Graph-based Algorithms for Lexical Semantics and its Applications

Wei Wu

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2012

Reading Committee:

Mari Ostendorf, Chair

Maya Gupta

Ye-Yi Wang

Luke Zettlemoyer

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Graph-based Algorithms for Lexical Semantics and its Applications

Wei Wu

Chair of the Supervisory Committee:
Professor Mari Ostendorf
Electrical Engineering

Lexical semantics studies the meaning of words, which is a useful tool for computer-based automatic natural language processing (NLP). This thesis explores graph-based algorithms to learn and apply distributional lexical semantics in NLP applications. One theory of lexical semanticists holds that semantic relations among words can be extracted from their textual context in natural languages. Based on this theory, we propose using graphs to represent natural language text according to the contextual relations of words in higher-level language units (e.g. sentences, definitions or documents). In these graphs, words and/or higher-level language units are represented with nodes, and edges are added between them according to their textual context to indicate their observed relatedness in a dictionary or a collection of documents. We explore two types of graph representations: the word-word graph which is used for modeling the semantic relations among words, and the instance-word bipartite graph which uses words as a medium to study the relatedness among higher-level language units. In this way, we can embed the semantic relations among words and optionally higher-level language units into the graph structure. We design algorithms to propagate semantic information through the graphs in order to recover the unobserved relatedness among words or higher-level language units, which is used in designing unsupervised, semi-supervised or active learning algorithms to reduce human supervision in NLP applications for harvesting or analyzing text data resources. Specifically, we design graph-based algorithms either for quantitatively assessing lexical semantic similarity, or for developing representativeness and

diversity criteria for selecting a characteristic subset of terms, which is useful for problems such as keyword summarization and query design for active learning. In particular, we focus on designing graph-based algorithms to apply lexical semantics in three NLP applications, including Wiktionary lexical semantic similarity extraction, Twitter user interest extraction, and active learning for semantic orientation classification.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor Professor Mari Ostendorf, for her encouragement and guidance in my study. She is an exceptional advisor and mentor, and it has been a privilege to work with her in the past five years. I also would like to thank the other members of my supervisory committee: Maya Gupta, Luke Zettlemoyer, and Ye-Yi Wang, for their valuable suggestions to my research and this dissertation.

I also want to thank our collaborators at SRI in the DARPA GALE project: Wen Wang, Xin Lei, Arindam Mandal and Andreas Stolcke, especially to Wen Wang for being my mentor when I did my first internship at SRI. I must thank my intern mentors at Microsoft Research: Y.C. Ju and Xiao Li, for the numerous inspiring discussions, and thank Ye-Yi Wang again for his guidance during my internship at Microsft Bing, which leads me to pursue my future career in the industry. And thank Munmun De Choudhury from Arizona State University for sharing the Twitter data.

There are many people in the SSLI lab I would like to thank for their help and support. I must thank Bin Zhang, Alex Marin, and Anna Margolis for kindly offering the help to annotate the Twitter data, thank Julie Medero for sharing the code of processing the Wiktionary data, and thank Lois Kim for the great job of labeling the implicit alignment data. I also want to thank Brian Hutchinson, Amittai Axelrod, Nicole Nichols for many useful discussions on my work. And many thanks to the fantastic dim sum and great vegan food.

Finally, I must give my deepest gratitude to my parents for all their support and encouragement throughout the past five years. As a Chinese poet put it, the love of the parents to their children can never be reciprocated.

Chapter 1

INTRODUCTION

Thanks to the rapid development of the internet and the online service industry, especially the emergence of web 2.0, there is a huge amount of user-generated text data on the internet, ranging from blogs, tweets, user reviews to wiki websites. Using computer-based automatic natural language processing (NLP) techniques to analyze these data can be very helpful for various commercial applications, such as improving information retrieval services, evaluating consumer feedback, or discovering user interests and preferences for targeted services or advertising. Lexical semantics, which studies the meaning of words [60], is a useful tool in related NLP techniques for these applications. Different from the techniques that work with the lexical form of words, lexical semantics looks beyond the "surface" of a word and explores the concepts and other semantic characteristics behind it, and thus enables deeper analysis of natural language text. Currently, lexical semantics is being applied in a wide range of NLP applications, such as paraphrase identification and generation [19], document summarization [62, 73], query expansion [118, 34, 61], information extraction [88, 16], and opinion mining [36].

The huge amount of text data on the internet poses both opportunities and challenges for applying lexical semantics in NLP tasks. On one hand, wiki websites such as wikipedia and wiktionary that are collaboratively edited by internet volunteers provide new resources to extract lexical semantic information, which enjoy larger coverage and more instant updates than traditional expert-edited linguistic resources. On the other hand, analyzing the huge amount of data demands efficient algorithms and requires minimizing human labor needed to annotate the data.

To tackle these challenges, this work focuses on the study of graph-based algorithms in a data driven approach to use lexical semantics for NLP applications. We propose to use graphs to represent natural language corpora and design graph-based algorithms to

extract, summarize or query semantic information in natural language text for specific NLP applications. The graph representations enable us to explore the relatedness among words and higher-level language units (sentences or documents), which is convenient to develop unsupervised, semi-supervised or active learning algorithms that can help to reduce human supervision in NLP applications. In addition, the graph-based algorithms explored in this work require relatively simple computation and can be extended to parallel computation for scaling up on huge data sets.

In the proposed graph representation framework, words and/or higher-level language units are represented with nodes, and edges are added between them according to their textual context to indicate their observed relatedness. We view each node in the graph as both a source and a receiver of semantic information, and design algorithms to propagate semantic information through the graph to recover the unobserved relatedness between words or higher-level language units. This relatedness is either used for quantitatively extracting semantic similarity between words, or for developing representativeness and diversity criteria for selecting a characteristic subset of terms, which is useful for problems such as keyword summarization and query design for active learning.

This work focuses the study of lexical semantics and graph algorithms on three NLP applications, including Wiktionary semantic similarity extraction, Twitter user interest summarization, and active learning for semantic orientation classification.

1.1 Semantic Concepts and Semantic Orientation

In lexical semantics, the meaning of words covers many loosely defined "dimensions" in linguistics [49, 60], this dissertation focuses on two of them that are closely related to natural language processing applications: *semantic concepts* and *semantic orientation*. Lexical semantic concepts refer to the cognitive concepts a word expresses, for instance "slim", "thin", and "skinny" all express the concept of "not well fleshed." In linguistics, semantic concepts can be characterized in terms of semantic fields, which is defined as a group of words closely related in meaning [49]. In the above example, "skinny", "thin", and "slim" can be grouped into the same semantic field. The semantic conceptual overlap of two words can be evaluated by semantic similarity, which have important applications in paraphrase

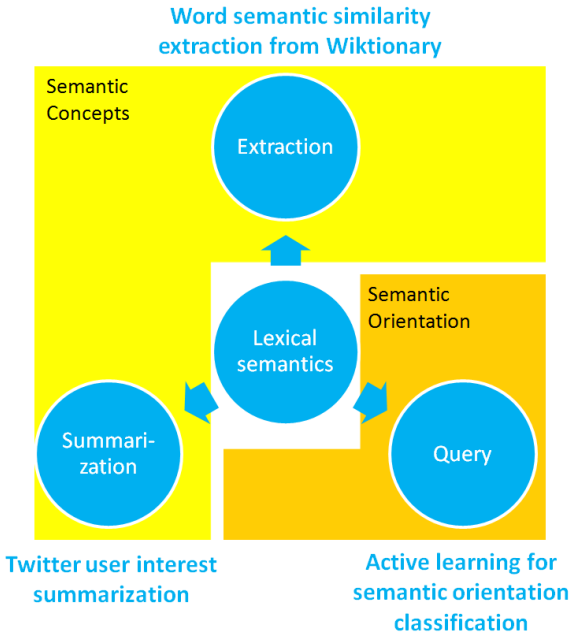


Figure 1.1: Three natural language processing applications to be studied in this dissertation

identification, generation, and text summarization. On the other hand, lexical semantic orientation refers to the sentiment associated with the words, which indicates the direction the word deviates from its semantic field [36] and constrains the word's usage in language due to its evaluative characteristics [7]. Taking the above example, "thin" is a neutral description, but "slim" has a positive sentiment associated with it, while "skinny" usually contains a negative sentiment. Although the three words belong to the same semantic field, and are nearly synonymous, yet they range from positive to negative in the semantic orientation dimension. Lexical semantic orientation is important for text sentiment analysis and opinion mining.

For the three NLP applications explored in this work, Wiktionary semantic similarity extraction and Twitter user interest summarization are in the domain of lexical semantic concepts, and active learning for semantic orientation classification is focused on the lexical semantic orientation (Figure 1.1).

1.2 Main Contributions

The goal of this work is to develop new models for graph-based algorithms to study and apply lexical semantics in NLP applications, including Wiktionary semantic similarity extraction, Twitter user interest summarization, and active learning for semantic orientation classification.

Cruse claims that semantic relations among words can be extracted from the context in natural languages [20]. Based on this theory, we apply traditional graph representations of a dictionary and text corpora to natural language text for NLP applications:

- For lexical semantic similarity extraction, we propose two homogenous graph representations of Wiktionary (directed and undirected WikNet) based on its *word sense definitions*. We also prove that the graph properties of both types of WikNet conform with that of a 'small world network' [121].

- For Twitter user interest summarization, we introduce a homogenous graph representation of user's *tweets* based on a combination of words/phrases semantic similarity and contextual co-occurrence.

- For active learning for semantic orientation classification, we use a bipartite graph to represent the relation between words in *product and movie reviews*, leveraging words as a medium for propagating semantic orientation information among the reviews.

We propose using graph algorithms to model the way semantic information is propagated through words and higher-level language units in natural language text to recover the unobserved relatedness between them for extracting, summarizing or querying semantic information. In particular, the recovered relatedness is used in NLP applications either for extracting semantic similarity, or for integrating representativeness and diversity criteria in summarization results or active learning queries.

For Wiktionary lexical semantic similarity extraction, we extend algorithms for social network link predictions to evaluate lexical semantic similarity in WikNet. We also develop a new semantic similarity measure from WikNet based on the traditional Jaccard

coefficient, which has the advantages of being able to model the semantic information propagated between mutually linked nodes in WikNet and incorporate synonym information in Wiktionary. The effectiveness of the new WikNet-based measure is demonstrated on three human-rated lexical semantic similarity datasets and two NLP tasks, including paraphrase detection and forum alignment identification.

For Twitter user interest summarization and active learning for semantic orientation classification, we introduce graph-based algorithms that combine random walk algorithms [12, 131] and graph-cut algorithms [55] for summarization or active learning, which have the advantage of taking both the representativeness and diversity criteria into account in the summarization results or active learning queries.

For Twitter user interest summarization, we develop a summarization system that combines PageRank [12] and maximum graph-cut [55] algorithms to extract key words or phrases from the user's tweets to summarize his/her interests. This system achieves high precision on the summarization results.

For active learning for semantic orientation classification, we propose a graph-based active learning framework that is convenient for both query design and graph-based semi-supervised learning. Based on this framework, we introduce two new active learning query strategies that outperforms the standard active learning baseline on the semantic orientation classification task.

1.3 Dissertation Outline

The remainder of this dissertation is organized as follows:

Chapter 2 gives a brief introduction to the lexical semantic theory which establishes the context of this work and provides us the foundation of our proposed graph representation for natural languages. We also give a brief overview of graph-based algorithms in the literature, and discuss their advantages and limitations in applications.

Chapter 3 presents our work on extracting lexical semantic similarity from Wiktionary. It begins with a background introduction of related work on lexical semantic similarity extraction. We then introduce the proposed graph representation for Wiktionary (WikNet) and the study of its graphic properties. Then we propose a series of WikNet-based lexical

semantic similarity measures. We compare their performance together with WordNet-based measures on human rated lexical semantic similarity datasets, followed by further comparisons in the paraphrase detection and alignment identification tasks.

Chapter 4 presents our work on Twitter user interest summarization. We first give a brief background introduction, including a literature review of keyword extraction work on general text genres and related work on natural language processing over Twitter corpora. Then we present the architecture of our Twitter user interest summarization system, together with the proposed graph representation and summarizing algorithms, followed by experimental results.

Chapter 5 presents our work on graph-based active learning framework for semantic orientation classification. It begins with a background introduction of related works in active learning and semantic orientation classification. Then we introduce the proposed active learning framework based on a bipartite graph representation, followed by the presentation of two new graph-based active learning query strategies. We give the experimental results of the proposed graph-based query strategies comparing with other commonly used active learning baseline strategies on the movie and product review corpora.

Chapter 6 summarizes our work in this dissertation and discusses directions of future work.

Chapter 2

BACKGROUND

In this chapter, we first give a brief overview of some of the necessary background models of lexical semantics, followed by a brief introduction to the motivation of our proposed graph-representations in NLP applications. Then an literature review of graph-based algorithms is presented, together with a discussions of their advantages and limitations. Finally, we introduce some practical issues that need to be accounted for in NLP applications. Introduction to prior work in task-specific computational methods of lexical semantics will be given in later chapters on applications of lexical semantics for the respective NLP tasks.

2.1 Lexical Semantics

2.1.1 Overview of Studies in Lexical Semantic Theory

As defined by Pustejovsky, lexical semantics "is a study of how and what the words of a language denote" [90]. Specifically, it covers the classification, decomposition and representation of the meaning of words, as well as the relation between word meaning and the meaning and structure of higher level language units in natural languages, such as sentence meaning and syntax [40]. It not only touches the structure and function of languages, but also involves psychology, philosophy and cognitive science. This work will focus on two aspects of lexical semantics: the semantic sense and semantic sentiment. We will study graph representations and algorithms for applying lexical semantics in computer-based automatic NLP applications. Though we will not work on the linguistic theory side of lexical semantics, here we present a brief overview of some of the important theoretical frameworks of lexical semantics in linguistics to give the readers a complete picture of lexical semantics.

One focus of lexical semantics is to explore the nature of the meaning of words. Cruse [20] and other researchers in this area believe that the meaning of words can be expressed by other words, thus studies on the relationship between words becomes one of the central parts

of lexical semantics. One way to study the relationship between words is through *lexical relations* in a language, which is defined as patterns of association between words, such as synonymy, antonyms, hyponymy and hypernymy. Lexical relations can be derived from four basic congruence relations, including identity, inclusion, overlap and disjunction. Structural semantics studies the relationship between words with the theory of semantic fields [60]. Lehrer [49] defines semantic field to be a group of words closely related in meaning, often subsumed under a general term. Studies in this area explore properties of semantic fields, and use them to suggest lexical strategies for selecting words, stretching word meanings, and interpreting unusual sentences. Researchers of structural semantics also believe the vocabulary of a language can be structured hierarchically in terms of hyponymy under several different roots [60]. One further step from this is to organize the vocabulary of a language into a taxonomy (an acyclic directed graph) based on the hyponymy and synonymy relations between words [20]. The meaning of a word can thus be viewed as being expressed through its relations with other words in the taxonomy. A typical example is WordNet [29], a widely used lexical semantic database edited by linguistic experts. In WordNet, nouns, verbs, adjectives and adverbs are grouped into sets of synonyms (synsets), each expressing a distinct concept, and synsets are interlinked by means of conceptual-semantic and lexical relations. Our work on Wiktionary semantic similarity extraction is related to the work on WordNet, but instead of using the expert-edited taxonomy, we extract lexical semantic information from the crowd-edited Wiktionary sense definitions.

Lexical semantics also studies the relationship between words and other lexical units in languages. For instance, Everaert [40] studies how idioms are lexically represented in a generative theory. Stubbs [106] explores the meaning of words within the context of phrases and the way words are used in predictable combinations in phrases. Our work does not study the relationship between words and other lexical units in linguistic theories, but we will use the contextual relation of words and higher-level language units (e.g. sentences or documents) to design the graph representations of natural language text.

Another important subfield of lexical semantics is focused on the relationship between lexical semantics and syntax. Grimshaw and Williams [89] explore the interaction between nominalization (the use of a verb, an adjective, or an adverb as the head of a noun phrase)

and the character of PP complements to nominals. Ingria and George [89] examine the mapping from surface syntactic constituents to semantic arguments. Dalrymple et.al. [22] work on using a deductive approach to the syntax-semantics interface by integrating logical deduction for semantic composition and the lexical functional grammar for analyzing linguistic structures. Our work does not consider the relation of syntax and semantics, focusing instead on extracting or analyzing lexical sense and sentiment.

Computational research in semantics [3, 45] may involve combining the semantic information of words to determine the meaning of a sentence or larger language unit. Alternatively, it may involve shallow word-level analysis to determine the sense or role of individual words for applications such as translation, paraphrasing or sentiment classification. The emphasis of this thesis is on the latter, where we focus on paraphrasing and sentiment classification applications. There are different approaches to representing lexical semantics, including use of word context statistics in general or in terms of a "frame" that includes human specified categories (e.g., arguments a verb can do). Our work concentrates on the general context statistics.

2.1.2 Lexical Semantics and Vector Representation

A widely used computational strategy for lexical semantics is through vector representation. This strategy uses a data-driven approach to extract the co-occurrence statistics from the natural language text and represent them with a vector. One approach extracts vector representation of lexical semantics from word distribution in a set of documents. For instance, latent semantic analysis [28] starts with a word-document matrix which describes word occurrence in documents, and applies singular value decomposition to find its low-rank approximation, where the concept vector of a word is produced by extracting a linear combination of the 'concept bases' in the low-rank space. Probabilistic latent semantic analysis [37] adopts a statistical technique and uses the probabilistic distribution of a word across a set of documents as its vector representation. Another approach extracts vector representation of a word using other words that co-occurs with some window of the target word in natural language context. Context vector [99] (a.k.a. the first order context vector)

constructs a word's vector representation by accumulating the co-occurrence frequency of a set of pivot words with it in context windows through a text corpus. The second order context vector [85] is also proposed by accumulating the first order context vectors according to the word gloss in a dictionary.

Vector representation of lexical semantics is not a focus of this work, but it is similar to our approach in the use of co-occurrence statistics. Therefore some of the related algorithms will be compared with or incorporated into the graph representations of lexical semantics proposed in our work. A key difference in our approach is the representation of dependencies between words through the graph.

2.1.3 Lexical Semantics and Graph Representation

In this section, we introduce two graph structures motivated by the contextual relation model in lexical semantics. From the linguistic perspective, words are basic units in natural languages to convey semantic information, which are combined to form the semantics of higher-level language units such as sentences or documents. The way the semantic information is passed among words and higher-level language units provides the foundation of using graphs to represent natural language and study lexical semantics. According to Cruse [20], an extremely useful model of the meaning of a word is to view it as being made up, at least in part, of the meaning of other words. Cruse believes that such semantic relations among words can be extracted from their textual context in natural languages. There are two types of context we are going to explore in this work to help construct the graph representations for studying lexical semantics. One type of context is word sense definitions in dictionaries, where the meaning of a word is explained by other words in its gloss. The other type of context is the higher-level language units where the word occurs, words in the same sentence or document share common semantic information, thus the meaning of a word can be constituted by these contextual relations.

Cruse's model provides a motivation to utilize the dictionary gloss or word co-occurrence relations among higher-level language units to design the graph representation of text corpora for exploring semantic information in natural languages.

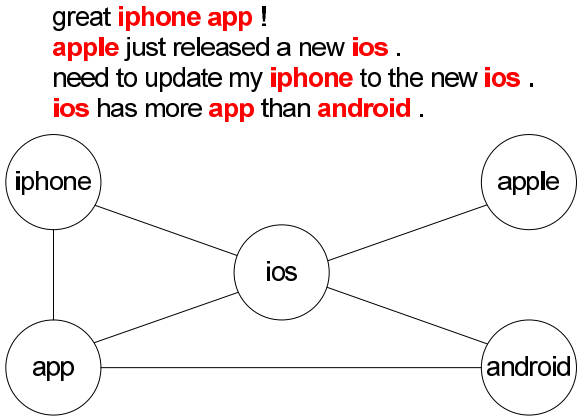


Figure 2.1: Word-word graph representation

One strategy is using a word-word graph representation, where each node represents a word, and an edge is added between two words if there is a contextual relation observed. For example, the edge could be used when one word appears in the dictionary gloss of the other, or if they co-exist in a higher-level language unit in the corpus (Figure 2.1). This graph representation provides us a way to explore the similarity or relatedness between words through their contextual relation in natural languages.

Alternatively, we can introduce higher-level units in the graph to construct a word-sentence or word-document bipartite graph representation. In this bipartite graph, the left side nodes represent sentences or documents, the right side nodes represent words, if one word appears in a sentence or document, an edge is added between them as in Figure 2.2. The bipartite graph representation enables us to use words as a medium to explore the relatedness between higher-level language units in natural languages.

Details on applying these two types of graph representations will be studied in later chapters on specific NLP tasks.

Two types of graphs, divided by the directional property of their edges, will be used in this work. One is the directed graph, in which the edges are directional pairs between two nodes: an edge (a, b) is considered to be directed from node a to node b, a is called b's predecessor, and b is said to be a's successor, as shown in Figure 2.3a. The other is

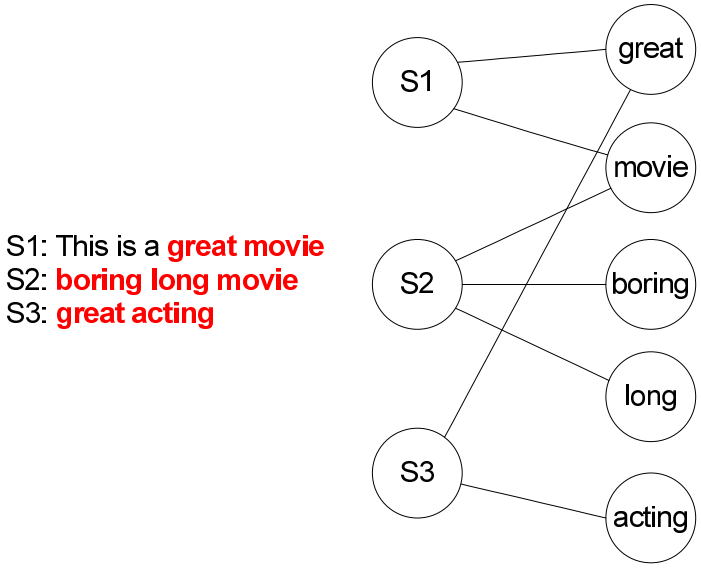
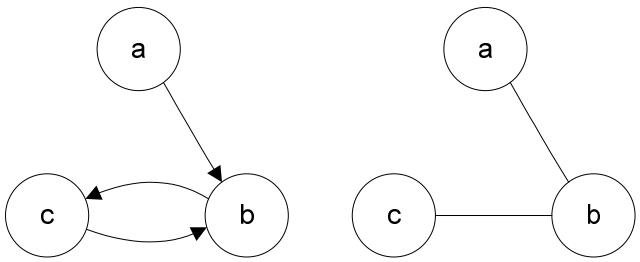


Figure 2.2: Bipartite graph representation



(a) directed graph (b) undirected graph

Figure 2.3: Two types of graphs divided by directional property of edges

the undirected graph, where the edges have no orientation. Edges in an undirected graph are unordered pairs, edge (a, b) is identical to edge (b, a), a and b are called each other's neighbor, as shown in Figure 2.3b. From the perspective of information propagation, in directed graphs, the information can be viewed as flowing along the edge directions, i.e. for edge (a, b), the information is propagated from node a to node b; there is no information being propagated from node b to node a unless there is a reverse edge (b, a) in the graph. In undirected graphs, we can view the information as being propagated along both directions of an edge.

