Automatic taxonomy construction from textual documents

Luu, Anh Tuan

2017

Luu, A. T. (2017). Automatic taxonomy construction from textual documents. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/69985>

<https://doi.org/10.32657/10356/69985>

**AUTOMATIC TAXONOMY CONSTRUCTION
FROM TEXTUAL DOCUMENTS**

Luu Anh Tuan

Ph.D Candidate

School of Computer Science and Engineering

Nanyang Technological University

**A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY**

April, 2017

Acknowledgement

First and foremost, I am deeply indebted to my supervisors, Prof. Hui Siu Cheung and Prof. Ng See Kiong, for their guidance, advice and encouragement as well as giving me extraordinary experiences throughout my doctoral program. Above all and the most needed, they provided me unflinching encouragement and support in various ways. Without their guidance, this thesis would not have been completed or written.

I am deeply grateful to Dr. Jung-jae Kim, who is my former supervisor. I thank him for introducing me to the exciting area of taxonomy construction. His critical comments contribute to the backbone of this research.

I would like to thank The Agency for Science, Technology and Research (A*STAR) in Singapore for providing me the scholarship to pursue this research, as well as supporting the finance for me to present papers in several conferences overseas.

Besides, I would also like to thank my fellow labmates: Dr. Watson Wei Khong Chua, Tay Yi, Han Xu, Tran Ha Nguyen, Ma Yukun, Clement Crepy, to name a few. I am grateful for their friendship throughout my study, and I really enjoyed my time with these brilliant people.

Lastly, I would like to thank sincerely and deeply my parents Luu Van Hai and Tran Thi Tuyet Van, who have taken care of me with great love in these years. I thank my wife Nguyen Thi Quynh Ngoc and my daughter Luu Nguyen Sophia, for all the love.

Abstract

The explosion of unstructured text data makes it difficult to find information for our interests. To provide access to information effectively, it is important to organize the unstructured data in a structured and meaningful manner. Taxonomies, which serve as the backbone for structured knowledge, are useful for many NLP applications such as question answering and document clustering by organizing domain knowledge into a hierarchy of ‘is-a’ relations between terms. Currently, there have been an increasing number of public hand-crafted taxonomies available such as WordNet and Freebase. However, it will be more effective to use taxonomies that are created specifically for the domain of interest in practice rather than re-using existing taxonomies created for other tasks or domains. As such, we often face the challenge of creating a brand new taxonomy for a specific domain from scratch.

In this thesis, we propose an effective framework for automatic domain-specific taxonomy construction from textual documents, which consists of three steps, namely domain term extraction, taxonomic relation identification and taxonomy induction. Domain term extraction aims to extract the relevant domain terms from a given text collection of specific domain. Taxonomic relation identification aims to identify the taxonomic relations (i.e. ‘is-a’ relations) among domain terms. Taxonomy induction aims to construct the taxonomy structure from the identified taxonomic relations. We use the big data approach which combines linguistics, statistical and deep learning methods to address the challenges in these steps. The main contributions of our research are summarized as follows:

- We proposed a Web-based method to extract domain terms from a given text collection. From that, we proposed a method to use the contextual information of the terms in syntactic structures to detect taxonomic relations across sentence boundary. In addition, we also proposed a novel graph-based algorithm to organize the extracted taxonomic relations into an optimal taxonomy tree. The experimental results show that the proposed method is well complementary to the previous methods of linguistic pattern matching and significantly improves recall

and F-measure.

- We studied two important aspects that can greatly affect the performance of taxonomy construction method. The first one is on the trustiness of individual source texts, which is important to filter out incorrect relations from unreliable sources. The second one is on the collective evidence from synonyms and contrastive terms, where synonyms provide additional supports to taxonomic relation identification, while contrastive terms may contradict them. We proposed an approach to incorporate these features into taxonomy construction, which can improve the performance on F-measure by up to 4%-10%.
- We proposed a time-aware approach to extract and integrate temporal information into the process of identifying taxonomic relations, by employing a timestamp contribution function to measure the evidence scores of source texts at a particular time. Experimental results show that our proposed approach outperforms the state-of-the-art methods on F-measure by up to 7%-20%. Furthermore, the proposed approach can incrementally and continuously update the taxonomy by adding fresh relations from new data and removing outdated relations, using a proposed information decay function. It thus avoids rebuilding the whole structure from scratch for every update and maintains the taxonomy up-to-date in order to keep up with the latest information trends quickly.
- We proposed a novel unsupervised approach to construct taxonomies based on word embedding clustering, using the following three word embedding measures: semantic clusters, taxonomic centroids and relative distances from the root, for identifying the semantic relationships between terms and their hypernyms. Our proposed approach significantly outperforms the state-of-the-art methods in terms of recall and F-measure.
- We proposed an approach to learn word embeddings for taxonomic relations based on the contextual words between the hypernym and hyponym using a dynamic weighting neural network. Our proposed approach significantly outperforms the state-of-the-art methods by 9% to 13% in terms of accuracy for both general and specific domain datasets.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Taxonomy Construction	2
1.2.1	Domain Term Extraction	3
1.2.2	Taxonomic Relation Identification	3
1.2.3	Taxonomy Induction	5
1.3	Objectives	5
1.4	Contributions	6
1.5	Organization of the Thesis	8
2	Related Work	9
2.1	Domain Term Extraction	9
2.1.1	Candidate Term Collection	9
2.1.2	Domain Term Filtering	10
2.2	Taxonomic Relation Identification	12

2.2.1	Statistical Approach	12
2.2.2	Linguistic Approach	16
2.3	Taxonomy Induction	18
2.4	Summary	20
3	Taxonomy Construction using Syntactic Contextual Evidence and Web Data	21
3.1	Proposed Approach	22
3.2	Domain Term Extraction	24
3.3	Taxonomic Relation Identification	25
3.3.1	String Inclusion with WordNet (SIWN)	26
3.3.2	Lexical-Syntactic Pattern (LSP)	26
3.3.3	Syntactic Contextual Subsumption (SCS)	27
3.3.4	Combined Method	31
3.4	Taxonomy Induction	31
3.5	Performance Evaluation	33
3.5.1	Datasets	34
3.5.2	Performance on Domain Term Extraction	35
3.5.3	Performance on Taxonomy Construction for AI and Terrorism Domains	36
3.5.4	Performance Comparison on Taxonomy Construction based on Curated Databases	38
3.6	Summary	41

4	Trustiness and Collective Synonym/Contrastive Evidence for Taxonomy Construction	42
4.1	Proposed Approach	43
4.2	Trustiness of the Evidence Data	44
4.2.1	Collecting Trustiness Score of the Evidence Data	44
4.2.2	Integrating Trustiness into Taxonomic Relation Identification Methods . . .	47
4.3	Collective Synonym Evidence	48
4.3.1	Synonymy Identification	48
4.3.2	Embedding Synonym Information	50
4.4	Contrastive Evidence	50
4.5	Combining Trustiness, Synonym and Contrastive Evidence	51
4.6	Performance Evaluation	52
4.6.1	Datasets	52
4.6.2	Performance on Taxonomy Construction for AI and Finance Domains . . .	53
4.6.3	Performance on Taxonomy Construction based on Curated Databases . . .	54
4.6.4	Performance on Trustiness and Synonymy Identification	56
4.6.5	Discussion	57
4.7	Summary	59
5	Temporal Information for Taxonomy Construction	60
5.1	Time-aware Taxonomy Construction	61
5.2	Proposed Approach	62

5.3	Temporal Information Processing	63
5.4	Taxonomy Construction	65
5.4.1	Incorporating Temporal Information into Taxonomic Relation Identification	65
5.4.2	Parameter Learning	68
5.4.3	Incremental Taxonomy Induction	68
5.5	Performance Evaluation	71
5.5.1	Datasets	71
5.5.2	Evaluation on Taxonomy Construction	72
5.5.3	Evaluation on Taxonomy Update	76
5.5.4	Parameter Tuning	81
5.6	Summary	81
6	Taxonomy Construction Using Word Embedding Clustering	83
6.1	Embedding-based Taxonomy Construction	84
6.2	Empirical Study on Taxonomic Relations	85
6.2.1	Data Preparation	85
6.2.2	Empirical Analysis and Observations	86
6.3	Proposed Approach	90
6.3.1	Taxonomic Relation Identification	91
6.3.2	Taxonomy Induction	93
6.4	Performance Evaluation	95

6.4.1	Datasets	95
6.4.2	Performance Based on Curated Databases	96
6.4.3	Performance Based on New Domains	98
6.4.4	Empirical Comparison with WordNet	98
6.4.5	Parameter Tuning	99
6.5	Summary	100
7	Learning Term Embeddings for Taxonomic Relation Identification	101
7.1	Proposed Approach	102
7.2	Learning Term Embeddings	102
7.2.1	Extracting Taxonomic Relations	103
7.2.2	Extracting Training Triples	104
7.2.3	Training Neural Network	104
7.3	Learning Neural Network	107
7.3.1	Weight Update Equation from Hidden Layer to Output Layer	107
7.3.2	Weight Update Equation from Input Layer to Hidden Layer	109
7.4	Supervised Taxonomic Relation Identification	109
7.5	Performance Evaluation	110
7.5.1	Datasets	110
7.5.2	Comparison Models	111
7.5.3	Performance on General Domain Datasets	112

7.5.4	Performance on Specific Domain Datasets	114
7.5.5	Tuning Vector Dimensions	116
7.6	Summary	118
8	Conclusion	119
8.1	Summary	119
8.2	Future Work	121
8.2.1	Extending the Proposed Taxonomy Construction Methods	121
8.2.2	Investigating Techniques for Non-taxonomic Relation Extraction and On- tology Learning	122
8.2.3	Investigating Probabilistic Model for Taxonomic Relations	123
8.2.4	Investigating Applications with Taxonomy	123
A	Authors Publications	125

List of Figures

1.1	An Example of Taxonomy for Animal Domain	2
1.2	Architecture of Domain-specific Taxonomy Construction	3
3.1	An Example of Taxonomy Induction	34
5.1	Proposed Approach for Time-aware Taxonomy Construction	63
5.2	Sample Report	64
5.3	Performance Results on Taxonomy Update over Time in the ‘MH370’ Domain	79
5.4	Top Three Hyponyms of ‘Search area’ in ‘MH370’ Taxonomies over Time	80
6.1	Clusters of Offset Vectors	87
6.2	Clusters of Offset Vectors between Pairs of Terms	88
6.3	Percentage of Taxonomic Relations $t_1 \gg t_2$ with $d(v_{\mathfrak{R}}, v_{t_1}) - d(v_{\mathfrak{R}}, v_{t_2}) < \alpha$	90
7.1	Proposed Approach for Learning Term Embeddings	103
7.2	Architecture of the Proposed Dynamic Weighting Neural Network Model	105

Chapter 1

Introduction

1.1 Motivation

With the popularity of the Web nowadays, there is an ever growing and explosion of textual information in electronic forms. However, most of them are in the form of natural languages with poor structure, which makes it difficult to find information of our interests. To provide access to information effectively, it is important to organize information in a structured and meaningful manner. Taxonomies, which organize domain knowledge into a hierarchy of taxonomic relations (i.e. ‘is-a’ relations) between terms, can help explore knowledge at different levels of granularity. Figure 1.1 shows an example of taxonomy for the animal domain.

Taxonomies serve as the backbone of structured knowledge, and are useful for many NLP applications such as question answering [27], document clustering [19] and textual entailment [24]. For example, knowing that “Gorilla” is an “ape” can help a question answering system answer the question such as “What kinds of apes are living in African rainforest?”.

Currently, there have been an increasing number of publicly available hand-crafted, well-structured taxonomies such as WordNet [51], OpenCyc [48], and Freebase [6]. However, the available taxonomies, which are originally created for general purposes, may be incomplete for new or special-

ized domains. Moreover, it is also time-consuming and cumbersome to create a new one manually. As such, methods for automatic taxonomy construction from text corpora are highly desirable. In this thesis, we aim to investigate new techniques for automatic taxonomy construction from a set of textual documents pre-clustered by a domain of interest.

Figure 1.1: An Example of Taxonomy for Animal Domain

1.2 Taxonomy Construction

Figure 1.2 shows an architecture for taxonomy construction for a specific domain, which consists of the following three steps:

- **Domain Term Extraction:** This step aims to extract domain terms from textual collections pre-clustered by a domain of interest.
- **Taxonomic Relation Identification:** This step aims to find all taxonomic relations among the identified domain terms.
- **Taxonomy Induction:** This step aims to construct the taxonomy structure from the identified taxonomic relations.

Figure 1.2: Architecture of Domain-specific Taxonomy Construction

In the following subsections, we discuss the challenges in each of the steps of taxonomy construction.

1.2.1 Domain Term Extraction

This step includes collecting all candidate terms in sentences and selecting terms that are relevant to the domain of interest. Previous methods for domain term extraction from textual documents often utilize statistical measures such as TF-IDF, document relevance (DR), and document consensus (DC) [54] to find the relevant terms in the domain. However, such measures are dependent on the selected corpora, and it is not trivial to construct balanced corpora for each domain.

1.2.2 Taxonomic Relation Identification

This is the most important step of taxonomy construction. A taxonomic relation is defined as a ‘is-a’ relation between a hypernym (i.e. generic term) and a hyponym (i.e. specific term). Many proven techniques from established fields such as information retrieval, machine learning and natural language processing have all contributed to the task of identifying taxonomic relations. However, there

are still some further challenges which are discussed as follows.

Trade-off between Precision and Recall: Previous methods for identifying taxonomic relations can be generally classified into two main categories: statistical and linguistic approaches. The former includes co-occurrence analysis [8], term subsumption [20] and clustering [77]. The main idea behind these techniques is that terms that frequently co-occur may have taxonomic relationships. Such approaches, however, usually suffer from low accuracy, but with relatively high coverage, and depend heavily on the choice of feature types and datasets. Most previous methods of the linguistic approach, on the other hand, rely on the lexical-syntactic patterns (e.g. *A is a B*, *A such as B*) [29]. These patterns can be manually created [38, 75], chosen via automatic bootstrapping [76, 25] or identified from machine-learned classifiers [54]. The pattern matching methods generally achieve high precision, but with low coverage due to the lack of contextual analysis across sentences.

Implicit Features: When identifying the taxonomic relation between a pair of terms, most previous methods often utilize only explicit features which are related directly to the two terms such as the linguistic patterns or distance between them, but ignore implicit features (e.g. synonym) which can help infer taxonomic relations effectively. For example, if we know that x is a hypernym of y , and y and z are synonyms, then we can infer that x is a hypernym of z .

Timestamp Information: When dealing with time-stamped and continuous changing textual documents, the taxonomic relation identification method will need to take into account the time-stamp information of the evident so that it can identify new taxonomic relations and get rid of obsolete relations effectively, as well as maintain the taxonomy up-to-date in order to keep up with the latest information trends. Previous works in taxonomic relation identification either ignored temporal information or used fixed time periods to discretize the time series of documents, thus may cause the resultant taxonomy to become incomplete, outdated and misleading.

Linguistic and Semantic Relations: The previous approaches for taxonomic relation identification focus on statistical learning and linguistic pattern matching methods, which require information from dictionary or Web data for training and learning. Moreover, the performance of these approaches may vary from different domains due to the fact that the learning process can be domain-

dependent. Word embedding [5], also known as distributed word representation, represents words with high-dimensional and real-valued vectors, and has been shown to be effective in exploring both linguistic and semantic relations between words. Word embedding has been successfully applied to many tasks such as machine translation, sentiment analysis and syntactic parsing. While it has gained popularity in recent NLP research, taxonomy construction studies have yet to catch on this trend.

1.2.3 Taxonomy Induction

After identifying taxonomic relations, the identified relations can be integrated into a graph for the task of taxonomy construction from scratch or associated with existing concepts of a given taxonomy via is-a relations [40]. In this step of taxonomic structure construction, there is a need for pruning incorrect and redundant relations. Previous methods for the pruning task [22, 42] treat the identified taxonomic relations equally, and the pruning task is thus reduced to finding the best balance between path length and nodes connectivity. This assumption, however, is not always true due to the fact that the identified taxonomic relations may have different confidence values, and relations with high confidence values can be incorrectly eliminated during the pruning process. In addition, when dealing with continuous changing documents, most previous taxonomy induction methods either rebuild the whole structure from scratch for every update or add new relations to old taxonomy without considering its appropriateness. As a result, it is a time consuming process for taxonomy induction and the constructed taxonomy may be outdated and inconsistent.

1.3 Objectives

The main objective of this research is to investigate the linguistics, statistical, graph-based and deep learning techniques to overcome the issues of current taxonomy construction methods as discussed in Section 1.2. In particular, this research will investigate techniques for domain term extraction, taxonomic relation identification and taxonomy induction as follows:

- *Investigate a new technique for domain term extraction.* To overcome the problem on domain dependency, we investigate a Web-based method for domain term extraction so that it can utilize the evidence from the Web rather than relying on the choice of balanced corpora.
- *Investigate new techniques for taxonomic relation identification.* To improve the performance of taxonomic relation identification, we investigate a new statistical method which can infer taxonomic relations beyond sentence boundary. Moreover, we also investigate some important aspects that can improve the performance of taxonomic relation identification. To deal with time-stamped documents, we investigate a time-aware method to extract and integrate temporal information into the process of taxonomic relation identification. To explore both linguistic and semantic relations, we focus on investigating a method to identify taxonomic relations based on word embedding clustering. To further improve the performance of the current embedding methods, we also investigate a method to learn word embeddings for the task of taxonomic relation identification.
- *Investigate new techniques for taxonomy induction.* We investigate a graph-based algorithm that utilizes not only the topological structure of the graph, but also the evidence information of the identified taxonomic relations. In addition, we also study an approach to update the taxonomy by adding fresh relations from new data and removing outdated relations in order to avoid rebuilding the whole structure from scratch for every update.

1.4 Contributions

As a result of this research, we proposed an effective framework for automatic domain-specific taxonomy construction from textual documents. The main contributions of our research are summarized as follows:

- We proposed a Web-based method to extract domain terms from a given text collection. From that, we proposed a method to use the contextual information of the terms in syntactic structures to detect taxonomic relations across sentence boundary. In addition, we also proposed

a novel graph-based algorithm to organize the extracted taxonomic relations into an optimal taxonomy tree. The experimental results show that the proposed method is well complementary to the previous methods of linguistic pattern matching and significantly improves recall and F-measure.

- We studied two important aspects that can greatly affect the performance of taxonomy construction method. The first one is on the trustiness of individual source texts, which is important to filter out incorrect relations from unreliable sources. The second one is on the collective evidence from synonyms and contrastive terms, where synonyms provide additional supports to taxonomic relation identification, while contrastive terms may contradict them. We proposed an approach to incorporate these features into taxonomy construction, which can improve the performance on F-measure by up to 4%-10%.
- We proposed a time-aware approach to extract and integrate temporal information into the process of identifying taxonomic relations, by employing a timestamp contribution function to measure the evidence scores of source texts at a particular time. Experimental results show that our proposed approach outperforms the state-of-the-art methods on F-measure by up to 7%-20%. Furthermore, the proposed approach can incrementally and continuously update the taxonomy by adding fresh relations from new data and removing outdated relations, using a proposed information decay function. It thus avoids rebuilding the whole structure from scratch for every update and maintains the taxonomy up-to-date in order to keep up with the latest information trends quickly.
- We proposed a novel unsupervised approach to construct taxonomies based on word embedding clustering, using the following three word embedding measures: semantic clusters, taxonomic centroids and relative distances from the root, for identifying the semantic relationships between terms and their hypernyms. Our proposed approach significantly outperforms the state-of-the-art methods in terms of recall and F-measure.
- We proposed an approach to learn word embeddings for taxonomic relations based on the contextual words between the hypernym and hyponym using a dynamic weighting neural

network. Our proposed approach significantly outperforms the state-of-the-art methods by 9% to 13% in terms of accuracy for both general and specific domain datasets.

1.5 Organization of the Thesis

This chapter has discussed briefly the motivation, objectives and contributions of this research. The rest of the thesis is organized as follows.

Chapter 2 reviews the related work on the three steps of taxonomy construction, namely domain term extraction, taxonomic relation identification and taxonomy induction.

Chapter 3 presents a proposed framework for taxonomy construction using syntactic contextual evidence and Web data.

Chapter 4 discusses two important aspects that can greatly affect the performance of taxonomy construction method, and a proposed method to incorporate them into taxonomy construction.

Chapter 5 presents a time-aware approach for constructing taxonomy from the time-stamped and continuous changing textual documents.

Chapter 6 discusses a novel unsupervised approach to construct taxonomies based on word embedding clustering.

Chapter 7 presents an approach to learn word embeddings using dynamic neural network for the purpose of identifying taxonomic relations.

Finally, Chapter 8 gives the conclusion and discusses the directions for further research work.

Chapter 2

Related Work

In this chapter, we review the related work on domain term extraction, taxonomic relation identification and taxonomy induction.

2.1 Domain Term Extraction

2.1.1 Candidate Term Collection

There are two main approaches to collect candidate terms from the domain corpus. The first approach relies on statistical methods such as mutual information measure, log-likelihood ratio and C-value. In mutual information measure [78], it begins with a list of seed terms, which is usually a set of single-word terms in the corpus. After that, multi-word terms are created by applying the maximum-likelihood estimation on those seed concepts. In log-likelihood ratio, it also starts with the list of single-words and try to form the multi-word terms by using the contingency table of the words to calculate the corresponding binominal distribution. In C-value [21], it takes into account of the long terms. In this method, not only the term's frequency and distribution but also the length of the terms in the corpus are taken into consideration. The terms are extracted using the term hood based on these factors. The main advantage of statistical approach is that it can derive terms that are

not explicitly stated in the documents. However, this approach usually suffers from low precision in term extraction.

The second approach to term extraction is based on linguistic properties, such as the work of Maedche et al. [46] and Navigli et al. [53]. The idea of this approach is to find the terms, including single-word terms and multi-word terms, captured in the syntactic structures of the sentence. To do this, some pre-defined linguistic patterns (e.g. POS tags) are created to extract all candidate terms (usually noun phrases). The advantage of this approach is that it can extract terms in documents with higher precision than the statistical approach.

2.1.2 Domain Term Filtering

The collected candidate terms are then filtered to select the terms that are most relevant to the domain of interest. Many statistical techniques have been developed for term filtering such as *tf-idf*, domain relevance, domain consensus, domain pertinence and lexical cohesion.

Term frequency-inverse document frequency (*tf-idf*) is a well-known measure to determine the domain relevance of terms. Specifically, *tf* measures the frequency of a term t in domain corpus D , while *idf* measures whether a term t is common or rare across all documents in D . They are defined as follows:

$$tf(t, D) = \frac{f(t, D)}{\max\{f(w, D) : w \in D\}} \text{ and}$$

$$idf(t, D) = \log \frac{|D|}{1 + |\{d \in D : t \in d\}|} ,$$

where $f(t, D)$ is the frequency of term t in domain corpus D , $|D|$ is the cardinality of D , i.e. number of documents in the corpus D .

Then *tf-idf* is calculated as follows:

$$tf-idf(t, D) = tf(t, D) \times idf(t, D).$$

The *tf-idf* method, however, has some issues in practice. Firstly, it is sensitive to corpus size such that if the number of the documents is small, *tf-idf* does not work effectively. Secondly, if a term

is quite significant and appears in most of the documents of a domain corpus, it may not be chosen as $idf(t) \approx 0$.

Domain relevance (DR) and domain consensus (DC) are another two measures introduced by Navigli and Velardi [53]. DR measures the amount of information that a term t captures within a domain of interest D_i , compared to other domains ($D_j, j : 1..n$), and is defined as follows:

$$DR(t, D_i) = \frac{P(t|D_i)}{\max_{1 \leq j \leq n} P(t|D_j)} ,$$

where $P(t|D_i)$ is the conditional probability that is estimated as follows:

$$P(t|D_i) = \frac{f_{D_i}(t)}{\sum_{t' \in D_i} f_{D_i}(t')} ,$$

where $f_{D_i}(t)$ is the frequency of term t in D_i .

In contrast, DC measures the distributed use of a term t across documents d in a domain D_i , and is defined as follows:

$$DC(t, D_i) = \sum_{d \in D_i} (P_t(d) \log \frac{1}{P_t(d)}) ,$$

where $P_t(d)$ is calculated as follows:

$$P_t(d) = \frac{f_d(t)}{\sum_{d_j \in D_i} f_{d_j}(t)} ,$$

where $f_{d_j}(t)$ is the frequency of term t in document d_j .

The two statistical measures DR and DC suffer from the following problems: Firstly, suppose a term t_1 appears one time in each document in domain D_i , and another term t_2 appears ten times in each document. Then, we have $DC(t_1, D_i) = DC(t_2, D_i)$. This is obviously unfair for important terms with high frequency. Secondly, the value of DR heavily depends on the choice of other contrasting domains. If the contrasting domains are, by accident, highly correlated to the domain of interest, then the DR measure can lead to the selection of many irrelevant domain terms.

Domain pertinence (DP) and lexical cohesion (LC) are the other two term filtering measurements proposed by Park et al. [55]. The definition of domain pertinence DP is similar to domain relevance

DR described above, which is used to measure the amount of information that a term captures within a domain of interest compared to other contrasting domains. The lexical cohesion LC of a term t in domain D measures the cohesion of all the words inside t and is defined as follows:

$$LC(t, D) = \frac{n \times f_D(t) \times \log(f_D(t))}{\sum_{w \in t} f_D(w)},$$

where n is the number of words in t , w is a word in t and $f_D(w)$ is the frequency of w in domain D .

2.2 Taxonomic Relation Identification

The techniques for identifying taxonomic relations can be generally classified into two main categories: statistical and linguistic approaches.

2.2.1 Statistical Approach

The main idea behind the statistical approach is that terms which frequently co-occur may have taxonomic relationships. Budanitsky [8] proposed a co-occurrence analysis method to extract the implicit relationship of lexical units in textual documents. The assumption is that related lexical units usually tend to occur together in the same sentence or document. In the study, many co-occurrence analysis techniques such as log-likelihood ratios (e.g., chi-square test) and dependency measures (e.g., mutual information) are applied for extracting the implicit relationships.

Fotzo and Gallinari [20] proposed term subsumption methods to find taxonomic relations. In their work, two methods are proposed to infer the subsumption relation between terms. The first method is based on terms hierarchy. Given two concepts C_1 and C_2 , the method computes the percentage x of instances of C_2 generalized by instances of C_1 , and the percentage y of instances of C_1 generalized by instances of C_2 . If $x > y > \text{threshold}$, then it can conclude that C_1 is the generalization of C_2 . The second method is based on the conditional probabilities of a pair of concepts. Given two concepts C_1 and C_2 , the conditional probability between C_1 and C_2 is defined as follows:

$$P(C_1|C_2) = \frac{\text{number of documents about concepts } C_1 \text{ and } C_2}{\text{number of documents about } C_2}.$$

If $P(C_1|C_2) > P(C_2|C_1) > \text{threshold}$, then C_1 is the generalization of C_2 . In addition, an Estimation Maximization (EM) is also proposed to determine whether a document is about a concept.

