Universität des Saarlandes
Max-Planck-Institut für Informatik
AG5

Deriving a Web-Scale Common Sense Fact Knowledge Base

Masterarbeit im Fach Informatik
Master's Thesis in Computer Science
von / by

Niket Tandon

angefertigt unter der Leitung / supervised by

Prof. Dr. Gerhard Weikum

betreut von / advised by

Dr. Gerard de Melo

begutachtet von / reviewers

Prof. Dr. Gerhard Weikum

Dr. Martin Theobald

August 2011

Hilfsmittelerklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben.

Non-plagiarism Statement

I hereby confirm that this thesis is my own work and that I have documented all sources used.

Saarbrücken, 29. August 2011,

(Niket Tandon)

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlichtwird.

Declaration of Consent

Herewith I agree that my thesis will be made available through the library of the Computer Science Department.

Saarbrücken, den 29. August 2011,

(Niket Tandon)

Acknowledgements

First and foremost, I owe this thesis to my spiritual mentors, my Babuji and my Bua. Life without them is unimaginable. My loving appreciation goes to my family especially my parents and my wife, Anjali for their unconditional love and encouragement all along. I derived a lot of confidence from my Fufaji, whose wonderful mentorship, is thanked.

I am very thankful to my advisor, Gerard de Melo, for several insightful technical conversations during the course of my M.S. Most importantly, he taught me with silent actions to do well by painstaking research and perfection. I consider myself exceedingly fortunate to have worked with him.

I would like to express my sincere gratitude to my supervisor, Prof. Dr. Gerhard Weikum, for giving me the opportunity to complete my thesis successfully. He taught me to step back, zoom-out in order to have a clearer vision of possibilities. I would consider my career a success if I can emulate during my entire career half of the flair, elegance, simplicity, vision, and enthusiasm he has displayed in the course of my M.S.

My M.S. was generously supported by the International Max Planck Research School for Computer Science. This offered me the rare opportunity for unfettered exploration.

I want to thank Laura Dietz and Ivan Titov for their excellent discussions on graphical models. Final words of appreciation are for Tuan, Tomasz, Adrian, Yagiz, Sairam, Christina, Mohamed, Andrey and Erdal for their invaluable friendship and for providing a stimulating work environment in MPI.

Abstract

The fact that birds have feathers and ice is cold seems trivially true. Yet, most machine-readable sources of knowledge either lack such common sense facts entirely or have only limited coverage. Prior work on automated knowledge base construction has largely focused on relations between named entities and on taxonomic knowledge, while disregarding common sense properties. Extracting such structured data from text is challenging, especially due to the scarcity of explicitly expressed knowledge. Even when relying on large document collections, pattern-based information extraction approaches typically discover insufficient amounts of information.

This thesis investigates harvesting massive amounts of common sense knowledge using the textual knowledge of the entire Web, yet staying away from the massive engineering efforts in procuring such a large corpus as a Web. Despite the advancements in knowledge harvesting, we observed that the state of the art methods were limited in terms of accuracy and discovered insufficient amounts of information under our desired setting.

This thesis shows how to gather large amounts of common sense facts from Web N-gram data, using seeds from the existing knowledge bases like ConceptNet. Our novel contributions include scalable methods for tapping onto Web-scale data and a new scoring model to determine which patterns and facts are most reliable.

An extensive experimental evaluation is provided for three different binary relations, comparing different sources of n-gram data as well as different algorithms. The experimental results show that this approach extends ConceptNet by many orders of magnitude (more than 200-fold) at comparable levels of precision.

Contents

Acknowledgements **v**

Abstract **vii**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Previous Work . . . 3
1.2.1 Our Approach . . . 4
1.3 Contribution . . . 6
1.4 Outline . . . 7

2 Related Work **9**
2.1 Background: Extracting Information from Data . . . 9
2.1.1 Information Extraction . . . 9
2.1.2 Knowledge Harvesting . . . 11
2.1.3 Pattern-Based IE . . . 11
2.2 Related Work . . . 16
2.2.1 Information Extraction . . . 16
2.2.2 Web-Scale Extraction . . . 17
2.2.3 Common Sense Knowledge Acquisition . . . 18
2.3 Summary . . . 18

3 Model **21**
3.1 Relying on N-Gram Statistics . . . 21
3.2 Our approach: Pattern-Based IE Model . . . 23
3.2.1 Overview of the System . . . 24
3.3 Seed Selection . . . 26
3.4 Pattern Induction . . . 28
3.5 Pattern Scoring . . . 29
3.5.1 Supervised Pattern Scoring . . . 30
3.5.2 Statistics Based Pattern Scoring . . . 31
3.6 Tuple Ranking . . . 33
3.6.1 Unsupervised Tuple Ranking . . . 34
3.6.2 Supervised Tuple Ranking: Method-1 . . . 34
3.6.3 Supervised Tuple Ranking: Method-2 . . . 36
3.6.4 Supervised Tuple Ranking: Method-3 . . . 36

3.7 Summary . . . 37

4 Experimental Setup **39**
4.1 Dataset . . . 39
4.2 Relations . . . 42
4.3 System Architecture . . . 43
4.3.1 Distributed Framework of Execution . . . 45
4.4 Evaluation Metric . . . 46

5 Evaluation Results **47**
5.1 Supervised Pattern Induction, Supervised Tuple Ranking: (approach-1) . . . 48
5.1.1 Supervised Pattern Induction . . . 48
5.1.2 Supervised Tuple Ranking . . . 49
5.1.3 Comparison with other resources . . . 51
5.1.4 Summary . . . 52
5.2 PMI Based Pattern Induction, Supervised Tuple Ranking: (approach-2) . . . 53
5.2.1 PMI Based Pattern Induction . . . 53
5.2.2 Labeled Data . . . 55
5.2.3 Accuracy and Coverage . . . 55
5.2.4 Detailed Analysis . . . 57
5.2.5 Summary . . . 58
5.3 A New Method Based Pattern Induction, Supervised Tuple Ranking: (approach-3) 61
5.3.1 Extraction . . . 61
5.3.2 Accuracy and Coverage . . . 61
5.3.3 Summary . . . 62
5.4 Conclusion of Evaluation . . . 63

6 Conclusions **67**
6.1 Conclusion . . . 67
6.2 Future Work . . . 68

List of Figures **68**

List of Tables **71**

Bibliography **73**

Chapter 1

Introduction

1.1 Motivation

Knowledge Acquisition For several decades, the *knowledge acquisition bottleneck* has been a major impediment to the development of intelligent systems. Knowledge acquisition involves the process of capturing knowledge and representing the knowledge in a machine consumable structure. It is a long-standing dream of Artificial Intelligence (AI) to make the world knowledge consumable by the machine. There have been several research efforts in the direction of constructing knowledge bases acquiring different kinds of knowledge. There are two broad goals of such knowledge bases, 1) encyclopedic and expert knowledge, 2) common sense knowledge.

Knowledge bases that contain encyclopedic knowledge are able to represent knowledge like "*New Delhi is the capital of India*", "*Mount Everest is 8,840 meters high*". Such knowledge bases form the backbone of question answering systems like Watson [Ferrucci et al., 2010]. These knowledge bases could potentially benefit the search engines because search engine users generally search for information, not for documents. In recent years, there has been a growing trend to go beyond standard keyword search over document collections by recognizing words and entity names and offering additional structured results when available [Paparizos et al., 2009]. For example, for a query like "*Pablo Picasso artwork*", the Bing search engine displays a list of Picasso paintings delivered by Freebase[1], a large database of structured data. Google has experimented with explicit answers to queries like "*capital of Germany*" or "*prime minister of India*". Additionally, structured data is useful for query expansion [Gong et al., 2005], semantic search [Milne et al., 2007], and faceted search [Bast et al., 2007], among other things. However, such fact knowledge bases are scarce despite the large amount of encyclopedic knowledge available in the form of text on the Web.

