill.

Saramäki, J., Kivelä, M., Onnela, J., Kaski, K., and Kertész, J. (2007). Generalizations of the Clustering Coefficient to Weighted Complex Networks. *Physical Review E*, 75(2):027105.

Schaeffer, S. E. (2007). Graph Clustering. *Computer Science Review*, 1(1):27–64.

Schiavo, S., Reyes, J., and Fagiolo, G. (2010). International Trade and Financial Integration: a Weighted Network Analysis. *Quantitative Finance*, 10(4):389–399.

Schütze, H. (1992). Context Space. In *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pages 113–120.

Schütze, H. (1998). Automatic Word Sense Discrimination. *Computational Linguistics*, 24:97–123.

Senellart, P. and Blondel, V. D. (2008). Automatic Discovery of Similar Words. In *Survey of Text Mining II*, pages 25–44. Springer.

Shakespeare, W. (1600). *Much Ado About Nothing*. Andrew Wise and William Aspley.

Shannon, C. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(379):379–423, 623–656.

Sinclair, J. (2004). *Trust the Text: Language, Corpus and Discourse*. Routledge.

Sinha, R. and Mihalcea, R. (2007). Unsupervised Graph-Based Word Sense Disambiguation Using Measures of Word Semantic Similarity. In *Semantic Computing, 2007. ICSC 2007*, pages 363–369. IEEE.

Sinha, R. and Mihalcea, R. (2011). Using Centrality Algorithms on Directed Graphs for Synonym Expansion. In *Twenty-Fourth International FLAIRS Conference*.

Snyder, B. and Palmer, M. (2004). The English All-Words Task. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43.

Sowa, J. (1976). Conceptual Graphs for a Data Base Interface. *IBM Journal of Research and Development*, 20(4):336–357.

Sowa, J. (1992). *Encyclopedia of Cognitive Science*, chapter Semantic Networks. Wiley Online Library.

Squartini, T., Fagiolo, G., and Garlaschelli, D. (2011). Rewiring World Trade. Part II: A Weighted Network Analysis.

Steinhaus, H. (1956). Sur la Division des Corp Materiels en Parties. *Bull. Acad. Polon. Sci*, 1:801–804.

Steyvers, M. and Tenenbaum, J. (2005). The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. *Cognitive Science: a Multidisciplinary Journal*, 29(1):41–78.

Stigler, S. (1989). Francis Galton's Account of the Invention of Correlation. *Statistical Science*, 4(2):73–79.

Stingl, K., Kullmann, S., Guthoff, M., Heni, M., Fritsche, A., and Preissl, H. (2010). Insulin Modulation of Magnetoencephalographic Resting State Dynamics in Lean and Obese Subjects. *Frontiers in Systems Neuroscience*, 4.

Stix, V. (2004). Finding all maximal cliques in dynamic graphs. *Computational Optimization and Applications*, 27(2):173–186.

Strogatz, S. (2004). *Sync: The Emerging Science of Spontaneous Order*. Penguin Press Science Series. Penguin.

Tan, P.-N., Steinbach, M., and Kumar, V. (2006). *Introduction to Data Mining*. Pearson Addison Wesley.

Tejada-Cárcamo, J., Calvo, H., Gelbukh, A., and Hara, K. (2010). Unsupervised WSD by Finding the Predominant Sense using Context as a Dynamic Thesaurus. *Journal of Computer Science and Technology*, 25(5):1030–1039.

Thomas, D. (1954). *Under Milk Wood: A Play for Voices*. New Directions Paperbook. New Directions.

Tijms, H. (2012). *Understanding Probability*. Cambridge University Press.

Tomita, E., Tanaka, A., and Takahashi, H. (2006). The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363(1):28–42.

Tommy Thumb's Pretty Song Book (1744). *Tommy Thumb's Pretty Song Book*. Mary Cooper.