Wong et al. [77] proposed a statistical approach which is based on clustering methods. In the proposed work, terms are grouped to form a hierarchy based on the similarity measurement. There are two main features used in the study: normalized Google distance and cosine similarity between articles of Wikipedia. The features are used in the following two clustering methods: agglomerative clustering and divisive clustering. In agglomerative clustering, it starts with individual terms. These terms are then clustered with other related terms based on the similarity score. In divisive clustering, it starts with all the terms, and then divides them into smaller groups based on the similarity score.

Sanchez and Moreno [61] proposed another statistical approach for exploiting taxonomic relations using search engine. First, an initial keyword is submitted to a search engine to retrieve related Web pages. Then, these Web pages are analyzed to find the related concepts of the domain. The identified concepts are then submitted to the search engine again to retrieve the second collection of Web pages. The important bigrams in the second collection are then identified and considered as the sub-concepts of those in the first collection. There are five attributes to consider for the selection of representative concepts from the corpus. These include the total number of occurrences, total number of Web pages containing the concept, total number of Web pages returned by the search engine, and total number of web pages returned by the search engine containing both the selected concept and the initial keyword.

Bansal et al. [2] presented a supervised learning approach to infer taxonomic relations from a list of given terms. Their method is based on the belief propagation approach with features selected from certain patterns. These patterns are automatically learned through the training from Web pages and Wikipedia abstracts. In this method, the hypernym and sibling relationships are also incorporated to help determine the evidences used for weighting and balancing in the probabilistic framework.

Knijff et al. [34] used two methods, namely statistics-based subsumption and hierarchical agglomer-

ative clustering, to identify taxonomic relations. The statistics-based subsumption method is based on the assumption on term co-occurrences. Given two terms x and y , the idea of the subsumption method is that if we have $P(x|y) > t$ and $P(y|x) < t$, where t is a co-occurrence threshold, then x is a potential parent term of y . The hierarchical agglomerative clustering method starts with n separate clusters, with each containing one term. For each subsequent step, two nearest clusters are combined until only one cluster that contains all the terms remains. Similar approach is also used in the work of Krishnan et al. [39] and Dietz et al. [13]. The distances between clusters are determined based on the average linkage, where the distances between terms are determined by document co-occurrence similarity and window-based similarity. More specifically, the document co-occurrence between two terms t_1 and t_2 is defined as follows:

$$doc_sim(t_1, t_2) = \frac{2 \times df(t_1, t_2)}{df(t_1) + df(t_2)},$$

where $df(x)$ is the number of documents containing x . The window-based similarity between two terms t_1 and t_2 is defined as follows:

$$win_sim(t_1, t_2) = \frac{2 \times wf(t_1, t_2)}{wf(t_1) + wf(t_2)},$$

where $wf(x)$ is the number of windows containing x , based on the window size.

Paukkeri et al. [56] proposed a two-step approach to learn the taxonomy from a set of textual documents. In the first step, three feature extraction methods are used to extract taxonomic relations: fuzzy combination of criteria, statistical keyphrase extraction, and tf-idf term weighting. In the second step, a hierarchical clustering method based on Self-Organizing Map (SOM) is used to learn the concept hierarchy. This study, however, has imposed a restriction on the input corpus, i.e. each document must be a description for only one domain term.

In [44], Liu et al. proposed an approach to extract taxonomic relations from a list of keyword phrases. This approach is a deterministic supervised learning method using Bayesian Rose Tree, which is a multi-branch hierarchical clustering approach. At the beginning, each keyword phrase is represented as a tree. Then, it finds pairs of trees to merge based on the likelihood of data and calculates the likelihood score of the merged tree. The steps are repeated until there is only one tree left.

Yao et al. [79] introduced a statistical method to construct taxonomy from dynamic tag space using two proposed graphs: tag co-occurrence graph and tag association rule graph. This approach has made the assumption that a tagging action must adhere to the following model: $\langle user, resource, tag, time \rangle$, meaning that the *user* tags the *resource* with *tag* at specific *time*. These graphs are based on the following three measurements: frequency, support and confidence. Frequency measures the number of tags appearing in the tag space. Support measures the number of tagging actions containing both specific tags tag_i and tag_j . Confidence measures how confidence the tag_i happens when tag_j is given. The definitions for frequency, support and confidence are given as follows:

$$Frequency(tag) = |\{(User, Resource, Tag) | tag \in Tag\}| ,$$

$$Support(tag_i, tag_j) = |\{(User, Resource, Tag) | tag_i, tag_j \in Tag\}| ,$$

$$Confidence(tag_i \rightarrow tag_j) = \frac{Support(tag_i, tag_j)}{Frequency(tag_i)} .$$

There are some other studies using distribution models for taxonomic relation identification such as the work of Weeds and Weir [73], Kotlerman et al. [36], Lenci and Benotto [41], Santus et al. [62], Roller et al., [59], Rimell [57] and Kruszewski et al. [40]. The main idea of these methods is based on the Distributional Inclusion Hypothesis, i.e. the contexts of a hyponym are most likely to be included in those of its hypernym. For example, in Weeds et al. [74], they proposed a simple metric to compute the weighted inclusion between the set of features X and set of features Y as follows:

$$WeedsInclusion(X, Y) = \frac{\sum_{f \in F_X \cap F_Y} w_Y(f)}{\sum_{f \in F_Y} w_Y(f)} ,$$

where F_X and F_Y are the sets of features of X and Y respectively, and $w_X(f)$ and $w_Y(f)$ are the weights of feature f for X and Y respectively.

In general, statistical approaches usually achieve a high recall as it can infer implicit taxonomic relations in the corpus. Such approaches, however, usually suffer from low accuracy, but with relatively high coverage, and heavily depend on the choice of feature types and datasets.

2.2.2 Linguistic Approach

Most of the linguistic approaches, which are inspired by the work of Hearst [29], rely on the lexical-syntactic patterns to find taxonomic relations.

Kozareva et al. [38] proposed an automatic method for identifying the hyponym-hypernym pair in a domain corpus by matching the sentences collected from the Web with pre-defined patterns using the bootstrapping approach. In this study, it uses the pattern “<ROOT> *such as* <seed> *and* *”, where *ROOT* is a general term (which is usually located in the high level of taxonomy), and *seed* is an input specific term which can be an instance, or an intermediate term of the taxonomy. The method is then processed in an iterative way. The pattern is submitted to a Web search engine, and all the terms located in the * location are collected. The learned new terms are then placed in the position of the *ROOT*, and this is repeated until no new term is discovered. The collection of terms are then ranked by the proposed *inDegree* and *outDegree* metric which explores the hypernym relations between terms based on co-occurrence frequency.

Wentao et al. [75] proposed a novel iterative learning framework to identify the taxonomic relations with high precision and recall. The method consists of two phases: information extraction, and data cleansing and integration. In information extraction, it relies on the pre-defined Hearst syntactic patterns [29] which are shown in Table 2.1. These patterns are then submitted to the Web search engine to gather all the pairs of terms/subterms.

Table 2.1: Hearst Patterns

<i>NP such as {NP,*}{(or and)} NP</i>
<i>such NP as {NP,*}{(or and)} NP</i>
<i>NP{,} including {NP,*}{(or and)} NP</i>
<i>NP{,NP}*{,} and other NP</i>
<i>NP{,NP}*{,} or other NP</i>
<i>NP{,} especially {NP,*}{(or and)} NP</i>

In data cleansing and integration, the senses of each term corresponding to the patterns are defined.

Let t_i denote an interpretation on term t with sense i . Two interpretations t_i and t_j are equivalent if and only if $i = j$. For an edge (x, y) , if (x_i, y_j) holds, then (x_i, y_j) is an interpretation of (x, y) and denoted as $(x, y) \vdash (x_i, y_j)$. The senses of the term in the sentence are identified using some syntactic properties and derivation rules. These senses are then horizontally and vertically merged using a similarity function (e.g. Jaccard metric) to form the hierarchy.

In the work of Widdows and Dorow [76], and Girju et al. [25], the syntactic patterns, however, are chosen automatically based on an automatic bootstrapping method. This method is a supervised learning process which consists of three steps. In the first step, it generalizes the training examples using the semantic classes *part* and *whole* of WordNet to build the generalized training corpus. Then, it learns the constraints for non-ambiguous examples using a machine learning algorithm (e.g. C4.5) with features from the semantic classes of the WordNet identified in the previous step. Finally, it specializes the ambiguous examples by specializing the ambiguous terms with specific values.

Navigli et al. [53] proposed another pattern matching method for taxonomic relation identification which relies on structural pattern recognition. In this method, they proposed a structural semantic interconnection algorithm which relies on WordNet synsets to solve the problem of word sense disambiguation. The sense of each word is assigned to the sentence by some annotated corpora such as SemCor and LDC/DSO. There are three kinds of senses used in their approach: *gloss* which is a textual sense definition; *hyperonymy links* which are the direct or indirect hypernyms of a term according to WordNet; and *meronymy relations* which are the *part-of* relations in WordNet.

Velardi et al. [71] proposed another approach to automatically learn the lexical-syntactic patterns using machine-learned classifiers. First, a set of upper terms is chosen from the topmost synsets in WordNet. Then, a *Word-Class Lattices (WCL)* algorithm is used to identify all the definition sentences of these upper terms in the corpus. To train the extraction algorithm, a data set of textual definitions is manually annotated with some pre-defined fields such as *DEFINIENDUM*, *DEFINITOR* or *DEFINIENS*. Table 2.2 shows some example patterns for the *DEFINITOR* part. After that, a star pattern matching algorithm and sentence clustering method are then applied to learn the model classifier, which will be used subsequently for pattern learning.

Table 2.2: Some Example Patterns for the *DEFINITOR* Field

is a term used to describe	is a specialized form of
is the genus of	was coined to describe
is a term that refers to a kind of	is a special class of
can denote	is the extension of the concept of
is commonly used to refer to	is defined both as

In [49], Medelyan et al. proposed an approach to extract taxonomic relations by linking and mapping the extracted terms to other sources such as Freebase, DBPedia, Wikipedia articles or other available domain taxonomies. Such approach, however, is very limited in term of coverage as terms do not appear in sources will not be counted. Another shortcoming of this approach is that it only involves terms which are the names of people, organizations, and locations.

Rios-Alvarado et al. [58] presented a method to construct taxonomy using Web content. This method uses two types of information: contextual information and supervised information. For contextual information, it is calculated based on the Pointwise Mutual Information (PMI) between two nouns from the corpus. For supervised information, given a term, it is extracted as the most closest synset of this term according to the WordNet. These information will be joined together with the original term and incorporated with the Hearst pattern to query a Web search engine. The search results are then used to determine the taxonomic relationships between terms.

In general, the pattern matching methods usually achieve high precision, but with low coverage due to the lack of contextual analysis across sentences.

2.3 Taxonomy Induction

Most of the taxonomic relation identification approaches end up with a list of taxonomic relations [18] or an existing taxonomy extended by inserting new concepts into the hierarchy [66]. Wentao et al. [75] built a connectivity graph for terms using the probabilistic method. However, as they

assume that the graph pruning activity may be harmful, all identified relations are kept in the graph, and the taxonomic relations between nodes are determined by applying Bayesian probability, using the following two proposed metrics: plausibility and typicality. As such, the graph may contain a lot of noise, and redundant and even incorrect edges.

Zhu et al. [82] proposed a graph-based method to construct a taxonomy with real time update support. The method is based on edge weighting using topic relatedness and an optimum branching algorithm. First, they find a set of ground terms (usually general meaning terms) in some sources such as WordNet and Wikipedia's pageants. Then, they make an assumption that if a term is related to many ground terms, it is more important and thus has more weight. The edge's weight is constructed based on the weight of two connected nodes. Their algorithm, however, repeatedly inserts one node into the graph at each step, which is ineffective for constructing large-sized taxonomy.

Kozareva and Hovy [37] proposed a graph-based algorithm to organize taxonomic relations into a graph, using the longest path search between the root and the input term. First, the algorithm eliminates all cycles, which contains incorrect results from taxonomic relation identification, in the graph. Then, they collect all the nodes that do not have any predecessor or successor. The idea comes from two observations: (1) nodes have no predecessor are usually located at higher level of the taxonomy; and (2) nodes have no successor are usually located at lower level of the taxonomy. With these observations, for each pair of nodes, the algorithm finds the list of all paths connecting them. Finally, the algorithm connects the pair of nodes using the longest path between them.

Velardi et al. [71] proposed another method to induce a taxonomy. The method contains four steps: graph trimming, edge weighting, optimal branching and pruning recovery. First, they manually find the list of initial terminologies, which contains mostly the high level terms in WordNet. The graph trimming step is to trim all the nodes in the graph which do not connect to those initial terminologies. In the edge weighting step, the edges connecting pair of nodes are weighted using the path length between them. The optimal branching pruning is thus reduced to find the balance between path length and node's connectivity.

A common assumption is generally made in most taxonomy induction methods is that all the identi-

fied taxonomic relations are treated equally. This assumption, however, is not always true due to the fact that the identified taxonomic relations may have different confidence values, and the relations with high confidence values can be incorrectly eliminated during the pruning process.

2.4 Summary

In this chapter, we have reviewed the different methods that are widely used for domain term extraction. Each algorithm has its own merits and limitations, but are mostly dependent on the selected corpora. We have also reviewed the current methods for taxonomic relation identification, which can be classified into two categories: statistical and linguistic approaches. In addition, this chapter has also reviewed the current techniques for taxonomy induction.

Chapter 3

Taxonomy Construction using Syntactic Contextual Evidence and Web Data

There are three general steps for creating a domain-specific taxonomy from scratch: (1) extracting domain terms for the taxonomy; (2) identifying taxonomic relations between the domain terms; and (3) constructing the taxonomy structure. Even though these steps have been extensively researched in previous work of taxonomy construction, there are still challenges for further improving their performance.

The previous methods for domain term extraction from text corpora often utilize statistical measures such as *tf-idf*, document relevance (DR), and document consensus (DC) [54]. However, these measures are dependent on the selected corpora, and it is not trivial to construct balanced corpora.

There are many proven techniques from the established fields, such as information retrieval, machine learning as well as natural language processing, which have all contributed to the task of identifying taxonomic relations. These techniques can be generally classified into two main categories: statistical and linguistic approaches. The former includes co-occurrence analysis [8], term subsumption [20] and clustering [77]. The main idea behind these techniques is that terms that frequently co-occur may have taxonomic relationships. Such approaches, however, usually suffer from

low accuracy, but with relatively high coverage, and are heavily dependent on the choice of feature types and datasets. Most previous methods of the linguistic approach, on the other hand, rely on the lexical-syntactic patterns (e.g. *A is a B*, *A such as B*) [29]. These patterns can be manually created [38, 75], chosen via automatic bootstrapping [76, 25], or identified from machine-learned classifiers [54, 65]. The pattern matching methods generally achieve high precision, but with low coverage due to the lack of contextual analysis across sentences.

The identified taxonomic relations can be integrated into a graph for the task of taxonomy construction from scratch or associated with existing concepts of a given taxonomy via is-a relations [66]. In this step of taxonomic structure construction, there is a need for pruning incorrect and redundant relations. Previous methods for the pruning task [37, 71] treat the identified taxonomic relations equally, and the pruning task is thus reduced to finding the best trade-off between path length and the connectivity of traversed nodes. This assumption, however, is not always true due to the fact that the identified taxonomic relations may have different confidence values, and the relations with high confidence values can be incorrectly eliminated during the pruning process.

In this chapter, we will discuss our proposed approach for domain term extraction, taxonomic relation identification and taxonomy induction, and present the performance evaluation of the proposed approach.

3.1 Proposed Approach

In this chapter, we use the big data approach to address the challenges in the three steps of taxonomy construction.

Firstly, we propose a Web-based method for domain term extraction so that it can utilize the evidence from the Web rather than relying on the choice of balanced corpora.

Secondly, we propose a novel statistical method and show that when combined with a pattern matching method, it shows significant performance improvement. The proposed statistical method, called syntactic contextual subsumption (SCS), compares the syntactic contexts of terms for taxonomic

relation identification, instead of using the bag-of-words model by previous statistical methods. We observe that the terms in taxonomic relations may not occur in the same sentences, but in similar syntactic structures of different sentences, and that the contexts of a specific term are often found in the contexts of a general term but not vice versa. By context of a term, we refer to the set of words that frequently has a particular syntactic relation (e.g. *Subject-Verb-Object*) with the term in a given corpus. Given two terms, the SCS method collects from the Web pre-defined syntactic relations of each of the terms and checks if the syntactic contexts of a term properly include that of the other term in order to determine their taxonomic relation. The method scores each taxonomic relation candidate based on the two measures of Web-based evidence and contextual set inclusion. As such, it is able to find implicit subsumption relations between terms across sentences. As will be discussed in Section 3.5.3, the SCS method is shown to be complementary to linguistic pattern matching method.

Thirdly, we propose a novel method for taxonomy induction by utilizing the evidence scores from the relation identification method and the topological properties of graph. We will show that it can effectively prune redundant edges and remove loops while preserving the correct edges of the taxonomy.

Finally, we apply the proposed methods to the task of constructing a taxonomy from a given text collection from scratch. The resultant system consists of three modules: Domain Term Extraction (Section 3.2), Taxonomic Relation Identification (Section 3.3), and Taxonomy Induction (Section 3.4). The outputs of the domain term extraction module are used as inputs of the taxonomic relation identification, such that the taxonomic relation identification module checks if there is a taxonomic relation between each pair of terms from the term extraction module. The taxonomy induction module gets the identified taxonomic relation set as the input, and outputs the final optimal taxonomy by pruning redundant and incorrect relations.

3.2 Domain Term Extraction

This step includes collecting candidate terms and selecting the relevant terms for the domain of interest. Following most of the linguistic approaches, we use pre-defined linguistic filters to collect candidate terms, including single-word terms and multi-word terms which are noun or noun phrases in sentences. These terms are then preprocessed by removing determiners and lemmatization.

The collected candidate terms are then filtered to select the terms that are most relevant to the domain of interest. Many statistical techniques are developed for the filtering process such as $tf-idf$, domain relevance (DR), and domain consensus (DC) [53]. DR measures the amount of information that a term t captures within a domain of interest D_i , compared to other contrasting domains (D_j), while DC measures the distributed use of a term t across documents d in a domain D_i . As discussed earlier, these measures are dependent on the selected corpora, and it is not trivial to construct balanced corpora. To overcome this problem, we propose a new Web-based method for calculating DR . The idea is that instead of calculating the frequency of a term in the selected domains, we find its relative frequency in the Web, i.e. the number of search results returned by a Web search engine such as Google and Bing, because Web data can be considered as a substantially large and domain-independent corpus. The new Web-based DR measure can be characterized as follows:

$$DR_{Web}(t, D_i) = \frac{P(t|D_i)}{W(t|D_i)},$$

where $P(t|D_i)$ is defined as follows:

$$P(t|D_i) = \frac{f_{D_i}(t)}{\sum_{t' \in D_i} f_{D_i}(t')},$$

where $f_{D_i}(t)$ is the frequency of term t in D_i , and $W(t|D_i)$ is defined as follows:

$$W(t|D_i) = \frac{\log \mathfrak{R}_t}{\sum_{t' \in D_i} \log \mathfrak{R}_{t'}},$$

where $\mathfrak{R}_t$ is the total number of search results returned by the Web search engine using the query “ t ”. In this study, we use the Google search engine.

Since the three measures have its own advantage and drawback, and might be complementary to each other, our term extraction method is thus a linear combination of them:

$$TS(t, D_i) = \alpha \times tf-idf(t, D_i) + \beta \times DR_{Web}(t, D_i) + \gamma \times DC(t, D_i). \quad (3.1)$$

We experimented (see Section 3.5) with different values of α , β and γ , and found that the method achieves the best performance when the values for α and β are 0.2 and 0.8 respectively, and the value for γ is between 0.15 and 0.35, depending on the size of the domain corpus.

3.3 Taxonomic Relation Identification

In this section, we present three methods for taxonomic relation identification. First, we discuss the two methods of string inclusion with WordNet and lexical-syntactic pattern matching, which were commonly used methods, with some modifications. Then, we propose our novel syntactic contextual subsumption method to find implicit relations between terms across sentences by using contextual evidence from syntactic structures and Web data. Finally, we combine these three methods linearly to form an integrated solution for taxonomic relation identification. Given two terms t_1 and t_2 , Table 3.1 summarizes the important notations used in this chapter.

Table 3.1: Notations

Notation	Meaning
$t_1 \gg t_2$	t_1 is a hypernym of t_2
$t_1 \approx t_2$	t_1 semantically equals or is similar to t_2
$t_1 \gg_{WN} t_2$	t_1 is a direct or inherited hypernym of t_2 according to WordNet
$t_1 \approx_{WN} t_2$	t_1 and t_2 belong to the same synset of WordNet

3.3.1 String Inclusion with WordNet (SIWN)

One simple way to check whether it is a taxonomic relation is to test string inclusion. For example, “terrorist organization” is a hypernym of “foreign terrorist organization”, as the former is a substring of the latter. We propose an algorithm to extend the string inclusion test by using WordNet, which is named as SIWN. Given a candidate general term t_g and a candidate specific term t_s , the SIWN algorithm examines t_g from left to right (designating each word in t_g to be examined as w_g) to check if there is any word w_s in t_s such that $w_g \approx_{WN} w_s$ or $w_g \gg_{WN} w_s$, and identifies the taxonomic relation between two terms if every word of t_g has a corresponding word in t_s (with at least one $\gg_{WN}$ relation). For example, consider the following two terms: “suicide attack” and “world trade center self-destruction bombing”. Because “attack” $\gg_{WN}$ “bombing” and “suicide” $\approx_{WN}$ “self-destruction”, according to the SIWN algorithm, we conclude that “suicide attack” is the hypernym of “world trade center self-destruction bombing”.

Given two terms t_1 and t_2 , the evidence score for the SIWN algorithm is calculated as follows:

$$Score_{SIWN}(t_1, t_2) = \begin{cases} 1 & \text{if } t_1 \gg t_2 \text{ via SIWN} \\ 0 & \text{otherwise} \end{cases} \quad (3.2)$$

3.3.2 Lexical-Syntactic Pattern (LSP)

Extending the ideas of Kozareva and Hovy [37] and Navigli et al. [54], we propose a method for extracting taxonomic relations by matching lexical-syntactic patterns to the Web data.

Definition 1 (Syntactic patterns) *Given two terms t_1 and t_2 , $Pat(t_1, t_2)$ is defined as the set of the following patterns:*

- “ t_1 such as t_2 ”;
- “ t_1 , including t_2 ”;
- “ t_2 is [a|an] t_1 ”;

- “ t_2 is a $[kind|type]$ of t_1 ”; and
- “ t_2 , $[and|or]$ other t_1 ”.

In the patterns, t_1 and t_2 can be replaced by the actual terms and $[a|b]$ denotes a choice between a and b .

Given a candidate general term t_1 and candidate specific term t_2 , the lexical-syntactic pattern (LSP) method works as follows:

1. Submit each phrase in $Pat(t_1, t_2)$ to a Web search engine as a query. Denote $C_{Web}(t_1, t_2)$ as the collection of search results.
2. Calculate the following evidence score:

$$Score_{LSP}(t_1, t_2) = \frac{\log(|C_{Web}(t_1, t_2)|)}{1 + \log(|C_{Web}(t_2, t_1)|)}. \quad (3.3)$$

3. If $Score_{LSP}(t_1, t_2)$ is greater than a threshold value, then $t_1 \gg t_2$.

While most lexical-syntactic pattern methods in the literature only consider the value of $|C_{Web}(t_1, t_2)|$ in checking $t_1 \gg t_2$ [75], we take into account both $|C_{Web}(t_1, t_2)|$ and $|C_{Web}(t_2, t_1)|$. The intuition of Equation (3.3) is that if t_1 is a hypernym of t_2 , then the size of $C_{Web}(t_1, t_2)$ will be much larger than that of $C_{Web}(t_2, t_1)$, which means the lexical-syntactic patterns are more applicable for the ordered pair (t_1, t_2) than (t_2, t_1) .

3.3.3 Syntactic Contextual Subsumption (SCS)

The LSP method performs well in recognizing the taxonomic relations between terms in the sentences containing those pre-defined syntactic patterns. This method, however, has a major shortcoming: it cannot derive taxonomic relations between two terms occurring in two different sentences.

We thus propose a novel syntactic contextual subsumption (SCS) method which utilizes contextual information of terms in syntactic structure (i.e. *Subject-Verb-Object* in this study) and Web data to infer implicit taxonomic relations between terms across sentences. Note that the chosen syntactic structure *Subject-Verb-Object* is identical to the definition of non-taxonomic relations in [9], where the *Verb* indicates non-taxonomic relations between *Subject* and *Object*. In this subsection, we first present the method to collect non-taxonomic relations. Then, we present in detail the ideas of the SCS method and how we can use it to derive taxonomic relations in practice.

Non-taxonomic Relation Identification

Following the previous approaches to non-taxonomic relation identification (e.g. [11]), we use the Stanford parser [33] to identify the syntactic structures of sentences and extract triples of (*Subject*, *Verb*, *Object*), where *Subject* and *Object* are noun phrases.

We further consider the following issues: First, if a term (or noun phrase) includes a preposition, we remove the prepositional phrase. However, if the headword of a term is a quantitative noun like “lot”, “many” or “dozen” and it is modified by the preposition “of”, we replace it with the headword of the object of the preposition “of”. For example, we can extract respectively the triples (*people*, *need*, *food*) and (*people*, *like*, *snow*) from the following sentences:

- “People in poor countries need food”, and
- “A lot of people like snow”.

Second, if the object of a verb is in a verb form, we replace it with, if any, the object of the embedded verb. For example, we can extract the triple (*soldier*, *attack*, *terrorist*) from the following sentence:

- “The soldiers continue to attack terrorists”.

Third, if a term has a coordinate structure with a conjunction like “and” or “or”, we split it into coordinated noun phrases and duplicate the triple by replacing the term with each of the coordinated

noun phrases. For example, we can extract the triples of $R(\text{girl}, \text{like}, \text{dog})$ and $R(\text{girl}, \text{like}, \text{cat})$ from the following sentence:

- “The girl likes both dogs and cats”.

Given two terms t_1, t_2 and a non-taxonomic relation r , some notations which will be used hereafter are given below:

- $R(t_1, r, t_2)$: t_1, r , and t_2 have a (*Subject, Verb, Object*) triple.
- $\Theta(t_1, t_2)$: The set of relations r such that there exists $R(t_1, r, t_2)$ or $R(t_2, r, t_1)$.