[1] `http://www.freebase.com/`

Common Sense Knowledge Acquisition The second types of knowledge bases are common sense knowledge bases. These knowledge bases contain knowledge like "*Flowers are soft*", "*the sky is blue*". Facts of this sort seem trivially true. Yet, knowledge that humans take for granted on a daily basis is not readily available in computational systems. The automatic construction of such common sense knowledge bases poses a bigger challenge than the encyclopedic knowledge base construction because humans rarely express the obvious. There is rarely a need to explicitly mention common sense facts to others e.g. we do not explicitly express in text that "*carrots are edible*". These are assumed to be trivial to state. Thus, it is challenging to automatically find common sense knowledge in text.

However, if such common sense knowledge was more easily accessible, applications could behave more in line with users' expectations. For example, a mobile device could recommend nearby coffee shops rather than ice cream vendors when users desire warm beverages because it would know that the property of a coffee is a warm beverage. A search engine would be able to suggest local supermarkets when a user wishes to buy soap, because it would know that soaps are sold in local supermarkets. Further applications include query expansion [Hsu et al., 2006], video annotation [Altadmri and Ahmed, 2009], faceted search [Bast et al., 2007], and distance learning [Anacleto et al., 2006], among other things. [Liu and Singh, 2004] provide a survey of applications that have made use of explicit common sense facts. Thus, common sense knowledge acquisition is an important problem.

Goal of the Thesis This thesis is driven with the goal of constructing a large-scale common sense knowledge base. Such a resource would pave the way for an entire ecosystem of novel intelligent applications. Figure 1.1 presents an example of the vision of this thesis.

It is important to ascertain that common sense knowledge can indeed be extracted from free text even when resorting to a large corpus. Fortunately, [Yu and Chen, 2010] corroborate that we can extract substantial amount of common sense facts from text by showing that more than 40% of common sense knowledge assertions are explicitly stated common sense knowledge in text. The remaining 60% of the common sense knowledge is too obscure to be stated. In general, such common sense knowledge may also be taken to encompass procedural knowledge or "*know-how*", e.g. knowing how to prepare a pasta meal. However, such knowledge is beyond the scope of this paper. We also disregard complex axiomatic rules like those given in formal knowledge bases like Cyc [Lenat, 1995], which presently for the most part are still considered too challenging to learn in an automated manner.

The scope of this thesis is to target the nearly 40% common sense knowledge assertions possible to find in text. However, we believe that by resorting to an extremely large text corpus, we can find several millions of common sense assertions and beat the number 40% by some margin.

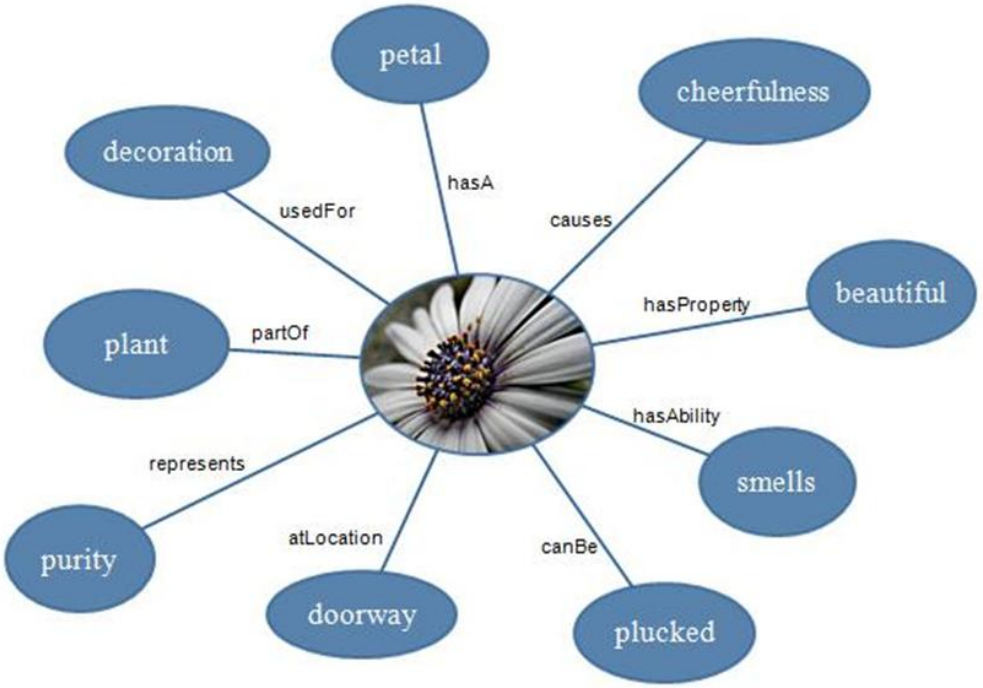

FIGURE 1.1: Some desirable common sense knowledge about a flower

The desiderata of such a common sense knowledge base that we envision is simple automatic construction, machine-comprehensible representation, extendible, accurate and high coverage. Unfortunately, previous approaches to construction of common sense knowledge bases have been restrictive in: 1) either scale (low coverage) or 2) involve human effort that is time-consuming and expensive. Therefore, the challenge involved in accomplishing the goals of this thesis is to automatically construct a robust and large-scale common sense knowledge base, overcoming the challenges of the previous approaches.

1.2 Previous Work

Previous work to formalize our common sense understanding of the world has largely been centered on

a) manual efforts, e.g. Cyc [Lenat, 1995], SUMO [Niles and Pease, 2001], and WordNet [Fellbaum, 1998], as well as resources like ConceptNet [Havasi et al., 2007] that rely on crowd-sourcing,

b) minimally supervised information extraction from text [Etzioni et al., 2005, Suchanek et al., 2009, Carlson et al., 2010].

Property	Desired	YAGO	Cyc	ConceptNet	Wordnet	TextRunner
Accuracy	high	high	high	high	high	low
Coverage	Web-scale	high	high	low	low	high
Ranking of Facts	scored	yes	no	no[1]	no	no[1]
Speed Of Collection	high	high	low[2]	low[2]	low[2]	high
Domain Extendibility	extendible	yes	yes	no	no	yes
Human Involvement	none	low	high	high	high	none
Consumable Facts	high	high	high	high	high	low
Type of facts	CSK	not-CSK	CSK	CSK	not-CSK	not-CSK

TABLE 1.1: Desired properties
(1: simple counting, 2: manual)

Both strategies are limited in scope or coverage. Human efforts are time-consuming and often fail to attract sufficient numbers of contributors.

Information extraction methods have been successful for taxonomic knowledge (`IsA` and `InstanceOf` relationships among classes and between entities and classes), and for relations between named entities (e.g. birthplaces of people). Most extraction systems rely on pattern matching, e.g. a string like "*... cities such as Paris ...*" matching the "<*X*> *such as* <*Y*>" pattern for the `IsA` relation leads to knowledge of the form `IsA(Paris,City)`. Previous work [Hearst, 1992] has shown that textual patterns can be surprisingly reliable but are generally very rare. For instance, in a 20 million-word New York Times article collection, Hearst found only 46 facts. This paucity has been an important bottleneck with respect to the goal of establishing large-scale repositories of structured data.