2.2 Review of Graph Algorithms

In this section, we provide an overview of typical graph-based algorithms in the machine learning area, and give a brief analysis of these algorithms from the perspective of information propagation through the graphs, followed by a discussion of their advantages and limitations. The graph-based algorithms can generally been divided into three groups: neighborhood-based algorithms, random walk algorithms and the maximum graph-cut algorithm.

2.2.1 Neighborhood-based Algorithms

Neighborhood-based graph algorithms are typically used to assess the relation between nodes in their graphs, they are widely applied in friend recommendations for social network services [75]. Neighborhood based graph algorithms assume the characteristics of a node can be expressed through its neighbors, and thus the overlap of two nodes' neighbors can be used to evaluate the relatedness between them. A few commonly used neighborhood-based graph algorithms are described briefly here.

Common-neighbors

Common-neighbors [75] counts the number of shared neighbors between two nodes i and j to evaluate their similarity. For the undirected WikNet. It is defined as

$$sim_{CN}(i, j) = |\Gamma(i) \cap \Gamma(j)| \tag{2.1}$$

where $\Gamma(i)$ is the neighbor set of node i.

Adam-Adar

Adamic and Adar [1] proposed a metric to evaluate the similarity between two nodes within a social network by counting their common neighbors weighted by the neighbor's frequency. It can be seen as a refined version of the common-neighbors by assigning larger weights to rare neighbors, which is defined as

$$sim_{AA}(i,j) = \sum_{k \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log |freq(k)|} \tag{2.2}$$

where $\Gamma(i)$ is the neighbor set of node i, and $freq(k)$ is the degree of node k.

Jaccard Coefficient

Another extended version of common-neighbors is the Jaccard coefficient [97], which is commonly used in information retrieval. Jaccard coefficient was originally proposed to measure the similarity between two web pages. For evaluating the similarity between two nodes i and j in a graph, it counts the number of neighbors that both nodes have over the number of neighbors that either of them has,

$$sim_{JC}(i,j) = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} \tag{2.3}$$

where $\Gamma(i)$ is the neighbor set of node i.

We can view the neighborhood-based graph algorithms as performing a 1-step information propagation from a node to its close neighbors, the overlap of two nodes' neighbors are thus the overlap of the information being propagated from them. Naturally, one can extend this information propagation to n-steps.

2.2.2 Random Walk Algorithms

This group of algorithms are based on a random walk model over the graph, and they propagate rank value or class distribution through a random walk for either ranking or classification tasks. Typical algorithms in this group include PageRank and label propagation.

PageRank

PageRank [12] is originally proposed for ranking web pages in information retrieval tasks. According to PageRank, the linking relations of webpages can be represented with a directed graph, in which each web page is represented with a node, and an directed edge is added between two nodes if one links to the other. Each node i has a rank value $R_i^{(n)}$ which will be updated iteratively according to the following equation,

$$R_i^{(n)} = (1-d) * R_i^{(0)} + d \sum_{j \in \Gamma(i)} \frac{w_{ji}}{\sum_{k \in \Gamma(j)} w_{jk}} R_j^{(n-1)} \tag{2.4}$$

where w_{ji} is the edge weight between node j and i, $\Gamma(i)$ is the node i's predecessor set, and d is a damping factor that is usually set to 0.85 [12]. $R_i^{(0)}$ is the initial rank value, which is set to be 1 for all nodes. This iteration continues until convergence.

We can view the PageRank iteration as a random walk among the graph nodes: at each node j, it has has a probability of $(1-d)$ to stay in the same node, and a probability of $d\frac{w_{ji}}{\sum_{k \in \Gamma(j)} w_{jk}}$ to walk into its successor node i. When the PageRank iteration converges, the normalized rank value for each node is the probability of the random walk arriving at that node.

Label Propagation

Label propagation [136, 131] is proposed as a semi-supervised learning algorithm for classification tasks. According to this algorithm, a graph is constructed to represent the corpus. In this graph, each node represents a data sample in the corpus, weighted edges are added between nearest neighboring nodes according to a closeness measure between data samples. Each node has a label distribution for the corresponding sample, which will be updated iteratively in the label propagation process. In Zhu's paper [136] the label distribution for each node is updated as follows,

$$F_i^{(n)} = (1-\alpha) F^{(0)} + \alpha \sum_{j \in \Gamma(i)} \frac{w_{ji}}{\sum_{k \in \Gamma(j)} w_{ki}} F_j^{(n-1)} \tag{2.5}$$

where $F_i^{(n)}$ is node i's label distribution in the n-th iteration, $F_i^{(0)}$ is node i's initial label distribution, $\Gamma(i)$ is node i's neighbor set, and α is a factor selected in $(0, 1)$. The update

process continues until $F_i^{(n)}$ converges.

It can be seen that the updating function for label propagation is quite similar to that for PageRank, except that the rank value is replaced with a label distribution. We can view the label propagation process as a random walk of labels, where in each iteration the label of node i has a $(1-\alpha)$ probability to keep as its initial label, and a $\alpha\frac{w_{ji}}{\sum_{k\in\Gamma(j)} w_{ki}}$ probability to take the label from its neighbors.

Both the PageRank and label propagation can been seen as performing a ∞-step information propagation from a node to its neighbors.

2.2.3 Maximum Graph-cut Algorithms

Maximum graph-cut algorithms split one graph into two subgraphs with the maximum cut. A widely used branch of the maximum graph-cut algorithms, which will be discussed in more details in this work, selects one subgraph with a fixed size N.

$$\begin{aligned} \arg\max_{S} & \sum_{i\in S, j\notin S} w_{ij} \\ & s.t. |S| = N \end{aligned} \tag{2.6}$$

where S is the N-size subgraph to be cut. Equation 2.6 is a submodular function, and its maximization can be solved with a greedy algorithm for a near-optimum solution [54]. This process can be viewed as selecting an N-sized node set that is the most closely related to the rest of nodes in the graph.

The maximum graph-cut based algorithms can also be viewed as a 1-step information propagation process. Different from the neighborhood-based or random walk algorithms where the source of information propagation are each individual node that propagates information to all the other nodes, the source of information propagation in maximum graph-cut algorithms is a node set in the selected subgraph, which only propagates information to nodes outside this set. Its advantage is to take into account the diversity criterion in addition to representative criterion when selecting the N-sized node set. It only counts the information propagated between the selected node set and its complement set in the graph, while disregarding the information propagated among nodes in the selected set to

avoid redundancy. This advantage makes it suitable for applications such as extraction or summarization [54, 55].

To compare the advantages and limitations of the above graph-based algorithms, and discuss their suitable scenarios to apply, we summarize their properties in Table 2.1.

2.3 Some Practical Issues in NLP

In this section, we present some practical issues that need to be accounted for in NLP applications that impact graph construction, including morphological variations and stopwords.

2.3.1 Morphological Variations

In many languages, words have morphological variations in different tense, plurality, or gender, etc. For example, in English, the word "begin" has "begins", "began", "begun" as its morphological variations in different tenses. In morphology, "begin" is called the *lemma* of this group of morphologic variations, also known as the canonical form, dictionary form or citation form; "begin", "begins", "began" and "begun" are called the *surface form*s of the lemma, which are the forms that appear in natural language text. Depending on specific NLP applications, words may need to be reduced from their surface forms to the lemma in order to reduce data sparsity. One way to achieve this is to use a morphological dictionary, which is a linguistic resource that contains the correspondences between surface forms and their lemma.

2.3.2 Stopwords

Stopwords are a group of words that need to be filtered out during the preprocessing of natural language data in some NLP applications in order to improve the performance. For example, in information retrieval and text classification, stopwords should be removed before putting the text to the training or classification stage.

A stopword set generally contains function words, such as "the", "a", "is", "on", "what", etc. Depending on specific applications, it may also include the most frequent content words, such as "want" and "need". Different NLP applications generally have different sets

Table 2.1: Comparison of typical graph-based algorithms

Algorithm		Information Propagation		Application	Pros	Cons	Suitable Scenarios
Group	Name	Sources	Steps				
Neighborhood	Common-neighbors	individual nodes	n-step	assessing similarity	simple in computation	only consider n steps	large graphs
	Adar-Adams						
	Jaccard coefficient						
Random walk	PageRank	individual nodes	∞-step	ranking	better performance	costly in computation	small or medium graphs
	Label propagation			classification			
Graph-cut	Maximum graph-cut	node set	1-step	extraction or summarization	accounts for redundancy	only consider one step	requiring to handle redundancy

of stopwords tailored for the specific tasks.

Chapter 3

WIKTIONARY LEXICAL SEMANTIC SIMILARITY EXTRACTION

In this chapter, we use a collaboratively edited on-line resource, Wiktionary (`http://www.wiktionary.org/`) as a source of lexical context to develop a new lexical semantic similarity measure. Lexical semantic similarity is a measure for evaluating the likeliness of the meaning (semantic concepts) of two given words, which can be interpreted as the question "how similar is word A to word B". Semantic similarity has many applications in natural language processing. For example, it can be employed in textual entailment, paraphrase identification for multi-document summarization, and paraphrase generation for text simplification. We propose graph representations of Wiktionary, and develop algorithms to extract lexical semantic similarity from the graph structure, and study their applications in two NLP tasks: paraphrase identification in news articles, and alignment identification in web forum discussions.

In Wiktionary, or any other dictionary, each word is explained by other words in the definitions for each of its senses, which we shall refer to as "word sense definitions." The referring relationship among words in sense definitions indicate a closeness in semantic concepts organized in an underlying graph: if we treat each word as a node, and add an edge between two words if one appears in any of the sense definitions of the other, then we can build a word closeness graph from Wiktionary, which is named *WikNet* in this thesis.[1] In other words, the semantic concepts of a word can be defined by the graphic structure of WikNet.

In social network studies, a widely accepted belief is that a person can be defined by his/her relationship with other people, similarity measures that estimate the closeness of two people can therefore be derived from the graph structure of social networks by propagating

[1]We use word nodes rather than sense nodes in order to reduce the size of the graph and because our applications do not require word sense disambiguation.

"friendship" through the social network. The analogy between WikNet and social networks inspires us to view each word in WikNet as a "person", and edges among words as "friend relationships". We can treat WikNet as a "social network" for words, since we show that it is a "small word graph" – a key property shared by social networks. Algorithms for evaluating the closeness of two people in social networks [80] can be borrowed to evaluate the semantic similarity of two words in WikNet. Instead of propagating "friendships", we use these algorithms to propagate semantic concept information through WikNet to assess the semantic similarity between words. These algorithms can also be enhanced by additional lexicon information in Wiktionary, such as the synonym lists. The main contributions of this chapter include: 1) we develop a new lexical semantic similarity measure that integrates an n-step semantic information propagation with Wiktionary synonym information; and 2) we demonstrate the proposed similarity measure's competitiveness on human-rated lexical semantic similarity datasets and two NLP applications.

The rest of this chapter is organized as follows: in Section 2, we give a brief introduction to related work in lexical semantic similarity extraction. In Section 3, we provide details on WikNet construction and give an analysis of its graphical properties. Section 4 introduces a series of WikNet-based lexical semantic similarity measures. Section 5 presents the experimental results and analysis of evaluating the proposed measures on human-rated lexical semantic similarity datasets. In Section 6 we apply the proposed lexical semantic similarity measures on a paraphrase identification task, and a forum discussion implicit alignment identification task. Finally, Section 7 summarizes the main findings and potential extensions.

3.1 Background

A related concept to semantic similarity is semantic relatedness, which is a measure of the relatedness of words, and thus is a broader concept than semantic similarity. It can be interpreted as "how much does word A have to do with word B". For instance, "bird" and "chicken", "car" and "vehicle" are semantically similar; while word pairs as "chicken" and "egg", "car" and "traffic" are not semantically similar, but are semantically related. Semantic relatedness can be used in tasks as word sense disambiguation and knowledge

based query expansion. Due to the close relation between semantic similarity and semantic relatedness, we will also include the latter in the background introduction.

Much of the previous work on computing word semantic similarity mainly focuses on extracting measures from the expert-edited structured resource – the WordNet [29]. A directed acyclic graph, the word sense taxonomy, can be built according to the synset, hypernym and hyponym relations defined in WordNet, in which nodes represent word sense synsets, and edges are added to link nodes with the hypernym-hyponym relation. Semantic similarity measures derived from this hierarchical structure can generally be divided into two groups, the path based measures and the information content based measures. For path based measures, Leacock and Chodorow [48] create a word semantic similarity measure based on the shortest path between word senses; Wu and Palmer [123] design a measure based on the depth of word senses and their least common subsumer (LCS); Yang and Powers [124] use weighted path counts between two words to assess their semantic similarity; and Alvarez and Lim [4] combine shortest path length, depth of LCS and WordNet gloss as the word semantic similarity measure. For information content based measures, Resnik [92] uses information content of the LCS as the measure. Jiang and Conrath [43] and Lin [53] improve Resnik's measure by evaluating the information content difference between the two word senses and their LCS. Other related work includes Hughes and Ramage [41], which builds the graph with richer semantic information in addition to the hypernym/hyponym relationship extracted from WordNet, and performs a random walk to assess word semantic relatedness.

The advantage of WordNet is its word sense taxonomy, which facilitates the definition of varies semantic similarity measures. But building and updating such a taxonomy by linguistic experts is expensive and time consuming. Recent work explores the use of on-line resources built by collaborative efforts, such as Wikipedia (`http://www.wikipedia.org/`) and its lexical counterpart Wiktionary. Ponzetto and Strube [87], Strube and Ponzetto [105], Zesch et al. [127] build a taxonomy from Wikipedia category labels, and apply the above semantic similarity measures originally designed for WordNet on the generated Wikipedia taxonomy; Yeh et al. [125] build a graph according to wikipedia page link relations and apply the random walk as proposed in [41] to extract semantic relatedness. Zesch et al. [128]

evaluate semantic relatedness by applying a concept vector based measure on Wiktionary pages.

Words in Wiktionary have yet to be organized into a taxonomy[2], which forces us to tackle the problem in a different way than the traditional taxonomy based semantic similarity measures designed for WordNet.

3.2 WikNet: Graph Representation of Wiktionary

3.2.1 Wiktionary

Wiktionary is a multilingual free dictionary that is written collaboratively by volunteers around the world. Since it can be edited freely, it enjoys more frequent updates and thus has broader coverage than WordNet, especially for newly invented words and word senses, such as *iPhone* and *tweet* (as an entry published on the microblog service). As of roughly April 2010, the English Wiktionary contained around 177k words and phrases, which exceeds the size of the latest WordNet version (v3.0) with 150k words and phrases. In addition, versions of Wiktionary exist for many languages, 18 of which contain more than 100k entries, so the semantic similarity measure we derived from the English Wiktionary can easily be extended to other languages, though the success will likely depend on how developed the resource is for that language. Compared with Wikipedia which mainly consists of nouns and proper nouns, Wiktionary has much better coverage on verbs, adjectives and adverbs, which makes it more suitable for tasks as paraphrase identification and generation.

On a typical Wiktionary entry page, there are pronunciations, part-of-speech (POS) and corresponding sense definitions of the word. Some of the word entries also contain other lexicon information as etymology, synonyms, antonyms, anagrams, related terms and derived terms, etc. Figure 3.1 shows an example English Wiktionary page for the word "rooster".

---

[2]Though Wiktionary has Wikisaurus as its subproject to build the taxonomy, it is still at an embryo stage, containing only around 1,258 entries by June 2012.

English

Pronunciation

- enPR: roo͡stər, IPA: /ˈruːstə(r)/, SAMPA: /"ru:st@(r)/
- Audio (US) (file)

Rhymes: -uːstə(r)

Noun

rooster (*plural* **roosters**)

1. A male domestic fowl, *Gallus gallus*.

Synonyms

- cock

Related terms

- roost

Derived terms

- roosterly
- roosterness
- roostertail

Figure 3.1: The English part of word “rooster” on the Wiktionary page

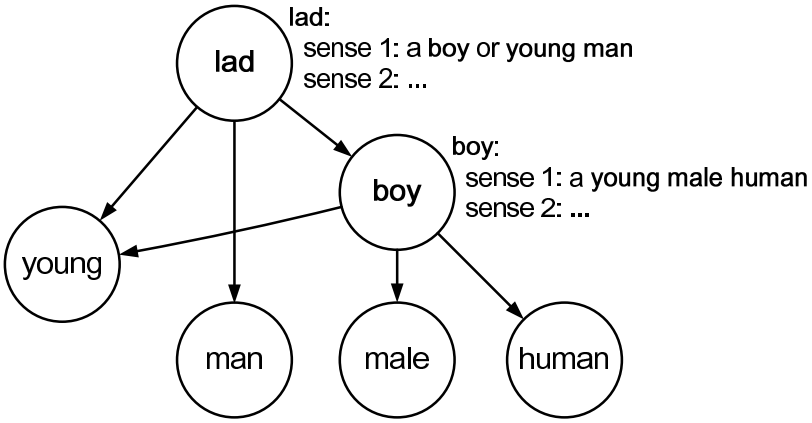


Figure 3.2: Part of the directed WikNet with words “boy” and “lad”.

3.2.2 WikNet

We construct the WikNet according to word sense definitions in English Wiktionary. In our work, each lexical content word (noun, verb, adjective and adverb) in the English Wiktionary is represented by one node in WikNet. The WikNet graph can be built as either directed or undirected. For the directed WikNet, if word j appears in any of the sense definitions of word i, one directed edge $i \rightarrow j$ is added, which represents “word i refers to word j” to explain its semantic meaning. Figure 3.2 shows part of the directed WikNet with words ”boy” and ”lad” as an example. For the undirected WikNet setting, we use the same approach to construct the WikNet graph except replacing directed edges with undirected edges. The edges are unweighted.

There are several issues we need to deal with when building the WikNet.

Map morphological variations to base forms

To avoid an overly large graph size and sparse links, we only add words in base forms to WikNet. Correspondingly, we need to map morphological variations in word sense definitions to their base forms when constructing the WikNet. In Wiktionary, a morphological variation entry has a link to its base form entry together with a tag labeling the morphological variation type. We use these morphological variation entries in Wiktionary to extract a

morphological form to base form map, which contains 119k map entries. We use this map to convert morphological variations in word sense definitions into their base forms.

Remove atypical word senses

Most word entries in Wiktionary contain more than one sense definitions, some of the sense definitions are rarely used, either because it is an archaic usage or only used in certain dialect. Adding these rarely used sense definitions to the WikNet may mislead the semantic similarity measures. We collect the sense context tags in Wiktionary, and manually picked 39 sense context tags, such as "rare", "obsolete", "dialect", "Jamaican English", to form an excluded sense context tag list. When we building the WikNet, sense definitions with these atypical context tags are ignored.

Remove stopwords

In Wiktionary, there are some words that appear in most of the word sense definitions; adding them to WikNet will lead to large number of edges with little semantic information. To deal with this issue, we build a stop word list for such words. Specifically, we treat each word sense definition as a document and compute the inverse document frequency (idf) for each word, and picked 88 words with the smallest idf values to form the stop word list. When building the WikNet, all words in the stop word list are ignored from the word sense definitions.

WikNet Statistics

We build both the directed and undirected WikNet from Wiktionary. In the undirected WikNet, there are roughly 177k nodes and 1.15M undirected edges. Its graph density[3] is 7.4×10^{-5}, which indicates that the undirected WikNet is a very sparse graph. The average node degree (number of edges of a node) of the undirected WikNet is 13.1. As shown in Figure 3.3, the degree distribution of the undirected WikNet conforms to the power law, which is an important property shared by social networks.

[3]The graph density for an undirected graph is defined as $D = \frac{2|E|}{|V|(|V|-1)}$, where $|E|$ denotes the number of edges, and $|V|$ denotes the number of nodes.

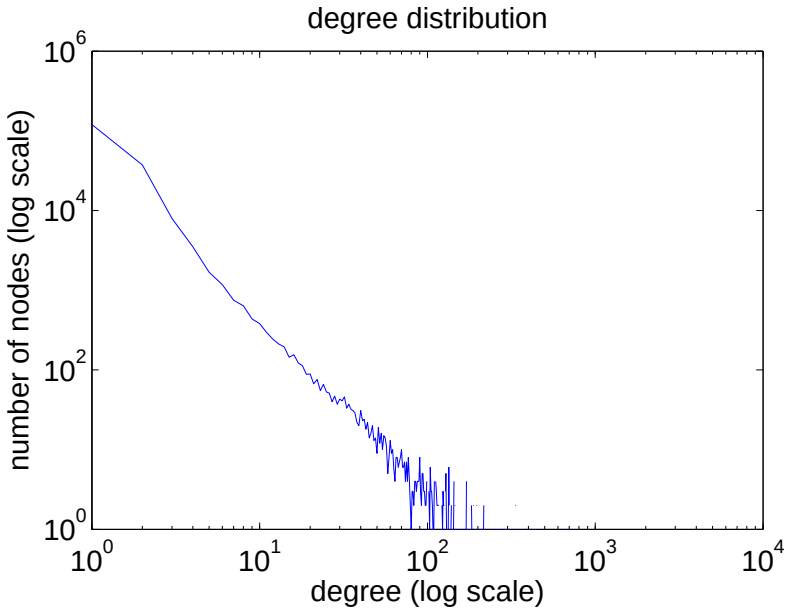


Figure 3.3: Degree distribution of undirected WikNet (both x-axis and y-axis are in log scale).

Another measure of the graph structure is the average cluster coefficient. The cluster coefficient of node n in an undirected graph is defined as

$$c_n = \frac{2T(n)}{deg(n)(deg(n)-1)} \tag{3.1}$$

where $T(n)$ is the number of edges among node n's neighbors, and $deg(n)$ is the degree of node n. It evaluates the local connection density of a graph. The average clustering coefficient of the undirected WikNet is 0.101, compared with that of a random graph 0.06.[4] The fact that the average clustering coefficient of the undirected WikNet is significantly larger than that of the random graph is an important property of a small world network [121], which is the type of graphs social networks belong to.

We also study the connectivity of undirected WikNet, the size of the largest connected component is 176k, which covers 99.3% nodes in WikNet. This shows that a large majority

[4]The average cluster coefficient of a random graph is computed according to the equation used in [138], $C_{random} = \frac{(<k^2>-<k>)^2}{|V|<k>^3}$, where k is the node degree, and $|V|$ is the number of nodes in the graph.

of words in WikNet are mutually connected, it is another indication that link-based node similarity algorithms which are commonly applied for person closeness evaluations in social network studies may also be feasible for word semantic similarity in WikNet.

In the directed WikNet, there are 1.17M directed edges; the average in-/out-degree (number of incoming/outgoing edges of a node) is 6.6. The size of the largest weakly connected and strongly connected components are 176k and 29k, respectively. Other properties of the directed WikNet are similar to those of the undirected WikNet.