Syntactic Contextual Subsumption Method

The idea of the SCS method is derived from the following two observations:

Observation 1 *Given three terms t_1, t_2, t_3 , and a non-taxonomic relation r , if we have two triples $R(t_1, r, t_3)$ and $R(t_2, r, t_3)$ (or $R(t_3, r, t_1)$ and $R(t_3, r, t_2)$), t_1 and t_2 may be in taxonomic relation.*

For example, given two triples $R(\text{Al-Qaeda}, \text{attack}, \text{American})$ and $R(\text{Terrorist group}, \text{attack}, \text{American})$, a taxonomic relation $\text{Terrorist group} \gg \text{Al-Qaeda}$ can be induced. However, it is not always guaranteed to induce a taxonomic relation from such pair of triples, for example, $R(\text{animal}, \text{eat}, \text{meat})$ and $R(\text{animal}, \text{eat}, \text{grass})$. The second observation introduced hereafter will provide a higher chance to infer taxonomic relationship.

Definition 2 (Contextual set of a term) *Given a term t_1 and a non-taxonomic relation r , $S(t_1, r, \text{“subj”})$ denotes the set of terms t_2 such that there exists a triple $R(t_1, r, t_2)$. Similarly, $S(t_1, r, \text{“obj”})$ is the set of terms t_2 such that there exists a triple $R(t_2, r, t_1)$.*

Observation 2 *Given two terms t_1, t_2 , and a non-taxonomic relation r , if $S(t_1, r, \text{“subj”})$ mostly contains $S(t_2, r, \text{“subj”})$ but not vice versa, then most likely t_1 is a hypernym of t_2 . Similarly, if*

$S(t_1, r, \text{"obj"})$ mostly contains $S(t_2, r, \text{"obj"})$ but not vice versa, then most likely t_1 is a hypernym of t_2 .

For example, assume that $S(\text{animal}, \text{eat}, \text{"subj"}) = \{\text{grass}, \text{potato}, \text{mouse}, \text{insects}, \text{meat}, \text{wild boar}, \text{deer}, \text{buffalo}\}$ and $S(\text{tiger}, \text{eat}, \text{"subj"}) = \{\text{meat}, \text{wild boar}, \text{deer}, \text{buffalo}\}$. Since $S(\text{animal}, \text{eat}, \text{"subj"})$ properly contains $S(\text{tiger}, \text{eat}, \text{"subj"})$, we can induce $\text{animal} \gg \text{tiger}$.

Based on Observation 2, our strategy to infer taxonomic relations is to first find the contextual set of terms via the evidence of syntactic structures and Web data, and then compute the score of the set inclusion. The detail of the method is presented hereafter.

Definition 3 Given two terms t_1, t_2 and a non-taxonomic relation r , $C(t_1, t_2, r, \text{"subj"})$ denotes the number of terms t_3 such that there exists both triples $R(t_1, r, t_3)$ and $R(t_2, r, t_3)$. Similarly, $C(t_1, t_2, r, \text{"obj"})$ is the number of terms t_3 such that there exists both relations $R(t_3, r, t_1)$ and $R(t_3, r, t_2)$.

Given a pair of a candidate general term t_1 and a candidate specific term t_2 , we extract their non-taxonomic relations from corpora extracted from the Web, and use them to determine the taxonomic relation between t_1 and t_2 as follows:

1. Find from a domain corpus the relation r and type Γ such that:

$$C(t_1, t_2, r, \Gamma) = \max_{\substack{r' \in \Theta(t_1, t_2) \\ \Gamma' \in \{\text{"subj"}, \text{"obj"}\}}} C(t_1, t_2, r', \Gamma').$$

2. If type Γ is "subj", collect the first 1,000 search results of the query " t_1 r " using the Google search engine, designated as $\text{Corpus}_{t_1}^\Gamma$. In the same way, construct $\text{Corpus}_{t_2}^\Gamma$ with the query " t_2 r ". If Γ is "obj", two queries " r t_1 " and " r t_2 " are submitted instead to collect $\text{Corpus}_{t_1}^\Gamma$ and $\text{Corpus}_{t_2}^\Gamma$, respectively.
3. Find the sets of $S(t_1, r, \Gamma)$ and $S(t_2, r, \Gamma)$ from $\text{Corpus}_{t_1}^\Gamma$ and $\text{Corpus}_{t_2}^\Gamma$, respectively, using the non-taxonomic relation identification method discussed earlier.

4. Calculate the following evidence score for the SCS method:

$$Score_{SCS} = \left[\frac{|S(t_1, r, \Gamma) \cap S(t_2, r, \Gamma)|}{|S(t_2, r, \Gamma)|} + \left(1 - \frac{|S(t_1, r, \Gamma) \cap S(t_2, r, \Gamma)|}{|S(t_1, r, \Gamma)|} \right) \right] \times \log(|S(t_1, r, \Gamma)| + |S(t_2, r, \Gamma)|). \quad (3.4)$$

The basic idea of the contextual subsumption score in our method is that if t_1 is a hypernym of t_2 , then the set $S(t_1, r, \Gamma)$ will mostly contain $S(t_2, r, \Gamma)$ but not vice versa. The intuition of Equation (3.4) is inspired by Jaccard similarity coefficient. We then multiply the score with the log value of total size of two sets to avoid the bias of small set inclusion.

5. If $Score_{SCS}(t_1, t_2)$ is greater than a threshold value, then we have $t_1 \gg t_2$.

3.3.4 Combined Method

In our study, we linearly combine the three methods as follows:

1. For each ordered pair of terms (t_1, t_2) , calculate the total evidence score:

$$\begin{aligned} Score(t_1, t_2) = & \alpha \times Score_{SIWN}(t_1, t_2) \\ & + \beta \times Score_{LSP}(t_1, t_2) \\ & + \gamma \times Score_{SCS}(t_1, t_2). \end{aligned} \quad (3.5)$$

2. If $Score(t_1, t_2)$ is greater than a threshold value, then we have $t_1 \gg t_2$.

We experimented with various combinations of values for α , β and γ , and found that the method achieves the best performance when the value of α is 0.5, β is between 0.35 and 0.45, and γ is between 0.15 and 0.25, depending on the domain corpus size.

3.4 Taxonomy Induction

The output of the taxonomic relation identification module is a set of taxonomic relations T . In this section, we will introduce a graph-based algorithm (Algorithm 3.1) to convert the set T into

Algorithm 3.1 Taxonomy Induction Algorithm**Input:** T : the taxonomic relation set**Output:** V : the vertex set of resultant taxonomy E : the edge set of resultant taxonomy

```

1: Initialize  $V = \{ROOT\}$ ,  $E = \emptyset$ 
2: for each taxonomic relation  $(t_1 \gg t_2) \in T$  do
3:    $E = E \cup \{e(t_1, t_2)\}$ 
4:   if  $t_1 \notin V$  then
5:      $V = V \cup \{t_1\}$ 
6:   end if
7:   if  $t_2 \notin V$  then
8:      $V = V \cup \{t_2\}$ 
9:   end if
10:  if  $\nexists e(t_3, t_1) \in E$  with  $t_3 \neq ROOT$  then
11:     $E = E \cup \{e(ROOT, t_1)\}$ 
12:  end if
13:  if  $\exists e(ROOT, t_2) \in E$  then
14:     $E = E \setminus \{e(ROOT, t_2)\}$ 
15:  end if
16: end for
17: edgeWeighting( $V, E$ )
18: graphPruning( $V, E$ )

```

an optimal tree-structured taxonomy, as well as to eliminate incorrect and redundant relations. Denote $e(t_1, t_2)$ as a directed edge from t_1 to t_2 , the algorithm consists of three steps which will be described hereafter with the corresponding lines in Algorithm 3.1.

Step 1 - Initial hypernym graph creation (lines 1 - 16): This step is to construct a connected directed graph from the list of taxonomic relations. The idea is to add each taxonomic relation $t_1 \gg t_2$ as a directed edge from parent node t_1 to child node t_2 , and if t_1 does not have any hypernym term, t_1 will become a child node of the *ROOT* node. The result of this step is a connected graph containing all taxonomic relations with the common *ROOT* node.

Step 2 - Edge weighting (line 17): This step is to calculate the weight of each edge in the hypernym graph. Unlike the algorithm of Velardi et al. [71] and Kozareva and Hovy [37] where every taxonomic relation is treated equally, we assume the confidence of each taxonomic relation is different, depending on the amount of evidence it has. Thus, the edges of the hypernym graph will be weighted as follows:

$$w(e(t_1, t_2)) = \begin{cases} 1 & \text{if } t_1 = ROOT \\ Score(t_1, t_2) & \text{otherwise} \end{cases} \quad (3.6)$$

Note that the *Score* value in Equation (3.6) is determined by the taxonomic relation identification process described in Section 3.3.4.

Step 3 - Graph pruning (line 18): The hypernym graph generated in Step 1 is not an optimal taxonomy as it may contain many redundant edges or incorrect edges, which may form a loop when joined together with other edges in the graph. In this step, we aim at producing an optimal taxonomy by pruning the graph based on our edge weighting strategy. A maximum spanning tree algorithm, however, cannot be applied as the graph is directed. For this purpose, we apply Edmonds' algorithm [16] for finding a maximum optimum branching of a weighted directed graph. Using this algorithm, we can find a subset of the current edge set, which is the optimized taxonomy where every non-root node has in-degree 1 and the sum of the edge weights is maximized. Figure 3.1 shows an example of the taxonomy induction process. Figure 3.1(a) shows the initial weighted hypernym graph. Figure 3.1(b) shows the final optimal taxonomy, where we prune two redundant edges (*group, International terrorist organization*) and (*Militant group, Hezbollah*), and eliminate the loop by removing the incorrect edge (*Al-Qaeda, Terrorist organization*).

3.5 Performance Evaluation

In this study, we have conducted three experiments for performance evaluation. The first experiment evaluates the performance of our proposed domain term extraction approach. The second experiment evaluates the performance of our proposed taxonomy construction approach on the task

Figure 3.1: An Example of Taxonomy Induction

of constructing new taxonomies for the Terrorism and AI domains. The third experiment evaluates the performance of our proposed taxonomy construction approach compared with the gold-standard WordNet sub-hierarchies.

Parameter Setting: In the experiments, the threshold value we used for $Score_{LSP}$ is 1.9, $Score_{SCS}$ is 1.5 and $Score$ is 2.1.

3.5.1 Datasets

We evaluate our approach for taxonomy construction based on the following five datasets of document collections obtained from different domains:

- Artificial Intelligence (AI) domain: The corpus consists of 4,976 papers extracted from the IJCAI proceedings from 1969 to 2014 and the ACL archives from 1979 to 2014. The same dataset used in the work of Velardi et al. [71].
- Terrorism domain: The corpus consists of 293 reports from “Patterns of Global Terrorism

(1988-2003)”¹ and “Country Reports on Terrorism (2004-2014)”² of the US state department. Each report contains about 1,500 words.

- Animal, Plant and Vehicle domains: The corpora are extracted from Web pages crawled by the bootstrapping algorithm described in Kozareva et al. [38]. Navigli et al. [54] and Kozareva and Hovy [37] used these datasets to compare their outputs against WordNet sub-hierarchies.

3.5.2 Performance on Domain Term Extraction

We compare the three term extraction and filtering methods, namely *tf-idf*, original *DR-DC* and *DR_{Web}-DC* with our proposed method *TS* described in Section 3.2. To estimate the precision, we manually validate randomly 300 filtered terms extracted by each method from the Terrorism and AI domains. Table 3.2 shows the estimated precision of the four methods in the two domains. The *TS* method achieves an average precision of 90%, much higher than that of *tf-idf* and *DR_{Web}-DC*, which may support the combination of the two methods. The *DR_{Web}-DC* improves the accuracy in the terrorism domain by 6% compared to the original *DR-DC* method, though the two methods show similar performance in the AI domain.

Table 3.2: Performance of Different Term Extraction Methods on Accuracy

Method	Terrorism Domain	AI Domain
<i>tf-idf</i>	51% (152/300)	63% (189/300)
<i>DR_{Orig}-DC</i>	64% (192/300)	80% (239/300)
<i>DR_{Web}-DC</i>	70% (211/300)	79% (237/300)
<i>TS</i>	92% (276/300)	91% (272/300)

¹<http://www.fas.org/irp/threat/terror.htm>

²<http://www.state.gov/j/ct/rls/crt/index.htm>

3.5.3 Performance on Taxonomy Construction for AI and Terrorism Domains

Referential taxonomy structures such as WordNet or OpenCyc are widely used in semantic analytics applications. However, their coverage is limited to common well-known areas, and many specific domains like Terrorism and AI are not well covered in those structures. Therefore, an automatic method which can induce taxonomies for those specific domains from scratch can greatly contribute to the process of knowledge discovery.

First, we apply our taxonomy construction approach to the AI domain corpus. Then, we compared the taxonomy constructed by our approach with that obtained by Velardi et al. [71]. The comparison results are shown in Table 3.3. Note that in this comparison, to be fair, we use the same set of terms that was used in [71]. The results show that our approach can extract 9.8% more taxonomic relations and achieve 7% better term coverage than Velardi’s approach.

Table 3.3: Performance Comparison with Velardi’s Approach for Taxonomy Construction

Taxonomy	Our Approach	Velardi’s Approach
#vertex	1839	1675
#edge	1838	1674
Average depth	6.2	6
Max depth	10	10
Term coverage	83%	76%

We also apply our proposed approach to the Terrorism corpus. The proposed taxonomic relation identification algorithm extracts a total of 976 taxonomic relations, from which the taxonomy induction algorithm builds the optimal taxonomy. The total number of vertices in the taxonomy is 281, and the total number of edges is 280. The average depth of the trees is 3.1, with the maximum depth 6. In addition, term coverage (the ratio of the number of terms in the final optimal trees to the number of terms obtained by the term extraction method) is 85%.

To judge the contribution of each of the taxonomic relation identification methods described in Section 3.3 to the overall proposed approach, we alternately run the approach for the AI and Terrorism

Table 3.4: Number of Extracted Taxonomic Relations for Different Taxonomic Relation Identification Methods

Method	Terrorism Domain	AI Domain
SCS	484	1308
SIWN	301	984
LSP	527	1537
SIWN + LSP	711	2203
SCS + SIWN + LSP	976	3122

domains with different combinations of the three methods (i.e. SIWN, LSP, and SCS). The results are shown in Table 3.4. Note that we employ only the first two modules on term extraction and taxonomic relation identification, excluding the last module of taxonomy induction for this experiment. Table 3.4 shows the number of the taxonomic relations extracted by each of the combinations. Since SIWN and LSP are commonly used by previous taxonomic relation identification methods, we consider the combination of SIWN + LSP as the baseline of the experiment. The results in Table 3.4 show that the three methods are all well complementary to each other. In addition, the proposed SCS method can contribute up to about 27% - 29% of all the identified taxonomic relations, which were not discovered by the other two baseline methods.

We further evaluate the precision of each individual taxonomic relation identification method. For AI and Terrorism domains, we again run the system with each of the three methods and with all together, and then randomly select 100 extracted taxonomic relations each time. These selected taxonomic relations are then examined by two domain experts to check the correctness. The evaluation results are given in Table 3.5. Note that only the first two modules on term extraction and taxonomic relation identification are employed for this experiment. The SIWN and LSP methods achieve high precision because they are based on the gold-standard taxonomy hierarchy WordNet and on the well-defined patterns, respectively. In contrast, the SCS method ambitiously looks for terms pairs that share similar syntactic contexts across sentences, though the contextual evidence is restricted to certain syntactic structures, and thus has a slightly lower precision compared to the

Table 3.5: Performance of Taxonomic Relation Identification Methods on Precision

Method	Terrorism Domain	AI Domain
SCS	91%	88%
SIWN	96%	91%
LSP	93%	93%
SIWN + LSP	94%	92%
SCS + SIWN + LSP	92%	90%

other two methods.

In short, the SCS method is complementary to the baseline methods, significantly improving the coverage of the combined method, when its precision is comparable to those of the baseline methods. We perform the next experiment to show that the SCS method has synergistic impact to improve the F-measure of the combined methods.

3.5.4 Performance Comparison on Taxonomy Construction based on Curated Databases

In this experiment, we construct taxonomies for the three domains, namely Animal, Plant and Vehicle, and then check whether the identified relations can be found in the WordNet, and which relations in WordNet are not found by our method. Note that in this comparison, to be fair, we have changed our proposed algorithm to avoid using WordNet in identifying taxonomic relations. Specifically, in the SIWN algorithm, all operations of “ $\approx_{WN}$ ” are replaced with normal string-matching comparison, and all “ $\gg_{WN}$ ” relations are falsified. The evaluation uses the following measures:

$$Precision = \frac{\#relations\ found\ in\ WordNet\ and\ by\ the\ method}{\#relations\ found\ by\ the\ method},$$

$$Recall = \frac{\#relations\ found\ in\ WordNet\ and\ by\ the\ method}{\#relations\ found\ in\ WordNet}, \text{ and}$$

$$F\text{-measure} = 2 \times \frac{Precision \times Recall}{Precision + Recall}.$$

We also compare our results with those obtained by the approaches of Navigli et al. [54] and Kozareva and Hovy [37], where they also compared their resultant taxonomies against WordNet. In this comparison, all the three approaches (i.e. ours and the two previous approaches) use the same corpora and term lists. The comparison results are given in Table 3.6. “N.A.” value means that this parameter is not applicable to the corresponding method. The results show that our proposed approach achieves better performance than the other two state-of-the-art methods, in terms of both the number of correctly extracted taxonomic relations and the term coverage. Our approach has a slightly lower precision than that of [54] and [37] due to the SCS method, but it significantly contributes to improve the recall and eventually the F-measure over the other two methods.

Table 3.6: Performance Comparison for Taxonomy Construction against WordNet Sub-hierarchies

	Animal Domain			Plant Domain			Vehicle Domain		
Taxonomy	Our	Kozareva	Navigli	Our	Kozareva	Navigli	Our	Kozareva	Navigli
#Correct relations	2427	1643	N.A.	1243	905	N.A.	281	246	N.A.
Term coverage	96%	N.A.	94%	98%	N.A.	97%	97%	N.A.	96%
Precision	95%	98%	97%	95%	97%	97%	93%	99%	91%
Recall	56%	38%	44%	53%	39%	38%	69%	60%	49%
F-measure	71%	55%	61%	68%	56%	55%	79%	75%	64%

To judge the effectiveness of our proposed taxonomy induction algorithm described in Section 3.4, we compare it with the graph-based algorithm of Velardi et al. [71]. Recall that in this algorithm, they treat all taxonomic relations equally, and the pruning task is reduced to finding the best trade-off between path length and the connectivity of traversed nodes. For each of the five domains (i.e. Terrorism, AI, Animal, Plant and Vehicle), we alternately run the two taxonomy induction algorithms over the same taxonomic relation set produced by our taxonomic relation identification process. For Terrorism and AI domains, we randomly pick up 100 edges in each resultant taxonomy and ask two domain experts to judge for the correctness. For Animal, Plant and Vehicle domains, we check the correctness of the edges in the resultant taxonomies by comparing them against the corresponding sub-hierarchies in WordNet. The results are given in Table 3.7. The results show that the proposed taxonomy induction algorithm can achieve better performance than the algorithm

of Velardi et al. [71]. This may be due to the fact that our algorithm considers the scores of the identified taxonomic relations from the relation identification module. Thus, it is more precise in eliminating incorrect relations during the pruning process.

Table 3.7: Performance Comparison with Velardi et al. [71] for Taxonomy Construction

Domain	Percentage of Correct Edges	
	Our Algorithm	Velardi's Algorithm
Terrorism	94%	90%
AI	93%	88%
Animal	95%	93%
Plant	95%	92%
Vehicle	93%	92%

In addition, when comparing Tables 3.5 and 3.7, we can find that the precision of taxonomic relations after the pruning process is higher than that before the pruning process, which proves that the proposed taxonomy induction algorithm can effectively trim the incorrect relations of the Terrorism and AI taxonomies, leveraging the percentage of correct relations by 2% - 3% up.

For the SCS method, besides the triple *Subject-Verb-Object*, we also explore other syntactic structures like *Noun-Preposition-Noun* and *Noun-Adjective-Noun*. For example, from the sentence “*I visited Microsoft in Washington*”, the triple (*Microsoft, in, Washington*) is extracted using the *Noun-Preposition-Noun* structure. Similarly, from the sentence “*Washington is a beautiful city*”, the triple (*Washington, beautiful, city*) is extracted using the *Noun-Adjective-Noun* structure. We then use the triples for the contextual subsumption method described in Section 3.3.3, and test the method against the Animal, Plant and Vehicle domains. The results are then compared against WordNet sub-hierarchies. The experimental results are given in Table 3.8 which shows that the triples of *Subject-Verb-Object* give the best performance compared to the other two syntactic structures. It can be explained that the number of triples from the other two structures *Noun-Preposition-Noun* and *Noun-Adjective-Noun* are smaller than that of *Subject-Verb-Object*, as the number of *Verb* is much greater than the number of *Preposition* or *Adjective*.

Table 3.8: Performance Comparison of Three Syntactic Structures for Syntactic Contextual Subsumption Method

	Animal Domain			Plant Domain			Vehicle Domain		
Taxonomy	<i>S-V-O</i>	<i>N-P-N</i>	<i>N-A-N</i>	<i>S-V-O</i>	<i>N-P-N</i>	<i>N-A-N</i>	<i>S-V-O</i>	<i>N-P-N</i>	<i>N-A-N</i>
Precision	95%	68%	72%	95%	63%	66%	93%	59%	60%
Recall	56%	52%	47%	53%	41%	43%	69%	45%	48%
F-measure	71%	59%	57%	68%	50%	52%	79%	51%	53%

3.6 Summary

In this chapter, we have proposed a novel method for identifying taxonomic relations using contextual evidence from syntactic structure and Web data. This method is proved to be well complementary with previous methods of linguistic pattern matching. We also present a novel graph-based algorithm to induce an optimal taxonomy from a given taxonomic relation set. The experimental results show that our proposed approach can generally achieve better performance than the state-of-the-art methods.

Chapter 4

Trustiness and Collective Synonym/Contrastive Evidence for Taxonomy Construction

Previous studies on taxonomy construction capture information about potential taxonomic relations between concepts, rank the candidate relations based on the captured information, and integrate the highly ranked relations into a taxonomic structure. When identifying the taxonomic relation between a pair of two terms, they often utilize explicit information of related features which are related directly to the two terms such as hypernym patterns (e.g. A is a B, A such as B) [38], syntactic dependency [15], definition sentences [54], co-occurrence [82], syntactic contextual similarity [70], and sibling relations [2]. However, implicit features are not considered in those work. In taxonomy construction, we should also consider the following three implicit features:

- Trustiness: Not all sources are trustworthy (e.g. gossip, forum posts written by non-experts) [14]. The trustiness of source texts is important in taxonomic relation identification because evidence from unreliable sources can be incorrect. For example, the invalid taxonomic relation between “American chameleon” and “chameleon” is mistakenly more popular in the Web

than the valid taxonomic relation between “American chameleon” and “lizard”, and statistical methods without considering the trustiness may incorrectly extract the invalid relation.

- **Synonyms:** A concept may be expressed in multiple ways, for example with synonyms. Most previous works assumed that a term represents an independent concept, and did not consider other information about the concept such as synonyms. The lack of evidence from synonyms may hamper the ranking of candidate taxonomic relations. Navigli and Velardi [53] combined synonyms into a concept, but only for those obtained from WordNet, called synsets.
- **Contrastive terms:** We observe that if two terms are often contrasted (e.g. A but not B, A is different from B) [32], they may not have a taxonomic relation.

In this chapter, we will discuss our proposed approach for incorporating the trustiness of source texts and the collective evidence from synonym/contrastive terms into the taxonomy construction approach introduced in Chapter 3 and present its performance evaluation.

4.1 Proposed Approach

In this research, we propose an approach to incorporate trustiness and collective synonym/contrastive evidence into taxonomy construction.

Firstly, we calculate the trustiness score of each data source with the following four weights: importance (if it is linked by many pages), popularity (if it is visited by many users), authority (if it is from a creditable Web site) and accuracy (if it has many facts) so that evidence for taxonomic relations from unreliable sources will be discarded.

Secondly, we identify synonyms of two terms (t_1, t_2), whose taxonomic relation is being scrutinized, by matching queries such as “ t_1 also known as” against the Web to find t_1 ’s synonyms located next to the query matches (e.g. t in “ t_1 also known as t ”). We then collect the evidence for all taxonomic relations between t'_1 and t'_2 , where t'_i is either t_i or its synonym ($i \in \{1, 2\}$), and combine them to calculate the evidence score of the candidate taxonomic relation between t_1 and t_2 . Similarly, for

each pair of two terms (t_1, t_2), we collect their contrastive evidence by matching queries such as “ t_1 is not a type of t_2 ” against the Web, and use them to proportionally decrease the evidence score for taxonomic relation between contrasting terms.

In the subsequent sections, we will discuss how to incorporate trustiness (Section 4.2), and collective evidence from synonyms (Section 4.3) and contrastive terms (Section 4.4) into the taxonomy construction method introduced in Chapter 3.

4.2 Trustiness of the Evidence Data

We introduce our approach for estimating the trustiness of a given source text in Section 4.2.1 and then explain how to incorporate it into the taxonomy construction method.

4.2.1 Collecting Trustiness Score of the Evidence Data

Given a data source (e.g. Web page), we consider four aspects of trustiness as follows:

- Importance: A data source may be important if it is referenced by many other sources.
- Popularity: If a data source is accessed by many people, it is considered popular.
- Authority: If the data is created by a trusted agency, such as government and education institute, it may be more trustful than others from less trusted sources such as forums and social media.
- Accuracy: If the data contains many pieces of accurate information, it seems to be trustful.

The trustiness score is calculated as follows:

Importance

To measure the importance of a Web page d as data source, we use the Google PageRank score¹ ($Score_{PageRank}(d)$) that is calculated based on the number and quality of links to the page. The PageRank scores have the scale from 0 to 9, where a higher score means it is more importance than a lower one. Using this score, the importance of a page is calculated as follows:

$$Trust_{Imp}(d) = \frac{1}{10 - Score_{PageRank}(d)}. \quad (4.1)$$

Note that we use the non-linearity for PageRank score rather than just normalizing PageRank to 0-1. The reason is to vary the gaps between the important sites (which usually have the PageRank score value from 7-10) and the majority of unimportant sites (which usually have the PageRank score value of less than 5).