Most information extraction systems have to date only been evaluated on small corpora, typically consisting of a couple of thousand [Suchanek et al., 2009] or perhaps a couple of hundred thousand documents [Agichtein and Gravano, 2000]. Previous studies [Cafarella et al., 2005] have found that much greater precision and recall is possible by turning to search engines to benefit from Web-scale document collections. This, however, is shown to slow down the extraction process by several orders of magnitude. Thus, the previous approaches are limited in scale and coverage or scope.

Table 1.1 shows that the existing state-of-the-art knowledge bases do not satisfy all of the desired properties of our ideal common sense knowledge (CSK) Harvesting system. Therefore, there is clearly a need for research on how to meet these requirements.

1.2.1 Our Approach

In this thesis, we consider an alternative strategy to tackle the problem of rarity of facts by turning to a very large corpus which is "*representative*" of the entire Web. Instead of operating on specific document collections, we turn to n-gram statistics computed from large fractions of the

Web, consisting of giga-scale numbers of documents. An n-gram is a sequence of n consecutive items in text. While character n-grams, i.e. sequences of consecutive characters, are useful for tasks like language identification and data compression, word n-grams, i.e. sequences of consecutive word tokens, are useful for a wide range of tasks like statistical machine translation [Brants et al., 2007], spelling correction [Islam and Inkpen, 2009], word breaking to split "*baseratesoughtto*" into the most likely "*base rates ought to*" [Wang et al., 2010] among other applications. An n-gram dataset is a resource that, for a given sequence of n words, lists the corresponding occurrence frequency or probability of that string in a large document collection. Some of the available n-gram datasets [Wang et al., 2010] are computed from petabytes of the documents, yet such n-gram statistics have not been used in information extraction systems in previous work. Thus, we address the first problem of a large-scale using a Web scale n-gram data.

We employ a pattern based information extraction approach for automatic and robust knowledge extraction. We start with a small set of facts (seeds) and apply these facts on the n-gram text. This produces a pattern like "*<X> is always <Y>*" that would discover the tuple `(grass,green)` in the n-gram (5-gram) phrase "*the grass is always green*". Further, we use linguistic annotation of the patterns in order to make the patterns more robust. Some common patterns for a few common sense relations are listed in Table 1.2. Finally, these patterns are applied to the n-gram data to yield a very large number of facts and the facts are ranked under supervision. However, there are several challenges that come up in this approach. We mention the challenges and then present how we overcome the challenges.

Relation Name	Example Patterns
hasProperty	NP is ADJ
isA	NP is a kind of NP
partOf	NP is part of NP
hasAbility	NP can VP
canBe	NP can be VP
atLocation	NP is found in NP
usedFor	NP is used for VP
causes	NP causes VP

TABLE 1.2: Sample CSK Relations

Challenges The first challenge was the requirement of a very large corpus given that humans rarely express the obvious. We already addressed this challenge by resorting to Web n- grams. However, extracting information from text is non-trivial and several challenges come up in our approach.

- The notorious variation of free text makes it difficult to specify significant numbers of textual patterns in advance. In addition, parsers are far from being perfect.

- More abstract linguistic patterns, relying for instance on dependency parsing, are able to overcome the first challenge, however it is computationally prohibitive to linguistically annotate substantial amounts of data.
- Additionally, as in our approach, it is vital to score the patterns reliably. This has multiple reasons:
  - Unlike previous approaches that relied on around very few seeds, we make use of the fact that significant efforts have already been made to collect instances of the most important common sense relations. We thus rely on a larger number of seeds taken from a knowledge base, ConceptNet [Havasi et al., 2007]. This is an advantage, but it also means that we find many more patterns than in previous studies.
  - Relying on a Web-scale n-gram dataset, we are faced with an enormous number of seed occurrences, and hence very large numbers of potential patterns.
  - Applying *all* of these candidate patterns for extraction would lead to tremendous scalability and noise challenges. Many patterns just coincidentally match certain seeds. For instance, a pattern like "<*X*> *and* <*Y*>" is very likely to match a number of seeds in ConceptNet. However, it is obvious that it is not a very reliable pattern. In an n-gram dataset derived from petabytes of data, the pattern will match many millions of n-grams but only few of the matches will correspond to any particular semantic relationship.

In order to design a robust knowledge base, we must deal with the several challenges just mentioned. Relying on a Web-scale n-gram dataset, we are faced with an enormous number of seed occurrences, and hence very large numbers of potential patterns. Applying *all* of these candidate patterns for extraction would lead to tremendous scalability and noise challenges. Many patterns just coincidentally match certain seeds. For instance, a pattern like "<*X*> *and* <*Y*>" is very likely to match a number of seeds. However, it is obvious that it is not a very reliable pattern. Therefore, unlike standard bootstrapping approaches, we rely on novel scoring functions to very carefully determine which patterns are likely to lead to good extractions. Unlike previous unsupervised outputs, we rely on a semi-supervised approach for scoring the output facts. The model is obtained from the input data, without any need for additional manual labeling. Thus, we simultaneously address both the problems of a large-scale and robustness.

1.3 Contribution

This thesis explores how large numbers of common sense properties like `CapableOf(dog,bark)`, `PartOf(room,house)` can be harvested automatically from the Web. A new strategy is proposed to overcome the robustness and scalability challenges of previous work.

- Rather than starting out with minimal numbers of seeds, we exploit information from an existing fact database, ConceptNet [Havasi et al., 2007].
- Rather than using a text corpus, we rely on a Web-scale n-gram dataset, which gives us a synopsis of a significant fraction of *all* text found on the Web. While people rarely explicitly express the obvious, we believe that "*a word is characterized by the company it keeps*" [Firth, 1957] and exploit the very large quantities of natural language text that are now available on the Web.
- Unlike standard bootstrapping approaches, we rely on novel scoring functions to very carefully determine which patterns are likely to lead to good extractions.
- Unlike previous unsupervised outputs, we rely on a semi-supervised approach for scoring the output facts. The model is obtained from the input data, without any need for additional manual labeling.

Some results of this thesis have been published in the proceedings of international conferences and in workshops, including:

- AAAI 2011 [Tandon et al., 2011]
- SIGIR 2010 N-gram Workshop [Tandon and de Melo, 2010]

1.4 Outline

The organization of this thesis is as follows. Chapter 2 begins by introducing background on information extraction and common sense knowledge extraction. Chapter 3 describes the pattern based information extraction model used for information extraction, in particular common sense knowledge extraction using n-grams with supervised and unsupervised approaches for pattern and tuple scoring. Chapter 4 provides details about the experimental setup, followed by experimental results in Chapter 5. Finally, chapter 6 concludes with discussion, applications and future directions of our research.

Chapter 2

Related Work

2.1 Background: Extracting Information from Data

2.1.1 Information Extraction

Information Extraction (IE) is the process of extracting relevant information from potentially large volumes of data. For example, from the text, `Berlin is the capital of Germany`, an IE system can extract a relation instance `isCapitalOf(Berlin, Germany)`. The data can be available in one of two structures or forms. These forms are unstructured and semi-structured. In this section, we start with a definition of IE, followed by a review of the different data forms. Following the definitions, we review some state-of-the-art IE systems. It is essential to discuss the IE systems with respect to the form of data, size of data, and the desired output.

Let us begin by understanding the word 'Information' in the context of IE. According to Wikipedia, information in its most restricted technical sense is an ordered sequence of symbols that record or transmit a message. Information is represented in several formats like textual or multimedia. However, we shall limit our discussion to textual IE. IE is the task of automatically translating a collection of input documents into structured data that is consumable by a machine. Unlike information retrieval(IR) where we identify relevant documents from a document collection, IE produces structured data that is consumable by scores of applications of Web mining and searching tools.