3.3 WikNet-based Lexical Semantic Similarity Measures

Thanks to the analogy between WikNet and social networks, the neighborhood-based methods introduced in Section 2.2.1, which are widely applied on evaluating user closeness for link prediction or friend recommendation in social networks, can be borrowed as our basis to develop lexical semantic similarity in WikNet. Specifically, we view each word represented in WikNet as a source that propagates semantic concept information to its neighbors, and the overlap of two words' neighbors can be viewed as the overlap of the semantic information propagated from these two words, and thus can be used to evaluate their semantic similarity. Below we present the WikNet-based lexical semantic similarity measures derived from the neighborhood-based methods.

3.3.1 Common-neighbors

The *Common-neighbors* measure [80] counts the number of shared neighbors between two words in WikNet to evaluate their semantic similarity. For undirected WikNet, it is defined as,

$$sim(i,j) = |\Gamma(i) \cap \Gamma(j)| \tag{3.2}$$

where $\Gamma(i)$ is the neighbor set of word i, and $|\cdot|$ indicates the number of members in the set. We extend this measure to apply on the direct WikNet, which is defined as

$$sim(i,j) = w|\Gamma_{suc}(i) \cap \Gamma_{suc}(j)| + (1-w)|\Gamma_{pre}(i) \cap \Gamma_{pre}(j)| \tag{3.3}$$

where $\Gamma_{suc}(i)$ and $\Gamma_{pre}(i)$ represent the successor set and predecessor set of word i's node, respectively; and $w \in (0, 1)$. The first part of Equation (3.3) counts the number of overlapped words in word i and j's sense definitions, the second part counts the number of words that both refer word i and j in their sense definitions, and w is the weight used to balance the contribution of the two part to semantic similarity. Without development data for the test, we set w to be 0.5, which is the same case for the following neighborhood-based measures.

3.3.2 Adamic & Adar

The *Adamic&Adar* measure [1] can be seen as a refined version of the *common-neighbors* measure, which assumes that a word propagates different amount of semantic information to different neighbors and the rare neighbors receive more semantic information. Its undirected WikNet version is defined as,

$$sim(i, j) = \sum_{k \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log |\Gamma(k)|} \tag{3.4}$$

where $\Gamma(i)$ is the neighbor set of word i. Similarity, we define the Adamic&Adar measure for directed WikNet as,

$$sim(i, j) = w \sum_{k \in \Gamma_{suc}(i) \cap \Gamma_{suc}(j)} \frac{1}{\log |\Gamma_{pre}(k)|} + (1 - w) \sum_{k \in \Gamma_{pre}(i) \cap \Gamma_{pre}(j)} \frac{1}{\log |\Gamma_{suc}(k)|} \tag{3.5}$$

where $\Gamma_{suc}(i)$ and $\Gamma_{pre}(i)$ represent the successor set and predecessor set of word i's node, respectively, and $w \in (0, 1)$. Similar to the *common-neighbors* measure, the first part in the measure definition is the weighted sum of the shared words in word i and j's sense definitions, and the weight is each word's inverse document frequency (idf) over Wiktionary if we treat the collection of a word's sense definitions as one document.

3.3.3 Jaccard coefficient and its variations

Jaccard

Another "extended version" of the *common-neighbors* measure is the *Jaccard coefficient* [97]. It assess the semantic similarity between two words by counting the number of neighbors that both words have over the number of neighbors that either of them has in WikNet. For undirected WikNet, the *Jaccard coefficient* is defined as

$$sim(i,j) = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} \tag{3.6}$$

where $\Gamma(i)$ is the neighbor set of word i.

Equation (3.6) well illustrates the underlining logic of the *Jaccard coefficient*: it not only considers the overlap of the semantic information propagated from the two words, but also takes into account of the differences. This definition conforms to a set-theoretic contrast similarity as per Tversky's definition [115].

Similarly, we have the version for the directed WikNet as

$$sim(i,j) = w\frac{|\Gamma_{suc}(i) \cap \Gamma_{suc}(j)|}{|\Gamma_{suc}(i) \cup \Gamma_{suc}(j)|} + (1-w)\frac{|\Gamma_{pre}(i) \cap \Gamma_{pre}(j)|}{|\Gamma_{pre}(i) \cup \Gamma_{pre}(j)|} \tag{3.7}$$

The advantage of the *Jaccard coefficient* is that its value is bounded within the range of [0,1].

Xjaccard

An extended version of the *Jaccard coefficient* is the *XJaccard* [31] which not only considers the immediate neighbors of i, j, but also nodes that can be reached in n steps. It can be seen as a result of an n-step semantic information propagation. This extension is useful for WikNet, because two related words may not share any neighbors in WikNet, but can be related by nodes within their n-step's reach ($n \geq 2$). For instance, "rooster" and "crane" share no words in their sense definition, but they can be related by "bird" which is within 2-step's reach of "rooster" ("rooster"→"fowl"→"bird") and 1-step's reach of "bird" ("crane"→"bird").

For the undirected WikNet, the *XJaccard coefficient* is defined as

$$sim_n(i,j) = \sum_{l=1}^{n} C^l(1-C)J_l(i,j) \tag{3.8}$$

where $J_l(i,j) = \frac{|\Gamma_l(i) \cap \Gamma_l(j)|}{|\Gamma_l(i) \cup \Gamma_l(j)|}$, $\Gamma_l(i)$ is the set of nodes that can be reached in l steps from word i, and $C \in (0,1)$ is a damping factor having a similar function as the one defined in Katz's measure. Without development data for tuning, we omit the damping factor and use $sim_n(i,j) = \sum_{l=1}^{n} J_l(i,j)$.

Accordingly, the directed WikNet version is

$$sim_n(i,j) = w \sum_{l=1}^{n} J_{suc,l}(i,j) + (1-w) \sum_{l=1}^{n} J_{pre,l}(i,j) \tag{3.9}$$

where $J_{suc,l}(i,j) = \frac{|\Gamma_{suc,l}(i) \cap \Gamma_{suc,l}(j)|}{|\Gamma_{suc,l}(i) \cup \Gamma_{suc,l}(j)|}$, $J_{pre,l}(i,j) = \frac{|\Gamma_{pre,l}(i) \cap \Gamma_{pre,l}(j)|}{|\Gamma_{pre,l}(i) \cup \Gamma_{pre,l}(j)|}$; $\Gamma_{suc,l}(i)$ and $\Gamma_{pre,l}(i)$ are word i's l-step successor/predecessor set, respectively.

J0-XJaccard

In Wiktionary, a word is referred by other words in their sense definitions. For instance, in Figure 3.2, "boy" is referred by "lad" in its sense definition. It indicates high semantic similarity between two words with such referring relationship. However, because the definition of a word usually does not include itself which would be a self-loop in WikNet, when computing the semantic similarity between this word ("boy") and a word referring to it ("lad"), this word ("boy") is not included in $|\Gamma_l(i) \cap \Gamma_l(j)|$, so the Jaccard and XJaccard measures miss this important information. To deal with this issue, we introduce a virtual J_0 term in the *XJaccard coefficient*. For undirected WikNet, $J_0(i,j) = 1$ if there is an edge between i and j, otherwise $J_0(i,j) = 0$. For directed WikNet

$$J_0(i,j) = \frac{e(i,j) + e(j,i)}{2} \tag{3.10}$$

where $e(i,j) = 1$ if there is an edge pointing from i to j, otherwise $e(i,j) = 0$.

Synonym J0-XJaccard

Many words in Wiktionary contains a synonym list, which is another information source we can leverage to improve the semantic similarity measure. First, if word i is in the synonym

list of word j, or vice versa, then they should have the maximum semantic similarity value. Second, when we count the common "features" of the two words, i.e., computing $|\Gamma_l(i) \cap \Gamma_l(j)|$ in the *Jaccard* or *XJaccard* equation, we can also incorporate the synonym information by counting two words as a common "feature" if one is on the other's synonym list or their synonym lists overlap. Below we use $|\Gamma_l(i) \cap_{synset} \Gamma_l(j)|$ to denote the common "feature" number counted with the synonym information.

Hence for undirected WikNet, we define the *Synonym J0-XJaccard* as

$$sim_n(i,j) = \begin{cases} maxsim & \text{if } i \in synset(j) \text{ or } j \in synset(i) \\ \sum_{l=0}^{n} J_l^{synset}(i,j) & \text{otherwise} \end{cases} \tag{3.11}$$

and

$$J_l^{synset}(i,j) = \frac{|\Gamma_l(i) \cap_{synset} \Gamma_l(j)|}{|\Gamma_l(i) \cup \Gamma_l(j)|} \tag{3.12}$$

where $synset(i)$ is the set of synonyms of word i, and $maxsim$ is the maximum possible value of *J0-XJaccard*, it is 3 with no damping and $n = 2$.

Similarly, we can define the *Synonym J0-XJaccard* for directed WikNet by replacing $J_l^{synset}(i,j)$ with the weighted sum of $J_{suc,l}^{synset}(i,j)$ and $J_{pre,l}^{synset}(i,j)$.

The *Synonym J0-XJaccard* integrates the Wiktionary synonym information into the n-step semantic information propagation.

3.3.4 Non-linear transformation of semantic similarity score

We apply a non-linear transformation over all semantic similarity measures we proposed for WikNet. Suppose the original semantic similarity score produced by these measures is x, then the transformed score is computed as

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{3.13}$$

This transformation was proposed in [52] for measuring semantic similarity. It scales scores to compress the difference between higher scores.

3.4 Experiments on Human-rated Lexical Semantic Similarity Datasets

In this section, we evaluate the proposed WikNet-based lexical semantic similarity measures against human-rated similarity scores on three standard datasets, and compare their performance with several commonly used WordNet-based lexical semantic similarity measures to demonstrate the feasibility of the WikNet-based approaches.

3.4.1 Datasets

The evaluation datasets for lexical semantic similarity are generally created by asking human annotators to rate the semantic similarity of given word pairs; and the averaged rating of these annotators are set as the golden standard score. Several such human rated datasets for word semantic similarity are available: Rubenstein and Goodenough [96] created a dataset with 65 noun pairs; Miller and Charles [68] and Resnik [92] used a subset (30 noun pairs) of Rubenstein and Goodenough's dataset for their experiments, and each provided a set of human ratings for this subset. Finkelstein et al. [30] has a larger set that contains 353 word pairs, but this dataset is mainly designed for semantic relatedness rather than semantic similarity. Hence, we adopt the three semantic similarity datasets, Rubenstein and Goodenough, Miller and Charles, and Resnik for our experiments. Although the correlation of human-rated similarity scores between these sets are high, we report results on all three sets, since that is standard in other work on semantic similarity measures.

3.4.2 Results and Analysis

We evaluate semantic similarity measures by computing *Pearson's correlation* between their scores and the corresponding human ratings, which is standard for reporting on these data sets. We first compare the performance of WikNet-based measures, including *common-neighbors*, *Adamic&Adar*, *Jaccard coefficient* and its variations. The performance of WikNet-based lexical semantic similarity measures are presented in Table 3.1.

It is shown that two relatively simple measures, *common-neighbors* and *Adamic&Adar*, achieve fairly good performance when computed with the directed WikNet. The *Synonym J0-XJaccard coefficient* in directed WikNet achieves the best performance, obtaining Pear-

Table 3.1: Comparison of WikNet-based semantic similarity measures on Resnik, Miller and Charles (M&C), Rubenstein and Goodenough (R&G) datasets under Pearson correlation coefficient

Semantic similarity measure	**Resnik**	**M&C**	**R&G**
Common-neighbors (undirected)	0.63	0.60	0.55
Common-neighbors (directed)	0.77	0.81	0.71
Adamic&Adar (undirected)	0.74	0.77	0.68
Adamic&Adar (directed)	0.76	0.77	0.76
Jaccard (undirected)	0.67	0.61	0.63
Jaccard (directed)	0.60	0.55	0.64
Synonym J0-XJaccard (n=2)(undirected)	0.85	0.83	0.77
Synonym J0-XJaccard (n=2)(directed)	**0.88**	**0.85**	**0.83**

son correlation with human ratings as 0.88, 0.85 and 0.83 on the three datasets respectively.

To further assess the relative contribution of different extensions from the basic *Jaccard coefficient* to the *Synonym J0-XJaccard* in directed WikNet, we added these extension one by one, and presented their performance in Table 3.2. It is shown that the *XJaccard*, which searches the given word's successors/predecessors beyond one step's reach, explores richer information, and thus outperforms the original *Jaccard coefficient*, which only considers the imminent successors/predecessors. The biggest improvement is achieved by incorporating the $J0$ term and synonym information. The $J0$ term extension includes the otherwise neglected link between the compared word pairs in the *Jaccard/XJaccard* measure. The synonym extension incorporates synonym information in computing the joint set size of the given word pair's successors/predecessors, which is essentially a makeup of the sparsity in WikNet.

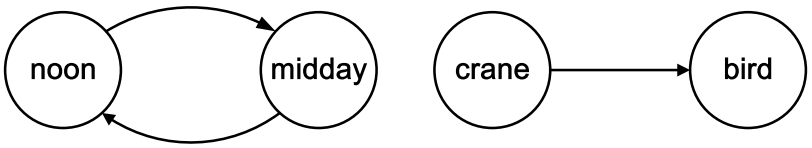
The *Synonym J0-XJaccard* measure used with undirected WikNet has the second best result among WikNet based measures. The fact that it performs worse than its directed

Table 3.2: Comparison of the performance of Jaccard coefficient with different extensions under Pearson correlation coefficient

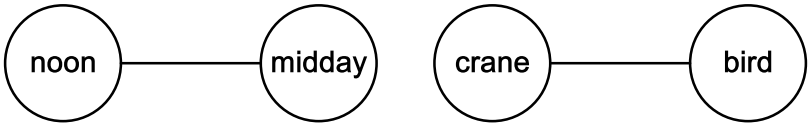
Semantic similarity measure	**Resnik**	**M&C**	**R&G**
Jaccard (directed)	0.60	0.55	0.64
XJaccard (n=2)(directed)	0.64	0.57	0.67
J0-XJaccard (n=2)(directed)	0.74	0.71	0.74
Synonym J0-XJaccard (n=2)(directed)	**0.88**	**0.85**	**0.83**

WikNet counterpart is due to the loss of directional information among a word's referring relationship in sense definitions, which can be critical for assessing word semantic similarity, especially on cases involving hyponym-hypernym pairs. For instance, in experimental results obtained on the undirected WikNet, we found that some hyponym-hypernym pairs, such as "crane" and "bird", "cock" and "bird", were assigned similarity scores comparable to those of synonym pairs, such as "noon" and "midday", "gem" and "jewel". In Wiktionary, hyponyms tends to refer to their corresponding hypernyms in sense definitions, but generally will not have links referring back; while synonym pairs tend to refer to each other in their definitions, and thus generate mutually linked node pairs in WikNet. The directed WikNet can distinguish hyponym-hypernym pairs from synonym pairs by checking whether the word pair is mutually linked or only linked by a single direction edge (Figure3.4a); however, the undirected WikNet has difficulty doing so due to its edges having no direction information (Figure3.4b).

Next, we compare the *Synonym J0-XJaccard* measure in directed WikNet with some of the most commonly used WordNet-based lexical semantic similarity measures, including the measures proposed by Wu and Palmer [123], Resnik [92], Jiang and Conrath [43], Lin [53] and Leacock and Chodorow [48]. The WordNet-based measures were implemented with the Perl package WordNet::Similarity [86]. Their results are presented in Table 3.3. It is shown that *Synonym J0-XJaccard* outperforms the above WordNet-based semantic



(a) synonym pair vs. hyponym-hypernym pair representation in the directed WikNet



(b) synonym pair vs. hyponym-hypernym pair representation in the undirected WikNet

Figure 3.4: Comparison of synonym pair vs. hyponym-hypernym pair representation in the directed and undirected versions of WikNet

Table 3.3: Comparison of WikNet-based and WordNet-based measures on Resnik, Miller and Charles (M&C), Rubenstein and Goodenough(R&G) datasets under Pearson correlation coefficient

Semantic similarity measure	**Resnik**	**M&C**	**R&G**
Synonym J0-XJaccard (n=2)(directed)	0.88	0.85	0.83
Gloss Vector	0.83	0.81	0.76
Wu and Palmer (1994)	0.78	0.76	0.80
Resnik (1995)	0.82	0.81	0.82
Jiang and Conrath (1997)	0.80	0.74	0.72
Lin (1998)	0.80	0.75	0.74
Leacock and Chodorow (1998)	0.83	0.78	0.84
Yang and Powers (2005)	n/a	0.92	0.90
Alvarez and Lim (2007)	0.92	0.91	0.90
Agirre et.al. (2009)	n/a	0.93	n/a

similarity measures on the three datasets except for Leacock and Chodorow on the R&G set, which obtains a correlation of 0.84, slightly higher than our result 0.83. Considering that WikNet lacks a well-organized hierarchical structure of word senses as in WordNet, this result which is obtained solely on the linking relationship of WikNet and synonym information from Wiktionary is very promising.

We also implement the *GlossVector*-based lexical semantic similarity measure for Wiktionary, which was originally proposed by Patwardhan and Pedersen [85] for WordNet. We would like to compare the GlossVector-based measure derived from the WordNet with our WikNet-based measure, however, since Patwardhan and Pedersen did not provide their result in Pearson correlation coefficient, we implement this method for Wiktionary as an alternative comparison. It is shown that the *Synonym J0-XJaccard* measure in directed WikNet also outperforms the GlossVector-based measure derived from Wiktionary.

We also list the Pearson correlation coefficient results obtained in some later work [124, 4, 2] on WordNet-based lexical semantic similarity measures in Table 3.3. These studies design more sophisticated models for extracting lexical semantic similarity and achieve better performance on some of the three human-rated lexical semantic similarity datasets. However, the size of these datasets is small, and they cover only nouns. In addition, there have been so many studies reported on these datasets that there is a high potential for over tuning to the set. Results are reported on these datasets since it is a standard for work in lexical semantic similarity, but the comparisons represent only a limited assessment that we view as mainly useful for development and validation. In the next section, we apply the lexical semantic similarity measures on NLP tasks to evaluate their performance in real application scenarios.

3.5 Experiments on NLP applications

In this section, we apply the WikNet and WordNet based lexical semantic similarity measures on two NLP applications: paraphrase identification in news articles, and implicit alignment identification in web forum discussions.

3.5.1 Paraphrase Identification

Paraphrase identification is a useful task for many natural language processing applications. For instance, it can be used for grouping semantically equivalent or similar sentences in multiple document summarization, or for automatically extracting paraphrased phrases or sentences as training data for text rewriting. It is also a good test bed for evaluating the effectiveness of word semantic similarity measures, Corley and Mihalcea [19] tested the performance of paraphrase identification with different semantic similarity measures derived from WordNet. To compare the effectiveness of Wiktionary derived semantic similarity measure, we redo the paraphrase identification experiments in [19] using the directed version of WikNet-based *Synonym J0-XJaccard*.

Lexical Semantic Similarity based Paraphrase Identification

We adopt the lexical semantic similarity based paraphrase identification algorithm proposed in [19]. The basic idea is that for a given sentence pair, we pair up words in each sentence that are most similar to each other and weight their semantic similarity to produce the similarity score of the sentence pair.

Specifically, for every given pair of sentences, we first perform part-of-speech (POS) tagging on each sentence and divide the words into several POS classes. We pick words in content word POS classes (including nouns, verbs, adjectives, adverbs) and cardinals to evaluate the semantic similarity between these two sentences. For each word with the chosen POS classes in one sentence, we find its best match in the same POS class from the other sentence.[5] For the four semantic content word POS classes, we use the directed WikNet-based *Synonym J0-XJaccard* measure; and for the cardinals, we apply lexical matching. This is a little bit different from the original method proposed in [19], where only verbs and nouns use semantic similarity derived from WordNet, while adjectives, adverbs and cardinals use lexical matching, since most semantic similarity measures for WordNet only apply to nouns and verbs. Since our proposed measure derived from Wiktionary has no such restrictions, we apply semantic similarity to all four semantic content word POS classes.

[5]Stemming is performed before the matching to transform the surface form of a word to its lexical form.

After obtaining the best match of each word in the sentence pair, the semantic similarity of the matched word pair is summed together as the sentence pair similarity score. Since words may have different contributions to determining the semantic content of a sentence, the semantic similarity of each matched word pair is weighted with the word idf. Hence, the directional similarity from sentence S_i to sentence S_j is defined as

$$Sim(S_i, S_j)_{S_i} = \frac{\Sigma_{w_k \in POS(S_i)} maxSim(w_k) * idf_{w_k}}{\Sigma_{w_k \in POS(S_i)} idf_{w_k}} \tag{3.14}$$

where $POS(S_i) = \{w_k \in S_i | POS(w_k) = noun, verb, adjective, adverb, cardinal\}$, and

$$maxSim(w_k) = \max_{w_l \in S_j and POS(w_l) = POS(w_k)} sim(w_k, w_l) \tag{3.15}$$

which is the semantic similarity (for noun/verb/adjective/adverb) or lexical match (for cardinal) between w_k and its best match in sentence S_j. Here we use the *Synonym J0-XJaccard* derived from the directed WikNet as the semantic similarity measure.

Consequently, the bidirectional sentence similarity is defined as the combination of the two directional sentence similarities,

$$Sim(S_i, S_j) = \frac{Sim(S_i, S_j)_{S_i} + Sim(S_j, S_i)_{S_j}}{2} \tag{3.16}$$

The obtained sentence similarity score is within the range $[0, 1]$.

Evaluation

We use the Microsoft Research Paraphrase Corpus [25] to evaluate the effectiveness of *Synonym J0-XJaccard* derived from directed WikNet for paraphrase identification. The Microsoft Research Paraphrase Corpus contains a training set with 4,076 sentence pairs, and a test set with 1,725 sentence pairs.

As in [19], we use the sentence similarity score defined above as the sole indicator for paraphrase identification, and conduct the experiments with two settings:

1. Unsupervised setting

A constant threshold of the sentence similarity score is set to be 0.5 for determining whether the given sentence pair is paraphrase or not.

2. Supervised setting

The threshold of sentence similarity is tuned by maximizing the paraphrase identification F1 measure over the training set.

Table 3.4 and Table 3.5 show the experimental results of our proposed semantic similarity measure and the ones obtained in [19] on the unsupervised and supervised setting, respectively. In [19], the paraphrase identification performance obtained with several WordNet-based semantic similarity measures were given, including Lin [53], Wu and Palmer [123], Leacock and Chodorow [48], Jiang and Conrath [43], Resnik [92] and their combination. In the supervised setting, the optimal threshold and weight (for the combined result) for various semantic similarity measures were learned from the training set.