Popularity

We use Alexa's Traffic Rank² as the measure of popularity, $Score_{Alexa}(d)$, which is based on the traffic data provided by users in Alexa's global data panel over a rolling 3 month period. The Traffic Ranks are updated daily. A site's rank is based on a combined measure of unique visitors and page views. Using this rank, the popularity of a data source is calculated as follows:

$$Trust_{Pop}(d) = \frac{1}{\log(Score_{Alexa}(d) + 1)}. \quad (4.2)$$

We use log transform in the popularity score instead of, for example, linear scoring because we want to avoid the bias of the large gap between the Alexa scores of different sites (e.g. one site may have an Alexa score of 1000, but another may have a score of 100,000).

Authority

We rank the authority of a data source based on the Internet top-level domain (TLD). We observe that pages with limited and registered TLD (e.g. *.gov*, *.mil*, *.edu*) are often more credible than those

¹<http://searchengineland.com/what-is-google-pagerank-a-guide-for-searchers-webmasters-11068>

²<http://www.alexa.com/>

with open domain (e.g. *.com*, *.net*). Therefore, the authority score of a data source is calculated as follows:

$$Trust_{Auth}(d) = \begin{cases} 1 & \text{if TLD of } d \text{ is } .gov, .mil \text{ or } .edu \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

Note that there are reasons why we choose such implementation of Authority in an elementary way. First, we tried finer categorization of various domains, e.g. *.int* has score 1/3, *.com* has score 1/4, etc. However, the experimental results did not show much difference in performance. In addition, there is controversy on which open TLD domains are more trustful than others, e.g. it is difficult to judge whether or not a *.net* site is more trustful than *.org*. Thus, we assign all open TLD domains to have the same score.

Accuracy

If the data source contains many pieces of accurate information, it will be trustful. Inspired by the idea of Dong et al. [14], we estimate the accuracy of a data source by identifying correct and incorrect information in the form of the triples (*Subject*, *Predicate*, *Object*) in the source, where *Subject*, *Predicate* and *Object* are normalized with regard to the knowledge base Freebase. The extraction of the triples includes six tasks: named entity recognition, part of speech tagging, dependency parsing, triple extraction, entity linkage (which maps mentions of proper nouns and their co-references to the corresponding entities in Freebase) and relation linkage. We use three information extraction (IE) tools (Angeli et al. [1], Manning et al. [47], MITIE³) for the first four tasks, and develop a method similar to Hachey et al. [26] for the last two tasks on entity linkage and relation linkage.

Since the IE tools may produce noisy or unreliable triples, we use a voting scheme for triple extraction as follows: A triple is only considered to be true if it is extracted by at least two extractors. After obtaining all triples in the data source, we use the closed world assumption as follows: Given subject *s* and predicate *p*, $O(s, p)$ denotes the set of such objects that a triple (*s*, *p*, *o*) is found in Freebase. Now given a triple (*s*, *p*, *o*) found in the data source, if $o \in O(s, p)$, we conclude that the

³<https://github.com/mit-nlp/MITIE>

triple is correct; but if $o \notin O(s, p)$ and $|O(s, p)| > 0$, we conclude that the triple is incorrect. If $|O(s, p)| = 0$, we do not conclude anything about the triple, and the triple is removed from the set of facts found in the data source.

Given a data source d , we define $cf(d)$ as the number of correct facts, and $icf(d)$ as the number of incorrect facts found in d . The accuracy of d is calculated as follows:

$$Trust_{Accu}(d) = \frac{1}{1 + icf(d)^2} - \frac{1}{1 + cf(d)^2}. \quad (4.4)$$

Combining Trustiness Scores

The final trustiness score of a data source is the linear combination of the four scores as follows:

$$\begin{aligned} Trust(d) = & \alpha \times Trust_{Imp}(d) + \beta \times Trust_{Pop}(d) \\ & + \gamma \times Trust_{Auth}(d) + \delta \times Trust_{Accu}(d). \end{aligned} \quad (4.5)$$

To estimate the optimal combination for parameters α , β , γ and δ , we apply the linear regression algorithm [28]. For parameter learning, we manually list 50 websites as trusted sources (e.g. *stanford.edu*, *bbc.com*, *nasa.gov*), and the top 15 gossip websites listed in a site⁴ as untrusted sources. Then, we use the scores of their individual pages by the four methods to learn the parameters in Equation (4.5). The learning results are given as follows: $\alpha=0.46$, $\beta=0.46$, $\gamma=2.03$ and $\delta=0.61$.

4.2.2 Integrating Trustiness into Taxonomic Relation Identification Methods

Given a data collection C , we define the average trustiness score of all data as follows:

$$AvgTrust(C) = \frac{\sum_{d \in C} Trust(d)}{|C|}.$$

Then, we integrate the trustiness score into the taxonomic relation identification methods described in Chapter 3.3 as follows:

⁴<http://www.ebizmba.com/articles/gossip-websites>

SIWN Method: This method does not use any evidence from the Web. So its measure does not change and is given as follows:

$$Score_{SIWN}^{Trust}(t_1, t_2) = Score_{SIWN}(t_1, t_2). \quad (4.6)$$

LSP Method: The LSP evidence score of the taxonomic relation between t_1 and t_2 is recalculated as follows:

$$\begin{aligned} Score_{LSP}^{Trust}(t_1, t_2) &= Score_{LSP}(t_1, t_2) \times \\ & (AvgTrust(C_{Web}(t_1, t_2)) + AvgTrust(C_{Web}(t_2, t_1))). \end{aligned} \quad (4.7)$$

The intuition of Equation (4.7) is that the original LSP evidence score is multiplied by the average trustiness score of all evidence documents for the taxonomic relation from the Web. If the number of Web search results is too large, we use only the first 1,000 results to estimate the average trustiness score.

SCS Method: Similarly, the SCS evidence score is recalculated as follows:

$$\begin{aligned} Score_{SCS}^{Trust}(t_1, t_2) &= Score_{SCS}(t_1, t_2) \times \\ & (AvgTrust(Corpus_{t_1}^\Gamma) + AvgTrust(Corpus_{t_2}^\Gamma)). \end{aligned} \quad (4.8)$$

Combined Method: The three measures of trustiness are also linearly combined as follows:

$$\begin{aligned} Score^{Trust}(t_1, t_2) &= \alpha \times Score_{SIWN}^{Trust}(t_1, t_2) + \\ & \beta \times Score_{LSP}^{Trust}(t_1, t_2) + \gamma \times Score_{SCS}^{Trust}(t_1, t_2). \end{aligned} \quad (4.9)$$

The values of α , β , and γ in Equation (4.9) are identical to those of Equation (3.5).

4.3 Collective Synonym Evidence

4.3.1 Synonymy Identification

We use the following three methods to collect synonyms: dictionaries, pattern matching, and supervised learning.

Dictionaries: Synonyms can be found in dictionaries like a general-purpose dictionary WordNet and also domain-specific ones. Since our domains of interest include virus, Animal, and Plant (see the next section for details), we also utilize MeSH⁵, a well-known vocabulary in biomedicine.

Pattern Matching: Given two terms t_1 and t_2 , we use the following patterns to find their synonymy evidence from the Web:

- t_1 also [known|called|named|abbreviated] as t_2 ;
- other common name[s] of t_1 [is|are|include] t_2 ;
- t_1 , or t_2 , is a; and
- t_1 (short for t_2).

The symbol $[a|b]$ denotes a choice between a and b . If the number of Web search results is greater than a threshold Ψ , t_1 will be considered as a synonym of t_2 .

Supervised Learning: We randomly pick 100 pairs of synonyms in WordNet, and for each pair, we use the Web search engine to collect sample sentences in which both terms of the pair are mentioned. If the number of collected sentences is greater than 2000, we use only the first 2000 sentences for training. After that, we extract the following features from the sentences to train a logistic regression model [28] for the synonymy identification:

- headwords of the two terms;
- average distance between the terms;
- sequence of words between the terms;
- bag of words between the terms;

⁵<http://www.ncbi.nlm.nih.gov/mesh>

- dependency path between the terms (using Stanford parser [33]); and
- bag of words on the dependency path.

The average F-measure of the obtained model with 10-fold cross-validation is 81%. We use the learned model to identify more synonym pairs in the next step.

4.3.2 Embedding Synonym Information

Given a term t , we denote $Syn(t)$ as the set of synonyms of t (including t itself). The evidence scores of the SCS and LSP methods are recalculated with synonyms as follows:

$$Score_{SCS}^{Synonym}(t_1, t_2) = \sum_{\substack{t'_1 \in Syn(t_1) \\ t'_2 \in Syn(t_2)}} Score_{SCS}(t'_1, t'_2), \quad (4.10)$$

where the variable X can be replaced with SCS and LSP.

The intuition of Equation (4.10) is that the evidence score of the taxonomic relation between two terms t_1 and t_2 can be boosted by adding all the evidence scores of taxonomic relations between them and their synonyms.

Again, as for the SIWN method, we do not change the evidence score which is given as follows:

$$Score_{SIWN}^{Synonym}(t_1, t_2) = Score_{SIWN}(t_1, t_2).$$

4.4 Contrastive Evidence

Given two terms t_1 and t_2 , we use the following patterns to find their contrastive (thus negative) evidence from the Web:

- t_1 is not a t_2 ;

- t_1 is not a [type|kind] of t_2 ;
- t_1 , unlike t_2 ;
- t_1 is different [from|with] t_2 ;
- t_1 but not t_2 ; and
- t_1 , not t_2 .

Let $WH(t_1, t_2)$ be the total number of Web search results on the contrastive evidence between t_1 and t_2 , the contrastive evidence score is then computed as follows:

$$Contrast(t_1, t_2) = \log(WH(t_1, t_2) + 1). \quad (4.11)$$

Similar to the collective synonym evidence, the contrastive evidence score of taxonomic relation between t_1 and t_2 is boosted with the contrastive evidence scores of taxonomic relations between the two terms and their synonyms as follows:

$$Score_{SCS}^{Contrast}(t_1, t_2) = \frac{\sum_{\substack{t'_1 \in Syn(t_1) \\ t'_2 \in Syn(t_2)}} Contrast(t'_1, t'_2)}{|Syn(t_1)| * |Syn(t_2)|}. \quad (4.12)$$

4.5 Combining Trustiness, Synonym and Contrastive Evidence

We combine all the three features into taxonomic relation identification discussed in Chapter 3 as follows:

$$Score_{SCS}^{Final}(t_1, t_2) = Score_{SCS}^{Trust}(t_1, t_2) + Score_{SCS}^{Synonym}(t_1, t_2), \quad (4.13)$$

where the variable X can be replaced with each of the three taxonomic relation evidence measures (i.e. SCS, LSP, SIWN). The final combined score is calculated as follows:

$$\begin{aligned} Score_{SCS}^{Final}(t_1, t_2) = & \alpha \times Score_{SCS}^{Trust}(t_1, t_2) \\ & + \beta \times Score_{SCS}^{Synonym}(t_1, t_2) - \gamma \times Score_{SCS}^{Contrast}(t_1, t_2) \\ & - \delta \times Score^{Contrast}(t_1, t_2). \end{aligned} \quad (4.14)$$

For each ordered pair of terms t_1 and t_2 , if $Score_{Combined}^{Final}(t_1, t_2)$ is greater than a threshold value, then t_1 is considered as a hypernym of t_2 .

We estimate the optimal values of parameters α , β , γ and δ in Equation (4.14) with the ridge regression technique [28] as follows: First, we randomly select 100 taxonomic relations in the Animal domain as the training set. For each taxonomic relation $t_1 \gg t_2$, its evidence score is estimated as $\tau + \frac{1}{Dist(t_1, t_2)}$, where τ is the threshold value for $Score_{Combined}^{Final}$, and $Dist(t_1, t_2)$ is the length of the shortest path between t_1 and t_2 found in WordNet. Then, we find evidence scores with taxonomic relation identification methods in Equation (4.12) and Equation (4.13). Finally, we build the training set using Equation (4.14), and use the ridge regression algorithm to learn that the best value for α is 1.31, β is 1.57, γ is 1.24 and δ is 0.79, where $\tau=2.3$.

4.6 Performance Evaluation

In this section, we have conducted two experiments for performance evaluation. The first experiment evaluates the performance of our proposed taxonomy construction approach on the task of constructing new taxonomies for the Finance and AI domains. The second experiment evaluates the performance of our proposed taxonomy construction approach in comparison with the curated databases. We also compare our current proposed approach (called Tuan2) with three other state-of-the-art methods in the literature, i.e. Kozareva et al. [37], Navigli et al. [54] and our taxonomy construction approach presented in Chapter 3 (called Tuan1). In addition, for Animal domain, we also compare with the reported performance of Bansal et al. [2], a recent work to construct taxonomy using belief propagation.

4.6.1 Datasets

We evaluate the performance of our proposed approach for taxonomy construction based on the following six datasets of document collections obtained from different domains:

- Artificial Intelligence (AI) domain: This is the same dataset used for performance evaluation in Chapter 3.
- Finance domain: The corpus consists of 1,253 papers extracted from the freely available collection of “Journal of Financial Economics” from 1995 to 2012 and “Review Of Finance” from 1997 to 2012.
- Virus domain: The corpus consists of 20,000 abstracts extracted from the search results which are retrieved by submitting the query “virus” to the PUBMED search engine ⁶.
- Animal, Plant and Vehicle domains: These are the same datasets used for performance evaluation in Chapter 3.

4.6.2 Performance on Taxonomy Construction for AI and Finance Domains

Table 4.1: Performance Results for the Finance and AI Domains

Method	Finance		AI	
	Precision	# Extracted Relations	Precision	# Extracted Relations
Kozareva	90%	753	94%	950
Navigli	88%	1161	93%	1711
Tuan1	85%	1312	90%	1838
Tuan2	88%	1570	92%	2168

To estimate the precision of a given method, we randomly choose 100 relations among the results of the method and manually check their correctness. The results summarized in Table 4.1 show that our proposed approach (named as Tuan2) extracts much more relations, though with slightly lower precision, than that of Kozareva et al. [38] and Navigli and Velardi [53]. Note that due to the lack of gold standards in these two domains, we do not compare the methods in terms of F-score, which will be used for the evaluation with curated databases in the next section. Compared to Tuan1

⁶<http://www.ncbi.nlm.nih.gov/pubmed>

approach, which can be considered as the baseline of our approach, our proposed method (Tuan2) has significant improvement in both precision and the number of extracted relations. It indicates that the three incorporated features of trustiness, and synonym and contrastive evidence are effective in improving the performance of existing taxonomy construction methods.

4.6.3 Performance on Taxonomy Construction based on Curated Databases

We evaluate automatically constructed taxonomies for four domains (i.e. Animal, Plant, Vehicle, Virus) against the corresponding sub-hierarchies of curated databases. For the Animal, Plant and Vehicle domains, we use WordNet as the gold standards, whereas for the Virus domain, we use MeSH sub-hierarchy of virus as the reference.

Note that in this comparison, to be fair, we change our algorithm to avoid using WordNet in identifying taxonomic relations (i.e. SIWN algorithm), and we only use the exact string-matching comparison without WordNet. The evaluation uses the following measures:

$$Precision = \frac{\#relations \text{ found in database and by the method}}{\#relations \text{ found by the method}},$$

$$Recall = \frac{\#relations \text{ found in database and by the method}}{\#relations \text{ found in database}}, \text{ and}$$

$$F\text{-measure} = 2 \times \frac{Precision \times Recall}{Precision + Recall}.$$

To understand the individual contribution of the three introduced features (i.e. trustiness, synonym, contrast), we also evaluate our approach only with one of the three features each time, as well as combining all the three features (denoted as Tuan2).

Table 4.2 and Table 4.3 summarize the experimental results. Our proposed approach (Tuan2) achieves significantly better performance than the previous state-of-the-art methods in terms of F-measure and Recall (t-test, p-value < 0.05) for all the four domains. For the Animal domain, it also shows higher performance than the reported performance of Bansal et al. [2]. In addition, the proposed approach improves the baseline (i.e. Tuan) by up to 4%-10% of F-measure.

Table 4.2: Performance Results for the Animal and Plant Domains

	Animal			Plant		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Kozareva	98%	38%	55%	97%	39%	56%
Navigli	97%	44%	61%	97%	38%	55%
Bansal	84%	55%	67%			
Tuan1	95%	56%	70%	95%	53%	68%
Trustiness	97%	61%	75%	97%	56%	71%
Synonym	92%	65%	76%	93%	58%	71%
Contrast	97%	55%	70%	97%	53%	69%
Tuan2	97%	65%	78%	96%	59%	73%

Table 4.3: Performance Results for the Vehicle and Virus Domains

	Vehicle			Virus		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Kozareva	99%	60%	75%	97%	31%	47%
Navigli	91%	49%	64%	99%	37%	54%
Tuan1	93%	69%	79%	93%	43%	59%
Trustiness	96%	72%	82%	97%	48%	64%
Synonym	91%	72%	80%	91%	53%	67%
Contrastive	97%	68%	80%	98%	42%	59%
Tuan2	95%	73%	83%	96%	54%	69%

Furthermore, we find that the three features have different contribution to the performance improvement. The trustiness feature contributes to the improvement on both precision and recall. The synonym feature has the tendency of improving the recall further than the trustiness, whereas the contrastive evidence improves the precision.

4.6.4 Performance on Trustiness and Synonymy Identification

We evaluate the individual methods for trustiness measurement and synonymy identification described in Sections 4.2.1 and 4.3.1. For this purpose, we evaluate our approach only with one of the individual methods at a time (i.e. importance, popularity, authority and accuracy for trustiness measure, and dictionary, pattern matching, and machine learning methods for synonymy identification).

As summarized in Table 4.4, the “Importance” and “Accuracy” methods for trustiness measurement based on PageRank and IE systems, respectively, have more contribution than the others. Similarly, the experimental results indicate that the “Machine learning” method has the most contribution among the three methods of synonymy identification.

Table 4.4: Performance Results on Trustiness and Synonymy Identification in Terms of F-measure

	Animal	Plant	Vehicle	Virus
Trustiness:				
Importance	74%	70%	81%	63%
Popularity	72%	69%	81%	61%
Authority	72%	69%	80%	61%
Accuracy	73%	70%	81%	62%
Importance + Accuracy	75%	70%	82%	64%
Synonym:				
Dictionaries	73%	69%	79%	62%
Pattern matching	74%	69%	80%	64%
Machine learning	74%	70%	80%	65%

In addition, we also examine the inter-dependence of the four aspects of trustiness by running the system with the combination of only two aspects, Importance and Accuracy. The results in all domains show that when combining only the Importance and Accuracy, the system almost achieves the same performance to that of the combined approach (i.e. Tuan2) with all four criteria, except for the Plant domain. It can be explained as the Importance aspect (which is expressed as the PageRank

score) may subsume the Popularity and Authority aspects. Another interesting point is that the performance of Accuracy, which is solely based on the local information from the website, when applied individually, is almost the same with that of Importance which is based on the distributed information. It shows that the method of ranking of the sites based on the knowledge-based facts can achieve the effectiveness as good as the traditional ranking method using PageRank score.

4.6.5 Discussion

Case Studies

We give two examples to illustrate how the proposed features help to infer correct taxonomic relations and filter out incorrect relations. Our baseline (Tuan1 from Chapter 3) extracts an incorrect taxonomic relation between ‘fox’ and ‘flying fox’ due to the following reasons: (1) ‘flying fox’ includes ‘fox’ (SIWN) and (2) untrusted sources such as a public forum⁷ support the relation. Using our proposed approach, this relation is filtered out because those untrusted sources are discouraged by the trustiness feature, and also because there are contrastive evidence⁸ saying that ‘flying fox’ is NOT a ‘fox’. Specifically, the average trustiness score of the LSP method of the sources for the invalid relation (i.e. $AvgTrust(C_{Web}(fox, flying\ fox)) + AvgTrust(C_{Web}(flying\ fox, fox))$) is 0.63, which is lower than the average trustiness score, 0.90. Also, the collective contrastive evidence score (i.e. $Score^{Contrast}(fox, flying\ fox)$) is 1.10, which is higher than the average collective contrastive score, 0.32.

On the other hand, the true taxonomic relation between ‘bat’ and ‘flying fox’ is not identified by the baseline, mainly due to the rare mention of this relation in the Web. However, our proposed approach can recognize this relation because of two reasons: (1) ‘flying fox’ has many synonyms such as ‘fruit bat’, ‘pteropus’, ‘kalong’, and ‘megabat’, and there are much evidence that these synonyms are kinds of ‘bat’ (i.e. using the collective synonym evidence). (2) The evidence for

⁷<http://redwall.wikia.com/wiki/User:Ferretmaiden/Archive3>

⁸<http://en.cc-english.com/index.php?shownews-1397>

the taxonomic relation between ‘flying fox’ and ‘bat’, though rare, is from trusted sites⁹ which are maintained by scientists. Thus, the trustiness feature helps to boost the evidence score for this relation over the threshold value. Specifically, the average trustiness score of the LSP method (i.e. $(AvgTrust(C_{Web}(bat, flying\ fox)) + AvgTrust(C_{Web}(flying\ fox, bat))))$, 2.84, is higher than the average trustiness score, 0.90.

We further investigate on 256 taxonomic relations that were missed by the baseline but correctly identified by the proposed approach. The average $Score_{LSP}$ and the average $Score_{SCS}$ of the relations by the baseline are 0.35 and 0.60, respectively, while those by the proposed approach are 1.17 and 0.82, respectively. We thus find that the proposed approach is more effective in correctly improving the LSP method than the SCS method.

Empirical Comparison with WordNet

By error analysis, we find that our results may complement WordNet. For example, in the Animal domain, our approach identifies ‘wild sheep’ as a hyponym of ‘sheep’, but in WordNet, they are siblings. However, many references^{10, 11} consider ‘wild sheep’ as a species of ‘sheep’. Another such example is that our approach recognizes ‘aquatic vertebrate’ as a hypernym of ‘aquatic mammal’, but WordNet places them in different subtrees incorrectly¹². Therefore, our results may help restructure and extend WordNet.

Threshold Tuning

Our scoring methods utilize several thresholds to select relations of high ranks. The threshold value Ψ for the pattern matching method in Section 4.3.1 controls the number of synonymy relations extracted from text. The threshold value for $Score_{Combined}^{Final}$ of Equation (4.14) in Section 4.5 controls

⁹<http://krjsoutheastasianrainforests.weebly.com/Animal-in-biome-and-habitat-structures.html>

¹⁰<http://en.wikipedia.org/wiki/Ovis>

¹¹<http://www.bjornefabrikken.no/side/norwegian-sheep/>

¹²http://en.wikipedia.org/wiki/Aquatic_mammal

the number of extracted taxonomic relations. In general, the larger these threshold values are, the higher number of synonyms and taxonomic relations we can get. In our experiments, we found that the threshold values for Ψ between 100 and 120, and those for $Score_{Combined}^{Final}$ between 2.3 and 2.5 generally help the approach achieve the best performance.

4.7 Summary

In this chapter, we have proposed an approach for taxonomic relation identification that incorporates the trustiness of source texts measured with techniques such as PageRank and knowledge-based trust, and the collective evidence of synonyms and contrastive terms identified by linguistic pattern matching and machine learning. The experimental results show that the proposed features can improve the performance of taxonomy construction by up to 4%-10% on F-measure.

Chapter 5

Temporal Information for Taxonomy Construction

Previous works on automatic construction of taxonomy from textual documents assumed that the datasets (that is, the document sets) and the underlying taxonomic relations are *static*. However, the datasets for certain domains may evolve over time, as new documents are added while older documents may be deleted or modified. As such, the taxonomic relations for these potentially fast-changing domains may not remain static but become *dynamic* over time as new domain terms emerge while some older ones disappear. For example, in the reports of WHO about *disease outbreak*, the term ‘*smallpox*’ used to be a hyponym of ‘*dangerous disease*’, but it has fallen off since 1980. On the other hand, the term ‘*Ebola*’ has now become an emerging hyponym of ‘*dangerous disease*’ since 2014. In another example, in the collection of US yearly reports of terrorism, the term ‘*Palestine Liberation Organization*’ used to be a hyponym of ‘*terrorist group*’ until 1992, but it is no longer true nowadays. ‘*Palestine Liberation Organization*’ should now be classified as a ‘*national organization*’ of Palestine.

When the temporal information in datasets are not captured, the resultant taxonomy may be incomplete or outdated and misleading. This could be caused by the overwhelming evidence of older patterns/contexts compared to emerging, but relatively small amount of, evidence of newer rela-

tions. For example, in the taxonomy of US yearly reports on terrorism, many previous methods might fail to recognize the taxonomic relation between the two terms ‘*ISIS*’ and ‘*terrorist group*’ due to relatively infrequent mentions of ‘*ISIS*’ (which only appears in reports from 2014). Meanwhile, ‘*Palestine Liberation Organization*’ could still be classified as a hyponym of ‘*terrorist group*’ because of the relatively larger number of mentions in the documents from the earlier years.

In this chapter, we discuss our proposed time-aware approach for taxonomy construction from a time series of textual documents for a particular domain and present the performance of the proposed approach.

5.1 Time-aware Taxonomy Construction

We define the root term of a taxonomy as a word or phrase that represents the domain of interest. It can be any informative concept such as an entity (‘*MH370*’) or an event (‘*Ebola outbreak*’). Given a root term $\mathcal{R}$, we define a corpus $\mathcal{C}$ as a pre-clustered set of a time series of textual documents based on $\mathcal{R}$.

Given two terms t_1 and t_2 , we denote $t_1 \rightarrow t_2$ as a taxonomic relation where t_1 is a hypernym of t_2 . In this work, we define a taxonomy as a triple $\mathcal{H} = (V, E, s)$, where:

- V is the set of taxonomy’s vertices, i.e. the set of terms, including the root term.
- E is the set of taxonomy’s edges, i.e. the set of taxonomic relations.
- s is the creation time of taxonomy. It can be the current date or any specified time.

Our task is formally defined as follows: *Given a root term $\mathcal{R}$, a corpus $\mathcal{C}$ and an optional existing taxonomy $\mathcal{H}_1 = (V_1, E_1, s_1)$ constructed at time s_1 , we aim to build a new taxonomy $\mathcal{H}_2 = (V_2, E_2, s_2)$ at time s_2 , where $s_2 > s_1$, so that we can process the document set in $\mathcal{C}$ up to time s_2 into the relevant terms in the taxonomy.*

If $\mathcal{H}_1$ does not exist, the problem becomes creating a taxonomy $\mathcal{H}_2$ for corpus $\mathcal{C}$. Otherwise, the problem is to update the existing taxonomy with the newly obtained data for corpus $\mathcal{C}$.