The foundations of IE can be found in the Natural Language Processing (NLP) community that was concerned with the task of recognizing named entities like names of people and organization from news articles in the Message Understanding Conference (MUC) and Automatic Content Extraction (ACE) competitions. With the plethora of accessible information, IE became an interesting and challenging research problem. With the consequent growth of the IE community,

researchers from various fields like machine learning, databases, IR, and NLP borrowed the techniques in these fields for various aspects of the information extraction problems.

We start with IE systems that extract information from semi-structured data such as web pages automatically generated with a schema in the background or web tables. Such documents follow a definite template because these are machine-generated pages. A popular source in this space is HTML documents dynamically generated via database-backed sites. The state-of-the-art in this field is Wrapper Induction [Kushmerick et al., 1997]. Information wrapping develops techniques that mainly make use of some structure like HTML tags in web pages in order to learn the underlying schema of the page. The schema that is learned is known as a wrapper. The wrappers can be used to extract a large amount of information from other semi-structured pages. In essence, wrappers are regular expressions that are typical of rule based IE systems. However, wrapper induction technique is not relevant in our setting. Wrapper induction techniques exploit the regularity of the document structure e.g. table layouts for tables automatically generated with a schema. The focus of this thesis is on free text (unstructured data), therefore we focus on information extraction from unstructured data.

Typical input data for an IE system are unstructured documents that contain free, natural language texts. Preprocessing and data cleaning is an important and challenging task for these systems. Nonetheless, the sheer amount of data available in unstructured format makes these IE approaches significant. Information extraction from unstructured documents has been a topic of interest in the research community. While unstructured data is found in plenty, it is a challenge to pre-process the free form text due to noise. This has led to several statistical NLP based approaches like pattern-based IE, in which, a pattern template is applied on data to extract information. For instance, the pattern "*<X> and other <Y>*" has been found to work well for the `isA` relation. A matching word sequence like "*... Stanford and other universities ...*" allows us to induce knowledge of the form `isA(Stanford,university)`.

The major challenge in IE systems deals with the tradeoff between accuracy of the extracted information, and scale of the extracted data. In order to provide a balance, we look for more data. People have resolved to the Web due to the enormous amount of data present on the Web. However, more data brings in more noise. In order to combat this problem, there has been growing interest in IE systems that leverage on structured and unstructured data simultaneously. These systems initialize with structured data because it is more accurate and relatively less prone to noise and subsequently extract information from unstructured pages. We adopt a similar strategy in this thesis of starting with a relatively clean structured database (ConceptNet) and then scaling up using the Web data (n-gram data).

2.1.2 Knowledge Harvesting

For a human, the richest knowledge base of the world is the Web. However, machines do not understand the knowledge as humans because the massive and valuable knowledge on the Web is a potpourri of noisy and clean, low quality and high quality, unstructured and structured text and media. According to [Weikum and Theobald, 2010], the proliferation of knowledge-sharing communities like Wikipedia and the advances in automated information extraction from Web pages open up an unprecedented opportunity: can we systematically harvest facts from the Web and compile them into a comprehensive machine-readable knowledge base about the worlds entities, their semantic properties, and their relationships with each other. The methods for harvesting relational harvesting broadly follow two different paradigms, rule based systems similar to wrapper induction and pattern-based techniques. The manually defined rule based systems suffer from the inherent problem that the rules are brittle. Although, the rules are generally accurate, they may fail if the structure of the data varies. In such cases, pattern-based techniques come to rescue because these techniques have a strong foundation in NLP and statistics.

The work in this thesis closely draws upon the pattern-based approach. In the following section, we discuss pattern-based techniques drawing on NLP techniques and statistics in detail. In order to understand the development of pattern-based IE systems, it is important to follow the incremental development of methods, starting with early pattern-based extraction systems [Agichtein and Gravano, 2000] [Hearst, 1992] [Brin, 1999].

2.1.3 Pattern-Based IE

Since much of the Web consists of unstructured text, *information extraction* techniques have been developed to harvest machine-readable, structured information from textual data. Often, these techniques rely on pattern matching with textual templates. As an example, the pattern "<*X*> *such as* <*Y*>" has been found to work well for the "`isA`" relation: A matching word sequence like "... *cities such as Paris* ..." allows us to induce knowledge of the form `isA(Paris,City)`.

The idea of searching for occurrences of specific textual patterns in text to extract information has a long history. Patterns for finding `isA` relationships were first discussed by Lyons [Lyons, 1977] and Cruse [Cruse, 1986]. Hearst [Hearst, 1992] empirically evaluated the reliability of such patterns on corpora, and proposed a method for finding them automatically using seed instances. Brin [Brin, 1999] presented a technique that exploits the duality between sets of patterns and relations to grow the target relation starting from a small sample. Further, [Agichtein and Gravano, 2000] cleaned up the approach of [Brin, 1999] with techniques

for evaluating the quality of the patterns and tuples generated at each step of the extraction process. In the following sections, we discuss these approaches in detail.

2.1.3.1 Hearst

Hearst [Hearst, 1992] describes a method for the acquisition of the hyponymy lexical relation from unrestricted text. An example of a hyponymy relation is `Paris, city`. The method identifies a set of lexico-syntactic patterns that are easily recognizable and occur frequently. The underlying technique of this approach is extendible and can be used beyond hyponymy.

For example, for the `isA` relation, one can automatically determine instances from noun phrases around a syntactic pattern like `NP such as NP` , where NP is the POS tag for proper nouns. This pattern can be used for extracting instances of hyponymy relation. Since one of the goals of building a lexical hierarchy automatically is to aid in the construction of a natural language processing program, this approach to acquisition is preferable to one that needs a complex parser and knowledge base. The trade off is that the reformation acquired is coarse-grained.

The paper provides an example of hyponym relation as `(broken bone, injury)` which indicates that the term `broken bone` can be understood at some level as an `injury` without having to determine the correct senses of the component words `broken bone` and `injury`. Note also that a term like `broken bone` is not likely to appear in a dictionary or lexicon although it is common.

Thus, the observation is that terms that occur in a list are often related semantically, whether they occur in a hyponymy relation or not. In order to find new patterns automatically, Hearst sketched the following procedure. First, we identify the lexical relation R whose instances we wish to extract. Our example considers R as the hyponym relation. The second step is to gather a list of term pairs (seeds) for which this relation holds. For example, `(Paris,city)`'. This list can be found automatically by bootstrapping from patterns found by hand, or by bootstrapping from an existing lexicon or knowledge base. The next step is to find occurrences in corpus where these expressions occur syntactically near one another and record the environment. Now, one finds commonalties among these environments and hypothesizes that common environments yield patterns that indicate R. This gives rise to new patterns and it is used to extract more instances of R. This process is repeated until one obtains the right number of extractions as needed.

There are both strengths and limitations of Hearst's approach. While this was a novel procedure for information extraction and paved the way for several research efforts, the technique is limited by low coverage. The approach yields cleaner patterns but these patterns are insufficient to extract a large number of tuples. Subsequently, another approach on similar lines attempted to overcome the problems of this approach.

2.1.3.2 DIPRE

Brin [Brin, 1999] introduced the following observation on the duality of facts and patterns: if one knows enough facts for a relation then it is possible to automatically find textual patterns and retain the best patterns, because some patterns are just noise. With good patterns, one can automatically find more facts in an iterative fashion, see Figure 2.1. Their method called, Dual Iterative Pattern Relation Extraction - DIPRE is a technique for extracting relations that makes use of pattern-relation duality.