Table 3.4: Comparison of paraphrase identification performance with different semantic similarity measures under unsupervised setting (the results of WordNet-based measures are obtained from [19])

Semantic similarity measure	Accuracy	Precision	Recall	F1 measure
WordNet-based measures				
Wu and Palmer (1994)	67.4	72.2	83.1	77.3
Resnik (1995)	67.2	72.5	81.5	76.8
Jiang and Conrath (1997)	68.3	72.4	84.6	**78.0**
Lin (1998)	67.9	71.7	85.5	**78.0**
Leacock and Chodorow (1998)	68.0	72.4	83.8	77.7
Combined	**68.8**	74.1	81.7	77.7
WikNet-based measure				
Synonym J0-XJaccard	**71.4**	70.5	97.8	**82.0**

For the paraphrase identification task, the WikNet-based *Synonym J0-XJaccard* outperforms each individual WordNet-based semantic similarity measure under both the unsupervised and supervised settings. Though it has slightly lower precision than its WordNet

Table 3.5: Comparison of paraphrase identification performance with different semantic similarity measures under supervised setting (the results of WordNet-based measures are obtained from [19])

Semantic similarity measure	Accuracy	Precision	Recall	F1 measure
WordNet-based measures				
Wu and Palmer (1994)	69.9	70.5	94.1	80.6
Resnik (1995)	69.2	70.5	92.1	79.9
Jiang and Conrath (1997)	69.9	70.7	93.5	80.5
Lin (1998)	70.2	70.6	94.7	80.9
Leacock and Chodorow (1998)	69.9	70.8	93.1	80.4
Combined	**71.5**	72.3	92.5	**81.2**
WikNet-based measure				
Synonym J0-XJaccard	**71.6**	71.4	95.8	**81.8**

counterparts, but its recall is much higher. Overall, it achieves F1 measure of 82.0 and 81.8 on both settings, which accomplishes a better or comparable result with the combined results of WordNet-based measures of 78.0 and 81.2 on the two settings, respectively.

We hypothesized that WikNet might have better vocabulary coverage than WordNet, but this is not the case. The vocabulary coverage of WikNet and WordNet on this corpus are similar, each covers 81.5% and 80.8% words appearing in this corpus. There are several possible advantages of the WikNet-based semantic similarity measure that contribute to its better performance on this paraphrase identification task. First, the WikNet-based measure can also apply on adjectives and adverbs, while WordNet-taxonomy-based measures do not support semantic similarity on these two types of content words very well. Second, the WikNet-based measure is derived from Wiktionary sense definitions, thus it can also evaluate the semantic similarity of two words with different POS; on the other hand, the hypernym and hyponym relation used to build WordNet taxonomy covers only words with

the same POS, thus the WordNet taxonomy based measures have poor performance to assess the semantic similarity between two words with different POS. For instance, the semantic similarity between "declare" and "declaration" under *Synonym J0-XJaccard* is 1, while their semantic similarity under the three WordNet taxonomy based measures are all 0.

One thing worth noting is that the WikNet-based *Synonym J0-XJaccard* obtains similar results under the unsupervised and supervised setting, while there are performance gaps of the WordNet-based measures under the two settings. This indicates that the WikNet-based paraphrase identification has less reliance on development data.

3.5.2 Implicit Alignment Identification

The second task we tested is the implicit alignment identification in web forum discussions. Alignments are statements made by participants in discussions to express their agreement or disagreement with other participants' opinions. Alignments are common moves in web forum discussions, which is an important feature for studying the discussion dynamics and dividing the participants into different opinion groups. Besides explicit alignments, where the participants use wording such as "I agree", "No, I don't think so" to signal their agreement or disagreement with others, implicit alignments also widely exist in web forum discussions. In an implicit alignment, a discussant responds with a rewording of another discussant's statement to either confirm or negate them, as shown by the examples in Table 3.6.

Lexical Semantic Similarity based Implicit Alignment Identification

Due to the semantic overlap between the two statements involved in the implicit alignments, the lexical semantic similarity based algorithms used for paraphrase identification can also be used for identifying implicit alignments in web forum discussions. In this section, we apply the same algorithm introduced in Section 3.5.1 to identify implicit alignments.

Table 3.6: Examples of implicit alignments in Wikipedia forum discussions

Agreement	A: **Invasion** is a **completely neutral word** as far as I'm concerned . B: **'Invasion'** is a **reasonably neutral word**
Disagreement	A: Its logical that in the first case the sanctions **caused more** people to **die**, and in the second case the **war** did. B: If **mortality rates** were **increasing** already, then a **war** begins, you cannot simply say everyone who **died** over last years number **died because** of the **war**.

Evaluation

We built an implicit alignment identification corpus from Wikipedia forums, where participants discuss the editing of corresponding Wikipedia pages. Adjacent sentence pairs with context overlaps larger than a threshold were extracted from 106 Wikipedia forum discussion threads. One annotator was hired to label each given sentence pairs as implicit agreement or disagreement or not an alignment. In this evaluation, we did not distinguish between agreement or disagreement cases, since here we are interested in semantic similarity and not polarity. There are 297 implicit alignment moves being identified among the 1445 extracted sentence pairs.

During the evaluation, each sentence pair is presented to the system, where a decision is made to judge whether the given sentence pairs involves implicit alignment moves or not.[6]

We applied the lexical semantic similarity based algorithms used in Section 3.5.1 to identify implicit alignments in this corpus. Due to a lack of development data, we used the unsupervised setting in this experiments, i.e., a constant threshold of the sentence similarity score is set to be 0.5 for determining whether the given sentence pair is implicit alignment or not. We compared the proposed *Synonym J0-XJaccard* semantic similarity derived from

[6]the polarity, i.e., agreement or disagreement, of the sentence pairs are not identified

the directed WikNet with three WordNet based semantic similarity measures that obtains the best performance in the paraphrase identification task under the unsupervised setting, including "Jiang and Conrath", "Leacock and Chodorow", and "Lin". These WordNet-based measures were implemented with the Perl package WordNet::Similarity [86].

Table 3.7: Comparison of implicit alignment identification performance with different semantic similarity measures

Semantic similarity measure	Accuracy	Precision	Recall	F1 measure
WordNet-based measures				
Jiang and Conrath (1997)	52.0	23.0	56.9	32.8
Lin (1998)	**68.1**	28.9	37.7	32.7
Leacock and Chodorow (1998)	43.5	22.4	71.0	**34.1**
WikNet-based measure				
Synonym J0-XJaccard	**72.5**	34.4	37.0	**35.7**

The experimental results are presented in Table 3.7. It is shown that the WikNet-based *Synonym J0-XJaccard* measure outperforms all three WordNet-based measures both in terms of accuracy and F1 score. Besides the advantages of WikNet semantic similarity measures we discussed in Section 6.1, the WikNet-based measures have one additional advantage over WordNet-based measures on this task. The WikNet has better vocabulary coverage on this implicit alignment identification corpus, which covers 89.8% words appearing in it, comparing with 86.6% for WordNet. The coverage advantage is more obvious on the web forum genre corpora, thanks to the collaborative work by Wiktionary volunteers, the Wiktionary can enjoy more frequent updates than the expert-edited WordNet, and thus have better coverage on new words, especially web terms.

The overall performance on the implicit alignment identification task is not as good as that on the paraphrase identification task, because the former is a more difficult task. First, the way the paraphrase corpus [25] was collected and filtered simplified the task, for

instance, the positive target (true paraphrase) rate in the test set is as high as 66.5%, which doesn't reflect the rate in the original data. On the other hand, the implicit alignment identification corpus was collected in a relatively "organic" way, its positive target (true implicit alignment) rate is 20.6%, which is closer to the discussion dynamics in the web forum, but the lower positive target rate makes it harder to get a high F1 score. Second the datasets for the two tasks involve different genres: the paraphrase identification data was extracted from news articles that are in formal written styles, while the implicit alignment identification data was collected from web forums that is in causal web language styles, which make it more difficult to analyze. But since the alignment detection task is more difficult, the WikNet-based measure obtains greater relative improvement over WordNet-based measures on this task than the paraphrase identification task.

3.6 Summary

In this chapter, we propose a method for representing semantic relations between words based on a graph constructed from Wiktionary word sense definitions. This graph shares many properties with social networks. We borrow algorithms for evaluating member closeness in social networks and enhance them by incorporating additional semantic information provided by Wiktionary, including the mutual reference between sense definitions and synonym lists, to apply to the problem of assessing word semantic similarity in WikNet. Based on experiments on three word semantic similarity datasets, the proposed *Synonym J0-XJaccard* applied on the directed WikNet obtains the best performance, which outperforms many of the WordNet derived semantic similarity measures and achieves Pearson correlation with human rates of 0.83-0.88. The *Synonym J0-XJaccard* also outperforms the WordNet-based measures in the paraphrase identification task on Microsoft Research Paraphrase Corpus, and the implicit alignment identification task on the Wikiforum discussion corpus.

Chapter 4

TWITTER USER INTEREST SUMMARIZATION

Twitter is a popular social media and social network service, which allows users to publish tweets with up to 140 characters on "what am I doing" or "what's happening right now". Twitter is growing at a fast pace. Currently it has 140 million active users, generating 340 million tweets per day. A large proportion of Twitter users update their tweets frequently, in which case the tweets compose a detailed log of their everyday life. These tweets contain rich information about the user's interest: people tweet about their hobbies, preferences and products they use, etc. Identifying a Twitter user's interest has wide commercial applications. For instance, it can be employed to produce personalized advertising for promoted tweets and promoted accounts. It can also be used as an important feature to improve the quality of "who to follow" suggestions, where users sharing similar interests and hobbies can be recommended as friends on Twitter.

In this chapter, we present an automatic system using graph-based algorithms to summarize a Twitter user's interest from their tweets in terms of key words or phrases. Keywords are used in mainstream content-based advertising systems, such as Google's Adsense or Twitter's promoted tweet system, where advertisements are shown to users based on whether related keywords are triggered in the content. We introduce a new two-stage graph-based algorithm to model the semantic concept information among a user's tweets, and extract key words or phrases to summarize his/her interests. This two-stage graph-based algorithm concatenates a graph random walk algorithm and a maximum graph-cut algorithm, which has the advantage of integrating the representativeness and diversity criteria into the summarization results.

4.1 Background

Studies that are related to our work include Zhao et al. [130] and Liu et al. [58], which use a keyword extraction technique to summarize Twitter topics contributed by the whole Twitter community. Our work is focused instead on summarizing an individual user's interests. Ou et al. [81] works on mining user's tweets and social network links for recommending Twitter users to join in groups sharing similar interests. Their work employs latent Dirichlet allocation [8] based topic models to represent a Twitter user with the topic vector for exploring similarity between users, but they do not directly extract keywords describing the user's interest.

Besides previous work on Twitter corpora, our work is also related to keyword extraction on academic papers, spoken documents and web pages. For supervised keyword extraction, a series of papers [112, 114, 42, 126, 57] employed TFIDF or its variants with Part-of-Speech (POS), capitalization, phrase and sentence length, etc., as features to train keyword extraction models, and discriminative training is usually adopted. Yih et al. [126] use logistic regression to extract keywords from web pages for content-targeted advertising, which has the most similar application to our work. However, due to the lack of human annotation on the Twitter corpus, we have to adopt an unsupervised strategy.

For unsupervised keyword extraction, TFIDF ranking is a popular method, and its effectiveness has been shown in the literature [42, 126]. TextRank and its variants [67, 119, 59] are graph-based text ranking models, which are derived from Google's PageRank algorithm [12]. It outperforms TFIDF ranking on traditional keyword extraction tasks. However, previous work on both TFIDF ranking and TextRank has been done mainly on academic papers, spoken documents or web pages, whose language style is more formal than that of Twitter messages. Tweets contain large amounts of "noise" like emoticons, internet slang words, abbreviations, and misspelled words. In addition, tweets are a casual log of a user's everyday life, which often lacks a coherent topic sequence compared to academic papers and most spoken documents.

4.2 Algorithm Outline and System Architecture

For an active Twitter user who tweets frequently, the collection of his/her tweets can essentially serve as a log of his/her everyday life, thus the semantic concept information expressed in this user's tweets convey his/her personal interests. If we can use a graph to represent the collection of a user's tweets, in which each node represents a word or phrase appearing in the tweets, and edges are added between nodes according to certain relatedness criterion, then the user interest summarization process can be modeled with semantic information propagation among this graph. Specifically, we can treat each word or phrase in the graph as both a source and a receiver of semantic information, then depending on the modeling strategy, the top-N words and phrases that receive or emit the most semantic information in the graph (i.e. have the strongest relatedness to the rest of the words and phrases) can be extracted to summarize the user's interests.

A typical graph-based algorithm for summarization is PageRank [12], also known as TextRank [67] when applied on text corpora. PageRank assigns each node in the graph with a rank value, which will be updated iteratively until it converges to a stationary distribution. This process can be viewed as an ∞-step semantic information propagation in the context of text summarization. The advantage of PageRank is that due to the ∞-step process, it thoroughly propagates the semantic information among words and phrases represented in the graph, thus can usually produce better performance in terms of the representativeness criterion for summarization results. However, PageRank does not take the diversity criterion into account during the summarization, one potential problem is that the summarization result may contain too much redundancy. For instance, if a user is a fan of football, he may mention many words and phrases related to football in his tweets. These words and phrases tend to coexist within tweets, and have high semantic similarity with each other, and thus form a closely connected group in the graph. Words and phrases in this group tend to promote each other's rank values during the PageRank iterations, and may end up with many related and phrases having high rank values, and crowd out words and phrases related to this user's other interest from the top-N result.

Another graph-based algorithm can be applied for summarization is maximum graph-cut

[56]. With the given graph representation, an N-size subgraph that has the maximum cut to the rest of the graph is extracted as the summarization. In the context of summarization, maximum graph-cut can be viewed as a 1-step information propagation: the subgraph that emits the most semantic information propagated from its neighbors is extracted as the summarization result. This algorithm has the advantage of integrating the diversity criterion for summarization, i.e., it only measures the semantic information propagation between the summarization subgraph and its complement subgraph, while disregards the propagation within the summarization subgraph. However, it only considers the result of 1-step propagation, and thus fails to take into account the semantic information propagated between words and phrases that are not directly connected in the graph. As a result, the maximum graph-cut algorithm integrates the diversity criterion at the expense of the representative criterion.

To take the advantages of both algorithms, we propose a new two-stage graph-based summarization algorithm that cascades PageRank and maximum graph-cut, which incorporates both the representativeness and diversity criteria in the summarization result. We use PageRank to perform the ∞-step semantic information propagation on the first stage, and then take the accumulated semantic information received by words and phrases in PageRank and further propagate it within 1-step during the maximum graph cut process as the second stage. Words and phrases in the subgraph that emit the most semantic information on the second stage are extracted as the summarization result.

Figure 4.1 shows the architecture of our Twitter user interest summarization system. In this system, tweets from the target user are collected and put through a preprocessing pipeline, where the tweets are cleaned and tokenized. Then the tokenized tweets will go through the two-stage graph-based algorithm to extract user interest keywords. In stage 1, an undirected token graph is built according to the co-occurrence relation and semantic similarity between words and phrases in tweets, and PageRank is applied over the graph to extract the top user interest summarization candidates. In stage 2, another smaller directed graph is built according to the PageRank result and pairwise semantic similarity between summarization candidates obtained in stage 1, and maximum graph-cut is applied to produce the final summarization results of user interests. In the following sections of this

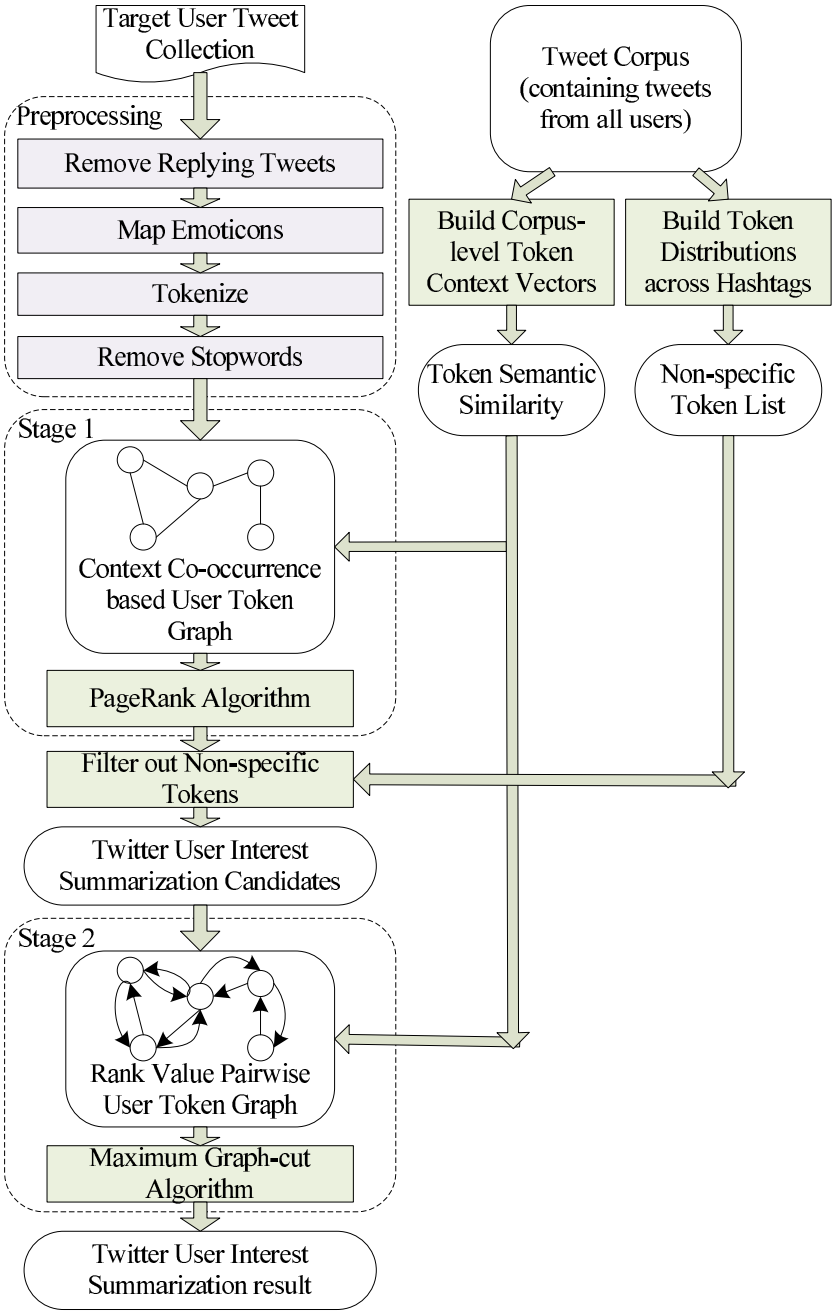


Figure 4.1: System architecture

chapter, details of each component in the system will be provided.

4.3 Preprocessing Pipeline

A tweet is from a very noisy text genre, which is filled with emoticons, various abbreviations and slang words. Useful information about user interest can be easily buried in this "noisy" background. We designed a preprocessing pipeline to remove the noise in replying tweets and tokenize each tweet into a collection of words or phrases.

4.3.1 Removing Replying Tweets

In addition to tweets describing "What am I doing" or "what's happening right now", Twitter users also write replying tweets to comment on other users' tweets. These kinds of tweets generally contains more information about the users they reply to than about themselves, and therefore are removed in the preprocessing pipeline.

4.3.2 Mapping Emoticons

Most emoticons in tweets can be classified as a laughing face or a crying face, hence we created a "_laugh_" and a "_cry_" token to represents the two groups of emoticons, and mapped the most frequent emoticons into these two tokens. Part of the mapping rules are presented in Table 4.1.

Table 4.1: Examples of Emoticon Mapping

Emoticons mapped to _laugh_	Emoticons mapped to _cry_
:)	:(
:-)	:-(
:D	T_T
:-D	
=)	
=D	

4.3.3 Tokenization

Tweets are tokenized into words and phrases according to a vocabulary containing 159k words and 14k phrases, words that are not in the vocabulary are discarded, and punctuation is removed. In this vocabulary, words are selected by filtering out infrequent words (frequency < 50) in the Twitter corpus, and phrases are extracted from the Twitter corpus using the mutual information and context dependency based algorithm proposed in [129]. Words outside this vocabulary are discarded in the preprocessing pipeline.

4.3.4 Removing Stopwords

The stopword list is built with multiple sources. It consists of a publicly available stopword list for general NLP tasks[1], a swear word list from `http://www.noswearing.com/`, a manually edited version of the slang word list from `http://www.noslang.com/` excluding content slang words,[2] and a Twitter-specific stopword list. The Twitter-specific stopword list is built by selecting words with low inverse document frequency (idf) in the Twitter corpus, where the idf is computed by taking each tweet as one document. After merging stopwords from these sources, we obtain a final stopword list containing 6.4k words.

4.4 Graph Representation of Tweets and Two-stage Summarization Algorithm

The proposed two-stage graph-based summarization algorithm concatenates PageRank and maximum graph-cut in order to incorporate the representativeness and diversity criteria in the summarization result. We design two different graph representations for the two stages in the summarization algorithm respectively. The graph representation for the PageRank stage is based on a combination of words/phrases context co-occurrence and their semantic similarity, and the graph representation for the maximum graph-cut stage integrates the PageRank outputs.

---

[1] `http://armandbrahaj.blog.al/2009/04/14/list-of-`english-stop-words/

[2] A few slang words are potentially useful content words, e.g. "bff" (best friend forever), "fone" (phone), so they are not included in the stopword list.

4.4.1 Stage 1 PageRank Summarization

Context Co-occurrence based Graph Representation

In the PageRank stage of the proposed summarization algorithm, we develop a graph representation for a user's tweet collection based on a combination of word and phrase context co-occurrence and their semantic similarity. Specifically, after tweets from the target user are cleaned and tokenized in the preprocessing pipeline, an undirected graph is constructed to represent this user's tweets. In this graph, each node represents a token (word or phrase) appearing in the user's tweets. An edge is added between two tokens if they co-exist within at least one tweet of this user.

Different from the traditional TextRank, where the edge weight is solely determined by two tokens' co-occurrence frequency, we introduce a weight that combines co-occurrence frequency of the two tokens in a user's tweet collection with a corpus-based semantic similarity score between them, where the semantic similarity serves as the corpus background information. The weight is set to be the multiplication[3] of the user-based co-occurrence frequency and the corpus-based semantic similarity for the token pair. The corpus-based score is included because the user-based co-occurrence counts are sparse and noisy. TextRank was originally proposed for keyword extraction from formal written documents as academic papers or news articles, and uses sentences as the context window for accumulating token co-occurrence frequency. In this graph representation, the context window is each tweet from the target user. In a formal written document, the topic is generally consistent, and the length of both the document and the sentences it contains are long enough for co-occurrence frequency to capture the semantic relations between tokens. However, a user's tweet collection is a casual log of his/her everyday life, which usually lacks a coherent topic. In addition, each individual tweet is short due to the 140-character limit and filled with various types of "noise". Hence, solely relying on context co-occurrence frequency to capture the semantic relations between tokens is not enough.

The token semantic similarity is computed with the cosine distance between their context

[3]From a check of some anecdotal examples, we choose multiplication over a linear combination of the scores for the token pair (user-based co-occurrence frequency and corpus-based semantic similarity).

vectors [99] derived from the Twitter corpus, resulting in a value in the $[0, 1]$ range, where 1 is the most similar. We use tweets as context windows to accumulate the context vector for each token. The token context co-occurrence frequency from a user's tweet collection reflects the user-specific semantic relations between tokens, while the Twitter corpus derived token semantic similarity can be viewed as user-independent background semantic relations between tokens.