5.2 Proposed Approach

In the proposed approach, we incorporate the timestamp information into the process of identifying taxonomic relations by computing evidence scores of the data sources weighted by a timestamp contribution function [17, 43] to capture the temporally-varying contributions of evidence from the various documents at a particular point in time. We assume that newer evidence is more important than older evidence. For example, the evidence that ‘*Palestine Liberation Organization*’ was a hyponym of ‘*terrorist group*’ occurred in 1990 is less important now than the evidence that ‘*ISIS*’ is a hyponym of ‘*terrorist group*’ occurred in 2014. We incorporate the timestamp contribution function into taxonomy construction to measure the weights of the evidence for both statistical and linguistic methods. With such built-in time-awareness for taxonomy construction, we ensure that the constructed taxonomies are up-to-date for the fast-changing domains found in newswire and social media, where users constantly search for updated relations and track information trends.

Most previous work requires re-running the taxonomy construction process whenever there are new incoming data. Our proposed approach enables incremental update of the constructed taxonomies to avoid costly reconstructions. We incorporate an information decay function [64] to manage outdated relations in the constructed taxonomy. The decay function measures the extent that the relation is out-of-date over time, and we incorporate it into a time-aware graph-based algorithm for taxonomy update for this work.

Figure 5.1 shows the proposed approach for time-aware taxonomy construction. There are two key processes in the proposed approach: temporal information processing and taxonomy construction.

Figure 5.1: Proposed Approach for Time-aware Taxonomy Construction

5.3 Temporal Information Processing

The aim of the temporal information processing process is to generate temporal information (or timestamp for short) for each sentence in the input document or Web data.

Previous taxonomy construction methods [82] only extract temporal information at the document level, i.e. all information in the document has the same timestamp as the document creation date. This assumption, however, is not always correct. Figure 5.2 shows a sample report about the flight *MH370* created on 30 July 2015. From the report, the timestamp of each sentence is very different from the document creation date. If we were to simply use the temporal information at document level, the timestamps for the search areas of *MH370* at different periods would be incorrect. Thus, we propose a method to extract timestamps (i.e. temporal expressions) at the sentence level. The method comprises the following three steps:

Document Creation Date Extraction: First, we extract the timestamp at the document level. The text corpus that we are using for this study consists of a collection of reports, scientific publications and Web search results. For the first two types of documents, the timestamp is the document creation date that can be extracted directly from the data source, i.e. the date of the report, or the date of the publication (e.g. journal/conference date). For the Web search results, we use Google advanced search with customized time range which returns the search results together with their creation dates at the beginning of search snippets.

[30 July 2015] Malaysia Airlines Flight 370

- (1) MH370 was an international passenger flight that disappeared on **8 March 2014**, while flying from Kuala Lumpur to Beijing.
- (2) A multinational search effort began in the Gulf of Thailand and the South China Sea, where the flight's signal was lost on surveillance radar, and was extended to the Strait of Malacca.
- (3) Australia took charge of the search effort on **17 March**, when the search shifted to the southern Indian Ocean.
- (4) The current phase of the search is on the northwest of Perth, Australia, which began in **April 2014**.
- (5) Despite the largest search in aviation history, nothing was found until **29 July 2015**, when a marine debris, later confirmed to be a flaperon from Flight 370, was found on Réunion Island.

Figure 5.2: Sample Report

Temporal Expression Extraction: Next, the second step proceeds to extract all temporal expressions (e.g. “2015 December 31”) in the document. Here, we use SUTime [10], a library for recognizing and normalizing time expression using deterministic rule-based method. The output of this step is a list of time expressions, together with their positions in the document.

Sentence Timestamp Extraction and Normalization: Finally, in the third step, we assign each sentence in a document with the time expression as follows:

- First, we assign a temporal value τ as the document creation date.
- For each sentence s in the document:
 - If s contains a temporal expression τ_1 , assign τ_1 as the timestamp of s and update $\tau = \tau_1$.
 - Otherwise, assign τ as the timestamp of s .

Note that we use the format ‘YYYY-MM-DD’ for the temporal expression. If the information of DD or MM is missing, it is replaced with the first day or first month respectively. For example, ‘December 2015’ will be normalized as ‘2015-12-01’. Using the proposed method, sentence (1) and sentence (2) in the sample report of Figure 5.2 will have the same timestamp of ‘2014-03-08’, while sentence (3) will have the timestamp of ‘2014-03-17’.

In Section 5.5.2, we will show that the extraction of timestamp at sentence level will improve the performance of the proposed taxonomy construction approach as compared to the extraction of timestamp at document level.

5.4 Taxonomy Construction

There are three general steps to construct a taxonomy: domain term extraction, taxonomic relation identification, and taxonomy induction. We make use of the taxonomy construction approach described in Chapter 3 for the first step, incorporate timestamp into the second step of identifying taxonomic relations (Section 5.4.1), and propose an incremental taxonomy induction algorithm for the third step (Section 5.4.3). In this chapter, we assume that the list of domain terms is extracted as discussed in Chapter 3 and we will focus only on discussing the second and third steps for taxonomy construction.

5.4.1 Incorporating Temporal Information into Taxonomic Relation Identification

Previous studies about taxonomic relation identification treated all evidence equally, i.e. an evidence that had occurred in 1950 is treated equally with another occurred in 2014. This assumption is not always appropriate, as discussed at the beginning of the chapter. We propose a time-aware approach to identify taxonomic relations by incorporating timestamp into the process of finding evidence, using the following timestamp contribution function:

Definition 4 (Timestamp contribution function) *Given a text sentence d with timestamp s_d , the timestamp contribution of d at time s_0 is defined as follows:*

$$T_d(s_0) = \xi e^{-\xi(s_0 - s_d)}, \quad (5.1)$$

where ξ is a control rate, $s_0 > s_d$ and $(s_0 - s_d)$ is the time lapse between s_d and s_0 .

Equation (5.1) describes the timestamp contribution of a sentence at a specific time by using an exponential distribution function T_d . The intuition of this function is that the evidence of taxonomic relations found in more recent sentences will be of higher relevance than those found in older sentences. This function is inspired by the work of Efron and Golovchinsky [17], and Li and Croft [43], in which it was used to effectively rank documents over time interval.

Using the timestamp contribution function, we incorporate the temporal information into the three taxonomic relation identification methods described in Chapter 3 as follows:

LSP Method: For each search result snippet d in $C_{Web}(t_1, t_2)$ collected from the Web search engine, we calculate the timestamp contribution score of d by using T_d : $T_d(s_0) = \xi e^{-\xi(s_0 - s_d)}$, where s_0 is a chosen specific time (i.e. the time of taxonomy construction) and s_d is the timestamp of d . Note that s_d has to be earlier than s_0 . The unit of time lapse $(s_0 - s_d)$ depends on the nature of corpus and can be, for instance, a day, a month or even a year. For example, if the corpus is from a fast-changing source such as social media, we can set the unit as day to keep up with the change of data on a daily basis. In contrast, for corpus from slower changing domains such as the scientific disciplines, the unit can be a year. The time-aware score for the LSP method is calculated as follows:

$$Score_{LSP}^{Time}(t_1, t_2) = Score_{LSP}(t_1, t_2) \times \frac{\sum_{d \in C_{Web}(t_1, t_2)} T_d(s_0)}{|C_{Web}(t_1, t_2)|}. \quad (5.2)$$

In Equation (5.2), the original LSP evidence score is multiplied by the average timestamp contribution score of all evidence sentences for the taxonomic relation from the Web. If the number of the returned search results is too large, we will use only the first 1,000 results to estimate the average timestamp contribution of evidence.

Note that the total timestamp contribution score of all evidence sentences $\sum_{d \in C_{Web}(t_1, t_2)} T_d(s_0)$ can be considered as the “weighted size” of $C_{Web}(t_1, t_2)$, i.e. we weigh each evidence sentence using Equation (5.1) and sum all these weights. However, if we use only the “weighted size” of $C_{Web}(t_1, t_2)$ for the time-aware score $Score_{LSP}^{Time}(t_1, t_2)$, there will have some issues. Firstly, the score $Score_{LSP}^{Time}$ will not be normalized with the number of evidence sentences. This may lead to potential bias due to large number of past evidence. If there were an obsolete and incorrect

taxonomic relation with many evidence sentences in the past, it would have overwhelmed the new taxonomic relations which would only have a small number of recent evidence sentences. Secondly, if we normalize the score, the information on the number of evidence sentences, which is important for the *LSP* method to recognize true taxonomic relation, will be lost. Therefore, we propose to use Equation (5.2), which combines both information on the number of evidence sentences (embedded inside the original $Score_{LSP}$ score) and the normalized “weighted size” of $C_{Web}(t_1, t_2)$.

SCS Method: Similarly, for each search result snippet d in $Corpus_{t_1}^\Gamma$ and $Corpus_{t_2}^\Gamma$, we calculate the timestamp contribution score of d using the function T_d : $T_d(s_0) = \xi e^{-\xi(s_0 - s_d)}$, where s_0 is a specific time and s_d is the timestamp of d . The time-aware score for the SCS method is calculated as follows:

$$Score_{SCS}^{Time}(t_1, t_2) = Score_{SCS}(t_1, t_2) \times \left(\frac{\sum_{d_1 \in Corpus_{t_1}^\Gamma} T_{d_1}(s_0)}{|Corpus_{t_1}^\Gamma|} + \frac{\sum_{d_2 \in Corpus_{t_2}^\Gamma} T_{d_2}(s_0)}{|Corpus_{t_2}^\Gamma|} \right). \quad (5.3)$$

In Equation (5.3), the original evidence score of $t_1 \rightarrow t_2$ is multiplied by the average timestamp contribution scores of the returned search snippets. Similar to Equation (5.2), Equation (5.3) combines both information on the number of evidence sentences (embedded inside the original score $Score_{SCS}$) and the normalized “weighted size” of them.

SIWN Method: Because WordNet does not contain information about timestamp, we set:

$$Score_{SIWN}^{Time}(t_1, t_2) = Score_{SIWN}(t_1, t_2). \quad (5.4)$$

Combined Evidence: The final combined evidence score for the time-aware method is calculated as follows:

$$\begin{aligned} Score^{Time}(t_1, t_2) = & \alpha \times Score_{SCS}^{Time}(t_1, t_2) \\ & + \beta \times Score_{LSP}^{Time}(t_1, t_2) \\ & + \gamma \times Score_{SIWN}^{Time}(t_1, t_2). \end{aligned} \quad (5.5)$$

If the value $Score^{Time}(t_1, t_2)$ is greater than a threshold value, we extract the relation $t_1 \rightarrow t_2$.

5.4.2 Parameter Learning

We need to estimate the optimal values for the parameters α , β and γ which are used in Equation (5.5). For this purpose, we apply the ridge regression technique [28]. First, we use the time-aware method to create the taxonomies for the ‘Animal’, ‘Plant’ and ‘Vehicle’ domains using the corpora constructed by a bootstrapping method [38]. Then, we ask two annotators to construct the gold standard taxonomies of the three domains (see Section 5.5.2 for more details) and use them to build the training sets. For each pair of terms (t_1, t_2) found in the gold standard taxonomies, its evidence score is estimated as $(\tau+1)$, where τ is the threshold value for $Score^{Time}$. Finally, we use Equation (5.5) to learn the best combination of α , β and γ using the ridge regression algorithm. Note that we learn the parameters only once and use them subsequently for the other domains.

5.4.3 Incremental Taxonomy Induction

To avoid reconstructing a taxonomy whenever there is new incoming data, we propose a novel incremental graph-based algorithm to update an existing taxonomy with a given set of taxonomic relations. The proposed algorithm updates a taxonomy automatically over time based on the information decay function defined below.

Definition 5 (Information decay function) *Given a taxonomic relation r , the information decay of r over period from time s_1 to time s_2 is computed by the following information decay function:*

$$D_r(s_1, s_2) = e^{-\lambda(s_2 - s_1)}, \quad (5.6)$$

where λ is a decay rate and $s_2 > s_1$.

The intuition of the information decay function is that the evidential value of a relation will decrease over time by an exponential function.

Given a root node $\mathcal{R}$, a set of taxonomic relations T and, optionally, an existing taxonomy $\mathcal{H}_1 = (V_1, E_1, s_1)$ created at time s_1 with vertex set V_1 and edge set E_1 , the proposed graph-based algorithm constructs a new taxonomy $\mathcal{H}_2 = (V_2, E_2, s_2)$ created at time s_2 with vertex set V_2 and edge

set E_2 . $t_1 \rightarrow t_2$ denotes the edge from t_1 to t_2 in a taxonomy, and $w(t_1 \rightarrow t_2)$ as the weight of this edge (i.e. evidence score). Algorithm 5.1 consists of four steps:

Algorithm 5.1 Taxonomy Induction Algorithm

Input: $\mathcal{R}$: root node of taxonomy

T : new taxonomic relation set

$\mathcal{H}_1 = (V_1, E_1, s_1)$: existing taxonomy created at time s_1 with vertex set V_1 and edge set E_1

Output: $\mathcal{H}_2 = (V_2, E_2, s_2)$: new taxonomy created at time s_2 with vertex set V_2 and edge set E_2

```

1: Set  $V_2 = V_1$  and  $E_2 = E_1$ 
2: for each edge  $(t_1 \rightarrow t_2) \in E_2, t_1 \neq \mathcal{R}$  and  $t_2 \neq \mathcal{R}$  do
3:    $w(t_1 \rightarrow t_2) = w(t_1 \rightarrow t_2) \times e^{-\lambda(s_2 - s_1)}$ 
4: end for
5: for each relation  $(t_1 \rightarrow t_2) \in T$  do
6:   if  $(t_1 \rightarrow t_2) \in E_2$  then
7:      $w(t_1 \rightarrow t_2) = w(t_1 \rightarrow t_2) + Score^{Time}(t_1, t_2)$ 
8:   else
9:      $E_2 = E_2 \cup (t_1 \rightarrow t_2)$ 
10:     $w(t_1 \rightarrow t_2) = Score^{Time}(t_1, t_2)$ 
11:    if  $t_1 \notin V_2$  then
12:       $V_2 = V_2 \cup \{t_1\}$ 
13:    end if
14:    if  $t_2 \notin V_2$  then
15:       $V_2 = V_2 \cup \{t_2\}$ 
16:    end if
17:    if  $\nexists (t_3 \rightarrow t_1) \in E_2$  and  $t_3 \neq \mathcal{R}$  then
18:       $E_2 = E_2 \cup (\mathcal{R} \rightarrow t_1)$ 
19:       $w(\mathcal{R} \rightarrow t_1) = 1$ 
20:    end if
21:    if  $\exists (\mathcal{R} \rightarrow t_2) \in E$  then
22:       $E_2 = E_2 \setminus (\mathcal{R} \rightarrow t_2)$ 
23:    end if
24:  end if
25: end for
26: edgeFiltering( $\mathcal{H}_2$ )
27: graphPruning( $\mathcal{H}_2$ )

```

Step 1 - Update existing taxonomy (lines 2 - 4): This step aims to update the existing taxonomy from time s_1 to s_2 . In this step, the weight of each edge $(t_1 \rightarrow t_2)$ in E_1 (except the edges connected to root $\mathcal{R}$) is reduced using the information decay function as follows:

$$w(t_1 \rightarrow t_2) = w(t_1 \rightarrow t_2) \times D_{t_1 \rightarrow t_2}(s_1, s_2).$$

Step 2 - Add new relations to existing taxonomy (lines 5 - 25): This step adds new taxonomic relations to the existing taxonomy and update their weights. It adds each relation $t_1 \rightarrow t_2$ as a directed edge from the parent node t_1 to child node t_2 if such edge does not exist in the existing taxonomy. Otherwise, we update the weight of this edge with new evidence score. If t_1 does not have any parent node, t_1 will become a child node of root $\mathcal{R}$. The edge's weight is updated as follows:

$$w(t_1 \rightarrow t_2) = \begin{cases} 1 & \text{if } t_1 = \mathcal{R} \\ w(t_1 \rightarrow t_2) + Score^{Time}(t_1, t_2) & \text{if } t_1 \rightarrow t_2 \in E_1 \\ Score^{Time}(t_1, t_2) & \text{otherwise.} \end{cases}$$

The result of this step is a weighted connected graph containing all taxonomic relations with root $\mathcal{R}$.

Step 3 - Edge filtering (line 26): The graph generated in Step 2 contains some edges with low evidence scores. The reason is that some relations in the existing taxonomy can become outdated during the period from s_1 to s_2 (according to the information decay function), and they do not exist in the new relation set. In this step, each edge $t_1 \rightarrow t_2$ in the graph is revisited, and if its weight is lower than the threshold value of $Score^{Time}$, it will be removed from the graph. In the case t_2 does not have any other parent node except t_1 , t_2 will be deleted from the vertex set, and edges from t_1 to t_2 's children will be added to the edge set with the weights that are equal to the weights of the edges from t_2 to t_2 's children. Then, all edges from t_2 to t_2 's children will be removed from the edge set.

Step 4 - Graph pruning (line 27): The graph generated in Step 3 is not an optimal tree as it may contain redundant edges or incorrect edges, for example, those edges that form a loop in the graph.

This step aims to produce an optimal tree of the taxonomy from the weighted graph in Step 3. For this purpose, we apply Edmonds’ algorithm [16] for finding the optimal branching out of a weighted directed graph. Using this algorithm, we can find a subset of the current edge set, which forms a taxonomy where every non-root node has in-degree 1 and the sum of the edge weights is maximized.

5.5 Performance Evaluation

We have conducted two experiments for performance evaluation. The first experiment evaluates the performance of our proposed time-aware approach on constructing taxonomy from a given list of terms without any prior knowledge (i.e. without any existing taxonomies). The second experiment evaluates the performance of our proposed approach on taxonomy update.

5.5.1 Datasets

We evaluate our approach for taxonomy construction based on the following four datasets of document collections obtained from different domains:

- Artificial Intelligence (AI) and Terrorism domains: They are the same datasets used in Chapter 3.
- MH370 domain: The corpus is about the root term ‘*MH370*’, the missing flight that went down in the ocean on Saturday, 8 March 2014. The corpus is created by querying the Google search engine with the keyword “MH370” from March 08, 2014 to April 30, 2014 and collecting the first 300 documents from the search results each day. After removing duplicates, the corpus contains a total of 12,307 documents.
- Disease domain: The corpus is about the root term ‘*Disease outbreak*’, created by collecting reports from “Disease outbreaks by year from 1996 to 2014”¹ of WHO, and the email archive

¹<http://www.who.int/csr/don/archive/year/en/>

of ProMed ² which is an Email based reporting system dedicated to reporting on disease outbreaks that affect human health. The corpus contains a total of 25,370 reports/emails.

Parameter Settings: For the rapidly changing domain ‘MH370’, we choose ‘day’ as the unit of the time lapse whereas for the other three domains, we use ‘year’ as the time lapse’s unit. We set the threshold value of $Score^{Time}$ in Equation (5.5) as 2.2, and the control rate ξ in Equation (5.1) and decay rate λ in Equation (5.6) as 0.15. The setting of these parameters will be discussed in Section 5.5.4.

5.5.2 Evaluation on Taxonomy Construction

Experiment 1

In this experiment, we compare our time-aware taxonomy construction approach with other state-of-the-art methods in the task of constructing a new taxonomy from a given list of terms without any prior knowledge (i.e. without any existing taxonomy). Three state-of-the-art methods in the literature selected for comparison are:

- Zhu’s method [82]: It constructs the taxonomy using evidence from multiple sources such as WordNet, Wikipedia and Web search engine. In their method, both statistical and linguistic approaches are used to infer taxonomic relations.
- Kozareva’s method [37]: It constructs the taxonomy using evidence from Web search engine by matching the search results with a predefined set of syntactic patterns.
- Tuan1: It is our taxonomy construction approach presented in Chapter 3. This approach ignores temporal information during taxonomy construction. We use Tuan1 instead of Tuan2 for this comparison as the proposed time-aware approach is developed based on Tuan1 by incorporating the temporal information.

²<http://www.promedmail.org/>

To evaluate the effectiveness of extracting timestamp at sentence level (as described in Section 5.3), we also conduct the experiment on the setting that uses timestamp at document level (i.e. all evidence in the document will have the same timestamp information as the document creation date). We use the subscript *docstamp* to denote this setting.

Evaluation Metric

In this experiment, we evaluate the constructed taxonomies against the manually created gold standard taxonomies. The gold standard taxonomies are created as follows. For each domain, two annotators are employed at the same time to create taxonomies independently using the list of terms obtained from the domain term extraction module, according to the following rules:

- Rule 1 (Relevancy): Every term in the taxonomy should be related to the root term.
- Rule 2 (Appropriateness): Each edge between two terms should be established at the time the taxonomy is created, if their relation is correct and not obsolete. A relation is obsolete if it is invalid at the time of consideration.
- Rule 3 (Hierarchical structure): The gold standard taxonomy of each domain should form a tree, without redundant paths or cycles.

Two annotators will then compare their constructed taxonomies. A taxonomic relation $t_1 \rightarrow t_2$ is counted as an agreement if and only if both annotators have t_1 and t_2 in their taxonomies, and there is a directed path from t_1 to t_2 . If an annotator has a taxonomic relation with one vertex not in the other annotator's taxonomy, it will be considered as a disagreement. After evaluation, the average inter-annotator agreement on edges of the constructed taxonomies between the two annotators is 87% using Cohen's kappa coefficient measurement. Finally, the two annotators discuss to come up with the gold standard taxonomies. As a result, the number of nodes and average depth of the taxonomies are summarized in Table 5.1.

Table 5.1: Analysis of Gold Standard Taxonomies

Domain	Number of Vertices	Average Depth
MH370	257	3.2
AI	925	5.3
Terrorism	312	3.6
Disease	459	4.0

We use precision, recall and F-measure to measure the performance of taxonomy construction. Let R and R_{gold} be the set of taxonomic relations of our constructed taxonomy and the gold standard taxonomy respectively, the metrics are given as follows:

$$precision = \frac{|R \cap R_{gold}|}{|R|} ; \quad recall = \frac{|R \cap R_{gold}|}{|R_{gold}|} ; \text{ and}$$

$$F\text{-measure} = 2 \times \frac{precision \times recall}{precision + recall} .$$

Experimental Results

The experimental results are given in Table 5.2 which shows that our time-aware approach achieves significantly better performance than the Kozareva's method and Zhu's method in terms of F-measure (t-test, p-value<0.05). Our approach shows slightly lower precision than that of Kozareva's method due to the SCS method, but with much higher recall and F-measure than Kozareva's method. In contrast, our approach shows slightly lower recall, but with much higher precision and F-measure than Zhu's method, which is based on statistical methods such as pointwise mutual information and cosine similarity. On average, our time-aware approach improves the F-measure by 20% compared to Kozareva's method, and by 10% compared to Zhu's method.

Moreover, the incorporation of timestamps into the time-aware approach also contributes to better performance as it helps identify new taxonomic relations effectively, while getting rid of obsolete and incorrect relations. As shown from the experimental results, the time-aware approach shows

Table 5.2: Experimental Results for Taxonomy Construction

Method	Domain	Precision	Recall	F-measure
Kozareva	MH370	89%	36%	51%
Zhu	MH370	60%	59%	59%
Tuan1	MH370	85%	49%	62%
Time-aware _{docstamp}	MH370	81%	54%	65%
Time-aware	MH370	86%	57%	69%
Kozareva	AI	90%	37%	52%
Zhu	AI	59%	57%	58%
Tuan1	AI	84%	51%	63%
Time-aware _{docstamp}	AI	80%	55%	65%
Time-aware	AI	84%	57%	68%
Kozareva	Terrorism	91%	32%	47%
Zhu	Terrorism	60%	64%	62%
Tuan1	Terrorism	86%	49%	62%
Time-aware _{docstamp}	Terrorism	80%	56%	66%
Time-aware	Terrorism	87%	60%	71%
Kozareva	Disease	90%	29%	44%
Zhu	Disease	57%	56%	56%
Tuan1	Disease	83%	50%	62%
Time-aware _{docstamp}	Disease	81%	53%	64%
Time-aware	Disease	85%	55%	67%

significantly better performance than the non time-aware approach (i.e. Tuan1) in all of the four domains in terms of F-measure (t-test, p-value<0.05). On average, our time-aware approach improves the F-measure by 7% compared to Tuan1.

For example, in the Terrorism domain, our approach recognizes ‘ISIS’ as a hyponym of ‘terror-

ist group', while the three state-of-the-art methods cannot recognize this. In addition, while the other three methods have extracted the outdated taxonomic relation between '*Palestine Liberation Organization*' and '*terrorist group*', our approach is able to ignore it. The reason is that the three state-of-the-art methods infer taxonomic relations using co-occurrence frequency, but '*ISIS*' has only appeared in reports since 2014. The occurrence frequency of '*ISIS*' is very low compared to the other terrorist groups such as '*Palestine Liberation Organization*' which was mentioned over the past many years. In contrast, by using the timestamp contribution function to better profile the relevance of evidence over time, our approach can recognize the recent relationship of '*terrorist group*' with '*ISIS*' while getting rid of the obsolete and incorrect relation with '*Palestine Liberation Organization*'.

From the experimental results of the time-aware and time-aware_{docstamp} approaches, we also observe that the use of timestamps extracted at sentence level is more effective than the use of timestamps at the document level. The timestamps extracted at sentence level can capture more precisely the temporal information of the facts in fast-changing domains than at the document level. The results have shown that the use of sentence-level timestamps can improve the precision and recall of our taxonomy construction approach, improving the F-measure by 4% on average, as compared to the use of timestamps at the document level.

5.5.3 Evaluation on Taxonomy Update

Experiment 2

For fast-changing domains, taxonomies should be frequently and quickly updated. In this experiment, we examine how the proposed time-aware approach can effectively update the constructed taxonomies over time to keep up with the latest information trends.

We use the case study of '*MH370*' domain for this experiment. During the search operation of the missing flight *MH370*, there were several turning points which could be captured into the following phases (according to well-known news agencies such as CNN, BBC and New York Times):

- Phase 1 (from March 08, 2014): The flight lost contact with the airport. The search started from the South China Sea and Gulf of Thailand, and was extended to the Strait of Malacca.
- Phase 2 (from March 13, 2014): Images from satellites indicated the plane might fall down in the Indian Ocean. The search focus was moved from the South of Sumatra to the South-West of Perth in the Southern Indian Ocean.
- Phase 3 (from March 28, 2014): The estimation of aircraft's remaining fuel and the radar track led the search shifting to a new area, the North-West of Perth in the Southern Indian Ocean.

We apply the proposed time-aware approach to construct and update the taxonomy for 'MH370' incrementally in every two days. We compare our time-aware approach with the following three methods:

- Zhu's method [82]: It applies a graph-based algorithm to update taxonomies incrementally with timestamp information.
- Baseline method 1: The taxonomy is updated with the newly obtained data every two days, but does not use any temporal information in both taxonomic relation identification and taxonomy induction (Section 5.4.1). Specifically, Step 1 (update existing taxonomy) in Section 5.4.3 is excluded since we are not using any temporal information, there is no updating of the weights of the existing taxonomy using the decay function.
- Baseline method 2: Here, we construct the taxonomies using temporal information every two days, but only with the new documents from these two days. This allows us to evaluate the effect of eliminating all the taxonomic relations built from the previous documents instead of the gradual decay approach in our proposed method.