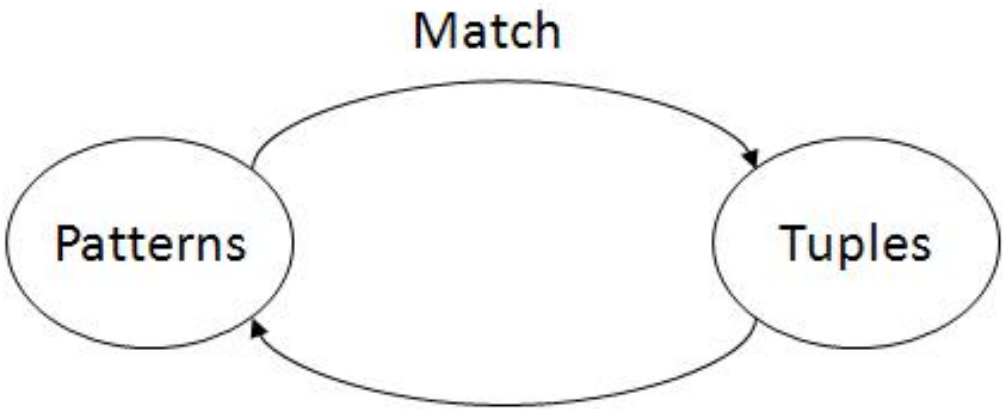

FIGURE 2.1: The duality that DIPRE harnesses

Comparison with Hearst's approach: Other than automating Hearst's approach, DIPRE's approach focused on duality of good tuples leading to good patterns and good patterns leading to good tuples. DIPRE starts with a similar approach however, while searching for patterns in text, only nearby occurrences of the seed pair was considered. In addition, the context of every occurrence (URL and surrounding text) is kept. In order to generate patterns based on the occurrence environment recorded, one must look for similar context. The patterns need to have a low error rate and thereby ensure that they are not over generic. The higher the coverage of the patterns, the better the accuracy of the system. On the other hand, highly specific patterns may lead to a low coverage but this can be compensated with a larger database. Once such specific patterns are obtained, the method proceeds to search the database for tuples matching any of the patterns. These steps are repeated until one find sufficient number of instances of the relation of interest.

One needs to combat the problem of a bad pattern leading to bad tuples and thus reducing the accuracy the subsequent iterations. The approach taken by DIPRE is to require the tuples to match multiple patterns. To this end, DIPRE introduces pattern specificity. The specificity of a pattern P roughly corresponds to $log(P(X \in \text{matching context of pattern}))$ where X is some random variable distributed uniformly over the domain of tuples of $R * R$. One must reject any pattern with too low specificity so that overly general patterns are not generated. Therefore, only patterns that satisfy a certain threshold count of tuple matches are kept.

While DIPRE is powerful, it is difficult to tune (regarding thresholds, weighting parameters) and susceptible to drifting away from the target. While high recall is easily possible, the precision would often be unacceptable. This led to Snowball [Agichtein, 2005].

2.1.3.3 Snowball

The Snowball system [Agichtein, 2005] extended DIPRE with a statistical approach to compute confidence of patterns and tuples. The overall flow of the system is very similar in spirit to DIPRE, as presented in 2.2.

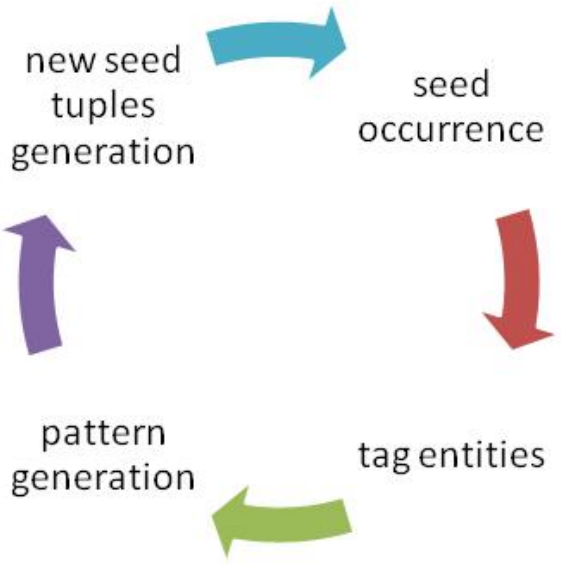

FIGURE 2.2: Snowball flow diagram

A Snowball pattern is a 5-tuple(left, tag1, middle,tag2, right), where tag1 and tag2 are named-entity tags, and left, middle, and right are vectors associating weights with terms. The weight of a term in each vector is a function of the frequency of the term in the corresponding context. These vectors are scaled so their norm is one. Finally, this is multiplied by a scaling factor to indicate each vectors relative importance. Their experiments with English-language documents found that the middle context is the most indicative of the relationship. Hence, they will typically assign the terms in the middle vector higher weights than the left and right vectors.

$$Match(Tuplet_P, Tuplet_S) = l_P l_S + m_P m_S + r_P r_S$$

In order to generate a pattern, Snowball groups occurrences of known tuples in documents, if the contexts surrounding the tuples are ”similar enough”. Snowball generates a 5-tuple for each string where a seed tuple occurs, and then clusters these 5-tuples using a simple single-pass clustering algorithm, using the Match function defined above to compute the similarity between

the vectors and some minimum similarity threshold τ^{sim}. The left vectors in the 5-tuples of clusters are represented by a centroid $\bar{l}_s$. Similarly, collapse the middle and right vectors into $\bar{m}_s$ and $\bar{r}_s$, respectively. The meaning of the centroid is intersection. These three centroids, together with the original tags (which are the same for all the 5-tuples in the cluster), form a Snowball pattern ($\bar{l}_s$,t_1, $\bar{m}_s$, t_2,$\bar{r}_s$).

Clustering is beneficial because all the over specific patterns are removed. It gives cleaner patterns more confidently. The basis of comparison is the context of the sentence so they get tuples in the same relation together. This helps in increasing efficiency and gives better results because they get clearer patterns, e.g., if they have ten patterns evaluated one by one as opposed to clustering, then they would get many rare and not useful patterns but clustering avoids such very specific patterns. In a way, this also helps in increasing recall if they consider a certain number of patterns.

The confidence of a pattern P is:

$$Conf(P) = \frac{P.positive}{(P.positive + P.negative)}$$

where $P.positive$ is the number of positive matches for P and $P.negative$ is the number of negative matches.

The confidence of tuple is measured in such a way that requires matching many patterns. The confidence of a candidate tuple T is:

$$Conf(T) = 1 - \prod_{i=1}^{|P|} (1 - (Conf(P_i).Match(C_i, P_i)))$$

where $P = P_i$ is the set of patterns that generated T and C_i is the context associated with an occurrence of T that matched P_i with degree of match $Match(C_i, P_i)$. During the calculation of the pattern confidence, Snowball does not ignore the confidence values from previous iteration. To control the learning rate of the system, they set the new confidence of the pattern as:

$$Conf(P) = Conf_{new}(P)W_{updt} + Conf_{old}(P)(1 - W_{updt})$$

The Snowball system was a novel system that draw upon statistics and NLP. It remains to be one of the state of the art approaches in pattern-based IE.

2.2 Related Work

Having equipped ourselves with the technical background of pattern-based IE, we consider the work that comes close to our goal of constructing a knowledge base that contains machine-readable common sense knowledge. There are three broad categories of related work, IE, Web-scale IE, and, common sense knowledge acquisition.

2.2.1 Information Extraction

With the novel approach of Hearst [Hearst, 1992], a large range of approaches have built upon these ideas like DIPRE [Brin, 1999] and Snowball [Agichtein, 2005] as discussed. Others extend these ideas to other relationships like `partOf` [Girju et al., 2006] as well as factual knowledge like birth dates of people and capital cities of countries [Cafarella et al., 2005].