PageRank Summarization Algorithm

After the context co-occurrence graph is constructed to represent the target user's tweet collection, we perform the PageRank algorithm to extract the user interest summarization candidates, as described in Section 2.2. To review, each token i has a rank value R_i initialized with an arbitrarily assigned value (e.g. 1.0), this rank value will be updated iteratively according to the following equation,

$$R_i = (1 - d) + d \sum_{j \in \Gamma(i)} \frac{w_{ji}}{\sum_{k \in \Gamma(j)} w_{jk}} R_j \tag{4.1}$$

where w_{ji} is the edge weight between token j and i, $\Gamma(i)$ is token i's predecessor set, and d is a damping factor that is usually set to 0.85 [12]. This iteration continues until convergence.

In the context of Twitter user interests summarization, we can view each token in the graph as both a source and a receiver of semantic information: each token propagates its current amount of semantic information (represented by the rank value) to its neighboring tokens in the graph. The split of the semantic information it propagates to a specific neighboring token is proportional to their edge weight $\frac{w_{ji}}{\sum_{k \in \Gamma(j)} w_{jk}}$, which is determined by their user-specific context co-occurrence frequency and corpus semantic similarity in our method. At the same time, it also receives the semantic information propagated from its neighboring tokens. When the PageRank process converges, the rank value of a token can thus be viewed as the amount of semantic information it eventually receives after ∞ steps of propagation. The ones that receives the most semantic information can be seen as the most representative tokens in the graph, and can therefore be selected as the user interest summarization candidates.

4.4.2 Stage 2: Maximum Graph-cut Summarization

Rank Value Pairwise Graph Representation

To reduce redundancy in the summarization results, we introduce a second stage of processing. After obtaining the PageRank result, we first filter out tokens with weak topic orientation and then select the top tokens with the largest rank values as the summarization candidates to build a new graph representation for the maximum graph-cut algorithm.

Tokens with weak topic orientation are too general to convey specific semantic information about user's interests, such as "new", "stop", "like", which are referred as non-specific tokens below. We keep them in the PageRank process in order to use them as a medium to help propagate semantic information among potential candidate tokens. Since most of them are content words, they are usually not contained in a general stopword list. Such kinds of tokens cannot be filtered by document frequency based measures such as idf either. This is because some of them have relatively high idf values, while some user interest keywords that represent hobbies shared by most users tend to have low idf values, such as "music", "movie" and "game". Document frequency based measures cannot separate these two groups of tokens.

We take advantage of hashtags as a form of metadata in tweets, and develop a strategy to filter out non-specific tokens based on their distribution across hashtags. Hashtags are a community-driven convention for marking the topics of tweets. They start with the "#" symbol, as "#ladygaga", "#vegan". Users add hashtags inline as a way to conveniently categorize tweets. Our assumption is that user interest is closely related with topics, a user interest keyword should mostly correlate only with hashtags on related topics, and thus tends to have a relatively sharp distribution across hashtags; while on the other hand, non-specific tokens generally do not have a strong correlation with any specific groups of hashtags, and thus should have a relatively flat distribution across hashtags. With this assumption, we design a strategy to build a non-specific token list based on token distribution across hashtags.

We group tweets with the same hashtag together to form a hashtag document. For each token t, we can have its count distribution across all hashtag documents as $P_t =$

$(p_1, p_2, ..., p_M)$, where M is the total number of hashtags in the Twitter corpus, and p_m is the probability that token t appears in hashtag document m, computed as

$$p_m = \frac{C(t,m)}{\sum_{m=1}^{M} C(t,m)} \tag{4.2}$$

where $C(t,m)$ is token t's count in hashtag document m. We compute the entropy of each token's hashtag distribution,

$$H_t = -\sum_{m=1}^{M} p_m \log(p_m) \tag{4.3}$$

Tokens with high hashtag distribution entropy are added to the non-specific token list. The size of the non-specific token list we used in this paper is 1k determined by inspecting the token list ranked by hashtag distribution entropy.

After filtering out non-specific tokens, we build a pairwise directed graph for the top-L summarization candidate tokens (L set to be 64 in experiments when available) using the PageRank result. In this graph, each token is represented by one node, and directed edges are added from it to all other candidate tokens in the graph. The edge weight is set to be the source token's rank value produced in stage 1 multiplied by the corpus-based semantic similarity between the two connected tokens, where the rank value serves as the user-specific information and the semantic similarity serves as the corpus background information. Similar to the edge weight setting in stage 1, the token PageRank value from a user's tweet collection reflects the user-specific semantic relations between this token to the other tokens, while the Twitter corpus derived token semantic similarity can be viewed as user-independent background semantic relations between them.

Maximum Graph-cut Summarization Algorithm

With the new graph-representation, we formulate the user interest summarization task as a graph cut problem: we choose top-N candidate tokens with the largest directed graph cut to the rest of the graph as the interest summarization for the target user,

$$\begin{aligned} &\arg\max_{S} \sum_{i \in S, j \notin S} w_{ij} \\ &s.t. |S| = N \end{aligned} \tag{4.4}$$

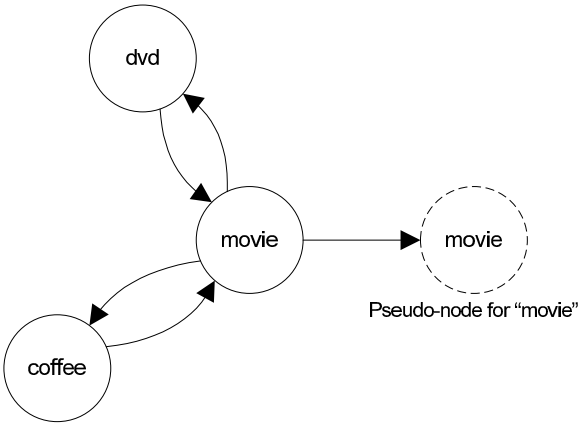


Figure 4.2: Pseudo-token node

where S is the user interest summarization token set, w_{ij} is the weight of the directed edge from token i to token j.

According to the weighting scheme of the graph, tokens with larger rank values and closer semantic connection with other tokens have more chance to be selected as user interest keywords. To account for the token's semantic coverage of itself, we also add a pseudo-token node to each of them, as in Figure 4.2. The pseudo-token node has no outgoing edges, it only has an incoming edge from the corresponding candidate node with the weight set to be the candidate's rank value (i.e., set the semantic similarity between the token and its pseudo-token to be 1). This maximum graph-cut process can also be seen as a semantic information propagation: each token in the graph has accumulated semantic information from the ∞-step propagation in PageRank measured by its rank value, now it will propagate this accumulated semantic information to its neighboring tokens in 1 step during the graph-cut process. The amount of semantic information it propagats to its neighboring tokens is weighted by their semantic similarity to account for their semantic closeness. Tokens in a subgraph that emit the most semantic information to its complement graph are extracted as the final user interest summarization result. Since the graph cut only counts the semantic information propagated from summarization tokens to the rest tokens in the graph, it avoids counting the redundant propagation within the summarization result.

To further penalize redundancy, we adopt the penalizing strategy proposed by [55] for multi-document summarization,

$$\arg\max_{S} \sum_{i \in S, j \notin S} w_{ij} - \lambda \sum_{i,k \in S} w_{ik} \quad s.t. |S| = N \tag{4.5}$$

where λ is a factor deciding the weight of the penalization, which is set to be 1 in experiments. Both Equation 4.4 and 4.5 are submodular functions, their maximization can be solved with greedy algorithm for a near-optimum solution [54].

4.5 Experimental Results

4.5.1 Data

In our experiments, we used the Twitter corpus introduced by Choudhury et al. [15]. This dataset is a focused snow ball crawl of Twitter data performed in late 2009. It consists of approximately 400K users and 29M tweets, whose timestamps of posting span between April 2006 and December 2009.

We extracted phrases and the hashtag-based non-specific token list, and computed the context-vector-based token semantic similarity on this corpus.

We evaluated the proposed user interest summarization algorithm on users with more than 180 tweets in this corpus. A heuristic rule was applied to filter out promotion and spam accounts, where users with more than 20% tweets containing http links or having fewer than 5% replying tweets or retweets are excluded from the evaluation set.

4.5.2 Experimental Settings

We adopted an evaluation measure similar to the one proposed in [126] for identifying advertising keywords on web pages, which emphasizes precision. We randomly selected 169 Twitter users to evaluate the top-N precision of their interest summarization tokens. After we obtained the top-N outputs from the system, three human evaluators were asked to judge whether the output user interest keywords reflected the corresponding user's interests or concerns according to the full set of his/her tweets. Each of them assessed a subset

of the user extraction results for all three systems.[4] We adopted a conservative standard in the evaluation: when a person's name is extracted as a user interest keyword, which is frequent among Twitter users, we judge it as a correct user interest keyword only when it is a name of a famous person (pop star, football player, etc). The percentage of the correct user interest keywords among the top-N selected output corresponds to the top-N precision of the system.

4.5.3 Results and Analysis

To evaluate the effectiveness of each component (Figure 4.1) in our Twitter user interest summarization system, we compared the results from the following three types of outputs: outputs after the PageRank summarization in stage 1, outputs after the non-specific token filtering (+ Non-specific token filtering) and outputs after the maximum graph-cut in stage 2 (+ GraphCut). Their top-N precision results are presented in Table 4.2. It is shown that the non-specific token filtering is very effective, it greatly improves the user interest summarization precision after PageRank by filtering out tokens that are too general to reveal a user's interests. This confirms our previous assumption that user interest tokens tend to have closer association with certain hashtags than other tokens, which can be used to extract the non-specific token list. The final output of our system achieves a top-N precision which greatly outperforms the system reported in [122] (top-1, 2, 5 and 10 precision as 67.3%, 66.0%, 63.0% and 58.3%, respectively), whose preprocessing baseline is based on part-of-speech tagging.

After applying maximum graph-cut, we obtained further improvement on top-1 to top-5 output, and kept comparable result on the top-10 output. This shows that although maximum graph-cut is mainly designed for reducing redundancy in the user interest summarization result, it can bring precision improvement for higher ranked user interest keywords. By checking some anecdotal examples, we hypothesize that this is because the diversity criterion in the maximum graph-cut summarization can help filter out some of the left-over non-specific tokens that co-exist with the correct user keywords. To evaluate the effective-

[4]The pairwise Kappa value for inter-evaluator agreement ranged from 0.62-0.65, showing substantial agreement.

Table 4.2: Top-N Precision of Twitter User Interest Summarization Results from Different System Components

Precision (%)	top-1	top-3	top-5	top-10
PageRank	39.1	37.3	38.5	43.3
+ Non-specific Token Filter	84.6	81.9	80.7	76.3
+ GraphCut	85.8	84.0	81.8	76.2

ness of the redundancy reduction by maximum graph-cut, we computed and compared the average pairwise semantic similarity[5] among the top-N user interest summarization results before and after maximum graph-cut is applied. The result is presented in Table 4.3. It is shown that maximum graph-cut can effectively reduce the redundancy among the user interest summarization result, on the top-10 result it achieves a semantic similarity reduction among the output up to 14.2%.

Table 4.3: Average Pairwise Semantic Similarity among the Top-N keywords of Twitter User Interest Summarization Results

Average Semantic Similarity	top-3	top-5	top-10
+ Non-specific Token Filter	0.052	0.089	0.167
+ GraphCut	0.047	0.082	0.144
Reduction	9.6%	7.2%	14.2%

4.6 Summary

In this chapter, we designed a system to automatically summarize Twitter user interests by extracting key words and phrases from his/her tweets. In this system, we introduce graph representations of a user's tweet collection that combine user-specific semantic information

[5]We use the same context-vector-based semantic similarity presented in previous sections, which has a range of 0 to 1.

and corpus-based background semantic information, and propose a two-stage graph-based summarization algorithm that cascades PageRank and maximum graph-cut algorithms to effectively incorporate the representativeness and diversity criteria in the summarization result. We also designed a strategy to filter out uninformative tokens for user interest based on their distributions across hashtags. The proposed system achieves promising results in experiments, with a top-5 precision up to 81.8%.

Since the graphs for both PageRank and maximum graph-cut algorithms are built with tweets from one user, their size is relatively small. Further, because both PageRank and maximum graph-cut algorithms are efficient in computation, the memory requirement and computational load for extracting one user's interests is relatively low. In addition, the algorithm process of extracting each user's interests is independent of each other, thus they can be run in parallel on different machines. Hence, the system has good scalability for handling a large number of user interest extraction tasks.

Chapter 5

ACTIVE LEARNING FOR SEMANTIC ORIENTATION CLASSIFICATION

Semantic orientation classification is an important technology for opinion mining over the large amount of user generated text on the internet, such as on-line forums, blogs, review web sites and tweets. It can be used either by product or service providers to obtain instant consumer feedbacks or by consumers to make well-informed shopping decisions. Most previous research treated semantic orientation classification as a supervised machine learning problem, applying text classification algorithms such as naive Bayes, maximum entropy models and support vector machines [84], which typically require a large amount of labeled data to train a model with high accuracy. While there are some genres for which semantic orientation labels are easy to collect (e.g. reviews with user ratings), there are many others for which there is little user-rated data. However, it can be expensive and time consuming to annotate training data. To deal with this issue, researchers have sought ways to minimize hand labeling efforts through active learning [17, 18], which aims to choose data to label that is expected to provide the biggest improvement in system performance. Active learning starts with a base model trained with a small labeled set and uses a query strategy to select the most informative data samples (referred to here as instances) from the unlabeled set. These instances are hand-labeled and added to the training set, and then the process may be repeated to add another batch of data. The key to active learning is the query strategy, which has received much attention in previous work; readers can refer to [100] for a detailed summary. The most commonly used query strategies choose individual instances that have the highest label uncertainty, e.g. as measured by entropy [101].

The strategy of choosing individual instances ignores the potential relatedness between samples. In this work, we take advantage of a graph-based representation – specifically an instance-word bipartite graph – to characterize the relatedness between data instances

for semantic orientation classification. In this bipartite graph, we use words as a medium to propagate semantic orientation information between data instances in order to explore the relatedness between them. Using the graph, we introduce two new query strategies that leverage relatedness. In one case, the relatedness is used to improve the uncertainty estimates associated with individual instances, tightly integrating semantic orientation label propagation as a semi-supervised learning algorithm with active learning. In the second case, the relatedness is used in a batch query to extract instances with both high uncertainty and diversity. In either case, the resulting models learned from the hand-labeled data can be integrated with semantic orientation label propagation in a final stage. In general, the proposed graph-based active learning framework has the advantages that it is easy to incorporate prior lexical semantic orientation information and it is simple to compute in a parallel framework such as MapReduce [24].

The rest of this chapter is organized as follows, Section 5.1 gives a literature review of work on semantic orientation classification and active learning; Section 5.3 presents the instance-word bipartitle graph representation we applied on semantic orientation classification corpora and an outline of the proposed active learning framework based on that; Section 5.4 introduces the label propagation algorithm in the context of semantic orienation classificaiton, and Section 5.5 presents the two new graph-based active learning query strategies. Experiments are presented and analyzed in Section 5.6, followed by a summary of this chapter in Section 5.7.

5.1 Background

Semantic orientation classification is a well-studied topic, with related work falling into two groups that leverage polarity lexicons [113, 47, 91, 117] vs. machine learning [84, 82]. This work falls into the second group. Previous research on machine-learning-based semantic orientation classification have explored features [93, 33, 69, 76], classifiers [104, 26, 72], and domain adaptation [9, 63]. Our work focuses on active learning.

The key to active learning is the query strategy, i.e., the criterion to evaluate the informativeness of data instances in order to select the most informative ones to label. A brief summarization of the query strategies explored in the existing literature is presented here.

The most commonly used query strategy is *uncertainty sampling*, which selects instances with the most uncertain result under the current model. For a model that produces a probability for each classification hypothesis, the uncertainty measure can be *least confident* (minimum top 1 result probability) [21], *margin sampling* (minimum gap between top 1 and top 2 result probability) [101], or *maximum entropy* of the hypothesis distribution [108, 14, 94, 134].

Version space reduction is another group of query strategies that has received extensive study. It selects instances that once being labeled can lead to the maximum reduction of the model's version space [116]. For a support vector machine, this is equivalent to selecting instances which are closest to the decision boundary [98, 109, 110]; for a Bayesian classifier of a binary problem, it is to select the instance whose posterior probability produced by the current learner is closest to 0.5 [50]. In some cases, these methods are equivalent to uncertainty sampling. Other than specializing this criterion for specific types of learners, another way to implement version space reduction selection is to create a committee of several learners, and select instances which have the largest disagreement within the committee. *Query-by-Committee* constitutes the committee by randomly sampling from the probability distribution of the model, i.e. the version space, resulted from the labeled training data. It has been applied to learners as perceptrons [32] and Naive Bayes [65]. *Multi-view active learning* generates the committee by building several learners with different sets of features, which was introduced as Co-Testing in [70, 71]. *Active-Decorate* [66] generates the committee by successively augmenting the training set with artificially-generated instances according to the distribution of the data space, and label instances with largest disagreement in the current committee to train a new model and adding it to the committee.

Other widely employed query strategies includes *expected error reduction*, which builds a loss function to estimate the error rate and selects instances that minimize the expected error once it is labeled and added to the training set [95, 137, 35], and *expected model change* which aims to select instances that once being labeled would bring the greatest change to the current model. For machine learning models trained with gradient ascending/desending, as conditional random field or maximum entropy, the expected gradient length can be used as the measure for model change [101]. Co-EM [78, 11] can also be integrated into active

learning as proposed in [44, 111].

Diversity and *density* are two query strategies that take the data distribution into consideration. They are generally applied together with one or several of the other query strategies introduced above. The diversity criterion selects instances from different region of the data space to ensure their representativeness. It can be implemented by selecting instances with the largest distance to the labeled data set [13, 6], or by clustering the unlabeled data first and then selecting instances from each of these clusters [108, 77, 103, 46]. The density criterion selects instances from the densely distributed region of the data space, so that labeling it can influence the classification results of more data instances. When a pre-clustering is performed, the density can be measured by the inverse of the average intra-cluster distance [108, 77]. It can also be implemented as selecting instances with maximum mutual information with the rest of the unlabeled data [46, 35].

In active learning, the relatedness of individual instances can be accounted for by retraining the model after each instance is labeled, but this is not practical in most applications. Instead, instances are usually added to the supervised set in a batch. Batch-mode active learning was first explored by Hoi and colleagues [39, 38], maximizing the Fisher information of the queried batch. Here, we use the graph structure to account for relatedness between instances in a batch query strategy that moves beyond prior work by integrating uncertainty with diversity and density.

The first query strategy we proposed is inspired by [51] which uses the maximum entropy model to regularize label propagation for query classification. In contrast to [51], which designs a co-training based semi-supervised learning strategy, we are aiming for an active learning framework. This query strategy is related to the *expected gradient length* query proposed for a conditional random field (CRF) in [102]. The latter uses the posterior probability produced by the CRF to compute the expectation of the gradient length for the CRF itself, while we take advantage of the framework that combines the maximum entropy model and label propagation, and use the label propagation result to estimate the gradient length.

The second query strategy we proposed is designed for batch-mode active learning settings. Batch-mode active learning was first explored by Hoi and colleagues [39, 38], max-

imizing Fisher information of the queried batch. Different from Hoi's work, our approach uses a graph-based corpus representation, which makes it easier to be incorporated with label propagation or extended for parallel computation. Lin and Bilmes [54] studied batch mode active learning with submodular graph functions for the problem of training hidden Markov models for speech recognition. In addition to differences in the structure of the graph (associated with differences in the applications), our approach differs in that it incorporates uncertainty, representativeness and diversity criteria as compared to the approach in [54] which is mainly designed for representativeness.

Semi-supervised learning [135] provides another mechanism for leveraging large amounts of unlabeled data in combination with some labeled data. For example, graph-based semi-supervised learning can be employed by applying label propagation [136, 131] over a graph built according to relatedness between data instances in the corpus [104], and passing the labeling information from labeled data instances to unlabeled ones. Zhu *et al.* [137] combined active learning with graph-based semi-supervised learning strategy using a Gaussian field to enhance an expected risk query strategy, assessed on both topic classification and other tasks. A key difference in our approach vs. [137] (and vs. earlier work on active learning integrated with EM-based semi-supervised learning [64, 27]) is the use of model combination: a maximum entropy model [79] trained from labeled data provides a regularizing prior in combination with label propagation for graph-based semi-supervised learning.

Other researchers have applied both semi-supervised and active learning to semantic orientation classification over the Amazon product review dataset, using an SVM [23] and Active Deep Network [133]. We obtain higher accuracy, but the results are not directly comparable since their work is on an early (smaller) version of that data set. While their work did obtain larger relative gains due to active learning, our work starts with a much better baseline (as well as consistently higher final performance). We claim that starting from a better baseline provides a stronger result. On the Amazon review dataset, our baseline achieves comparable performance with the result presented in [10], when the same amount of training data are used.

5.2 Base Classification Model

The semantic orientation of a piece of text (e.g. sentence or document) is expressed through the semantic orientation of words in it. It can be viewed as a collaborative result of sentiment evaluative characteristics of words, especially for words with strong semantic orientation, as shown by examples in Figure 5.1.

> Yes, it is time to **appreciate** again James Cameron's **glorious epic**, for its scope, **underappreciated** script and **perfect** casting.
>
> Simply one of 20th century cinema's **greatest**, old-fashioned love stories, full of still **impressive** effects (pre 3D) and **decent**, if theatrical acting
>
> Rose and Jack are **burdened** with **simplistic** characterisation and **atrocious** dialogue.

Figure 5.1: Examples of lexical semantic orientation in sentence semantic orientation, where red font indicates words with positive semantic orienation and blue font indicates words with negative semantic orienation.

A simple but effective model for text semantic orientation classification is the "bag-of-words" assumption, in which a sentence or document is represented as an unordered collection of words, disregarding grammar and word order. It views the semantic orientation of a piece of text as an accumulated result of lexical semantic orientation. The success of the "bag-of-words" assumption in semantic orientation classification has been demonstrated in a series of previous literature [84, 69, 117].

In this work, we also adopt this assumption and use maximum entropy model [79] as our base model for semantic orientation classification. The maximum entropy model is a discriminative classification model, formulated as:

$$P(y|x;\lambda) = \frac{\exp(\sum_k \lambda_k f_k(x,y))}{\sum_y \exp(\sum_k \lambda_k f_k(x,y))}, \tag{5.1}$$

where $f_k(x,y)$ is the k-th feature of the model, and λ_k is the model parameter associated with it. The parameters of the maximum entropy model are estimated using gradient ascent with L_2 regularization.

In this work, we use unigram and bigram occurrences as features for the maximum entropy model, which are widely used for semantic orientation classification. We filter out unigrams and bigrams with frequency smaller than a given threshold (< 4) in the corpus, and remove stopwords. To handle negations, we use the heuristic proposed in [84], where a "NOT_" prefix is added to all unigrams occurring between a negation word and the closest punctuation following it.

5.3 Bipartite Graph Representation

The "bag-of-words" assumption also provides us a way of using graphs to represent text, in which a sentence or document can be represented by a node connecting to a group of words that appear in it. A simple example of the graph representation of a sentence (S1) with positive semantic ordination is illustrated in Figure 5.2. Note stopwords have been removed before constructing this graph representation. Undirected edges in the graph indicate that the semantic orientation information can be propagated in both directions: according to the "bag-of-words" assumption, lexical semantic orientation forms the semantic orientation of a sentence, which corresponds to the semantic orientation information propagation from word nodes to the sentence node. On the other hand, the semantic orientation of a sentence constrains its choice of words, e.g. a sentence with positive semantic orientation tends to contain positive words, which corresponds to the semantic orientation information propagation from the sentence node to word nodes.