Here, we have chosen the time period of two days because the 'MH370' domain was a truly fast-changing domain. As we shall see shortly, even using only the new documents within 2 days to build the taxonomy in our baseline method 2, there were new taxonomic relations updated from the latest information (as illustrated in the example in Figure 5.4).

Evaluation Metric

When constructing the gold standard taxonomies using the same rules described in Section 5.5.2, we have asked the annotators to select for each parent term at most three sub-terms that are most related to it at the time of taxonomy construction. We denote the set of gold-standard taxonomic relations as S_{gold} . In the same way, when applying the methods of taxonomy construction, we select for each parent term at most three sub-terms with the highest evidence scores. We denote the set of those automatically extracted taxonomic relations as S . We use the following metrics to evaluate the update of taxonomy:

$$precision = \frac{|S \cap S_{gold}|}{|S|}; \quad recall = \frac{|S \cap S_{gold}|}{|S_{gold}|}; \text{ and}$$

$$F\text{-measure} = 2 \times \frac{precision \times recall}{precision + recall}.$$

The intuition for limiting the sub-term number to three for the evaluation is that if a taxonomy can keep up with the newly updated data, it should be able to detect the emerging terms and relations, and add them to the taxonomy with high evidence scores so that the user can easily observe the emerging trend of information in the domain as it occurs. In addition, the approach should also have the capability to remove any obsolete relations in the taxonomy when they are no longer valid.

Experimental Results

From the results shown in Figure 5.3, we can see that our time-aware approach achieves the best performance and significantly outperforms the two baseline methods and Zhu's method in terms of F-measure (t-test, p-value<0.05). One interesting point to observe is that there are two periods when the time-aware approach shows much higher F-measure than the baseline methods and Zhu's method: from March 12 to March 14, and from March 28 to March 30. During these periods, the performance of baseline method 1 (which does not use any timestamp information) and Zhu's method have dropped quite significantly, while our time-aware approach's performance has increased slightly.

Figure 5.3: Performance Results on Taxonomy Update over Time in the ‘MH370’ Domain

One plausible explanation is that there are some turning points happened on March 13 and March 28, which fall within these periods as described above. During these periods, many new terms/relations such as search area, search focus and search device were added to the corpus. Our time-aware approach was able to assign higher weights to the new taxonomic relations than the older relations due to their recent timestamps, even though the frequencies of these new relations were fewer than that of the older relations. In contrast, Zhu’s method and baseline method 1 were unable to recognize these new relations due to their relatively low frequencies in the corpus. In addition, incorrect relations in the existing taxonomy were also removed from the new taxonomy using the information decay function by our time-aware approach, whereas the other two methods still kept them in the taxonomy. In short, our time-aware approach can update the taxonomy faster with the latest information trends, as well as remove the incorrect relations effectively, as compared to the other methods.

Also, from the experimental results of our time-aware approach and the baseline method 2, we can observe that updating the existing taxonomy with new taxonomic relations is more effective than rebuilding a new taxonomy using only the new data. The reason is that although some older taxonomic relations are mentioned occasionally in the new data, they are still valid. Therefore, if we ignore the older data, their taxonomic relations will be lost in the new taxonomy when it is constructed with only the new data. In addition, there are also many taxonomic relations that may need longer time period to establish.

Figure 5.4: Top Three Hyponyms of 'Search area' in 'MH370' Taxonomies over Time

Figure 5.4 shows an example on the changes of hyponym list of the term '*search area*' over time using different methods. We observe that both the time-aware approach and baseline method 2, which utilized the temporal information, can quickly update the relations with the latest information as compared to Zhu's method and baseline method 1, which ignore temporal information for taxonomy construction. For example, in the taxonomy constructed on March 14, the time-aware

approach and baseline method can quickly recognize the change of the search area to ‘*Southern Indian Ocean*’ and ‘*Sumatra*’, thereby ranking them at the top of the hyponym list of ‘*search area*’, whereas the Zhu’s method and baseline method 1 both missed this update until March 26. Another interesting point is that due to the lack of temporal information, both Zhu’s method and baseline method 1 still ranked ‘*South China Sea*’ at the top from the taxonomies constructed on April 30, while this term has earlier been removed from the hyponym list of ‘*search area*’ by our time-aware approach and even baseline method 2 using temporal information.

5.5.4 Parameter Tuning

In our approach for taxonomy construction, some parameters are tuned to optimize performance.

The threshold value for $Score^{Time}$ in Equation (5.5) controls the number of extracted taxonomic relations. In general, the larger this threshold value is, the higher number of true taxonomic relations we can get. However, the higher number of incorrect relations may also incur. From our experiments, we found that the threshold value for $Score^{Time}$ can be set between 2.1 to 2.3 for the time-aware approach to achieve the best performance.

The control rate ξ in Equation (5.1) and decay rate λ in Equation (5.6) have the effect on the contribution of old and new data. Specifically, smaller values of control rate and decay rate will allow newer data to contribute more evidence of taxonomic relations than older data, whereas larger values will cause the old and new data to have similar evidence contributions. According to our experiments, the time-aware approach shows the best performance when the values of these rates are set between 0.15 to 0.20.

5.6 Summary

In this chapter, we have proposed a novel time-aware approach for taxonomy construction given a time series of textual documents from a domain that could be fast-changing with emerging concepts

or events. By using the timestamp contribution and information decay functions, our approach can effectively utilize temporal information for both taxonomic relation identification and taxonomy update. The experimental results show that our proposed approach achieves better performance than the state-of-the-art methods. In addition, the proposed approach can be used to update the taxonomy incrementally over time and keep the taxonomy up-to-date with the latest information trends for the domain.

Chapter 6

Taxonomy Construction Using Word Embedding Clustering

Word embedding [5], also known as distributed word representation, represents words with high-dimensional and real-valued vectors, and has been shown to be effective in exploring both linguistic and semantic relations between words. In recent years, word embedding has been used quite extensively in NLP research, ranging from syntactic parsing [67], machine translation [83] to sentiment analysis [68]. However, taxonomy construction studies have yet to catch on this trend. The few works on applying word embedding for this purpose such as the studies of [23] and [69] are either domain-dependent or cover only few contexts of taxonomic relations, and are not well-suited for constructing taxonomies for large or new domains.

Tan *et al.* [69] recently proposed a method to construct taxonomy based on word embedding. The method is derived from the observation that a hypernym-hyponym pair often occurs in a sentence with an ‘*is-a*’ phrase. It learns a vector for the phrase ‘*is-a*’ and hypothesizes that if t_1 is a hypernym of t_2 then $v(t_1) = v(\text{‘is-a’}) \times v(t_2)$, where $v(t)$ is the embedding vector of term t . However, given that taxonomic relations can be inferred from various other contexts in the sentence, or even beyond the sentence boundary [70], the method only achieves fairly low recall in the experiments.

In the study by Ruiji et al. [23], they proposed a method to use word embedding to construct taxonomy by learning projection matrices for taxonomic relations. However, the method has some limitations. First, as the projection matrix is domain-dependent, different matrices will have to be learnt for different domains. This may not be feasible in practice, as the projection matrices are learnt using a supervised learning method, which requires labeled data for training. Such training data are not always available, especially for new domain areas. Furthermore, the method is designed to construct a taxonomy for a list of hypernyms of a term. The size of such hypernym list, however, is usually not more than ten terms. As such, the method is not effective for constructing larger size of taxonomies.

In this chapter, we discuss an empirical study for identifying three characteristics of taxonomic relations in the vector space of word embeddings, namely semantic cluster, taxonomic centroid and relative distances from the root. Based on these characteristics, we propose a novel method based on word embedding clustering to infer taxonomic relations. We also propose a graph-based algorithm which utilizes the evidence scores from word embeddings to organize taxonomic relations into a taxonomic structure.

6.1 Embedding-based Taxonomy Construction

The problem of our embedding-based taxonomy construction is specified as follows.

Given two terms t_1 and t_2 , we denote $t_1 \gg t_2$ as a taxonomic relation where t_1 is a hypernym of t_2 .

A taxonomy is defined as a tuple $\mathcal{H} = (\mathfrak{R}, V, E)$, where:

- $\mathfrak{R}$ is the root of the taxonomy.
- V is the set of taxonomy's vertices, i.e. the set of terms, including the root $\mathfrak{R}$.
- E is the set of taxonomy's edges, i.e. the set of taxonomic relations.

We define the goal as follows: *Given a list of terms L which is the hyponyms of the root $\mathfrak{R}$, we aim to construct a taxonomy $\mathcal{H} = (\mathfrak{R}, V, E)$ where $V \subseteq L \cup \{\mathfrak{R}\}$.*

In this chapter, we propose a word embedding clustering approach for constructing taxonomy from a list of hyponyms of a term, which can contain hundreds or thousands of terms. Inspired by the observations from our empirical study on the characteristics of the spatial distributions of terms with taxonomic relations in the vector space of their word embeddings, we have derived three word embedding measures, namely semantic cluster, taxonomic centroid and relative distances from the root, to capture the relations of the hypernoms (or hyponyms) of a term, the distribution of taxonomic relations, as well as the relations of the taxonomy’s root term to the hypernoms and hyponyms in the vector space. Based on these word embedding measures, we devise a three-step unsupervised method to identify taxonomic relations. This method is domain-independent and does not require any training data. Thus, it can be applied effectively to construct taxonomies for various new or large domain areas. We also introduce a graph-based algorithm that uses the evidence scores from word embeddings to organize taxonomic relations into a taxonomy structure, and remove redundant and incorrect relations.

For the rest of this chapter, we will discuss the proposed approach and its performance evaluation.

6.2 Empirical Study on Taxonomic Relations

In this section, we discuss our empirical study on taxonomic relations in the vector space of word embeddings.

6.2.1 Data Preparation

Word Embedding Training: We apply the Skip-gram model [50] which has been shown to be efficient for learning word embeddings from large-scale corpus because of its low time complexity. Given a word, the model adopts the log-linear classifier to predict the context words. The embeddings are learnt by maximizing the log-likelihood over the entire dataset using stochastic gradient descent method. In this study, we use the entire English Wikipedia corpus for learning, resulting

in the embeddings of around 12 million of words in vocabulary. Each word is represented as a 300-dimensional vector in the vector space.

Sample Taxonomies: We randomly extract from WordNet 100 taxonomies in various domains. The extraction is carried out as follows. Firstly, we randomly choose 100 terms located at middle levels of WordNet (e.g. “bird”, “fish” or “woody plant”) and consider them as roots of the domains. Secondly, starting from each root, we randomly walk in WordNet following hyponym relations and collect all terms in walking paths to form the taxonomy for this root. The random walk will stop if we are unable to move anymore, or the number of terms is higher than 150. Finally, we obtain 100 taxonomies with average depth of around 4.7 and the number of terms in each taxonomy ranges from 50 to 150. Note that the obtained taxonomy for each domain is just a relative taxonomy, i.e. it may not be a full taxonomy containing all possible terms of this domain. These taxonomies are used for the analysis of the characteristics of taxonomic relations in our study.

Dataset Collection: For each term t appearing in the sample taxonomies, we look up its embedding in the collection of word embeddings obtained from the Skip-gram model and denote this embedding as $v(t)$. Note that if t is a multiword term, $v(t)$ is calculated as the average of embeddings of all words in t . The set of terms together with their embeddings are used as the dataset for empirical analysis.

6.2.2 Empirical Analysis and Observations

From the study, we have analyzed the dataset of terms and their word embeddings, and derived the following three observations:

Observation 1 (Semantic Cluster) *Terms under a semantic hierarchy are usually clustered together in terms of their word embeddings.*

In each sample taxonomy, for each possible (direct or indirect) taxonomic relation $t_1 \gg t_2$, we calculate the offset of embeddings ($v_{t_1} - v_{t_2}$) and represent the offsets in a two-dimensional space

Figure 6.1: Clusters of Offset Vectors

using the t-SNE algorithm [45] for visualization. One example of the offset vectors is shown in Figure 6.1. The hypernyms (or hyponyms) of a term are usually located close together in the vector space. The left cluster shows some taxonomic relations about ‘*chordate*’. The right cluster shows some taxonomic relations about ‘*shipworm*’. We observe that for more than 97% of the cases, the offset vectors form clusters, and each cluster is either a set of offsets between a term and its hypernyms, or a set of offsets between a term and its hyponyms. Formally, if t is a term and $t_1, t_2, \dots, t_k$ are hypernyms (or hyponyms) of t , the vectors $(v_t - v_{t_1}), (v_t - v_{t_2}), \dots, (v_t - v_{t_k})$ tend to be located close together, or equally, $v_{t_1}, v_{t_2}, \dots, v_{t_k}$ tend to be located close together. It means that *the terms under the same semantic hierarchy (i.e. terms that are hypernyms or hyponyms of another term) are usually located close together in the vector space*. We call these *semantic clusters*.

One plausible explanation for this phenomenon is that because of the inheritable characteristic of taxonomic relations, a term and its hypernyms (or hyponyms) will usually share the same or similar contexts in the corpus. In other words, if there is a sentence containing a term, then similar sentences that have this term replaced with its hyponyms (or hypernyms) are also highly likely to occur. For example, if we have the following sentence:

- The food of the **feline** is meat.

Then the following sentences are highly likely to occur:

- The food of the **tiger** is meat.
- The food of the **wildcat** is meat.
- The food of the **puma** is meat.

Observation 2 (Taxonomic Centroid) *Terms that are taxonomically related are usually located near the centroids of the clusters formed by their word embedding offsets.*

Figure 6.2: Clusters of Offset Vectors between Pairs of Terms

For each sample taxonomy, we calculate all possible embedding offsets ($v_{t_1} - v_{t_2}$) between any pairs of terms (t_1, t_2) , regardless of whether they are taxonomically related or not. Again, we use the t-SNE algorithm to represent them in the two-dimensional space. One example on the taxonomy of ‘bird’ is given in Figure 6.2. In the figure, the offsets of true taxonomic relations are represented by blue star symbols. The offsets of false taxonomic relations are represented by red dots. We observe that these offsets also form clusters, and for more than 91% of the cases, *the offsets ($v_{t_1} - v_{t_2}$) between the embeddings of two terms which have taxonomic relation $t_1 \gg t_2$ are located at or near*

the centroid of these clusters. Note that not all nodes located at or near the centroid have taxonomic relations.

This phenomenon can be explained as follows. The offsets in each cluster are usually obtained from pairs of terms $(t_{i_1}, t_j), (t_{i_2}, t_j), \dots, (t_{i_k}, t_j)$ (or pairs of terms $(t_j, t_{i_1}), (t_j, t_{i_2}), \dots, (t_j, t_{i_k})$). Assuming that $(t_{i_x} \gg t_j)$ (or $(t_j \gg t_{i_x})$, correspondingly) is a taxonomic relation from these pairs. According to the study of Kommers et al. [35], a hypernym, which is the more general concept, generally has more neighbors on the vector plane than a term which is not hypernym of any other terms. Therefore, the embedding offset $(v_{t_{i_x}} - v_{t_j})$ (or $(v_{t_j} - v_{t_{i_x}})$) will have more neighbors than the embedding offset $(v_{t_{i_y}} - v_{t_j})$ (or $(v_{t_j} - v_{t_{i_y}})$, correspondingly) if t_{i_y} is not a hypernym. As a result, the embedding offset $(v_{t_{i_x}} - v_{t_j})$ (or $(v_{t_j} - v_{t_{i_x}})$) obtained from the taxonomic relation $t_{i_x} \gg t_j$ (or $(t_j \gg t_{i_x})$) tends to be located near the centroid of the cluster where it will have more neighbors than those located near the boundary of the cluster.

Observation 3 (Relative Distance from the Root) *The semantic distances of hypernyms and hyponyms with respect to the root are usually conserved in terms of their word embeddings.*

In this observation, we measure the relative distances of the hypernym and hyponym with respect to the root node. Formally, given a taxonomic relation $t_1 \gg t_2$ and the root node $\mathfrak{R}$ of the taxonomy, we measure the difference in cosine distance between v_{t_1} and v_{t_2} to $v_{\mathfrak{R}}$. The cosine distance between the two vectors v_1 and v_2 is defined as:

$$d(v_1, v_2) = 1 - \text{cosine_similarity}(v_1, v_2).$$

An analysis of over 5000 random taxonomic relations is given in Figure 6.3. We observe that for more than 89% of the cases, the distance from the hypernym to the root is less than the distance from the hyponym to the root, i.e.

$$d(v_{\mathfrak{R}}, v_{t_1}) - d(v_{\mathfrak{R}}, v_{t_2}) < 0. \quad (6.1)$$

In fact, for almost 100% of the cases, we have:

$$d(v_{\mathfrak{R}}, v_{t_1}) - d(v_{\mathfrak{R}}, v_{t_2}) < 0.2. \quad (6.2)$$

Figure 6.3: Percentage of Taxonomic Relations $t_1 \gg t_2$ with $d(v_R, v_{t1}) - d(v_R, v_{t2}) < \alpha$

The explanation for this observation is similar to observation 2. In the vector space, the root of a domain, which is the most general term, is usually located near the centroid of the cluster formed by all terms in this domain. Therefore, the hypernym, which usually has more neighbors than the hyponym, will tend to be located closer to the root where it has more neighbors than those that are located further.

This observation can be used to serve as a measure to filter out incorrect taxonomic relations. As far as we know, the use of the root node for filtering taxonomic relations was not investigated in previous studies about taxonomy construction using word embedding.

6.3 Proposed Approach

In this section, we propose an approach for taxonomy construction based on the observations introduced in Section 6.2.2. First, we use an unsupervised algorithm to identify taxonomic relations using word embedding clustering (in taxonomic relation identification). Then, a graph-based algorithm is proposed to build the taxonomy structure from the list of taxonomic relations (in taxonomy

induction).

6.3.1 Taxonomic Relation Identification

Based on the observations given in Section 6.2.2, we propose a three-step algorithm (Algorithm 6.1) to extract taxonomic relations using word embedding clustering as follows:

Step 1 - Dividing terms into clusters (line 2): This step is based on observation 1 that terms under a semantic hierarchy are usually clustered together in terms of their word embeddings, thereby forming sub-taxonomies. The aim of this step is therefore to find the clusters of terms for such potential sub-taxonomies. We apply the affinity propagation clustering algorithm [22] to the embeddings of terms in the list L to find the set C_1 of clusters of terms, and each cluster is a potential candidate to form a sub-taxonomy. Note that the affinity propagation clustering algorithm creates clusters by sending messages between pairs of embeddings until convergence. Therefore, we do not need to declare the number of clusters at the beginning. The detail of this clustering algorithm can be found in [22].

Step 2 - Identifying taxonomic relations in each cluster (lines 13 - 25): In this step, we aim to identify the taxonomic relations in the clusters obtained in Step 1. We achieve this based on observations 2 and 3. For each cluster, we calculate all possible embedding offsets $(v_{t_1} - v_{t_2})$ between pairs of terms (t_1, t_2) , and then use the affinity propagation clustering algorithm again to divide these offsets into the set of clusters C_2 . According to observation 2, the embedding offsets from taxonomic relations are usually located at or near the centroid of new clusters. Therefore, for each cluster in C_2 with the centroid Θ , if the offset $(v_{t_i} - v_{t_j})$ satisfies:

$$d(v_{t_i} - v_{t_j}, \Theta) \leq \epsilon, \quad (6.3)$$

where ϵ is a threshold value, then we consider $t_i \gg t_j$ as a candidate of taxonomic relation.

Algorithm 6.1 Taxonomic Relation Identification

Input: L : the list of terms
 $\mathfrak{R}$: the root of taxonomy

Output: T : the set of taxonomic relations

```

1: Initialize  $T = \emptyset$ 
2:  $C_1 = \text{Affinity\_Propagation\_Clustering}(L)$ 
3: for each cluster  $c_1 \in C_1$  do
4:    $L_{c_1}$  = list of terms existing in  $c_1$ 
5:    $L_1 = \emptyset$ 
6:   for each term  $t_1 \in L_{c_1}$  do
7:     for each term  $t_2 \in L_{c_1}$  do
8:       if  $t_1 \neq t_2$  then
9:          $L_1 = L_1 \cup (v_{t_1} - v_{t_2})$ 
10:      end if
11:    end for
12:  end for
13:  $C_2 = \text{Affinity\_Propagation\_Clustering}(L_1)$ 
14: for each cluster  $c_2 \in C_2$  do
15:    $L_2 = \emptyset$ 
16:    $\Theta = \text{centroid}(c_2)$ 
17:   for each offset  $(v_{t_i} - v_{t_j}) \in c_2$  do
18:     if  $d(v_{t_i} - v_{t_j}, \Theta) \leq \epsilon$  then
19:       if  $d(v_{\mathfrak{R}}, v_{t_i}) - d(v_{\mathfrak{R}}, v_{t_j}) < \xi$  then
20:          $L_2 = L_2 \cup (t_i \gg t_j)$ 
21:       end if
22:     end if
23:   end for
24:    $T = T \cup L_2$ 
25: end for
26:  $L_{root} = \emptyset$ 
27: for each cluster  $c_2 \in C_2$  do
28:    $L_2$  = taxonomic relation list found in  $c_2$ 
29:   for each relation  $(t_1 \gg t_2) \in L_2$  do
30:     if  $\nexists (t_3 \gg t_1) \in L_2$  then
31:        $L_{root} = L_{root} \cup t_1$ 
32:     end if
33:   end for
34: end for
35:  $T_{root}$  = apply Step 1 to 25 of Algorithm 6.1 with list  $L_{root}$  and root  $\mathfrak{R}$ 
36:  $T = T \cup T_{root}$ 
37: end for

```

The identified candidates are then filtered using observation 3. For candidates $t_i \gg t_j$ that have

$$d(v_{\mathfrak{R}}, v_{t_i}) - d(v_{\mathfrak{R}}, v_{t_j}) \geq \xi , \quad (6.4)$$

where $\mathfrak{R}$ is the given root in the list of terms and $\xi \geq 0$ is a threshold value, they will be removed from the set of taxonomic relations.

Step 3 - Connecting root terms of clusters (lines 26 - 36): In Step 2, we have identified the taxonomic relations in each cluster of terms. In this step, we collect the root terms of each clusters (i.e. the term t_i such that it has the relation $t_i \gg t_j$ but not any other relation $t_k \gg t_i$). Then, from this set of root terms, we repeat Step 1 and Step 2 of Algorithm 6.1 to find the taxonomic relations between these roots. Here, we aim to identify the linkage between the directed graphs found in the clusters of terms to form larger graphs by connecting the roots of these directed graphs together.

6.3.2 Taxonomy Induction

As mentioned, after performing the above taxonomic relation identification, we will obtain a list of taxonomic relations, which may contain redundant relations (e.g. if $t_1 \gg t_2$ and $t_2 \gg t_3$, then $t_1 \gg t_3$ is redundant) or incorrect relations (e.g. if $t_1 \gg t_2$ and $t_2 \gg t_1$, then it will form a loop between the two terms). In this section, we propose a graph-based algorithm to organize the taxonomic relations into an optimal tree-structured taxonomy, as well as to eliminate incorrect and redundant relations. Let's denote $t_1 \rightarrow t_2$ as a directed edge from t_1 to t_2 . The proposed algorithm (Algorithm 6.2) consists of the following three steps:

Step 1 - Initial hypernym graph creation (lines 1 - 16) : First, we construct a connected directed graph from the list of taxonomic relations. The idea is to add each taxonomic relation $t_1 \gg t_2$ as a directed edge from the parent node t_1 to child node t_2 , and if t_1 does not have any hypernym term, t_1 will become a child node of the root node $\mathfrak{R}$. The result of this step is a connected graph containing all taxonomic relations with the common root $\mathfrak{R}$.

Algorithm 6.2 Taxonomy Induction Algorithm**Input:** T : the set of taxonomic relations**Output:** V : the vertex set of resultant taxonomy E : the edge set of resultant taxonomy

```

1: Initialize  $V = \{\mathfrak{R}\}$ ,  $E = \emptyset$ 
2: for each taxonomic relation  $(t_1 \gg t_2) \in T$  do
3:    $E = E \cup \{t_1 \rightarrow t_2\}$ 
4:   if  $t_1 \notin V$  then
5:      $V = V \cup \{t_1\}$ 
6:   end if
7:   if  $t_2 \notin V$  then
8:      $V = V \cup \{t_2\}$ 
9:   end if
10:  if  $\nexists t_3 \rightarrow t_1 \in E$  with  $t_3 \neq \mathfrak{R}$  then
11:     $E = E \cup \{\mathfrak{R} \rightarrow t_1\}$ 
12:  end if
13:  if  $\exists \mathfrak{R} \rightarrow t_2 \in E$  then
14:     $E = E \setminus \{\mathfrak{R} \rightarrow t_2\}$ 
15:  end if
16: end for
17: edgeWeighting( $V, E$ )
18: graphPruning( $V, E$ )

```

Step 2 - Edge weighting (line 17): Next, we calculate the weight of each edge in the hypernym graph. Based on observation 3 in Section 6.3.1, the evidence score of each edge can be measured as a difference between the distances from its vertices to the root node $\mathfrak{R}$. Specifically, the weight of edge $t_1 \rightarrow t_2$ is calculated as:

$$w(t_1 \rightarrow t_2) = \begin{cases} 1 & \text{if } t_1 = \mathfrak{R} \\ d(v_{\mathfrak{R}}, v_{t_2}) - d(v_{\mathfrak{R}}, v_{t_1}) & \text{otherwise} \end{cases}$$

where $d(v_i, v_j)$ is the cosine distance between v_i and v_j as defined in Equation (6.2.2).

Step 3 - Graph pruning (line 18): The hypernym graph generated in Step 1 is not an optimal taxonomy as it may contain many redundant or incorrect edges which may form a loop when combined together. In this step, we aim to produce an optimal taxonomy by pruning the graph based on our edge weighting strategy. A maximum spanning tree algorithm, however, cannot be applied as the graph is directed. For this purpose, we apply Edmonds’ algorithm [16] for finding a maximum optimum branching of a weighted directed graph. Using this algorithm, we can find a subset of the current edge set, which is an optimized taxonomy where every non-root node has in-degree 1 and the sum of the edge weights is maximized.

6.4 Performance Evaluation

For the evaluation, we conduct two experiments as follows. The first experiment is to evaluate the performance of our proposed word embedding clustering (WEC) approach based on the curated databases’ sub-taxonomies (Section 6.4.2). We also compare our WEC approach with four other state-of-the-art methods in the literature, namely Kozareva et al. [37], Navigli et al. [54], Ruiji et al. [23] and Tuan1 approach (in Chapter 3). In addition, for the Animal domain, the reported performance of Bansal et al. [2], a recent work that constructs taxonomy using belief propagation, is also compared. The second experiment is to evaluate the performance of our WEC approach by constructing new taxonomies for the Finance and AI domains (Section 6.4.3). An empirical comparison with WordNet is also presented in Section 6.4.4.