To overcome the sparsity of pattern matches, pattern-based iterative bootstrapping approaches attempt to re-use extraction results as seeds [Pantel and Pennacchiotti, 2006]. Unfortunately, the extraction quality often degrades very quickly after a few iterations. Our approach ensures that significant amounts of seeds and pattern matches are available in the first iteration, so additional rounds are not necessary.

Recent work [Suchanek et al., 2009] has used consistency constraints on fact hypotheses (e.g., among several birthplace candidates for a person, only one can be correct) to improve precision. However, these techniques are computationally much more expensive and it is an open issue if and how constraints could be formulated or learned for common sense properties. NELL [Carlson et al., 2010] relies on humans to filter the rules proposed by its rule learner.

Several projects studied using much more sophisticated techniques to extract very specific information from within single web pages or even smaller units of text. The goal was usually to extract information from the text as completely as possible. Some used deeper NLP techniques to discover more complex extraction rules [Riloff, 1993]. Other systems investigated segmentation algorithms like CRFs to partition citation references into individual parts (author names, title, year, etc.) [Sarawagi and Cohen, 2004]. Wrapper induction systems [Kushmerick et al., 1997] attempted to discover rules for template-based web sites, e.g. to extract pricing information from vendor product pages, as used by Google's Product Search.

2.2.2 Web-Scale Extraction

A different line of research focused instead on obtaining better results by scaling up the size of the document collection. The idea is to avoid making risky decisions for individual pieces of information, but instead exploit the redundancy of information in large document collections. Pantel et al. [Pantel et al., 2004] considered techniques to scale pattern-based approaches to larger corpora, however even these corpora represent only a very small fraction of the Web. Since Web-scale document collections are not easily obtainable, other approaches relied on web search engines to find instances [Etzioni et al., 2004a, Pantel and Pennacchiotti, 2006]. Approaches that directly rely on search engines interfaces face major challenges. Those that use search engines to discover new tuples will generally only retrieve top-k results for a limited k without being able to exploit the large amounts of facts that are in the long tail. For example, a query like "*such as*" cannot be used on its own to retrieve very large numbers of `isA` pairs. Those approaches that use search engines to derive more information for specific output tuples first need to obtain the set of candidates from some other source, usually a much smaller document collection or perhaps pre-existing lists of named entities or words, which is likely to be limited in coverage. Cafarella et al. [Cafarella et al., 2005] investigated the costs of relying on slow commercial search engine interfaces in detail. Their study shows how using a local document collection can speed up the time for extracting information from days to minutes. However, it also shows that this entails a significant loss of precision and recall.

This is why n-gram data is often a better alternative. Downey et al. [Downey et al., 2007] presented a post-processor that double-checks output tuples delivered by an information extraction system using n-gram language models. Their system checks if the arguments have similar contexts as seed arguments to ensure that they are of the correct type (e.g. city names). Additionally, they use general similarity of contexts to aid in assessing the overall validity of output pairs.

N-gram frequencies have also been used for other tasks. Previous work used online search engines to obtain string frequencies on-the-fly [Lapata and Keller, 2004] for tasks like context-sensitive spelling correction, ordering of prenominal adjectives, compound bracketing, compound interpretation, and countability detection. Pre-computed n-gram frequencies have several advantages, however, including much greater scalability and reproducibility of experimental results. N-gram datasets have been used for statistical machine translation [Brants et al., 2007], spelling correction [Islam and Inkpen, 2009], word breaking to split "*baseratesoughtto*" into the most likely "*base rates ought to*" [Wang et al., 2010], and search query segmentation to segment web search engine queries like "*mike siwek laywer mi*" into "*mike siwek*", "*lawyer*", "*mi*" [Huang et al., 2010]. In our study, the n-gram data is used to find patterns and extract structured information.

2.2.3 Common Sense Knowledge Acquisition

There have been several efforts to collect a large database of common-sense knowledge. Most previous approaches have relied on paid experts or unpaid volunteers [Lenat, 1995]. However, these approaches are limited e.g. after two decades, much less than five million facts were collected far from the estimated hundreds of millions that are required. Others, [von Ahn et al., 2006] emphasized on creating a game system to collect facts. These games are appealing to a large audience of people and are able to collect several million facts in few weeks. However, these approaches are limited by two factors. First, it is hard to aggregate the outputs from several users. Since the ground truth is unknown, the challenge is to know which outputs to trust or discard. In addition, it is required to merge the outputs. Second, with time, people tend to lose interest in the games [von Ahn, 2011]. Few attempts [Suh et al., 2006a] were made to use natural language processing tools to extract statements regularly from Wikipedia. However, these approaches are severely limited by the amount of facts collected.

Therefore, most previous work has relied on human-supplied information [von Ahn et al., 2006, Havasi et al., 2007, Speer et al., 2009a] or used a smaller corpus and acquired only few relations of common sense. Rather than explicitly soliciting contributions, we instead attempt to make use of the large amounts of information that humans have already implicitly revealed on the Web. [Matuszek et al., 2005] used Web search engines to extend Cyc. There have been studies on applying hard-coded rules to parse trees of text [Schubert, 2002, Clark and Harrison, 2009], achieving a precision of around 50-70%.

2.3 Summary

Having reviewed the necessary background and related-work relevant to this thesis, it is important to revisit the distinguishing work of this thesis from the related work. Let us consider the three bodies of work discussed in the related work.

- First, unlike the standard bootstrapping based IE approaches , we rely on novel scoring functions to very carefully determine which patterns are likely to lead to good extractions. Secondly, rather than starting out with minimal numbers of seeds, we exploit information from an existing fact database, ConceptNet [Havasi et al., 2007].
- In the Web-Scale IE paradigm, rather than using a text corpus that is very difficult to obtain on a Web-scale, we rely on a Web-scale n-gram dataset. We exploit the very large quantities of natural language text that are now available on the Web through n-grams.

- In the common sense knowledge acquisition paradigm, rather than relying on experts or unpaid volunteers, we automatically construct a knowledge that is nearly 200 times larger than the existing state-of-the-art common sense knowledge base.

Chapter 3

Model

3.1 Relying on N-Gram Statistics

An n-gram dataset f is a resource that accepts n-gram query strings $s = s_1 \cdots s_n$ consisting of n consecutive tokens, and returns scores $f(s)$ based on the occurrence frequency of that particular string of tokens in a large document collection. In the simplest case, $f(s)$ values will just be raw frequency counts, e.g. $f($"*major cities like London*"$)$ would yield the number of times the string "*major cities like London*" occurs in the document collection. Some n-gram resources instead provide probabilities based on smoothed language models [Wang et al., 2010]. Additionally, we also consider n-gram datasets supporting wildcards "*?*" or regular expressions like "*", in which case f returns a set of string-score tuples.

Some of the dangers of relying on Web n-grams include:

- Influence of spam and boilerplate text: Large portions of the Web consist of automatically generated text. This can be due to simple boilerplate text in templates, which is repeated on many pages, or due to malicious (spam) web sites, consisting of meaningless text, often replicated millions of times on the same server or an entire network of servers.
- Less control over the selection of input documents: With conventional extraction systems, one typically chooses a document collection likely to contain relevant information. N-gram data is usually not limited to a specific domain. Among other things, this may mean that issues like polysemy become more of a problem.
- Less context information: Word sense disambiguation and part-of-speech tagging are more difficult when considered out-of-context.

- Less linguistically deep information: Approaches that analyze entire documents can rely on sophisticated NLP techniques like pronoun resolution and dependency parsing to analyze sentences.