With the graph representation of a sentence or document, we can construct a bipartite graph representation of a text corpus. As shown in Figure 5.3, the left side nodes represent instances (sentences or documents), the right side nodes represent words. If a word occurs in an instance, an edge is added between them. The edge weight can be set according to prior knowledge of the word's discriminative power for semantic orientation classification. In experiments presented in this chapter, we set the weight of all edges as 1 due to lack of sufficient prior knowledge on semantic orientation of words. Assuming there are m instances and n unique words in the corpus, the graph can be represented by an $(m + n) \times (m + n)$ adjacency matrix. Due to the bipartite structure, two blocks in this matrix will have zero

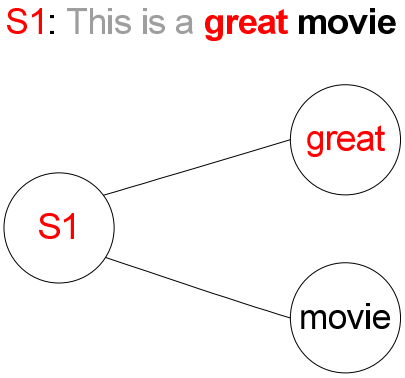


Figure 5.2: Text graph presentation with "bag-of-words" assumption (stopwords are removed in the graph representation)

values as shown below,

$$\begin{bmatrix} 0 & W \\ W^T & 0 \end{bmatrix} \tag{5.2}$$

where W is an $m \times n$ matrix representing the connection weight from m instances to n words.

This bipartite graph representation enables us to build a graph-based active learning framework that captures the relatedness between instances in the corpus by using words as a medium to propagate semantic orientation information among the instances. With this graph, we can view each node (including instance nodes and word nodes) as both a source and a receiver of semantic orientation information and design active learning query strategies based on the semantic orientation information propagation result. We propose two active learning query strategies for semantic orientation classification. The first query strategy integrates label propagation as a semi-supervised learning method into the active learning process. We use label propagation to fully propagate the semantic orientation information from labeled instances to unlabeled ones, and use the result to query instances that are expected to bring the most improvement to the maximum entropy model. The second query strategy is specifically designed for a batch-mode active learning by incorporating uncertainty criterion with the representativeness and diversity criteria. It queries a batch of

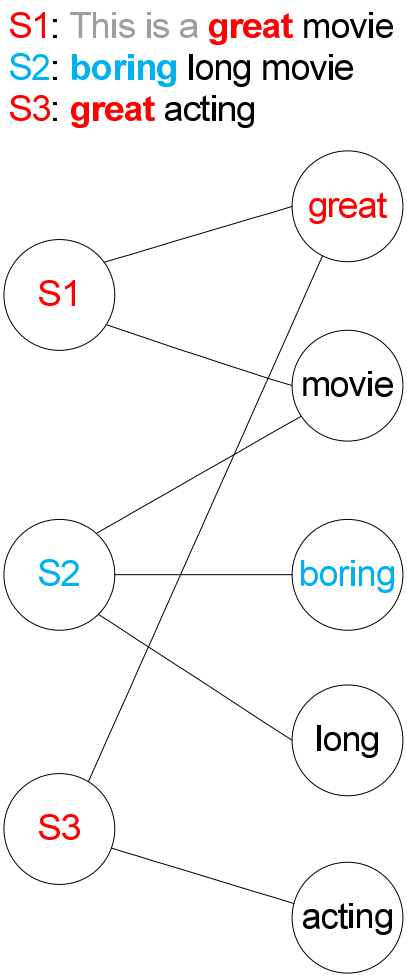


Figure 5.3: Bipartite graph presentation of the text semantic orientation classification corpus

instances that once being labeled will propagate the most semantic orientation information to the rest of the unlabeled instances in the corpus. Both of the two active learning query strategies share the same graph representation and algorithm infrastructure as the label propagation algorithm, which makes it convenient to integrate the actively learned models with label propagation in the classification stage.

5.4 Semantic Orientation Label Propagation

Label propagation [136, 131] is a graph-based semi-supervised learning strategy, which has been studied in a number of works for various machine learning problems [132, 5, 107, 51, 120]. It is an important component in our proposed graph-based active learning framework. We use label propagation to propagate semantic orientation information from labeled instances or words to unlabeled ones. It serves as a key part in one of our proposed active learning strategy, and will also be concatenated to the actively learned models in the classification stage to obtain further gain through this semi-supervised learning process. This section gives a detailed introduction to our adoption of label propagation for propagating semantic orientation information among instances in the corpus.

5.4.1 Label Propagation

Graph-based data representations have been studied extensively in previous work on label propagation as a semi-supervised learning method [136, 131]. Early work was based on homogeneous graphs, where only instances are represented in the graph according to their nearest-neighbor relations. Later studies applied it on bipartite graphs for various applications, including Youtube video recommendation (user-video graph) [5] and query classification (query-link graph) [51, 120]. The instance-word bipartite graph data representation used in our work is inspired by these studies and other work on semi-supervised semantic orientation classification [104]. Below we gives a brief introduction to the label propagation algorithm in the context of the semantic orientation classification.

Given the instance-word bipartite graph presented in the previous section, according to label propagation, each node i has an adjustable semantic orientation class "likelihood" vector I_i (for an instance node) or F_i (for a word node), where each element $I_{i,y}$ and $F_{i,y}$ is

Algorithm 1 Label Propagation

Initialize $I^{(0)}$ and $F^{(0)}$;

repeat

$F^{(n)} = \alpha\tilde{W}^T I^{(n-1)} + (1-\alpha)F^{(0)}$;

$I^{(n)} = \alpha\tilde{W}F^{(n-1)} + (1-\alpha)I^{(0)}$;

until $\|I^{(n)} - I^{(n-1)}\| < \epsilon$

Normalize $I^* = I^{(n)}$ and output the semantic orientation posterior probability $P_{prop}(y|x)$ for each instance node.

a non-negative real number that when normalized gives the posterior probability of a node belonging to a semantic orientation class y. Let I denote a matrix with the i-th row as I_i and $I^{(0)}$ as the semantic orientation class prior used to initialize I, and similarly for F, F_i, and $F^{(0)}$. The label propagation process can then be defined as shown in Algorithm 1, where $\tilde{W} = D^{-1/2}W(D^{-1/2})^T$ is the normalized adjacency matrix from the instance nodes to word nodes and D is a diagonal matrix with its (i,i) element equal to the sum of the i-th row of W. According to this algorithm, in each iteration $I^{(n)}$ and $F^{(n)}$ are updated as a weighted sum ($0 < \alpha < 1$) of the propagation from the node's neighbors and its prior. The iteration continues until it converges, and the normalized semantic orientation class vector of an instance node gives the posterior semantic orientation distribution of that instance.

This label propagation process is an efficient computation that is convenient to implement in a MapReduce system for parallel programming, which enables it to handle large data corpora.

5.4.2 Regularizing Semantic Orientation Label Propagation with Prior Knowledge

As shown by [131], the label propagation defined in Algorithm 1 will converge to the optimum result by minimizing the objective function

$$\begin{aligned} Q(N) &= \alpha \sum_{i,j=1}^{n+m} w_{i,j} \| \frac{1}{\sqrt{D_{ii}}} N_i - \frac{1}{\sqrt{D_{jj}}} N_j \|^2 \\ &\quad + (1-\alpha) \sum_{i=1}^{n+m} \|N_i - N_i^{(0)}\|^2 \end{aligned}$$

where N_i represents node i's semantic orientation class likelihood vector (i.e., it is I_i for an instance node and F_i for a feature node). The first term in this objective function is the smoothness constraint, preferring a result that does not change too much between neighboring nodes, i.e. instances sharing the same words tend to have the same semantic orientation. The second term can be seen as a regularization term; it prefers a result that does not deviate too far from its semantic orientation prior. With proper priors, this regularization can help prevent propagating errors through the graph [51].

In the semantic orientation classification experiments, we use the maximum entropy model result and lexicon semantic orientation knowledge to regularize the label propagation. The two types of nodes in the bipartite graph (instances and words) are treated differently when setting the priors for label propagation:

For the instance nodes, the label propagation is regularized by a maximum entropy model. Specifically, if the instance is unlabeled, its prior $F^{(0)}(x)$ is set to be the posterior semantic orientation distribution $P(y|x;\lambda)$ produced by the maximum entropy model; otherwise, it is set as their label indicator vector $l(y|x)$.

For the word nodes, the setting of their priors provides a chance to introduce an outside source of lexicon semantic orientation information. The word node's prior can either be set with a knowledge-based strategy, using a polarity lexicon from a human-edited semantic orientation dictionary, or with a data-driven strategy, using a polarity lexicon obtained by mining on-line resources [117]. In this work, we adopt the knowledge-based strategy. We use the *Inquirer*[1] semantic orientation lexicon list, which contains 1,915 positive words (*Inquirer Positiv* category) and 2,291 negative words (*Inquirer Negativ* category). For words on this list, we set the semantic orientation prior as their semantic orientation label indicator vector. For words outside this list, they are initialized with the uniform distribution and then updated with the label propagation semantic orientation posterior obtained in the previous active learning iteration.

---

[1] `http://www.wjh.harvard.edu/~inquirer`

5.5 Graph-based Active Learning Query Strategies

Most traditional uncertainty-based active learning query strategies are optimized for selecting one instance at a time, ideally the model should be retrained after querying every single instance. However, this will lead to too many active learning iterations (a query→label →retrain cycle), which is usually not practical in application scenarios. First, many machine learning models do not support on-line training or perform less effectively under on-line training settings, and frequently retraining the model with all labeled data can be expensive. Second, small sample active learning iterations may pose difficulties for managing annotator working hours. Hence, in our proposed framework, the active learning is carried on in a batch mode, where a group of instances are queried at each active learning iteration.

Let L be the labeled set of instances, U be the unlabeled instance pool, λ be the maximum entropy model and G be the graph representation. The batch mode active learning will select a batch set B for human labeling from U according to a query strategy $\psi(B, \lambda, G)$. We assume that the batch size at each algorithm set is $|B| = N$. The overall active learning process is described in Algorithm 2. We propose two graph-based query strategies within this framework: i) maximum gradient length, and ii) batch network gain, the latter is specifically optimized for this batch mode active learning. Label propagation based semi-supervised learning is incorporated in the query strategy when the maximum gradient length approach is used. It can also be used in the classification stage for either approach with the maximum entropy model providing regularization.

5.5.1 Maximum Gradient Length Query

The first active learning query strategy we proposed is the *maximum gradient length* query. It incorporates label propagation as a graph-based semi-supervised learning method to leverage large amounts of unlabeled data to help select the most informative instances for the maximum entropy model. This query strategy views labeled instances and words as the original sources of semantic orientation information. During each active learning iteration, we apply label propagation on the bipartite graph representation of the given corpus, which propagates semantic orientation information from currently labeled instances

Algorithm 2 Graph-based Active Learning Framework

Given: the initial labeled set L, the unlabeled set U, the graph G, the query strategy $\psi(B, \lambda, G)$, and batch size N;

repeat

// train maximum entropy model λ

$\lambda = train(L)$;

// find B according to the query strategy

$B^* = \text{argmax}_{|B|=N, B \subset U}\, \psi(B, \lambda, G)$;

$L \leftarrow L \cup B^*$; $U \leftarrow U - B^*$;

until meet the labeling budget

// Train model with labeled samples

$\lambda = train(L)$

and words to unlabeled ones. The propagation results on the unlabeled instances are used to help query the instances that once being labeled will bring the largest improvement to the maximum entropy model. We assume that the instances that bring large changes to the current model tend to improve it the most. For a maximum entropy model, parameters are estimated by maximizing the log likelihood $\mathcal{L}(y|x;\lambda) = \log P(y|x;\lambda)$ with gradient ascent, thus the gradient length induced by an instance can be used as measure of the change it brings to the model during the training process. The gradient of the maximum entropy model log likelihood induced by instance x is computed as

$$\frac{\partial \mathcal{L}(y|x;\lambda)}{\partial \lambda_k} = E_{\tilde{P}(x,y)}[f_k(x,y)] - E_{P(y|x;\lambda)}[f_k(x,y)] \tag{5.3}$$

where $\tilde{P}(x,y)$ is the empirical distribution of the data and $E_P[\cdot]$ denotes expectation with respect to distribution P. Thus

$$E_{\tilde{P}(x,y)}[f_k(x,y)] = \sum_y l(y|x) f_k(x,y) \tag{5.4}$$

where $l(y|x)$ is the label indicator of instance x. Then we can select instances that has the largest the gradient length $\| \bigtriangledown \mathcal{L}(y|x;\lambda) \|$ to query for their labels.

However, for unlabeled instances, we do not know their true labels to estimate the empirical distribution $E_{\tilde{P}}(x,y)[f_k(x,y)]$. Here we propose to approximate $E_{\tilde{P}}(x,y)[f_k(x,y)]$ with the label propagation output $P_{prop}(y|x)$. We have two options to utilize the label propagation result in computing the maximum entropy model gradient: First, we can use the posterior classification distribution $P_{prop}(y|x)$ produced by label propagation as a *soft decision* to replace $l(y|x)$ in Equation 5.4:

$$E_{\tilde{P}_{prop}(x,y)}[f_k(x,y)] = \sum_y P_{prop}(y|x) f_k(x,y) \tag{5.5}$$

Note that $\| \bigtriangledown \mathcal{L}(y|x;\lambda) \|$ computed with Equation 5.5's approximation corresponds to the length of the expected gradient, which is not strictly the same as the expected gradient length used in [102], which would use the posterior distribution of the current maximum entropy model to estimate the gradient length. Alternatively, we can use the classification decision indicator vector $D(P_{prop}(y|x))$ of label propagation as a *hard decision* to replace $l(y|x)$ in Equation 5.4:

$$E_{\tilde{P}_{prop}(x,y)}[f_k(x,y)] = \sum_y D(P_{prop}(y|x)) f_k(x,y) \tag{5.6}$$

With the estimated gradient length $\| \bigtriangledown \mathcal{L}(y|x;\lambda) \|$, we can select top-$N$ instances (indicated using the N_argmax notation) with *maximum gradient length* to query for their labels according to algorithm 3, i.e. $\psi(B,\lambda,G) = \sum_{x \in B} \| \bigtriangledown \mathcal{L}(y|x;\lambda) \|$.

Algorithm 3 Maximum gradient length query strategy

Given: labeled set L, unlabeled set U, and batch size N

// Run regularized label propagation over $L \cup U$

$P_{prop}(y|x) = propagation(L \cup U | P(y|x;\lambda))$;

// Map unlabeled samples to soft or hard labels $l(y|x)$

$l(y|x) = P_{prop}(y|x)$ or $D(P_{prop}(y|x))$;

$B = N\_\text{argmax}_{x \in U} \| \bigtriangledown \mathcal{L}(y|x;\lambda) \|$;

5.5.2 Maximum Batch Network Gain Query

The maximum batch network gain query is a query strategy optimized for the batch mode active learning; it combines representativeness and diversity criteria with the traditional uncertainty criterion. Representativeness and diversity are two complementary criteria to the uncertainty criterion, which take into account the interdependence between instances in the corpus. For example, an instance with high classification uncertainty can also be an outlier point, in which case labeling it will give little help for classifying other instances. Hence, we would like to have the labeled instances be representative in the corpus, i.e., choosing instances located in an area of the feature space with densely distributed instances. In batch mode active learning, we also want to increase the instance diversity in the queried batch to reduce labeling redundancy. The proposed maximum batch network gain query incorporates the two criteria and selects the batch set by maximizing the network gain associated with labeling instances in the batch, leveraging the bipartite graph representation. This query strategy views instances in the queried batch as the original source of semantic orientation information. In each active learning iteration, it selects a batch of instances that once being labeled will propagate the most semantic orientation information to the rest of the unlabeled instances in the corpus.

For each unlabeled instance i in S, we use its semantic orientation class distribution entropy to measure the *individual gain* of semantic orientation information by querying its label:

$$H(i) = -\sum_{y} P(y|x_i;\lambda)\log(P(y|x_i;\lambda)), \tag{5.7}$$

where $P(y|x_i;\lambda)$ is i's semantic orientation class distribution produced by the maximum entropy model.

Inspired by label propagation, we approximate the gain of querying a batch by propagating the *individual gain* of querying instances in it through the graph to other unlabeled instances. We compute the effective weights between instances $a_{ji} = [\tilde{W}\tilde{W}^T]_{ji}$, where $\tilde{W}$ is the normalized adjacency matrix from the instance nodes to feature nodes. In other words, a_{ji} sums all normalized paths from queried instance i to unlabeled instance j in the bipartite graph, incorporating all words they have in common. Computing a_{ji} is analogous

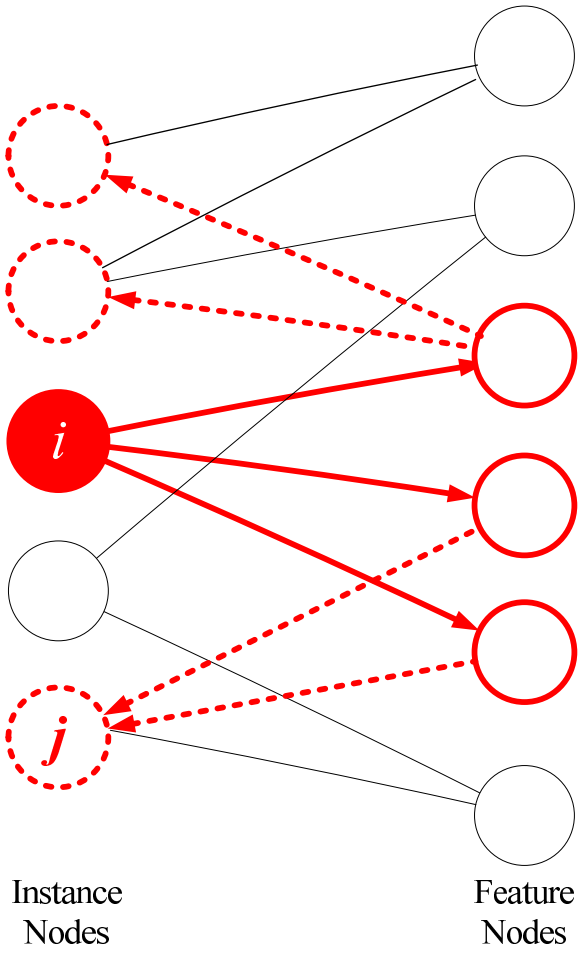


Figure 5.4: Propagation of semantic orientation information gain through the bipartite graph.

to one step of label propagation, as shown in figure 5.4, the words are used as a medium to propagate the gain from labeling individual instances.

The effective weights between instances can be used to evaluate their relatedness. With this idea, we define the *batch network gain* $NG(B)$ as:

$$NG(B) = \sum_{j \in U-B, i \in B} a_{ji} H(i) - \mu \sum_{i,k \in B, i \neq k} a_{ki} H(i) \tag{5.8}$$

where the first term represents the impact of the information gain from querying the batch on the rest of the unlabeled set, i.e., the semantic information propagated from nodes in the

queried batch to the rest of the unlabeled instances, and the second term compensates for the redundancy between nodes in the batch with penalty term μ. For experiments in this paper, we set $\mu = 1$. The batch network gain criterion integrates uncertainty (through the use of the node entropy term $H(\cdot)$) with representativeness (via the first term) and diversity (via the second term).

In each active learning iteration, we select the batch with maximum $NG(B)$ to query for their labels, i.e. $\psi(B, \lambda, G) = NG(B)$. To solve this maximizing problem, we can see $A = [a_{ij}]$ as an adjacency matrix of a new graph consisting of only the instances nodes, then maximizing the first term of Equation (5.8) is equivalent to maximize the graph cut between B and $U - B$ given B's size. This is a submodular problem [74], which can be solved near-optimally with a greedy algorithm [54]. Similarly, Equation (5.8) is also submodular, therefore we can select the active learning batch with the following greedy algorithm 4.

Algorithm 4 Maximum batch network gain query

Given: unlabeled set U, batch size N

Initialize $B = \emptyset$;

repeat

$k = \text{argmax}_{i \in U} NG(B \cup \{i\})$

$B \leftarrow B \cup \{k\}$, $U \leftarrow U - \{k\}$

until $|B| = N$

5.6 Experimental Results

To assess the effectiveness of the proposed active learning framework, we applied it using two semantic orientation review datasets, including the document-level Amazon product review dataset that includes multiple domains, and the sentence-level movie review dataset. Two sets of experiments were run to study different configurations of the two proposed graph-based query strategies. For conciseness, only results on the most difficult task (Amazon book reviews) are presented. Then we present trends for the proposed system compared to several contrasting cases over various domains from the two semantic orientation review

datasets.

5.6.1 Data

1. Amazon product reviews:

The Amazon product review dataset is the "Multi-domain semantic orientation dataset v2.0" introduced in [10]. This dataset consists of Amazon product reviews from 4 domains, including books, dvds, electronics, and kitchen. Each domain contains 5k-6k user reviews with balanced positive/negative classes. In our experiments, we conducted a 5-fold cross validation for each of the 4 domains. In each fold, we divided the domain dataset into a test set (1/5 of the dataset documents), an initial training set (500 documents), and an active learning instance pool with the rest of documents in this corpus. The batch size for each active learning iteration is 250.

2. Movie reviews:

The movie review dataset is the "sentence polarity dataset v1.0" introduced in [83]. This dataset consists of sentences extracted from `http://www.rottentomatoes.com/` user reviews. It has two tasks, the positive/negative classification task and the subjective/objective classification task, each has 10k sentences with balanced classes. In our experiments, we performed a 5-fold cross-validation for each of the two tasks. In each fold, we divided the task dataset into a test set (1/5 of the dataset sentences), an initial training set (1k sentences), and an active learning instance pool with the rest of sentences in this corpus. The batch size for each active learning iteration is 500.

5.6.2 Maximum Gradient Length Query: Settings for Estimating Gradient Length

For the maximum gradient length query proposed in Section 5.5.1, we compare there possible settings to estimate the gradient length in this query strategy: *soft decisions* (Equation 5.5), *hard decisions* (Equation 5.6), and *maxent prob* which is to use the posterior distribution produced by the current maximum entropy model instead of label propagation to estimate the gradient length. The results on the positive/negative task in the Amazon book review domain are presented in Figure 5.5, with a comparison to random selection as a baseline.