6.4.1 Datasets

We evaluate our proposed approach for taxonomy construction using the collections from six different domains, namely Artificial Intelligence (AI), Finance, Virus, Animal, Plant and Vehicle. They are the same dataset used in Chapter 3 and Chapter 4.

From the documents in these corpora, we apply the term extraction and filtering techniques to extract the list of domain-relevant terms. The number of extracted terms in each domain is given in Table

6.1.

Table 6.1: Number of Terms in Each Domain

Domain	Number of Domain Terms
AI	2914
Finance	2348
Virus	282
Animal	659
Plant	520
Vehicle	117

Parameter Settings: We set the threshold value ϵ in Equation (6.3) as 0.25 and the threshold value ξ in Equation (6.4) as 0.15. The determination of these parameters will be discussed in Section 6.4.5.

6.4.2 Performance Based on Curated Databases

We evaluate the performance of the constructed taxonomies for four domains (i.e. Animal, Plant, Vehicle and Virus) against the corresponding sub-taxonomies of curated databases. For the Animal, Plant and Vehicle domains, we use WordNet as the gold standard, whereas for the Virus domain, we use MeSH sub-taxonomy of virus as the reference. The evaluation uses the following measures:

$$Precision = \frac{\#relations\ found\ in\ database\ and\ by\ the\ method.}{\#relations\ found\ by\ the\ method.};$$

$$Recall = \frac{\#relations\ found\ in\ database\ and\ by\ the\ method.}{\#relations\ found\ in\ database};\ and$$

$$F-measure = 2 \times \frac{Precision \times Recall}{Precision + Recall}.$$

Tables 6.2 and 6.3 summarize the experimental results. Our proposed word embedding clustering (WEC) approach achieves significantly better performance than the methods of Kozareva et al.

Table 6.2: Experimental Results for the Animal and Plant Domains

	Animal			Plant		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Kozareva	98%	38%	55%	97%	39%	56%
Navigli	97%	44%	61%	97%	38%	55%
Tuan1	95%	56%	70%	95%	53%	68%
Fu	81%	43%	56%	84%	36%	50%
WEC	82%	67%	74%	83%	63%	72%
Bansal	84%	55%	67%			

Table 6.3: Experimental Results for the Vehicle and Virus Domains

	Vehicle			Virus		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Kozareva	99%	60%	75%	97%	31%	47%
Navigli	91%	49%	64%	99%	37%	54%
Tuan1	93%	69%	79%	93%	43%	59%
Fu	89%	44%	59%	85%	30%	44%
WEC	87%	77%	82%	85%	53%	65%

[38], Navigli et al. [54] and Tuan1 in terms of recall and F-measure (t-test, p-value < 0.05) for all the four domains. The intuitive explanation is that these three state-of-the-art methods mainly use pattern-based approach to find taxonomic relations. The patterns can be manually defined, or learned automatically. However, due to the wide range of complex linguistic structures, it is difficult to discover all possible patterns of taxonomic relations. Instead of explicitly discovering these patterns, our proposed WEC approach can encode this information through the process of embedding learning. Thus, it is able to detect a wider range of taxonomic relations.

For the Animal domain, it also shows better performance than the reported performance of Bansal et al. [2]. One explanation is that Bansal’s method learns for taxonomy induction with belief propagation. Their learning method, however, requires information of patterns between term pairs. Thus, it has similar limitation to pattern-based methods.

In addition, our proposed WEC approach outperforms the word embedding method of Fu et al. [23] by up to 18%-23% on F-measure. It indicates that our WEC approach is more effective in exploiting word embedding information to infer taxonomic relations than Fu’s method in various domains.

6.4.3 Performance Based on New Domains

As the taxonomies for Finance and AI domains are much larger than those of the curated domains, in order to estimate the precision of a given method, we randomly choose 100 taxonomic relations among the results of the method and manually check their correctness. The results summarized in Table 6.4 show that our proposed WEC approach extracts much more relations, though with slightly lower precision, than the other state-of-the-art methods. Note that due to the lack of gold standards in these two domains, we do not compare the methods in terms of F-measure. When compared to the word embedding method of Fu et al. [23], our WEC approach has achieved similar precision, but with significant improvement in the number of extracted relations (up to 2.5 to 3 times). It indicates that for new and large sized domains, our proposed WEC approach is effective for taxonomy construction, whereas Fu’s approach suffers from deficiency when applying to those domains.

6.4.4 Empirical Comparison with WordNet

By error analysis, we found that our results may complement WordNet. For example, in the Animal domain, our approach identifies ‘*wild sheep*’ as a hyponym of ‘*sheep*’, but in WordNet, they are siblings. However, many references ^{1, 2} consider ‘*wild sheep*’ as a species of ‘*sheep*’. Another

¹<http://en.wikipedia.org/wiki/Ovis>

²<http://www.bjornefabrikken.no/side/norwegian-sheep/>

Table 6.4: Experimental Results for the Finance and AI Domains

	Finance		AI	
	Precision	# of Extracted Relations	Precision	# of Extracted Relations
Kozareva	90%	753	94%	950
Navigli	88%	1161	93%	1711
Tuan1	85%	1312	90%	1838
Fu	82%	637	83%	815
WEC	81%	1605	83%	2461

such example is in the Plant domain, our approach recognizes ‘*lily*’ as a hyponym of ‘*flowering plant*’, but WordNet places them in different subtrees incorrectly³. Therefore, our results may help restructure and even extend WordNet.

6.4.5 Parameter Tuning

In our approach for taxonomy construction, some parameters are tuned to optimize performance.

The threshold value ϵ in Equation (6.3) controls the number of extracted taxonomic relations. In general, the larger this threshold value is, the higher number of correct taxonomic relations we can get, but at the same time, the higher number of incorrect relations may incur. For our experiments, we found that the threshold value ϵ between 0.20 to 0.25 generally helps the approach achieve the best performance.

The threshold value ξ in Equation (6.4) is used to control the number of incorrect taxonomic relations which are filtered out. In general, the smaller value of this threshold is, the higher number of incorrect relations is filtered out, but at the same time, the more number of correct relations are also removed. According to our experiments, the proposed WEC approach shows the best performance if the value of ξ is between 0.10 to 0.15.

³<https://en.wikipedia.org/wiki/Lilium>

6.5 Summary

In this chapter, we have proposed an approach for taxonomy construction using word embedding clustering. Our proposed word embedding clustering approach is based on three measures of taxonomic relations in the vector space of word embeddings, namely semantic cluster, taxonomic centroid and relative distances from the root. Based on these measures that are inspired by our empirical observations of the spatial distributions of taxonomic relations in the word embedding vector space, we have proposed an unsupervised method using word embeddings to extract taxonomic relations from a list of given terms. We have also proposed a graph-based algorithm to organize the relations into a taxonomy structure, as well as to remove the redundant and incorrect relations. The experiments show that our word embedding approach has significantly outperformed the state-of-the-art methods in various domains.

Chapter 7

Learning Term Embeddings for Taxonomic Relation Identification

In recent years, there are a few studies on taxonomic relation identification using word embeddings such as the work of Tan et al. [69] and Fu et al. [23]. These studies are based on word embeddings from the Word2Vec model [50], which is mainly optimized for the purpose of analogy detection using co-occurrence based similarity learning. As such, these studies suffer from poor performance on low accuracy for taxonomic relation identification.

Recently, Yu et al. [80] proposed a supervised method to learn term embeddings based on pre-extracted taxonomic relation data. In the approach, a distance-margin neural network is proposed to learn term embeddings based on the pre-extracted taxonomic relations from the Probase database [75]. However, the neural network is trained using only the information of the term pairs (i.e. hypernym and hyponym) without considering the contextual information between them, which has been shown to be an important indicator for identifying taxonomic relations from previous studies [42, 70, 72]. Moreover, if a pair of terms is not contained in the training set, there is high possibility that it will become a negative example in the learning process, and will likely be recognized as a non-taxonomic relation. The key assumption behind the design of this approach is not always true as no available dataset can possibly contain all taxonomic relations.

In this chapter, we discuss the proposed term embedding learning approach and its performance results.

7.1 Proposed Approach

To learn term embeddings, we propose a novel approach based on *dynamic weighting neural network* to encode not only the information of hypernym and hyponym, but also the contextual information between them for the purpose of taxonomic relation identification. We then apply the identified embeddings as features to find the positive taxonomic relations using the supervised method SVM. The experimental results show that our proposed term embedding learning approach outperforms other state-of-the-art embedding learning methods for identifying taxonomic relations with much higher accuracy for both general and specific domains. In addition, another advantage of our proposed approach is that it is able to generalize from the training dataset the taxonomic relation properties for unseen pairs. Thus, it can recognize some true taxonomic relations which are not even defined in dictionary and training data.

7.2 Learning Term Embeddings

Figure 7.1 shows the proposed approach for learning term embeddings which consists of the following three steps: (i) extracting taxonomic relations; (ii) extracting training triples; and (iii) training neural network. First, we extract from WordNet all taxonomic relations as training data. Then, we extract from Wikipedia all sentences which contain at least one pair of terms involved in a taxonomic relation in the training data, and from that we identify the triples of hypernym, hyponym and contextual words between them. Finally, using the extracted triples as input, we propose a *dynamic weighting neural network* to learn term embeddings based on the information of these triples.

Figure 7.1: Proposed Approach for Learning Term Embeddings

7.2.1 Extracting Taxonomic Relations

This step aims to extract a set of taxonomic relations for training. For this purpose, we use WordNet hierarchies for extracting all (direct and indirect) taxonomic relations between noun terms in WordNet. However, based on our observation, the relations involving with top-level terms such as ‘object’, ‘entity’ or ‘whole’ are usually ambiguous and become noise for the learning purpose. Therefore, we exclude from the training set all relations which involve with those top-level terms. As a result, the total number of extracted taxonomic relations is 236,058.

7.2.2 Extracting Training Triples

This step aims to extract the triples of hypernym, hyponym and the contextual words between them. These triples will serve as the inputs to the neural network for training. In this research, we define contextual words as *all words located between the hypernym and hyponym in a sentence*. We use the latest English Wikipedia corpus as the source for extracting such triples.

Using the set of taxonomic relations extracted from the first step as reference, we extract from the Wikipedia corpus all sentences which contain at least two terms involved in a taxonomic relation. Specifically, for each sentence, we use the Stanford parser [47] to parse it, and check whether there is any pair of terms which are nouns or noun phrases in the sentence having a taxonomic relationship. If yes, we extract the hypernym, hyponym and all words between them from the sentence as a training triple. In total, we have extracted 15,499,173 training triples from Wikipedia.

Here, we apply the Stanford parser rather than matching the terms directly in the sentence in order to avoid term ambiguity as a term can serve for different grammatical functions such as noun or verb. For example, consider the following sentence:

- *Many supporters book tickets for the premiere of his new publication.*

The triple (*'publication'*, *'book'*, *'tickets for the premiere of his new'*) may be incorrectly added to the training set due to the occurrence of the taxonomic pair (*'publication'*, *'book'*), even though the meaning of *'book'* in this sentence is not about the *'publication'*.

7.2.3 Training Neural Network

Contextual information is an important indicator for detecting taxonomic relations. For example, in the following two sentences:

- *Dog is a type of animal which you can have as a pet.*

Figure 7.2: Architecture of the Proposed Dynamic Weighting Neural Network Model

- *Animal such as dog is more sensitive to sound than human.*

The occurrence of contextual words ‘*is a type of*’ and ‘*such as*’ can be used to identify the taxonomic relation between ‘*dog*’ and ‘*animal*’ in the sentences. Many works in the literature [37, 54, 75] attempted to manually find these contextual patterns, or automatically learn them. However, due to the wide range of complex linguistic structures, it is difficult to discover all possible contextual patterns between hypernyms and hyponyms in order to detect taxonomic relations effectively.

In this chapter, instead of explicitly discovering the contextual patterns of taxonomic relations, we propose a *dynamic weighting neural network* to encode this information, together with the hypernym and hyponym, for learning term embeddings. Specifically, the target of the neural network is to predict the hypernym term from the given hyponym term and contextual words. The architecture of the proposed neural network is shown in Figure 7.2, which consists of three layers: input layer, hidden layer and output layer.

In our setting, the vocabulary size is V , and the hidden layer size is N . The nodes on adjacent layers are fully connected. Given a term/word t in the vocabulary, the input vector of t is encoded as a one-hot V -dimensional vector x_t , i.e. x_t consists of 0s in all elements except the element used to uniquely identify t which is set as 1. The weights between the input layer and output layer are represented by a $V \times N$ matrix W . Each row of W is a N -dimensional vector representation v_t of the associated word/term t of the input layer.

Given a hyponym term $hypo$ and k context words $c_1, c_2, \dots, c_k$ in the training triple, the output of hidden layer h is calculated as:

$$\begin{aligned} h &= W^\top \cdot \frac{1}{2k} (k \times x_{hypo} + x_{c_1} + x_{c_2} + \dots + x_{c_k}) \\ &= \frac{1}{2k} (k \times v_{hypo} + v_{c_1} + v_{c_2} + \dots + v_{c_k}), \end{aligned} \quad (7.1)$$

where v_t is the vector representation of the input word/term t .

The weight of h in Equation (7.1) is calculated as the average of the vector representation of hyponym term and contextual words. Therefore, this weight is not based on a fixed number of inputs. Instead, it is dynamically updated based on the number of contextual words k in the current training triple, and the hyponym term. This model is called *dynamic weighting neural network* to reflect its dynamic nature. Note that to calculate h , we also multiply the vector representation of hyponym by k to reduce the bias problem of high number of contextual words, so that the weight of the input vector of hyponym is balanced with the total weight of contextual words.

From the hidden layer to the output layer, there is another weight $N \times V$ for the output matrix W' . The j^{th} column of W' is a N -dimensional vector v'_{t_j} representing the output vector of t_j . Using these weights, we can compute an output score u_{t_j} for each term/word t_j in the vocabulary:

$$u_{t_j} = v'_{t_j}^\top \cdot h, \quad (7.2)$$

where v'_{t_j} is the j -column of the matrix W' .

We then use soft-max, a log-linear classification model, to obtain the posterior distribution of hypernym terms as follows:

$$\begin{aligned}
 y_j &= p(hype|hypo, c_1, c_2, \dots, c_k) \\
 &= \frac{e^{u_{hype}}}{\sum_{i=1}^V e^{u_i}} \\
 &= \frac{e^{v_{hype}^\top \cdot \frac{1}{2k} (k \times v_{hypo} + \sum_{j=1}^k v_{c_j})}}{\sum_{i=1}^V e^{v_i^\top \cdot \frac{1}{2k} (k \times v_{hypo} + \sum_{j=1}^k v_{c_j})}} ,
 \end{aligned} \tag{7.3}$$

where y_j is the output of the j^{th} node (hypernym) of the output layer.

The objective function is then defined as:

$$O = \frac{1}{T} \sum_{t=1}^T \log(p(hype_t|hypo_t, c_{1t}, c_{2t}, \dots, c_{kt})) , \tag{7.4}$$

where T is the number of training triples; $hype_t$, $hypo_t$ and c_{it} are hypernym term, hyponym term and contextual words respectively in the training triple t .

After maximizing the log-likelihood objective function in Equation (7.4) over the entire training set using stochastic gradient descent, the term embeddings are learned accordingly.

7.3 Learning Neural Network

7.3.1 Weight Update Equation from Hidden Layer to Output Layer

For each training triple, the objective is to maximize Equation (7.3), the conditional probability of observing the actual hypernym (with its index in the output layer as j^*) given the hyponym and context words.

We have:

$$\begin{aligned}
\max y_{j^*} &= \max(p(hype|hypo, c_1, c_2, \dots, c_k)) \\
&:= \max(\log((p(hype|hypo, c_1, c_2, \dots, c_k)))) \\
&= u_{t^*} - \log \sum_{t=1}^V e^{u_t} \\
&:= -L,
\end{aligned} \tag{7.5}$$

where $L = -\log p(hype|hypo, c_1, c_2, \dots, c_k)$ is the loss function, and we need to minimize it.

Take the derivation of L on output score u_t , we have:

$$\frac{\partial L}{\partial u_{t_j}} = y_j - t_j := e_j, \tag{7.6}$$

where $t_j = 1$ iff $j = j^*$, otherwise $t_j = 0$. e_j is considered as the prediction error of the j -th node in the output layer.

Take the derivation of L on each element w'_{ij} of the output matrix W' :

$$\begin{aligned}
\frac{\partial L}{\partial w'_{ij}} &= \frac{\partial L}{\partial u_{t_j}} \cdot \frac{\partial u_{t_j}}{\partial w'_{ij}} \\
&= e_j \cdot h_i,
\end{aligned} \tag{7.7}$$

where h_i is the i -th element of the hidden layer h .

Using stochastic gradient descent, we have the weight updating equation for the hidden layer to output layer as follows:

$$w'_{ij} = w'_{ij} - \tau \cdot e_j \cdot h_i, \tag{7.8}$$

or

$$v'_{t_j} = v'_{t_j} - \tau \cdot e_j \cdot h, \tag{7.9}$$

where τ is the learning rate.

7.3.2 Weight Update Equation from Input Layer to Hidden Layer

Take the derivation of L on the hidden layer, we have:

$$\begin{aligned}\frac{\partial L}{\partial h} &= \sum_{j=1}^V \frac{\partial L}{\partial u_{t_j}} \cdot \frac{\partial u_{t_j}}{\partial h} \\ &= \sum_{j=1}^V e_j \cdot v'_{t_j}.\end{aligned}\tag{7.10}$$

Now take the derivation of L on each row v_i of the input matrix W :

$$\begin{aligned}\frac{\partial L}{\partial v_i} &= \frac{\partial L}{\partial h} \cdot \frac{\partial h}{\partial v_i} \\ &= \left(\sum_{j=1}^V e_j \cdot v'_{t_j} \right) \cdot \frac{\partial h}{\partial v_i}.\end{aligned}\tag{7.11}$$

Therefore, using stochastic gradient descent, we have the weight updating equation for the input layer to hidden layer as follows:

$$\begin{aligned}v_{hypo} &= v_{hypo} - \frac{1}{2} \tau \sum_{j=1}^V e_j \cdot v'_{t_j}, \\ v_{c_i} &= v_{c_i} - \frac{1}{2k} \tau \sum_{j=1}^V e_j \cdot v'_{t_j},\end{aligned}\tag{7.12}$$

where τ is the learning rate and e_j is the prediction error.

7.4 Supervised Taxonomic Relation Identification

To decide whether a term x is a hypernym of term y , we build a classifier that uses embedding vectors as features for taxonomic relation identification. Specifically, we use Support Vector Machine (SVM) [12] for this purpose. Given an ordered pair (x, y) , the input feature is the concatenation of embedding vectors (v_x, v_y) of x and y . In addition, our term embedding learning approach has the property that the embedding of hypernym is encoded based on not only the information of hyponym

but also the information of contextual words. Therefore, we add one more feature to the input of SVM, i.e. the offset vector $(v_x - v_y)$, to contain the information of all contextual words between x and y . In summary, the feature vector is a $3d$ dimensional vector $\langle v_x, v_y, v_x - v_y \rangle$, where d is the dimension of term embeddings. As will be shown later in the experimental results, the offset vector plays an important role in the task of taxonomic relation identification of our approach.

7.5 Performance Evaluation

We conduct experiments to evaluate the performance of our term embedding learning approach on the general domain areas as well as the specific domain areas. In performance evaluation, we compare our approach with two other state-of-the-art supervised term embedding learning methods in Yu et al. [80] and the Word2Vec model [50].

7.5.1 Datasets

There are five datasets used in the experiments. Two datasets, namely BLESS and ENTAILMENT, are general domain datasets. The other three datasets, namely Animal, Plant and Vehicle, are specific domain datasets.

- BLESS [4] dataset: It covers 200 distinct, unambiguous concepts (terms); each of which is involved with other terms, called *relata*, in some relations. We extract from BLESS 14,547 pairs of terms for the following four types of relations: taxonomic relation, meronymy relation (a.k.a. part-of relation), coordinate relation (i.e. two terms having the same hypernym), and random relation. From these pairs, we set taxonomic relations as positive examples, while the other relations form the negative examples.
- ENTAILMENT dataset [3]: It consists of 2,770 pairs of terms, with equal number of positive and negative examples of taxonomic relations. Altogether, there are 1,376 unique hyponyms and 1,016 unique hypernyms.

- Animal, Plant and Vehicle datasets: These datasets are also used in Chapter 3 and Chapter 4 for performance evaluation. The positive examples are created by extracting all possible (direct and indirect) taxonomic relations from the taxonomies. The negative examples are generated by randomly pairing two terms which are not involved in any taxonomic relation.

The number of terms, positive examples and negative examples extracted from the five datasets are summarized in Table 7.1.

Table 7.1: Datasets used in the Experiments

Dataset	# terms	# positive	# negative
BLESS	5229	1337	13210
ENTAILMENT	2392	1385	1385
Animal	659	4164	8471
Plant	520	2266	4520
Vehicle	117	283	586

7.5.2 Comparison Models

In the experiments, we use the following supervised models for comparison:

- SVM+TuanTE: This model uses SVM and the term embeddings obtained by our learning approach. The input is a 3d-dimensional vector $\langle v_x, v_y, v_x - v_y \rangle$, where d is the dimension of term embeddings, x and y are two terms used to check whether x is a hypernym of y or not, and v_x, v_y are the term embeddings of x and y respectively.
- SVM+Word2Vec: This model uses SVM and the term embeddings obtained by applying the Skip-gram model [50] on the entire English Wikipedia corpus. The input is also a 3d-dimensional vector as in the SVM+TuanTE model. Note that the results of the Skip-gram model are word embeddings. So if a term is a multiword term, its embedding is calculated as the average of all words in the term.

- **SVM+Yu:** This model uses SVM and the term embeddings obtained by using Yu et al.'s method [80]. According to the best setting stated in [80], the input is a $2d+1$ dimensional vector $\langle O(x), E(y), \|O(x)-E(y)\|_1 \rangle$, where $O(x)$, $E(y)$ and $\|O(x)-E(y)\|_1$ are hyponym embedding of x , hypernym embedding of y and 1-norm distance of the vector $(O(x)-E(y))$ respectively.

Parameter Settings: The SVM in the three models is trained using a RBF kernel with $\lambda=0.03125$ and penalty term $C=8.0$. For term embedding learning, the vector's dimension is set to 100. The tuning of the dimension will be discussed in Section 7.5.5.

7.5.3 Performance on General Domain Datasets

For the general domain datasets, we have conducted two experiments to evaluate the performance of our proposed approach.

Experiment 1

For the BLESS dataset, we hold out one concept for testing and train on the remaining 199 concepts. The hold-out concept and its relatum constitute the testing set, while the remaining 199 concepts and their relatum constitute the training set. To further separate the training and testing sets, we exclude from the training set any pair of terms that has one term appearing in the testing set. We report the average accuracy across all concepts.

For the ENTAILMENT dataset, we use the same evaluation method: hold out one hypernym for testing and train on the remaining hypernyms, and we also report the average accuracy across all hypernyms.

Furthermore, to evaluate the effect of the offset vector to taxonomic relation identification, we deploy a setting that removes the offset vector in the feature vectors of SVM. Specifically, for SVM+TuanTE and SVM+Word2Vec, the input vector is changed from $\langle v_x, v_y, v_x - v_y \rangle$ to $\langle v_x, v_y \rangle$. We use the subscript *short* to denote this setting.

Table 7.2: Performance Results for the BLESS and ENTAILMENT Datasets

Model	Dataset	Accuracy
SVM+Yu	BLESS	90.4%
SVM+Word2Vec _{short}	BLESS	83.8%
SVM+Word2Vec	BLESS	84.0%
SVM+TuanTE _{short}	BLESS	91.1%
SVM+TuanTE	BLESS	93.6%
SVM+Yu	ENTAIL	87.5%
SVM+Word2Vec _{short}	ENTAIL	82.8%
SVM+Word2Vec	ENTAIL	83.3%
SVM+TuanTE _{short}	ENTAIL	88.2%
SVM+TuanTE	ENTAIL	91.7%

Table 7.2 shows the performance of the three supervised models in Experiment 1. Our approach achieves significantly better performance than Yu’s method and Word2Vec method in terms of accuracy (t-test, p-value < 0.05) for both BLESS and ENTAILMENT datasets. Specifically, our approach improves the average accuracy by 4% compared to Yu’s method, and by 9% compared to the Word2Vec method. The Word2Vec embeddings have the worst result because it is based only on co-occurrence based similarity, which is not effective for the classifier to accurately recognize all the taxonomic relations. Our approach performs better than Yu’s method and it shows that our approach can learn embeddings more effectively. Our approach encodes not only hypernym and hyponym terms but also the contextual information between them, while Yu’s method ignores the contextual information for taxonomic relation identification.

Moreover, from the experimental results of SVM+TuanTE and SVM+TuanTE_{short}, we can observe that the offset vector between hypernym and hyponym, which captures the contextual information, plays an important role in our approach as it helps to improve the performance in both datasets. However, the offset feature is not so important for the Word2Vec model. The reason is that the Word2Vec model is targeted for the analogy task rather than taxonomic relation identification.

Experiment 2

This experiment aims to evaluate the generalization capability of our extracted term embeddings. In the experiment, we train the classifier on the BLESS dataset, test it on the ENTAILMENT dataset and vice versa. Similarly, we exclude from the training set any pair of terms that has one term appearing in the testing set. Table 7.3 shows the performance results for the general domain datasets when using one domain for training and another for testing. As shown in Table 7.3, our term embedding learning approach performs better than other methods in accuracy. It also shows that the taxonomic properties identified by our term embedding learning approach have great generalization capability (i.e. less dependent on the training set), and can be used generically for representing taxonomic relations.

Table 7.3: Performance Results for the General Domain Datasets

Model	Training	Testing	Accuracy
SVM+Yu	BLESS	ENTAIL	83.7%
SVM+Word2Vec _{short}	BLESS	ENTAIL	76.5%
SVM+Word2Vec	BLESS	ENTAIL	77.1%
SVM+TuanTE _{short}	BLESS	ENTAIL	85.8%
SVM+TuanTE	BLESS	ENTAIL	89.4%
SVM+Yu	ENTAIL	BLESS	87.1%
SVM+Word2Vec _{short}	ENTAIL	BLESS	78.0%
SVM+Word2Vec	ENTAIL	BLESS	78.9%
SVM+TuanTE _{short}	ENTAIL	BLESS	87.1%
SVM+TuanTE	ENTAIL	BLESS	90.6%

7.5.4 Performance on Specific Domain Datasets

Similarly, for the specific domain datasets, we have conducted two experiments to evaluate the performance of our proposed approach.