Tsuruoka, Y., Tateishi, Y., Kim, J., Ohta, T., McNaught, J., Ananiadou, S., and Tsujii, J. (2005). Developing a Robust Part-of-Speech Tagger for Biomedical Text. *Advances in Informatics*, pages 382–392.

Tuggy, D. (1993). Ambiguity, Polysemy, and Vagueness. *Cognitive Linguistics*, 4(3):273–290.

van Benthem, J. and Meulen, A. (2010). *Handbook of Logic and Language*. Elsevier Science.

Van de Cruys, T. (2008). Using Three Way Data for Word Sense Discrimination. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 929–936.

Van de Cruys, T. and Apidianaki, M. (2011). Latent Semantic Word Sense Induction and Disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL/HLT)*, pages 1476–1485.

van den Heuvel, M., Mandl, R., Stam, C., Kahn, R., and Hulshoff Pol, H. (2010). Aberrant Frontal and Temporal Complex Network Structure in Schizophrenia: a Graph Theoretical Analysis. *The Journal of Neuroscience*, 30(47).

van Dongen, S. (2000). A Cluster Algorithm for Graphs. *Report-Information Systems*, (10):1–40.

Veenman, C. J., Reinders, M. J. T., and Backer, E. (2002). A Maximum Variance Cluster Algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(9):1273–1280.

Velldal, E. (2003). *Modeling Word Senses with Fuzzy Clustering*. Cand.Philol. Thesis.

Velldal, E. (2005). A Fuzzy Clustering Approach to Word Sense Discrimination. In *Proceedings of the 7th International Conference on Terminology and Knowledge Engineering*.

Véronis, J. (2004). Hyperlex: Lexical Cartography for Information Retrieval. *Computer Speech & Language*, 18(3):223–252.

Vickrey, D., Biewald, L., Teyssier, M., and Koller, D. (2005). Word-Sense Disambiguation for Machine Translation. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 771–778. Association for Computational Linguistics.

Wang, L., Yu, C., Chen, H., Qin, W., He, Y., Fan, F., Zhang, Y., Wang, M., Li, K., Zang, Y., et al. (2010). Dynamic Functional Reorganization of the Motor Execution Network After Stroke. *Brain*, 133(4).

Wasserman, S. and Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Cambridge University Press.

Watts, D. and Strogatz, S. (1998). Collective Dynamics of 'Small-World' Networks. *Nature*, 393(6684):440–442.

Weaver, W. (1955). Translation. *Machine Translation of Languages: Fourteen Essays*, 14:15–23.

Weeds, J. (2003). *Measures and Applications of Lexical Distributional Similarity*. PhD thesis, University of Sussex.

Widdows, D. (2003). Unsupervised Methods for Developing Taxonomies by Combining Syntactic and Statistical Information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 197–204. Association for Computational Linguistics.

Widdows, D. (2004). *Geometry and Meaning*. CSLI Lecture Notes. CSLI Publications, Center for the Study of Language and Information.

Widdows, D. and Dorow, B. (2002). A Graph Model for Unsupervised Lexical Acquisition. In *Proceedings of the 19th International Conference on Computational Linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Wittgenstein, L. (1953). *Philosophical Investigations*. Blackwell.

Yarowsky, D. (1995). Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Yuret, D. (1998). *Discovery of Linguistic Relations Using Lexical Attraction*. PhD thesis, Massachusetts Institute of Technology.

Zadeh, L. (1965). Fuzzy Sets. *Information and Control*, 8(3):338–353.

Zahn, C. T. (1971). Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters. *Computers, IEEE Transactions on*, 100(1):68–86.

Zhang, B. and Horvath, S. (2005). A General Framework for Weighted Gene Co-Expression Network Analysis. *Statistical Applications in Genetics and Molecular Biology*, 4(1):17.

Zhang, Z. and Sun, L. (2011). Improving Word Sense Induction by Exploiting Semantic Relevance. *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 1387–1391.