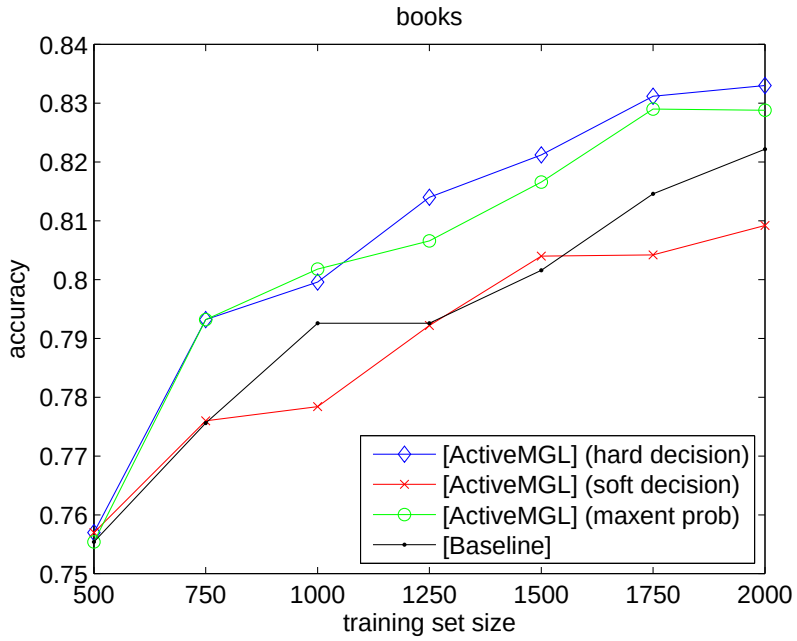


Figure 5.5: Performance using *maximum gradient length* query with *soft decisions*, *hard decisions* and *maxent prob* compared to the random baseline.

It is shown that estimating the gradient length from label propagation with hard decisions significantly outperforms the baseline, while the soft decision version obtains performance worse than the baseline. To study the reason for the soft decision's failure, we analyzed the label propagation results from the first active learning iteration. Let $P_{prop}(y = 1|x)$ denote the posterior probability of being positive semantic orientation produced by label propagation. In Figure 5.6, we show the relative frequency of positive semantic orientation labels for different bins of the posterior $P_{prop}(y = 1|x)$. An unbiased posterior would have relative frequencies that match the diagonal line. We see that label propagation tends to converge with distributions that are not confident and thus biased, leading to poor estimates of the gradient length. The bias is such that the decisions associated with these posteriors are more often correct than predicted, which may explain why hard decisions being more effective than soft decisions.

Figure 5.5 also shows that estimating the gradient length from label propagation with

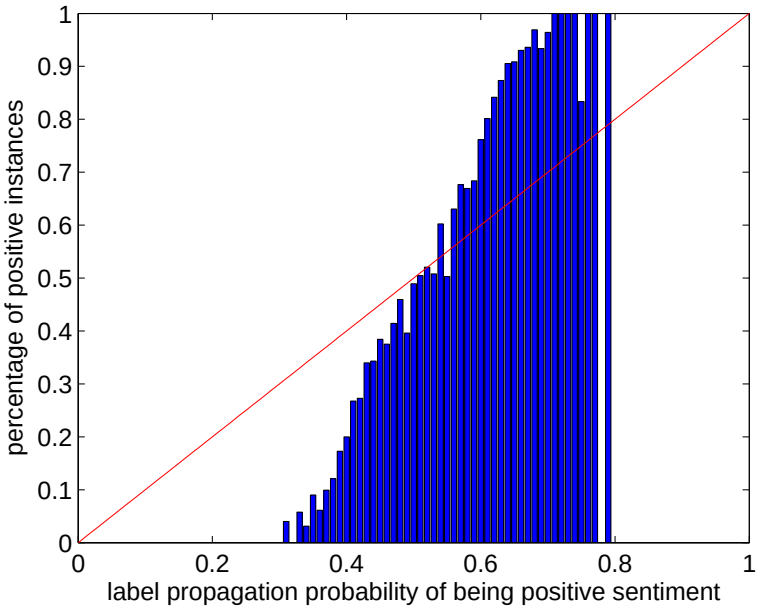


Figure 5.6: Bias of posterior probability produced by label propagation.

hard decisions achieves better performance than with the maximum entropy model posterior distribution. This is because the former combines the outputs from both the label propagation and maximum entropy model, while the latter only uses the output from the maximum entropy model along and thus has less accurate judgement on sentiment classes of unlabeled instances when estimating the gradient length.

Hence, in subsequent experiments, we only present results using the maximum gradient length query with a hard decision.

5.6.3 Maximum Graph-cut vs. Maximum Batch Network Gain

Recall that the *maximum batch network gain* query integrates the uncertainty criterion with the representativeness and diversity criteria in the query strategy by propagating the individual instance gain $H(i)$ through the graph (equation 5.8). If we modify Equation (5.8) setting $H(i)$ to be 1 for all instances in the queried batch, we eliminate the uncertainty component of the criterion and obtain a simple "diversity and density" criterion that is the

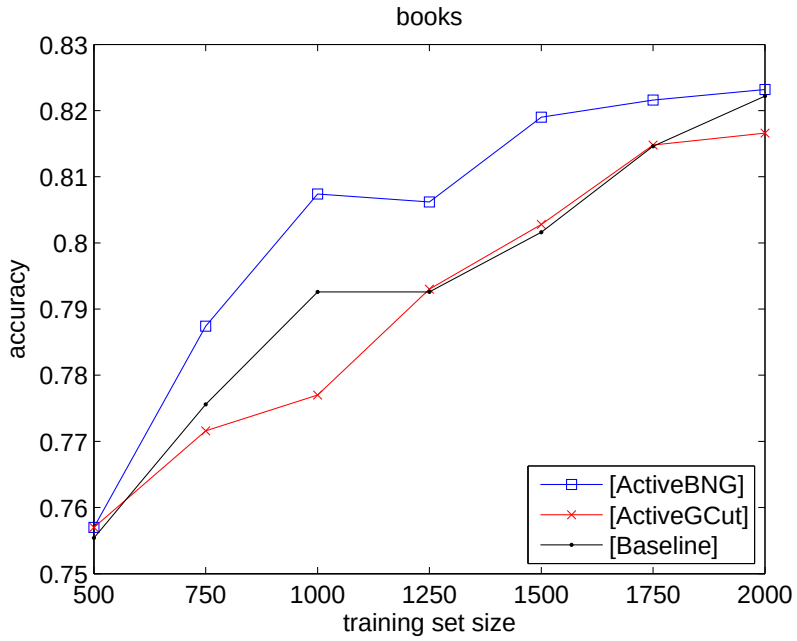


Figure 5.7: Performance of *maximum batch network gain* and *maximum graph-cut* queries, compared to a random baseline.

counterpart of the *maximum graph-cut* query [54] in the general text classification setting. Figure 5.7 compares these two methods on the Amazon book review data, together with the random selection baseline. As expected, the addition of the uncertainty component leads to improved results. The maximum graph-cut query alone does no better than the random baseline, indicating that relying on graph structure alone is not effective for active learning for this task where the graph is determined by feature co-occurrence.

5.6.4 Comparison across Different Domains

In this section, we systematically evaluate the proposed query strategies for active learning over different domains, including two tasks in the movie review dataset, and the four domains from the Amazon product review dataset. For each domain, we conducted six active learning iterations, which used 10%-40% of the data to train the model.

Comparison of Query Strategies

We first made a direct comparison of the different query strategies. In our experiment, the following four systems are compared, which are all based on the maximum entropy model and use its outputs as the final decisions:

- *Random query* [Baseline]
  This system uses random selection to choose instances to label in each iteration. This is a standard baseline for active learning; building on uninformed labeling of data is essentially not using active learning.

- *Uncertainty query* [ActiveEnt]
  This system applies the widely used uncertainty query, specifically the variation based on entropy (equation 5.7), which is equivalent to the *least confident* and *marginal sampling* query strategies for binary classification problems.

- *Maximum gradient length query* [ActiveMGL]
  This system applies the maximum gradient length query with a hard decision based on label propagation results.

- *Maximum batch network gain query* [ActiveBNG]
  This system applies the *maximum batch network gain* query with redundancy penalties as in equation (5.8).

Table 5.1 summarizes the average absolute accuracy gain and relative error reduction of the three other systems over the [Random] baseline system in the first six active learning batches. Both the proposed *maximum gradient length* and *maximum batch network gain* queries outperform the random query baseline. By leveraging the larger dataset through label propagation, the *maximum gradient length* query obtains better performance than *uncertainty query* on all domains except on the Amazon electronics domain. The *maximum batch network gain* query consistently outperforms the *uncertainty query*, which does not account for interdependence between individual instances in the batch. It also outperforms

maximum gradient length on all the tested domains except the Amazon book domain, for which none of the differences are significant.

Classification Enhanced with Label Propagation

In this section, we integrate label propagation into the semantic orientation classification process to enhance the performance of the four systems described above, and compare different query strategies under this new setting. Specifically, we apply the regularized label propagation after obtaining the maximum entropy posterior, and use the label propagation result as the final system output. The *random query* system with label propagation corresponds to semi-supervised learning alone, i.e. without active learning. The four semi-supervised systems are again compared to the baseline system (random query without label propagation).

Figures 5.8-5.13 present the experimental results in each domain respectively. The horizontal dash line in each figure represents the *random query* baseline system's perform after the sixth active learning iteration. Table 5.2 summarizes the average absolute accuracy gain and relative error reduction of the four other systems over the [Random] baseline system in the first six active learning batches. Compared with results in Table 5.1, systems with all query strategies get a performance boost with the label propagation. With label propagation, both *maximum gradient length* and *maximum batch network gain* queries significantly outperform the baseline across all domains. With one exception, all strategies combining active learning and semi-supervised learning improve over semi-supervised learning alone,

The *maximum batch network gain* query has relatively stable performance, which consistently outperforms all other systems on the tested domains, except for the *maximum gradient length* query on the Amazon book domain. The *maximum batch network gain* not only benefits the maximum entropy model (indicated by the results in Table 5.1), but also label propagation by choosing well-connected instances in the "key" paths of the label propagation.

The improvement margin associated with integrating label propagation is relatively larger on the movie review dataset than the other four domains from the Amazon review

Table 5.1: Average classification accuracy gain (absolute) and error reduction (relative) in the first 6 active learning batches (using 10%-40% training data) over the [Baseline] for different active learning strategies. (*** significant on 0.001; ** significant on 0.01; * significant on 0.05; others, not significant on 0.05)

Average Accuracy Gain (%)	**Books**	**DVDs**	**Electronics**	**Kitchen**	**Movie**	**Movie (sub/obj)**
ActiveEnt	1.14	0.60	1.50*	0.85	1.21	1.16*
ActiveMGL	**1.55**	0.93	1.03	1.10	1.48*	1.29**
ActiveBNG	1.09	**1.51***	**1.57***	**1.46***	**1.78****	**1.69*****
Average Error Reduction (%)	**Books**	**DVDs**	**Electronics**	**Kitchen**	**Movie**	**Movie (sub/obj)**
ActiveEnt	5.68	3.01	8.61	5.47	3.94	7.40
ActiveMGL	**7.74**	4.64	5.91	7.06	4.84	8.26
ActiveBNG	5.46	**7.51**	**9.03**	**9.38**	**5.82**	**10.81**

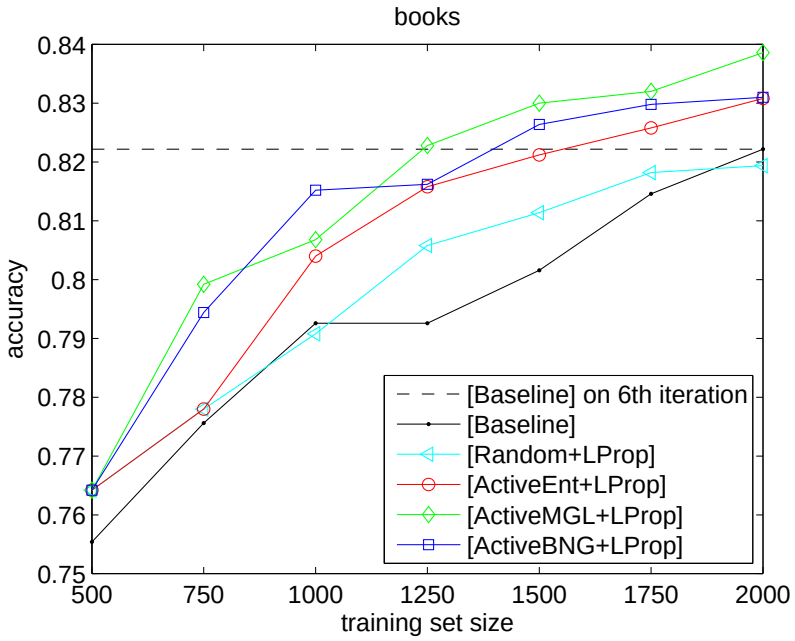


Figure 5.8: Performance on Amazon product review (books)

dataset. This is because the movie review data is on the sentence-level, which on average has a much smaller number of features as clues for semantic orientation judgment per instance, thus the data sparseness problem is more severe for this dataset. Hence, leveraging the unlabeled data with label propagation can bring more "mileage" to this dataset than the Amazon review dataset, which is on the document-level.

Figure 5.14 compares the percentage of the required training data amount relative to the random query [Baseline] system for the other 4 compared systems to reach the performance of the [Baseline] system on the 6th iteration. It is shown that the both the [ActiveMGL+LProp] and [ActiveBNG+LProp] systems need only 50%-75% of the data required by the [Random] baseline system to reach its performance on the 6th iteration; and they also require less training data than the other systems to achieve this goal on most of the tested domains.

Table 5.2: Average classification accuracy gain (absolute) and error reduction (relative) in the first 6 active learning batches (using 10%-40% training data) over the [Baseline] for different active learning strategies. (*** significant on 0.001; ** significant on 0.01; * significant on 0.05; others, not significant on 0.05)

Average Accuracy Gain (%)	**Books**	**DVDs**	**Electronics**	**Kitchen**	**Movie**	**Movie (sub/obj)**
Random+LProp	0.41	0.54	0.45	0.65	2.13***	2.24***
ActiveEnt+LProp	1.27*	1.10	1.52*	1.23*	2.98***	2.93***
ActiveMGL+LProp	**2.17****	1.53*	1.50*	1.64**	3.40***	1.94***
AcitveBNG+LProp	1.90**	**2.22****	**1.93***	**1.91****	**3.52*****	**3.33*****
Average Error Reduction (%)	**Books**	**DVDs**	**Electronics**	**Kitchen**	**Movie**	**Movie (sub/obj)**
Random+LProp	2.03	2.69	2.60	4.16	6.96	14.30
ActiveEnt+LProp	6.36	5.48	8.71	7.90	9.75	18.74
ActiveMGL+LProp	**10.84**	7.65	8.61	10.58	11.11	12.37
AcitveBNG+LProp	9.48	**11.07**	**11.06**	**12.32**	**11.52**	**21.26**

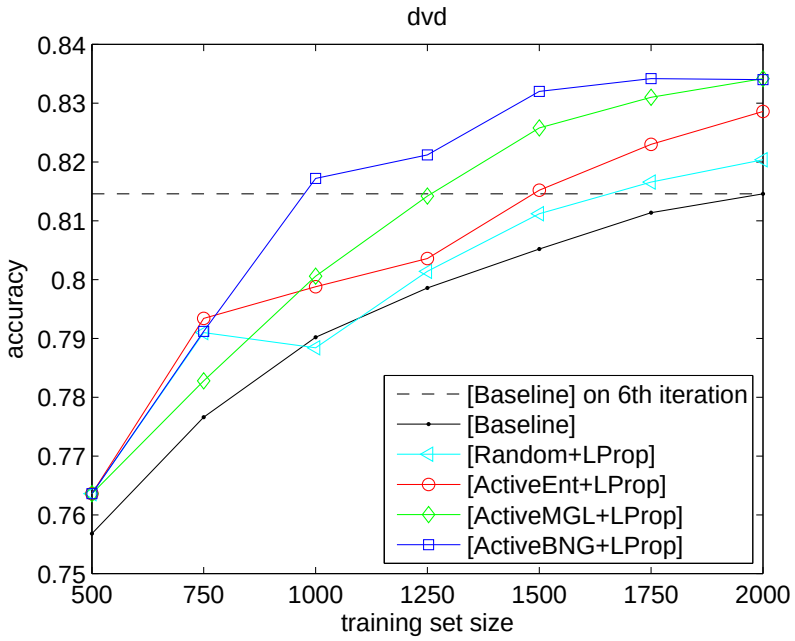


Figure 5.9: Performance on Amazon product review (dvd)

5.7 Summary

In this chapter, we have proposed a graph-based active learning framework that integrates label propagation based semi-supervised learning with a maximum entropy model. Based on this framework, we design two active learning query strategies that aim to account for interdependence between samples through a bipartite graph structure. One method uses label propagation through the graph to improve the evaluation of usefulness of individual samples. The other method uses the graph to incorporate diversity and representativeness into the selection criterion referred to as batch network gain. Experiments with and without label propagation in testing stage in order to understand the impact of semi-supervised learning. The success of the batch network gain method in both cases suggests that it is not semi-supervised learning per se that is leading to the performance gains, but rather the representation of interdependence between samples. This finding would support investigation of different graph structures for this and other tasks, as well as tight integration of semi-supervised learning into the network gain model.

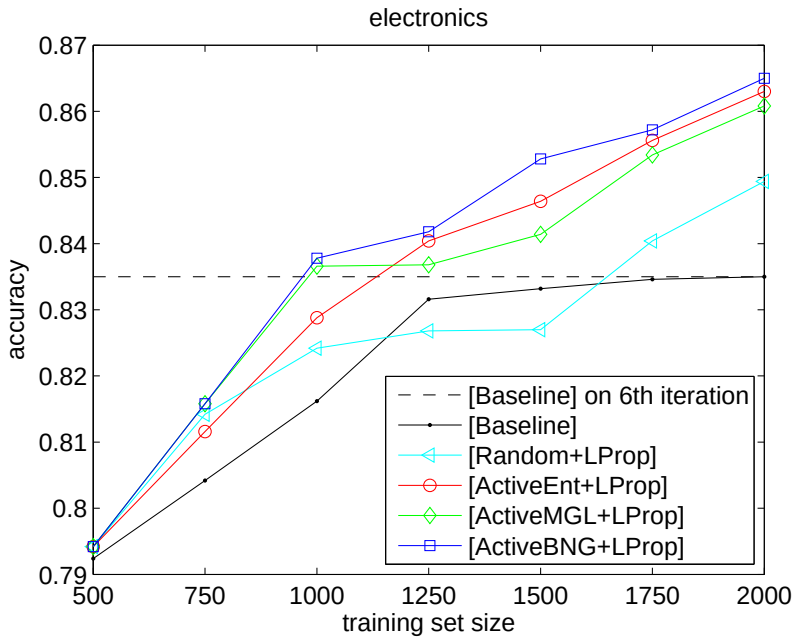


Figure 5.10: Performance on Amazon product review (electronics)

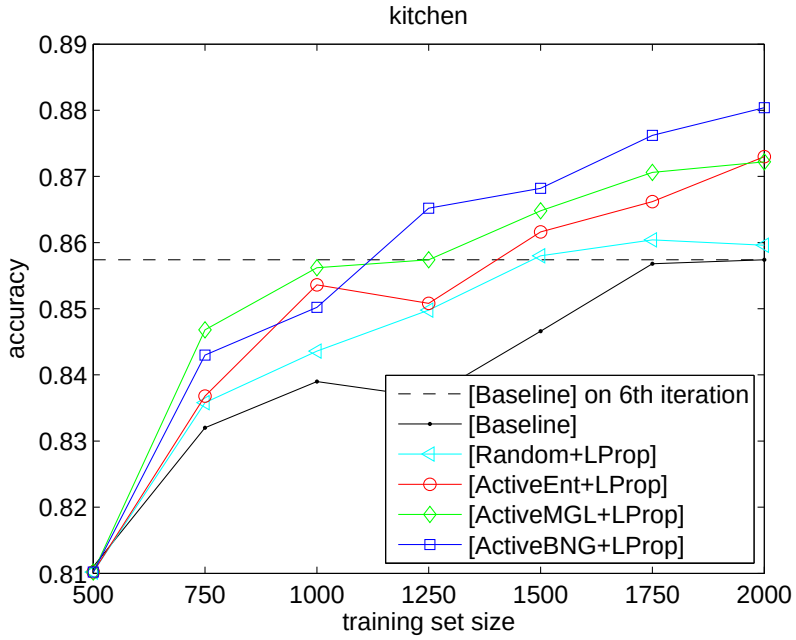


Figure 5.11: Performance on Amazon product review (kitchen)

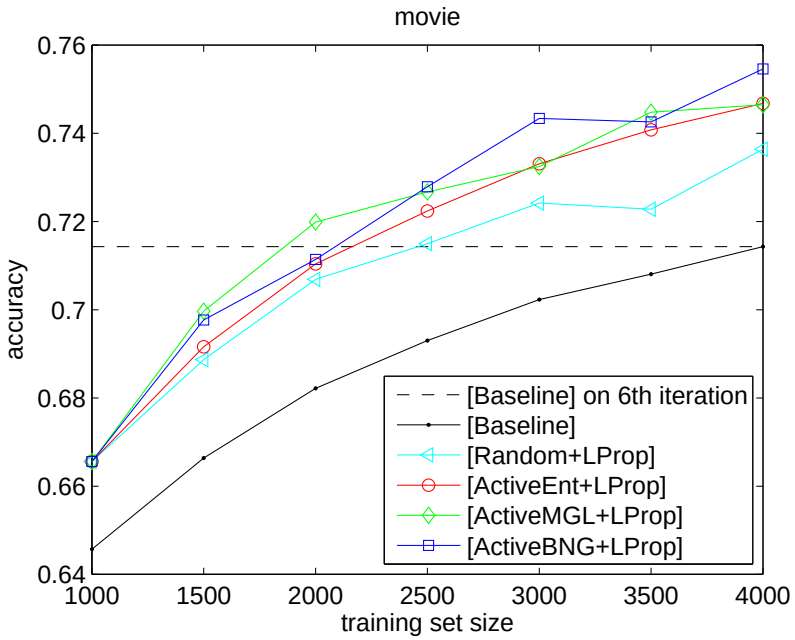


Figure 5.12: Performance on movie review (positive/negative task)

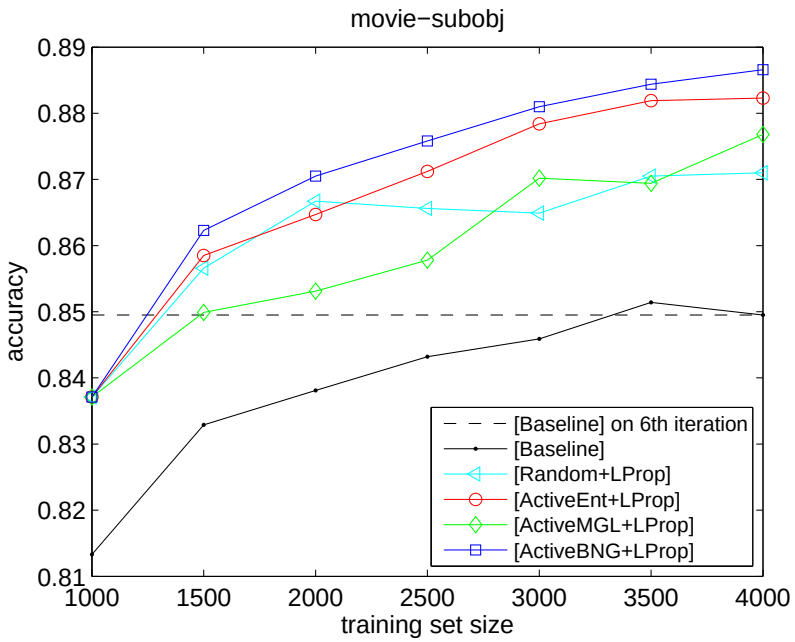


Figure 5.13: Performance on movie review (subjective/objective task)

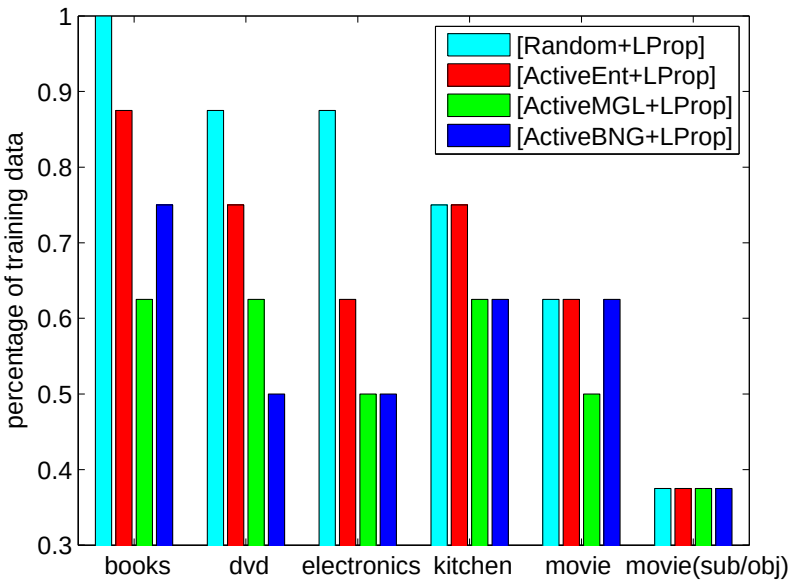


Figure 5.14: Percentage of the required training data relative to the [Random] baseline system for [Random+LProp], [ActiveEnt+LProp], [ActiveMGL+LProp] and [ActiveBNG+LProp] to reach the baseline system's performance on the 6th iteration

Chapter 6

SUMMARIZATION AND FUTURE DIRECTIONS

6.1 Summarization of Contributions

6.1.1 General Contributions

This thesis focuses on graph representation and algorithms for applying lexical semantics in computer-based automatic natural language processing tasks. We explore strategies for using graphs to represent natural language text based on contextual relations between words and/or other higher-level natural language units (e.g. sentences and documents). We work with two types of contextual relations when building graph representations for specific NLP applications: the word sense definitions in dictionary and word co-occurrence in higher-level language units. In the graphs, words and/or higher-level language units are represented with nodes, and edges are added between them according to their textual context to indicate their relatedness in semantics. We design two types of graph representations for specific NLP applications: the word-word graph which is used for modeling the semantic relations among words, and the instance-word bipartite graph which uses words a medium to study the relatedness among higher-level language units in natural languages. In this way, we can embed the semantic relations among words and/or higher-level language units into the graph structure.