Experiment 3

For each of the Animal, Plant and Vehicle datasets, we also hold out one term for testing and train on the remaining terms. The positive and negative examples which contain the hold-out term constitute the testing set, while other positive and negative examples constitute the training set. We also exclude from the training set any pair of terms that has one term appearing in the testing set. The experimental results are given in Table 7.4. We can observe that not only for general domain datasets but also for specific domain datasets, our term embedding learning approach has achieved significantly better performance than Yu’s method and the Word2Vec method in terms of accuracy (t-test, p-value < 0.05). Specifically, our approach improves the average accuracy by 22% compared to Yu’s method, and by 9% compared to the Word2Vec method.

Table 7.4: Performance Results for the Animal, Plant and Vehicle Datasets

Model	Dataset	Accuracy
SVM+Yu	Animal	67.8%
SVM+Word2Vec	Animal	80.2%
SVM+TuanTE	Animal	89.3%
SVM+Yu	Plant	65.7%
SVM+Word2Vec	Plant	81.5%
SVM+TuanTE	Plant	92.1%
SVM+Yu	Vehicle	70.5%
SVM+Word2Vec	Vehicle	82.1%
SVM+TuanTE	Vehicle	89.6%

Another interesting point to observe is that the accuracy of Yu’s method drops significantly in specific domain datasets (as shown in Table 7.4) when compared to the general domain datasets (as shown in Table 7.2). One possible explanation is the accuracy of Yu’s method depends on the training data. As Yu’s method learns the embeddings using pre-extracted taxonomic relations from Probase, and if a relation does not exist in Probase, there is high possibility that it becomes a nega-

tive example and be recognized as a non-taxonomic relation by the classifier. Therefore, the training data extracted from Probase plays an important role in Yu's method. For general domain datasets (BLESS and ENTAILMENT), there are about 75%-85% of taxonomic relations in these datasets found in Probase, while there are only about 25%-45% of relations in the specific domains (i.e. Animal, Plant and Vehicle) found in Probase. Therefore, Yu's method achieves better performance in general domain datasets than the specific ones. Our approach, in contrast, depends less on the training relations. Therefore, it can achieve high accuracy in both the general and specific domain datasets.

Experiment 4

Similar to experiment 2, this experiment aims to evaluate the generalization capability of our term embeddings. In this experiment, for each of the Animal, Plant and Vehicle domains, we train the classifier using the positive and negative examples in each domain and test the classifier in other domains. Table 7.5 shows the performance results for the specific domain datasets using one domain for training and another domain for testing. As shown in Table 7.5, our approach achieves the best performance compared to other state-of-the-art methods for all the datasets. As also shown in Table 7.3, our approach has achieved high accuracy for both general and specific domain datasets, while in Yu's method, there is a huge difference in accuracy between these domain datasets.

7.5.5 Tuning Vector Dimensions

We also conduct experiments to learn term embeddings from the general domain datasets with different dimensions (i.e. 50, 100, 150 and 300) using our proposed approach. We then use these embeddings to evaluate the performance of taxonomic relation identification based on training time and accuracy, and show the results in Table 7.6. The experiments are carried out on a PC with Intel(R) Xeon(R) CPU at 3.7GHz and 16GB RAM.

In general, when increasing the vector dimension, the accuracy of our term embedding learning approach will be increased gradually. More specifically, the accuracy improves slightly when the

Table 7.5: Performance Results for the Specific Domain Datasets

Model	Training	Testing	Accuracy
SVM+Yu	Animal	Plant	65.5%
SVM+Word2Vec	Animal	Plant	82.4%
SVM+TuanTE	Animal	Plant	91.9%
SVM+Yu	Animal	Vehicle	66.2%
SVM+Word2Vec	Animal	Vehicle	81.3%
SVM+TuanTE	Animal	Vehicle	89.5%
SVM+Yu	Plant	Animal	68.4%
SVM+Word2Vec	Plant	Animal	81.8%
SVM+TuanTE	Plant	Animal	91.5%
SVM+Yu	Plant	Vehicle	65.2%
SVM+Word2Vec	Plant	Vehicle	81.0%
SVM+TuanTE	Plant	Vehicle	88.5%
SVM+Yu	Vehicle	Animal	70.9%
SVM+Word2Vec	Vehicle	Animal	79.7%
SVM+TuanTE	Vehicle	Animal	87.6%
SVM+Yu	Vehicle	Plant	66.2%
SVM+Word2Vec	Vehicle	Plant	78.7%
SVM+TuanTE	Vehicle	Plant	87.7%

dimension is increased from 50 to 150. But after that, increasing the dimension has very little effect on the accuracy. We observe that the vector dimension for learning term embeddings can be set between 100 to 150 to achieve the best performance, based on the trade-off between accuracy and training time.

Table 7.6: Performance Results based on Training Time and Accuracy of the SVM+TuanTE Model using Different Vector Dimensions

Dimension	Dataset	Training time	Accuracy
50	BLESS	1825s	87.7%
100	BLESS	2991s	89.4%
150	BLESS	4025s	89.9%
300	BLESS	7113s	90.0%
50	ENTAIL	1825s	88.5%
100	ENTAIL	2991s	90.6%
150	ENTAIL	4025s	90.9%
300	ENTAIL	7113s	90.9%

7.6 Summary

In this chapter, we have proposed a novel approach to learn term embeddings using dynamic weighting neural network. This model encodes not only the hypernym and hyponym terms, but also the contextual information between them. Therefore, the extracted term embeddings have good generalization capability to identify unseen taxonomic relations which are not even defined in dictionary and training data. The experimental results show that our approach significantly outperforms other state-of-the-art methods in terms of accuracy in identifying taxonomic relation identification.

Chapter 8

Conclusion

In this chapter, we first summarize the work that has been done in this research. Then, we give the directions for further research work.

8.1 Summary

Taxonomy plays an important role in many research areas. Even though there are many available general hand-crafted taxonomies, there is still a need to build taxonomies for specific domains.

In this thesis, we have presented the motivation, objectives and related work of this research for automatic domain-specific taxonomy construction from textual documents. In this research, we focus on investigating linguistics, statistical and deep learning methods for the purpose of creating an effective framework for taxonomy construction.

The main contributions of this research are summarized as follows:

- We proposed a Web-based method to extract domain terms from a given text collection. From that, we proposed a method to use the contextual information of the terms in syntactic structures to detect taxonomic relations across sentence boundary. In addition, we also proposed

a novel graph-based algorithm to organize the extracted taxonomic relations into an optimal taxonomy tree. The experimental results show that the proposed method is well complementary to the previous methods of linguistic pattern matching and significantly improves recall and F-measure.

- We studied two important aspects that can greatly affect the performance of taxonomy construction method. The first one is on the trustiness of individual source texts, which is important to filter out incorrect relations from unreliable sources. The second one is on the collective evidence from synonyms and contrastive terms, where synonyms provide additional supports to taxonomic relation identification, while contrastive terms may contradict them. We proposed an approach to incorporate these features into taxonomy construction, which can improve the performance on F-measure by up to 4%-10%.
- We proposed a time-aware approach to extract and integrate temporal information into the process of identifying taxonomic relations, by employing a timestamp contribution function to measure the evidence scores of source texts at a particular time. Experimental results show that our proposed approach outperforms the state-of-the-art methods on F-measure by up to 7%-20%. Furthermore, the proposed approach can incrementally and continuously update the taxonomy by adding fresh relations from new data and removing outdated relations, using a proposed information decay function. It thus avoids rebuilding the whole structure from scratch for every update and maintains the taxonomy up-to-date in order to keep up with the latest information trends quickly.
- We proposed a novel unsupervised approach to construct taxonomies based on word embedding clustering, using the following three word embedding measures: semantic clusters, taxonomic centroids and relative distances from the root, for identifying the semantic relationships between terms and their hypernyms. Our proposed approach significantly outperforms the state-of-the-art methods in terms of recall and F-measure.
- We proposed an approach to learn word embeddings for taxonomic relations based on the contextual words between the hypernym and hyponym using a dynamic weighting neural

network. Our proposed approach significantly outperforms the state-of-the-art methods by 9% to 13% in terms of accuracy for both general and specific domain datasets.

8.2 Future Work

For future work, we will focus on the following directions: extending the proposed taxonomy construction methods, investigating techniques for non-taxonomic relation extraction and ontology learning, investigating probabilistic model for taxonomic relations and investigating applications with taxonomy.

8.2.1 Extending the Proposed Taxonomy Construction Methods

In our proposed methods for taxonomy construction, further extensions can be carried out for future work. For example, in the proposed method using syntactic contextual evidence in Chapter 3, we use the syntactic structure *Subject-Verb-Object* as it gives the best performance compared to other syntactic structures such as *Noun-Adjective-Noun* or *Noun-Preposition-Noun* (see Section 3.5.4 for details). One possible extension is to identify other syntactic structures, tune the threshold values of different patterns and investigate the performance. If possible, we can try to find the combination of different structures that can give the best performance.

For the method of taxonomy construction which utilizes timestamp information (see Chapter 5), we can elaborate the method with the following extensions:

- Try different timestamp contribution and decay functions to see how they affect the performance of the method. We can also refer to other works about time modeling in the information retrieval community to learn different models of time representation.
- Instead of using average time as in Equations (5.2) and (5.3) (see Section 5.4.1), which tends to be biased in some cases (e.g. one document is one day ago vs. one document is one year ago), we can use other statistical method such as median, probability distribution, etc.

- Combine domain term ranking with taxonomic relation ranking in the taxonomy induction algorithm (see Algorithm 3 in Section 5.4.3). For example, in the edge filtering step, if a term which is a vertex of a filtered edge is an emerging term (i.e. has high evidence score in domain term extraction method), we may need to keep it in the hierarchy even though it does not have any connections to other edges in the hierarchy.

For the method of learning term embeddings for the taxonomy construction task (see Chapter 7), rather than using the one-hidden layer neural network, we can explore more sophisticated sequence/recurrent models for hypernym prediction such as Long Short Term Memory (LSTM) network [30] or Hierarchical Recurrent Neural Network [63] to improve the accuracy of the taxonomic relation identification.

8.2.2 Investigating Techniques for Non-taxonomic Relation Extraction and Ontology Learning

Taxonomy is the backbone of ontology. However, to achieve a complete ontology, we also need other relations such as “part-of”, “is-derived-from” or “located-in” rather than taxonomic relations. There are some previous studies about ontology learning such as [31, 60]. However, those works are still limited on the performance of relation extraction. In future work, we may extend the non-taxonomic relation extraction method to infer relations in other patterns rather than just using *Subject-Verb-Object* pattern as in our current work. For example, from the following sentence: “*The car’s wheel is out of order*”, we may need to infer a non-taxonomic relation “part-of” between “*wheel*” and “*car*”.

We can then combine the non-taxonomic relation extraction with the proposed taxonomic relation extractions using some available taxonomies to present a novel framework for ontology learning. Some other tasks may need to be done in ontology learning such as word sense disambiguation [52], term clustering [7] and relation labeling [81].

8.2.3 Investigating Probabilistic Model for Taxonomic Relations

Information is not only black or white. If we eliminate all the uncertainties, it will be very harmful. We may need to have another way to live with those uncertainties, and probabilistic model for taxonomic relations can be an option. This model can significantly contribute to some important applications such as semantic web search or short text understanding [75]. It can conceptualize from a set of words using Bayesian calculation based on the probability $p(\text{concept}|\text{sub-concept})$. For example, given a word *India*, we can infer its most typical concepts such as *country* or *region*. Given two words, *India* and *China*, the most typical concepts become *Asian country* or *developing country*, etc. Adding another word, *Brazil*, the top concepts may become *BRIC* or *emerging market*, etc. To build such probabilistic model, we may rely on the evidence score in the proposed taxonomic relation extraction method, combine it with joint probability in Bayesian model, propose some characterized features and use available taxonomies such as WordNet to learn the model.

8.2.4 Investigating Applications with Taxonomy

With the constructed taxonomy, we can investigate the following applications: semantic web search and short text understanding.

Most of the current search engines rely on keyword-based search. For example, given the query “*terrorist group in Middle East countries*”, the keyword-based search engines cannot deliver good results as they return only pages with exact, word-for-word matches with terms such as “*terrorist group*” or “*Middle East countries*”. One strategy to improve search performance is to search not only the original query but also the refined queries where we substitute each term in the query with its sub-terms found in taxonomy. Because there are many combinations of the substitutions of query terms, we only replace term with its most popularity sub-terms based on the edge scores in taxonomy. For example, according to the Terrorism taxonomy created by our approach as described in Chapter 3, we can replace “*terrorist group*” with its most popularity terms “*Al-Qaeda*”, “*Hezbollah*”, “*Hamas*”, etc., and replace “*Middle East countries*” with “*Iraq*”, “*Iran*”, “*Lebanon*”, etc. The refined queries will be “*Al-Qaeda in Iraq*”, “*Hezbollah in Iran*”, etc. We conduct a simple

experiment on Google and Bing search engine by alternately submitting the original query and refined queries, and check the first 100 search results to see whether they are relevant to the query. The evaluation shows that on average, 81% of the results from the refined queries are relevant, compared to 54% of those results from the original query. It has shown that the weighted taxonomy can help to bring more relevant search results to users and improve the quality of search engine.

Understanding short text (e.g. web search, tweets, anchor texts) is important to many applications. Statistical approach such as topic model treats text as a bag of words, and discovers latent topics from the text. However, the bag-of-word approach does not work well if we do not have enough features. Weighted taxonomy can be leveraged for this application. For example, given two words “*lion*” and “*hyena*”, according to the Animal taxonomy, we can infer that they are “*carnivore*” since it is the common ancestor which has the shortest weighting distance to these two terms. Adding another word “*whale*”, the topic may change to “*mammal*”. Unlike the bag-of-word approach that returns the set of words as a latent topic, taxonomy can return the concepts which are more meaningful.

Appendix A

Authors Publications

1. **Luu Anh Tuan**, Jung-jae Kim, See-Kiong Ng. Taxonomy Construction using Syntactic Contextual Evidence. *In proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pp. 810-819, 2014.
2. **Luu Anh Tuan**, Jung-jae Kim, See-Kiong Ng. Incorporating Trustiness and Collective Synonym/Contrastive Evidence into Taxonomy Construction. *In proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pp. 1013-1022, 2015.
3. **Luu Anh Tuan**, Yi Tay, Siu Cheung Hui, See-Kiong Ng. Learning the Term Embeddings for Taxonomic Relation Identification. *The Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2016 (accepted).
4. **Luu Anh Tuan**, Siu Cheung Hui, See-Kiong Ng. Utilizing Temporal Information for Topic Hierarchy Construction. *Transactions of the Association for Computational Linguistics (TACL)*, 2016 (accepted).
5. **Luu Anh Tuan**, Siu Cheung Hui, See-Kiong Ng. Word Embedding Clustering for Taxonomy Construction. *Submitted to the ACL Conference*, 2017.

Bibliography

- [1] G. Angeli, J. Tibshirani, J. Y. Wu, and C. D. Manning. Combining Distant and Partial Supervision for Relation Extraction. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 1556–1567, 2014.
- [2] M. Bansal, D. Burkett, G. D. Melo, and D. Klein. Structured learning for taxonomy induction with belief propagation. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1041–1051, 2014.
- [3] M. Baroni, R. Bernardi, N.-Q. Do, and C.-c. Shan. Entailment above the word level in distributional semantics. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, 2012.
- [4] M. Baroni and A. Lenci. How we blessed distributional semantic evaluation. *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, 2011.
- [5] Y. Bengio, R. Ducharme, and P. Vincent. A Neural Probabilistic Language Model. *Proceedings of the Conference on Neural Information Processing Systems*, pages 932–938, 2001.
- [6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, 2008.
- [7] D. Bourigault and C. Jacquemin. Term extraction+ term clustering: An integrated platform for computer-aided terminology. *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 15–22, 1999.
- [8] A. Budanitsky. Lexical semantic relatedness and its application in natural language processing. 1999.
- [9] P. Buitelaar, D. Olejnik, and M. Sintek. A Protégé Plug-in for Ontology Extraction from Text Based on Linguistic Analysis. *Proceedings of the 1st European Semantic Web Symposium*, pages 31–44, 2004.

- [10] A. X. Chang and C. D. Manning. Sutime: A library for recognizing and normalizing time expressions. *Proceedings of the Conference on Language Resources and Evaluation*, pages 3735–3740, 2012.
- [11] M. Ciaranita, A. Gangemi, E. Ratsch, J. Saric, and I. Rojas. Unsupervised Learning of Semantic Relations Between Concepts of a Molecular Biology Ontology. *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 659–664, 2005.
- [12] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.
- [13] E. Dietz, D. Vandic, and F. Frasincar. Taxolearn: A semantic approach to domain taxonomy learning. *Web Intelligence and Intelligent Agent Technology*, 1:58–65, 2012.
- [14] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *Proceedings of the VLDB Endowment*, 8(9), 2015.
- [15] L. Drumond and R. Girardi. Extracting ontology concept hierarchies from text using markov logic. *Proceedings of the ACM Symposium on Applied Computing*, pages 1354–1358, 2010.
- [16] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71:233–240, 1967.
- [17] M. Efron and G. Golovchinsky. Estimation methods for ranking recent information. *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 495–504, 2011.
- [18] S. Faralli and R. Navigli. A Java framework for multilingual definition and hypernym extraction. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 103–108, 2013.
- [19] S. Fodeh, B. Punch, and P. N. Tan. On ontology-driven document clustering using core semantic features. *Knowledge and information systems*, 28(2):395–421, 2011.
- [20] H. N. Fotzo and P. Gallinari. Learning “generalization/specialization” relations between concepts - application for automatically building thematic document hierarchies. *Proceedings of the 7th International Conference on Computer-Assisted Information Retrieval*, 2004.
- [21] K. T. Frantzi and S. Ananiadou. Extracting nested collocations. *Proceedings of the 16th conference on Computational linguistics*, pages 41–46, 1996.
- [22] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

- [23] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu. Learning semantic hierarchies via word embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1199–1209, 2014.
- [24] M. Geffet and I. Dagan. The distributional inclusion hypotheses and lexical entailment. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 107–114, 2005.
- [25] R. Girju, A. Badulescu, and D. Moldovan. Learning semantic constraints for the automatic discovery of part-whole relations. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 1–8, 2003.
- [26] B. Hachey, W. Radford, J. Nothman, M. Honnibal, and J. R. Curran. Evaluating entity linking with Wikipedia. *Artificial intelligence*, 194:130–150, 2013.
- [27] S. M. Harabagiu, S. J. Maiorano, and M. A. Pasca. Open-domain textual question answering techniques. *Natural Language Engineering*, 9(3):231–267, 2003.
- [28] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning*. Springer-Verlag, 2009.
- [29] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. *Proceedings of the 14th Conference on Computational Linguistics*, pages 539–545, 1992.
- [30] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [31] X. Jiang and A. H. Tan. CRCTOL: A semantic?based domain ontology learning system. *Journal of the American Society for Information Science and Technology*, 61(1):150–168, 2010.
- [32] J. J. Kim, Z. Zhang, J. C. Park, and S. K. Ng. Biocontrasts: extracting and exploiting protein-protein contrastive relations from biomedical literature. *Bioinformatics*, 22(5):597–605, 2006.
- [33] D. Klein and C. D. Manning. Accurate Unlexicalized Parsing. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, 2003.
- [34] J. Knijff, F. Frasincar, and F. Hogenboom. Domain taxonomy learning from text: The subsumption method versus hierarchical clustering. *Journal of Data and Knowledge Engineering*, 83:54–69, 2013.
- [35] C. Kommers, V. Ustun, A. Demski, and P. Rosenbloom. Hierarchical reasoning with distributed vector representations. *Proceedings of the 37th Annual Conference of the Cognitive Science Society*, 2015.

- [36] L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389, 2010.
- [37] Z. Kozareva and E. Hovy. A Semi-supervised Method to Learn and Construct Taxonomies Using the Web. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1110–1118, 2010.
- [38] Z. Kozareva, E. Riloff, and E. H. Hovy. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1056, 2008.
- [39] N. Krishnan, D. J. Cook, and Z. Wemlinger. Learning a taxonomy of predefined and discovered activity patterns. *Journal of Ambient Intelligence and Smart Environments*, 5(6):621–637, 2013.
- [40] G. Kruszewski, D. Paperno, and M. Baroni. Deriving boolean structures from distributional vectors. *Transactions of the Association for Computational Linguistics*, 3:375–388, 2015.
- [41] A. Lenci and G. Benotto. Identifying hypernyms in distributional semantic spaces. *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 75–79, 2012.
- [42] O. Levy, S. Remus, C. Biemann, I. Dagan, and I. Ramat-Gan. Do supervised distributional methods really learn lexical inference relations. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1390–1397, 2014.
- [43] X. Li and B. W. Croft. Time-based language models. *Proceedings of the 12th ACM International Conference on Information and Knowledge Management*, pages 469–475, 2003.
- [44] X. Liu, Y. Song, S. Liu, and H. Wang. Automatic taxonomy construction from keywords. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1433–1441, 2012.
- [45] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11):2579–2605, 2008.
- [46] A. Maedche and S. Staab. Mining ontologies from text. *Proceedings of the Conference on Knowledge Acquisition, Modeling and Management*, pages 189–202, 2000.
- [47] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 55–60, 2014.

- [48] C. Matuszek, J. Cabral, M. J. Witbrock, and J. DeOliveira. An introduction to the syntax and content of cyc. *Proceedings of the AAAI Spring Symposium*, pages 44–49, 2006.
- [49] O. Medelyan, S. Manion, J. Broekstra, and A. Divoli. Constructing a focused taxonomy from a document collection. *The Semantic Web: Semantics and Big Data. Springer Berlin Heidelberg*, pages 367–381, 2013.
- [50] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.
- [51] G. A. Miller. WordNet: a Lexical Database for English. *Communications of the ACM*, 38(11):39–41, 1995.
- [52] A. Moro, A. Raganato, and R. Navigli. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244, 2014.
- [53] R. Navigli and P. Velardi. Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites. *Computational Linguistics*, 30(2):151–179, 2004.
- [54] R. Navigli, P. Velardi, and S. Faralli. A Graph-based Algorithm for Inducing Lexical Taxonomies from Scratch. *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1872–1877, 2011.
- [55] Y. Park, R. J. Byrd, and B. K. Boguraev. Automatic Glossary Extraction: Beyond Terminology. *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, 2002.
- [56] M. S. Paukkeri, A. P. Garcia-Plaza, and V. Fresno. Learning a taxonomy from a set of text documents. *Journal of Applied Soft Computing*, 12(3):1138–1148, 2012.
- [57] L. Rimell. Distributional lexical entailment by topic coherence. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 511–519, 2014.
- [58] A. B. Rios-Alvarado, I. Lopez-Arevalo, and V. Sosa-Sosa. A Taxonomy Construction Approach Supported by Web Content. *Distributed Computing and Artificial Intelligence. Springer Berlin Heidelberg*, pages 461–468, 2012.
- [59] S. Roller, K. Erk, and G. Boleda. Inclusive yet selective: Supervised distributional hypernymy detection. *Proceedings of the COLING conference*, pages 1025–1036, 2014.
- [60] M. Sabou, C. Wroe, C. Goble, and G. Mishne. Learning domain ontologies for web service descriptions: an experiment in bioinformatics. *Proceedings of the 14th international conference on World Wide Web*, pages 190–198, 2005.

- [61] D. Sanchez and A. Moreno. Creating ontologies from Web documents. *Recent Advances in Artificial Intelligence Research and Development*, 113:11–18, 2004.
- [62] E. Santus, A. Lenci, Q. Lu, and S. S. Im Walde. Chasing hypernyms in vector spaces with entropy. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 38–42, 2014.
- [63] J. Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 4(2):234–242, 1992.
- [64] M. D. Smucker and C. L. Clarke. Time-based calibration of effectiveness measures. *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104, 2012.
- [65] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*, 2004.
- [66] R. Snow, D. Jurafsky, and A. Y. Ng. Semantic Taxonomy Induction from Heterogenous Evidence. *Proceedings of the 21st International Conference on Computational Linguistics*, pages 801–808, 2006.
- [67] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. Parsing with compositional vector grammars. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 932–937, 2013.
- [68] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 1631–1642, 2013.
- [69] L. Tan, R. Gupta, and J. van Genabith. Usaar-wlv: Hypernym generation with deep neural nets. *Proceedings of the SemEval*, pages 932–937, 2015.
- [70] L. A. Tuan, J. J. Kim, and S. K. Ng. Taxonomy Construction using Syntactic Contextual Evidence. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 810–819, 2014.
- [71] P. Velardi, S. Faralli, and R. Navigli. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707, 2012.
- [72] P. Velardi, S. Faralli, and R. Navigli. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707, 2013.

- [73] J. Weeds, D. Clarke, J. Reffin, D. J. Weir, and B. Keller. Learning to distinguish hypernyms and co-hyponyms. *Proceedings of the 24th International Conference on Computational Linguistics*, pages 2249–2259, 2014.
- [74] J. Weeds, D. Weir, and D. McCarthy. Characterising measures of lexical distributional similarity. *Proceedings of the 20th International Conference on Computational Linguistics*, pages 1015–1025, 2004.
- [75] W. Wentao, L. Hongsong, W. Haixun, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 481–492, 2012.
- [76] D. Widdows and B. Dorow. A graph model for unsupervised lexical acquisition. *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7, 2002.
- [77] W. Wong, W. Liu, and M. Bennamoun. Tree-traversing ant algorithm for term clustering based on featureless similarities. *Data Mining and Knowledge Discovery*, 15(3):349–381, 2007.
- [78] F. Xu, D. Kurz, J. Piskorski, and S. Schmeier. A Domain Adaptive Approach to Automatic Acquisition of Domain Relevant Terms and their Relations with Bootstrapping. *Proceedings of the 3rd International Conference on Language Resources an Evaluation*, 2002.
- [79] J. Yao, B. Cui, G. Cong, and Y. Huang. Evolutionary taxonomy construction from dynamic tag space. *World Wide Web*, 15(5):581–602, 2012.
- [80] Z. Yu, H. Wang, X. Lin, and M. Wang. Learning term embeddings for hypernymy identification. *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 1390–1397, 2015.
- [81] M. Zhang, J. Su, D. Wang, G. Zhou, and C. L. Tan. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. *International Conference on Natural Language Processing*, pages 378–389, 2005.
- [82] X. Zhu, Z. Y. Ming, and T.-S. Chua. Topic hierarchy construction for the organization of multi-source user generated contents. *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 233–242, 2013.
- [83] W. Y. Zou, R. Socher, D. M. Cer, and C. D. Manning. Bilingual word embeddings for phrase-based machine translation. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 1393–1398, 2013.