We focus on binary relationships between entities. N-gram statistics are usually limited to rather small n, more complex m-ary relations between more than $m > 2$ entities usually cannot be extracted very well, unless they can be broken down into multiple independent binary relationships. For example, the founding year and founding location of a company can be treated independently, but the fact that country "*<V> imported <W> million dollars worth of <X> from country <Y> in year <Z>*" could not be broken down in a similar manner.

The following additional conditions apply:

- Very short items of interest: The words or entities of interest that should fill the placeholders in patterns should be very short units, preferably just unigrams. For instance, it is possible to search for a relationship between "*Kyoto*" and "*Japan*" or between "*Mozart*" and "*1756*". However, it may not be possible to search for relationships between multiple people with their full names, or to find instances of the `hasMotto` relationship from YAGO [Suchanek et al., 2007], which connects an entity like `Connecticut` to a slogan like "*Full of Surprises*". Moving to such longer units requires long n-grams that are not captured in a normal n-gram dataset.

- Knowledge that is typically expressed using very short patterns: This approach will work only for relations that are often expressed using very short expressions. There are limitations when such knowledge is usually spread out over several sentences or expressed using longer patterns, e.g. "*<X> has an inflation rate of <Y>*". Even if only a small fraction of all theoretically possible n-grams actually occur in document collection, scaling n-gram datasets up to large n becomes unpractical very quickly due to the combinatorial explosion. There may be work-arounds like using a pre-classifier that acts as a filter and selects only those sentences for further indexing that are likely to contain useful knowledge, based on document topic, sentence interestingness (e.g. the presence of certain keywords in the sentence), etc.

The primary motivation for working with n-gram data is to provide information extraction systems with more input in order to obtain a better coverage as well as accuracy. A larger document collection as input not only means more output, but also output of higher quality. Due to the greater amount of redundancy, the system has more information to base its assessment on. This may also have implications on the model that the system is based on. Pantel et al. [Pantel et al., 2004] showed that scaling to larger text collections alone could allow a rather simple technique to outperform much more sophisticated algorithms.

Certainly, one could obtain similar effects by running traditional information extraction systems on corpora as large as the document collections used to compute the n-gram statistics. However, there are several reasons why one may not choose that option:

- Availability: The carefully edited, balanced corpora traditionally used in NLP are comparatively small, e.g. the British National Corpus (BNC) consists of 100M tokens, and even Web document collections like the GigaWord Corpus [Graff and Cieri, 2003] are several orders of magnitude smaller than the web-scale document collections that some of the available n-gram datasets are based on. For instance, Google's Web1T dataset is computed from a tera-word document collection. For most researchers, crawling the Web to obtain similar web-scale collections would be a slow and expensive process requiring sophisticated infrastructure (e.g. web link farm detection).
- Re-usability of N-Gram Data: N-gram statistics can be used for a variety of different purposes, apart from information extraction, including examples such as machine translation and spelling correction. Hence, there are many incentives to create such highly reusable n-gram data.

We rely on 5-grams to extract information for two reasons. First, 5-grams statistics also contain information about 3- and 4-grams. Second, our pattern length ranges from 1-3 words, however the n-gram text oftentimes contains punctuation marks, thus requiring longer context even when pattern length is one word. Thus, 5-grams are more reliable source of extraction for CSK.

3.2 Our approach: Pattern-Based IE Model

According to [Weikum and Theobald, 2010], the methods for harvesting relational facts pursue different paradigms: 1) rule-based with declarative querying as a key asset, 2) pattern-based drawing on NLP techniques and statistics, and 3) learning-based with joint inferencing by coupled learners and logical reasoning about hypotheses. The rule-based paradigm is best suited for semi-structured inputs such as Web tables as discussed in Chapter 2. The pattern-based paradigm is suited for natural-language inputs, and the learning/reasoning-oriented paradigm aims to combine the best of both worlds.

Having mentioned the approaches for harvesting relations facts, let us discuss which approach fits in our context of common sense knowledge extarction. Our input is natural language text, therefore, the rule-based approach does not fit the framework, as it is more suited to semi-structured text. Approach (3) requires reasoning rules that are difficult to procure for common sense knowledge unlike factual knowledge. However, reasoning can be applied to further clean

the output facts once the knowledge is extracted. The main objective of this thesis is to construct a common sense knowledge base, therefore, we employ pattern-based paradigm.

Our approach is to gather a set of patterns that allow us to identify candidate tuples. The n-gram frequency statistics for the candidate tuples occurring with specific patterns are then used to derive a vector representation for each candidate tuple. Based on a training set of labeled tuples, a learning algorithm finally determines which candidate tuples should be accepted. We give an overview of the overall system in the next subsection.

3.2.1 Overview of the System

Our system consists of three key modules:

- Seed processor and Pattern Inducer: Given an existing fact knowledge base having a mix accuracy of tuples, the Seed processor inspects seeds that can lead to potentially good patterns. These seeds are collected in tunes of several hundred for every relation described in the previous section. The seeds are used to induce more patterns. The patterns are constructed using text matching approach as well with linguistic constraints.
- Pattern ranker: The patterns obtained from the pattern inducer are noisy. The pattern ranker adopts a statistical approach that is language independent as well as relation independent. The pattern ranker produces a score for each pattern. All patterns above a fixed threshold are considered for the fact extraction.
- Fact extractor and ranker: The fact extractor applies the high ranked patterns to the n-grams. This leads to a very large collection of tuples. However, the tuples contain correct as well as incorrect information. Therefore, the fact ranker scores the facts based on a supervised approach. This leads to a knowledge base with facts having associated confidence weight.

An overview of the system flow is depicted in Figure 3.1.

Seed Processor and Pattern Inducer We use ConceptNet, an existing common sense knowledge base constructed using crowd sourcing, as the resource for seeds. We begin with a seed set from ConceptNet and process the seed set in order to induce better patterns. Section 3.3 provides details of this module. The next phase of this module is the pattern inducer. The pattern inducer takes as input the clean and normalized seeds processed by the seed processor. These seeds are then applied to the n-grams text in order to obtain patterns. We make two notes here: the first concerns the context of text match, while, the second concerns leveraging part of speech tagging

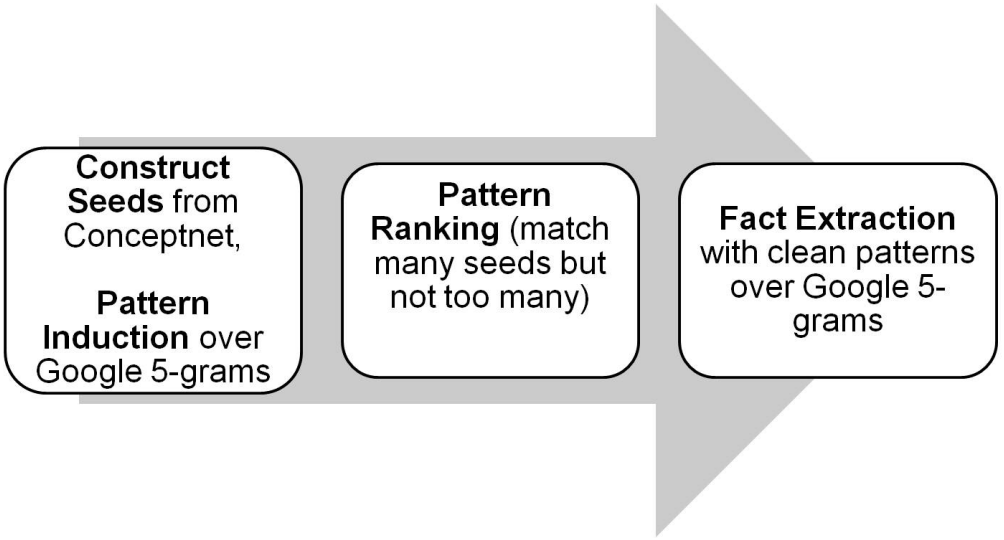

FIGURE 3.1: System Flow Diagram

constraints in order to churn out misleading text matches. To address the two notes mentioned, we support different contexts in which a pattern can occur. At the same time, we use linguistic knowledge about the occurrences of seed in a pattern match to further improve the scope of pattern match. Details of the pattern induction are presented in section 3.4.