With the proposed graph representations, we develop a series of algorithms to propagate semantic information among words or among higher-level language units via words in graphs, in order to explore the relatedness between words or higher-level language units in natural language text. The relatedness is used in designing unsupervised, semi-supervised or active learning algorithms to reduce human supervision in NLP applications for harvesting or analyzing text data resources. Specifically, we design graph-based algorithms either for quantitatively assessing lexical semantic similarity, or for developing representativeness and diversity criteria for selecting a characteristic subset of terms, which is useful for problems

such as keyword summarization and query design for active learning.

6.1.2 Application Advances

This work focuses the study of lexical semantics and graph algorithms on three NLP applications, including Wiktionary semantic similarity extraction, Twitter user interest summarization and active learning for semantic orientation classification. The graph-based representation and algorithms for lexical semantics lead to advances in all three cases.

For *Wiktionary lexical semantic similarity extraction*, we design graph-based algorithms to develop lexical semantic similarity measures from Wiktionary. We propose a word-word graph representation of Wiktionary based on its word sense definitions (WikNet), and study its graph properties. We apply a series of graph-based similarity measures to assess lexical semantic similarity in WikNet, and develop a new lexical semantic similarity measure (*Synonym J0-XJaccard*) based on a combination of n-step semantic similarity propagation and synonym information in Wiktionary. We demonstrate this new lexical semantic similarity measure obtains comparable performance with its WordNet-based counterparts on three human-rated lexical semantic similarity datasets. We further evaluate the performance of the WikNet-based lexical semantic similarity measure with the WordNet-based measures in two NLP applications, including paraphrase identification in news articles and alignment identification in web forum discussions. Experiments show that the WikNet-based Synonym J0-XJaccard measures consistently outperforms its WordNet-based counterparts.

For *Twitter user interest summarization*, we develop a system based on graph algorithms to summarize a Twitter user's interest and preferences in terms of keywords extracted from his/her tweets. We propose graph representations of the collection of a user's tweets based on lexical semantic similarity and word co-occurrence in tweet context, and design a new summarization algorithm that cascades an ∞-step random walk based semantic information propagation algorithm (PageRank) and a maximum graph-cut based 1-step semantic information propagation algorithm, which integrates both representativeness and diversity criteria in the summarization results. Experiments demonstrate that the proposed summarization system achieves high precision in the keyword summarization result.

For *active learning for semantic orientation classification*, we propose a graph-based active learning framework. We apply an instance-word bipartite graph representation for the semantic orientation classification corpora, and design two new graph-based query strategies that account for the instance relatedness for batch-mode active learning. The first query strategy tightly integrates an ∞-step random walk based semantic information propagation as a semi-supervised learning method to improve the uncertainty estimates associated with individual instances. The second query strategy is designed with a maximum graph-cut based 1-step semantic information propagation that combines the uncertainty criterion together with the representativeness and diversity criteria. The proposed graph-based active learning framework outperforms the standard active learning baseline on the semantic orientation classification task over a movie review dataset and an Amazon product review datasets covering four different product domains.

6.2 Future Directions

There are several directions we can pursue as future work of this thesis. First, we can extend the work in this thesis to more languages. Although the NLP corpora we studied in this work is in English, the proposed graph-based frameworks and algorithms for exploring lexical semantics in NLP applications are general across languages, and online dictionaries (including Wiktionary) are available in many languages. Second, we can develop new types of graph representations that combine the word-word graph and word-sentence (or other higher-level language unit) graph to leverage semantic information from difference sources. For example, starting with a word-sentence bipartite graph built from a text classification corpus, we can also connect the word nodes according to their sense definitions in Wiktionary. In this way, we can incorporate the semantic information from Wiktionary to help the text classification task.

Besides the general graph representation and algorithm framework, We can also extend our work in each of the three NLP applications.

For future work with WikNet, we can incorporate more information from Wiktionary to assess lexical semantic similarity. For instance, we can integrate the "derived terms" and "related terms" information from Wiktionary and the hyponym-hypernym relation from

Wiksaurus into the lexical semantic similarity measure. In addition, the current version of the WikNet-based semantic similarity measures is based on word-level nodes that merge links from all senses of a word; in future work we can extend this to sense-dependent nodes. We can apply the sense-dependent lexical semantic similarity measure derived from WikNet in applications such as word sense disambiguation. Alternatively, we can use a weighted combination of scores for different senses depending on context.

For future work with Twitter user interest summarization, we can place an inference layer on top of the word-level summarization. For instance, if "Manchester United" and "Arsenal" are extracted as user keywords, our system should also output "soccer" as a derived keyword for this user, though it may not appear in his/her tweets. This inference layer can improve the generalization of the summarization result, which can be more helpful for applications as following recommendation. In addition, we can incorporate semantic orientation classification into the user interest summarization result, so that we can associate the user interest keywords together with the user's affection towards them.

For future work with active learning for semantic orientation classification, we can apply and evaluate the proposed framework to genres other than movie or product reviews, such as tweet and facebook status. Second, we can develop active learning strategies for more resource-limited scenarios, for example, building a semantic orientation classification starting with only a semantic orientation lexical list without any annotated instances. Finally, we can extend the active learning framework to other text classification tasks, such as topic classification, alignment identification, political orientation classification, and so on.

VITA

Wei Wu was born in Beijng, a city with ancient royal palaces and great food but occasionally harassed by sand storms. He earned a Bachelor in Computer Science from Tsinghua University in 2004 and became a proud 9# alumnus. He spent three more years in Tsinghua University and earned a Master in Computer Science. In 2007, he arrived in Seattle and started his PhD study at University of Washington, and earned a PhD in Electrical Engineering in 2012.

BIBLIOGRAPHY

[1] Lada A. Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[2] Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of NAACL-HLT*, pages 19–27, 2009.

[3] James Allen. *Natural Lanuage Understanding*. The Benjamin/Cummings Publishing Company, 1995.

[4] Marco A. Alvarez and SeungJin Lim. A graph modeling of semantic similarity between words. In *Proceedings of the International Conference on Semantic Computing*, pages 355–362, 2007.

[5] Shumeet Baluja, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of WWW*, pages 895–904, 2008.

[6] Yoram Baram, Ran E. Yaniv, and Kobi Luz. Online choice of active learning algorithms. *J. Mach. Learn. Res.*, 5:255–291, 2004.

[7] Edwin L. Battistella. *Markedness: The Evaluative Superstructure of Language*. State University of New York Press, 1990.

[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(4-5):993–1022, March 2003.

[9] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jenn Wortman. Learning Bounds for Domain Adaptation. In *Proceedings of NIPS*, 2008.

[10] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, Bollywood, Boomboxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of COLING*, pages 440–447, 2007.

[11] Ulf Brefeld and Tobias Scheffer. Co-EM support vector learning. In *Proceedings of the twenty-first international conference on Machine learning*, Proceedings of ICML, 2004.

[12] Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[13] Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *Proceedings of the 20th International Conference on Machine Learning*, pages 59–66, 2003.

[14] Jinying Chen, Andrew Schein, Lyle Ungar, and Martha Palmer. An empirical study of the behavior of active learning for word sense disambiguation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 120–127, 2006.

[15] Munmun D. Choudhury, Yu-Ru Lin, Hari Sundaram, K. Selcuk Candan, Lexing Xie, and Aisling Kelliher. How Does the Sampling Strategy Impact the Discovery of Information Diffusion in Social Media? In *Proceedings of In the 4th Int'l AAAI Conference on Weblogs and Social Media*, 2010.

[16] Massimiliano Ciaramita and Yasemin Altun. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 594–602. Association for Computational Linguistics, 2006.

[17] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[18] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active Learning with Statistical Models. *Journal of Artificial Intelligence Research*, 4(1):129–145, 1996.

[19] Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. In *EMSEE '05: Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, 2005.

[20] D.A. Cruse. *Lexical Semantics*. Cambridge University Press, 1986.

[21] A. Culotta and A. McCallum. Reducing labeling effort for stuctured prediction tasks. In *Proceedings of the National Conference on Artificial Intelligence*, pages 746–751, 2005.

[22] Mary Dalrymple. *Semantics and syntax in lexical functional grammar : the resource logic approach*. MIT Press, 1999.

[23] Sajib Dasgupta and Vincent Ng. Mine the Easy, Classify the Hard: A Semi-Supervised Approach to Automatic Sentiment Classification. In *Proceedings of ACL-IJCNLP*, 2009.

[24] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *Proceedings of the Sixth Symposium on Operating System Design and Implementation*, 2004.

[25] Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: exploit massively parallel news sources. In *Proceedings of COLING*, 2004.

[26] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of ICML*, pages 264–271, 2008.

[27] Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proceedings of EMNLP*, 2009.

[28] Susan T. Dumais. Latent semantic analysis. *Annual Review of Information Science and Technology*, (188), 2005.

[29] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[30] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin.

[31] Dániel Fogaras and Balázs Rácz. Scaling link-based similarity search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 641–650, 2005.

[32] Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Mach. Learn.*, 28(2-3):133–168, 1997.

[33] Michael Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of COLING*, 2004.

[34] Julio Gonzalo, Felisa Verdejo, Irina Chugur, and Juan Cigarran. Indexing with WordNet synsets can improve text retrieval. *Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, 1998.

[35] Yuhong Guo and Russ Greiner. Optimistic active learning using mutual information. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 823–829, 2007.

[36] Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 174–181, 1997.

[37] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999.

[38] Steven C. H. Hoi, Rong Jin, and Michael R. Lyu. Large-scale text categorization by batch mode active learning. In *Proceedings of the 15th international conference on World Wide Web*, pages 633–642, 2006.

[39] Steven C. H. Hoi, Rong Jin, Jianke Zhu, and Michael R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 417–424, 2006.

[40] Malka Rappaport Hovav, Edit Doron, and Ivy Sichel. *Lexical Semantics, Syntax, and Event Structure*. Oxford University Press, 2010.

[41] Thad Hughes and Daniel Ramage. Lexical semantic relatedness with random graph walks. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[42] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of EMNLP*, pages 216–223, 2003.

[43] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, 1997.

[44] Rosie Jones. *Learning to Extract Entities from Labeled and Unlabeled Text*. PhD thesis, 2005.

[45] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: an Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, 2000.

[46] Seokhwan Kim, Yu Song, Kyungduk Kim, Jeong W. Cha, and Gary G. Lee. Mmr-based active machine learning for bio named entity recognition. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 69–72, 2006.

[47] Soo M. Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of COLING*, 2004.

[48] Claudia Leacock and Martin Chodorow. Combining local context and wordnet sense similiarity for word sense disambiguation. In *In WordNet, An Electronic Lexical Database*, 1998.

[49] Adrienne Lehrer. *Semantic Fields and Lexical Structure*. North-Holland Publishing Company, 1974.

[50] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, 1994.

[51] Xiao Li, Ye-Yi Wang, and Alex Acero. Learning query intent from regularized click graphs. In *Proceedings of SIGIR*, pages 339–346, 2008.

[52] Yuhua Li, Zuhair A. Bandar, and David McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882, 2003.

[53] Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, 1998.

[54] Hui Lin and Jeff Bilmes. How to Select a Good Training-data Subset for Transcription: Submodular Active Selection for Sequences. In *Proceedings of Conference of the International Speech Communication Association*, 2009.

[55] Hui Lin and Jeff Bilmes. Multi-document Summarization via Budgeted Maximization of Submodular Functions. In *Proceedings of NAACL-HLT*, 2010.

[56] Hui Lin, Jeff Bilmes, and Shasha Xie. Graph-based Submodular Selection for Extractive Summarization. In *Proceedings of IEEE Automatic Speech Recognition and Understanding*, 2009.

[57] Fei Liu, Feifan Liu, and Yang Liu. Automatic keyword extraction for the meeting corpus using supervised approach and bigram expansion. In *Proceedings of IEEE SLT*, pages 181–184, 2008.

[58] Fei Liu, Yang Liu, and Fuliang Weng. Why is SXSW trending? Exploring Multiple Text Sources for Twitter Topic Summarizationr. In *Proceedings of Workshop on Language in Social Media*, 2011.

[59] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of NAACL-HLT*, pages 620–628, 2009.

[60] John Lyons. *Semantics*. Cambridge University Press, 1977.

[61] Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka. Combining multiple evidence from different types of thesaurus for query expansion. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–197, 1999.

[62] Inderjeet Mani and Eric Bloedorn. Multi-document summarization by graph search and matching. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, pages 622–628, 1997.

[63] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain Adaptation with Multiple Sources. In *Proceedings of NIPS*, 2009.

[64] Andrew McCallum and Kamal Nigam. Employing EM n pool-based active learning for text classification. In *Proceedings of ICML*, pages 359–367, 1998.

[65] Andrew K. Mccallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *In Proceedings of the 15th International Conference on Machine Learning*, 1998.

[66] Prem Melville and Raymond J. Mooney. Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.

[67] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In *Proceedings of EMNLP*, 2004.

[68] George A. Miller and Walter G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1), 1991.

[69] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of EMNLP*, 2004.

[70] Ion Muslea, Steven Minton, and Craig A. Knoblock. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 435–442, 2002.

[71] Ion Muslea, Steven Minton, and Craig A. Knoblock. Active learning with multiple views. *J. Artif. Int. Res.*, 27(1):203–233, 2006.

[72] Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. Dependency tree-based sentiment classification using CRFs with hidden variables. In *Proceedings of NAACL-HLT*, 2010.

[73] Vivi Nastase. Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 763–772, 2008.

[74] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14(1):265–294, 1978.

[75] M. E. J. Newman. Clustering and preferential attachment in growing networks. *Physical Review E*, 64(2), 2001.

[76] Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of the ACL*, pages 611–618, 2006.

[77] Hieu T. Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.

[78] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93, 2000.

[79] Kamal Nigam, John Lafferty, and Andrew McCallum. Using maximum entropy for text classification. In *Proceedings of IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

[80] David L. Nowell and Jon Kleinberg. The link prediction problem for social networks. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.

[81] Zhonghua Ou and Yang Liu. Interactive Group Suggesting for Twitter. In *Proceedings of ACL-HLT*, 2011.

[82] Bo Pang and Lillian Lee. A Sentimental Education: Subjectivity Summarization Based on Minimum Cuts. In *Proceedings of ACL*, pages 271–278, 2004.

[83] Bo Pang and Lillian Lee. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124, 2005.

[84] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86, 2002.

[85] Siddharth Patwardhan and Ted Pedersen. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. In *Proceedings of the EACL*, pages 1–8, 2006.

[86] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. Wordnet::similarity: measuring the relatedness of concepts. In *Proceedings of NAACL*, 2004.

[87] Simone P. Ponzetto and Michael Strube. Deriving a large scale taxonomy from wikipedia. In *Proceedings of the 22nd national conference on artificial intelligence*, pages 1440–1445, 2007.

[88] Borislav Popov, Atanas Kiryakov, Damyan Ognyanoff, Dimitar Manov, and Angel Kirilov. KIM – a semantic platform for information extraction and retrieval. *Nat. Lang. Eng.*, 10(3-4):375–392, 2004.

[89] James Pustejovsky. *Semantics and the Lexicon*. Kluwer Academic Publishers, 1993.

[90] James Pustejovsky. *Generative Lexicon*. MIT Press, 1995.

[91] Delip Rao and Deepak Ravichandran. Semi-supervised polarity lexicon induction. In *Proceedings of EACL*, pages 675–682, 2009.

[92] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence*, pages 448–453, 1995.

[93] Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe. Feature subsumption for opinion analysis. In *Proceedings of EMNLP*, pages 440–448, 2006.

[94] Eric Ringger, Peter Mcclanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, and Deryle Lonsdale. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *Proceedings of the Linguistic Annotation Workshop*, 2007.

[95] Nicholas Roy and Andrew Mccallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of 18th International Conf. on Machine Learning*, pages 441–448, 2001.

[96] Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, October 1965.

[97] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[98] Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 839–846, 2000.

[99] Hinrich Schütze. Automatic word sense discrimination. *Comput. Linguist.*, 24(1):97–123, 1998.

[100] Burr Settles. Active Learning Literature Survey. Technical report, 2010.

[101] Burr Settles and Mark Craven. An Analysis of Active Learning Strategies for Sequence Labeling Tasks. In *Proceedings of EMNLP*, 2008.

[102] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Proceedings of NIPS*, pages 1289–1296, 2008.

[103] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew L. Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, 2004.

[104] Vikas Sindhwani and Prem Melville. Document-Word Co-Regularization for Semi-supervised Sentiment Analysis. In *Proceedings of ICDM*, 2008.

[105] Michael Strube and Simone P. Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *Proceedings of the 21st national conference on artificial intelligence*, pages 1419–1424, 2006.

[106] Michael Stubbs. *Words and Phrases: Corpus Studies of Lexical Semantics*. Blackwell Publisher, 2001.

[107] Partha P. Talukdar, Joseph Reisinger, Marius Paşca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of EMNLP*, pages 582–590, 2008.

[108] Min Tang, Xiaoqiang Luo, and Salim Roukos. Active learning for statistical natural language parsing. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 120–127, 2002.

[109] Simon Tong and Edward Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 107–118, 2001.

[110] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, 2002.

[111] Gokhan Tur, Dilek Hakkani-tür, and Robert E. Schapire. Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, 5(2), 2004.

[112] Peter D. Turney. Learning algorithms for keyphrase. *Information Retrieval*, 2(4):303–336, 2000.

[113] Peter D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL*, pages 417–424, 2002.

[114] Peter D. Turney. Coherent keyphrase extraction via web mining. In *Proceedings of IJCAI*, pages 434–439, 2003.

[115] Amos Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

[116] Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data*. 1982.

[117] Leonid Velikovich, Sasha B. Goldensohn, Kerry Hannan, and Ryan McDonald. The viability of web-derived polarity lexicons. In *Proceedings of NAACL-HLT*, pages 777–785, 2010.

[118] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69, New York, NY, USA, 1994.

[119] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of ACL*, pages 552–559, 2007.

[120] Ye-Yi Wang, Raphael Hoffmann, Xiao Li, and Jakub Szymanski. Semi-supervised learning of semantic classes for query understanding: from the web and for the web. In *Proceedings of CIKM*, pages 37–46, 2009.

[121] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[122] Wei Wu, Zhang Bin, and Mari Ostendorf. Automatic generation of personalized annotation tags for twitter users. In *Proceedings of NAACL-HLT*, 2010.

[123] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, 1994.

[124] Dongqiang Yang and David M. W. Powers. Measuring semantic similarity in the taxonomy of WordNet. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science*, pages 315–322, 2005.

[125] Eric Yeh, Daniel Ramage, Christopher D. Manning, Eneko Agirre, and Aitor Soroa. Wikiwalk: random walks on wikipedia for semantic relatedness. In *Proceedings of the Workshop on Graph-based Methods for Natural Language Processing*, pages 41–49, 2009.

[126] Wen-Tau Yih, Joshua Goodman, and Vitor R. Carvalho. Finding advertising keywords on web pages. In *Proceedings of 15th International Conference on World Wide Web*, pages 213–222, 2006.

[127] Torsten Zesch, Iryna Gurevych, and Max Mühlhäuser. Comparing wikipedia and german wordnet by evaluating semantic relatedness on multiple datasets. In *NAACL-HLT*, pages 205–208, 2007.

[128] Torsten Zesch, Christof Müller, and Iryna Gurevych. Using wiktionary for computing semantic relatedness. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 861–866, 2008.

[129] J. Zhang, J. Gao, and M. Zhou. Automatic generation of personalized annotation tags for twitter users. In *Proceedings of CLPW*, 2000.

[130] Wayne X. Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. Topical Keyphrase Extraction from Twitter. In *Proceedings of ACL-HLT*, 2011.

[131] Dengyong Zhou, Olivier Bousquet, Thomas N. Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Proceedings of NIPS*, pages 321–328, 2004.

[132] Dengyong Zhou, Bernhard Schölkopf, and Thomas Hofmann. Semi-supervised learning on directed graphs. In *Proceedings of NIPS*, 2005.

[133] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. Active Deep Networks for Semi-Supervised Sentiment Classification. In *Proceedings of COLING*, 2010.

[134] Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K. Tsou. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1137–1144, 2008.

[135] Xiaojin Zhu. Semi-supervised learning literature survey. 2008.

[136] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of ICML*, 2003.

[137] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and Semi-Supervised learning using gaussian fields and harmonic functions. In *Proceedings of ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 58–65, 2003.

[138] V. Zlatić, M. Božičević, H. Štefančić, and M. Domazet. Wikipedias: Collaborative web-based encyclopedias as complex networks. *Physical Review E*, 74(1), 2006.