Pattern Ranker The pattern ranker judges the reliability of a pattern. The part of speech tagger mentioned just mentioned, oftentimes, makes judges incorrect part of speech predictions. This amounts in an additional noise that creeps into the patterns despite the linguistic constraints. The noise at the seed as well as the pattern induction level in addition to the noise in the text itself, makes it very important to churn out the noisy patterns. This is handled by the next module i.e. the pattern ranker. We investigate three different approaches for pattern ranking.

1. Supervised pattern ranking: This approach classifies patterns as useful or not useful based on a defined set of features. Section 3.5 describes this approach.

2. Pointwise Mutual Information (PMI) based pattern ranking : In this approach, we rank patterns based on PMI. Section 3.5 describes this approach.

3. A new method for pattern ranking : We propose a novel method that overcomes the limitations of the previous pattern ranking approaches that use supervised and PMI. This approach is presented in Section 3.5.2.

Fact Extractor and Ranker The final module of the system is the fact extractor. The fact extractor applies the reliable patterns obtained from the previous step on text(n-grams) and collects facts with certain statistics. These statistics are passed on to the fact ranker. The fact judges

the correctness of a fact using a supervised approach. We explore a statistical and three different supervised methods for fact ranking in Section 3.6.

Having gained an overview of the approach, we now describe the different modules of the system in more detail. We begin with the first module i.e. selection of seeds. All pattern-based IE approaches start with a set of seeds in order to induce patterns as described in Section 2.

3.3 Seed Selection

Seed selection is the first step in inducing patterns. However, there are several challenges involved in choosing the right seeds. Let us discuss the challenges in detail.

1. **Automatic selection of good seeds**: Manually creating and maintaining large list of entities is expensive and laborious. In response, many automatic and semi-automatic techniques have been developed. Among them, semi-supervised techniques are most popular because they allow the users to expand specific target classes without the need for large amounts of training data. The most popular semi-supervised method is pattern-based techniques (discussed in Chapter 2). Typical semi-supervised methods start with a small set of seed instances for a specific target relation. This seed list can be automatically expanded by essentially running a small cycle of pattern-based IE approach. However, even state-of-the-art systems inevitably produce seed sets containing errors. One of the reasons is drifting of the concept.

2. **Quality of seeds**: Accuracy of the seeds severely influences the quality of extraction results. Good seeds lead to good patterns and good patterns lead to good tuples. Therefore, it is critical that the seeds are good in quality.

3. **Rare seeds vs. typical seeds**: A typical seeds gives rise to several patterns during pattern induction. However, this leads to drifting to meaning. Typical seeds occur in several contexts that are not related to the relation that is desired. On the other hand, rare seeds generally tend to match the desired context, however these rare seeds are not likely to find many matches in text. Subsequently, it provides only few patterns. The balance is again a challenge.

4. **Quantity of seeds**: What is the right amount of seeds that is enough to initiate the process in such a way that a lower number of iteration are required? The answer to this perhaps lies in conjunction with the problem on ”how much iteration” discussed next.

5. **How much iteration?** In general, pattern-based IE approaches require several iterations for a larger coverage of facts as discussed in Chapter 2. On the other hand, more iterations

degrade performance and possibly cause drift of the concept with subsequent iterations. This poses a question as to what is the optimum number of iterations. Although there is no clear answer, but it depends on the number of seeds available. An observation is that much iteration is not required when the initial seed set taken is good and large enough, as the seeds themselves are capable of collecting more patterns and hence more facts.

6. **Source of seeds**: It is very difficult to get a reliable source for obtaining seeds. If one turns to very accurate systems like WordNet, the problem is that it is not easy to obtain typical seeds from Wordnet. Thus, this problem is one of finding a source that balances reliability of source and typicality of seeds in that source.

7. **Ambiguity of seeds**: Some seeds can have more than one meaning e.g. the word `bank` is used as a riverbank or financial institution. Polysemy of a seed can introduce semantic ambiguity resulting in errors during pattern induction such that patterns of two different relations would belong to the same relation. This produces noise in the next of the iterations.

8. **Relation hierarchy**: Consider two relations that form subclass, superclass hierarchy i.e. one relation contains the other more specific relation. Seed selection is very difficult in these cases.

   In our system, as in other pattern-based IE approaches, seed selection plays a crucial role. Therefore, it is critical to churn the incorrect seeds. Our approach is to use ConceptNet, an existing common sense knowledge base as the seed source. Using ConceptNet addresses many of the issues in seed selection. Firstly, in a crowd sourced knowledge base, several high ranked seeds are moderately typical and at the same time, these are rare enough to prevent semantic drift. ConceptNet has a large collection of facts, therefore, even little iteration could lead to a desirable number of extractions. This further avoids semantic drift, and helps maintain a good accuracy.

   However, as with any crowd sourced database, there are several incorrect tuples, see Section 4.1. For all relations that we consider, we observe that ConceptNet tuples with high confidence score were sufficiently accurate. The concern is that some relations have many times more tuples than others do. One choice of seed selection would be to set this threshold at a different value for each relation. In this way, we can get much cleaner seeds. However, this approach is limited due to hand coding. Therefore, we settle on a score of a sufficiently high score for all relations in order to employ sufficient accurate seeds. We let the pattern ranker address the noise where the seeds are not absolutely accurate. An important point to consider is normalization of seeds. Due to the inherent nature of natural language, several variations of the same word are possible. There are plural forms, singular forms. Hence, we normalize the seeds in four combinations obtained by pluralizing the subject and object respectively. The limitation by considering stemming is that

it leads to several forms that are not correct and the rules of stemming cannot handle all those variation thus leading to errors. Thus, the seed processing step produces a seed set that are passed to the pattern induction step.

3.4 Pattern Induction

After the seed processing step, we begin the pattern induction step that attempts to bootstrap the extraction starting out with a set of correct seed instances of each relation under consideration. For instance, for the `PartOf` relation, it could use a list including `(finger,hand)`, `(leaves,trees)`, and `(windows,houses)`. For the `IsA` relation, seed patterns can include `(dogs,animals)` and `(gold,metal)`. The goal of this step is to obtain a list of patterns that can then be used to harvest further knowledge from the corpora.

We iterate over the n-gram dataset and look for n-grams that contain the two words that make up a seed. Given a match, we can obtain a pattern by replacing the seed words with wildcards and optionally pruning preceding and following words. For example, given a seed pair `(dogs,animals)` for the `IsA` relation, we may encounter an n-gram like "*with dogs and other animals*". This n-gram gives rise to two patterns: "*with* <*X*>$_{\text{NNS}}$ *and other* <*Y*>$_{\text{NNS}}$" and "<*X*>$_{\text{NNS}}$ *and other* <*Y*>$_{\text{NNS}}$", where NNS represents the part-of-speech tag of the words. Generally, we retain all words between the two seed words, as well as all combinations of the following:

- 0,. . . ,n-2 preceding words before the first seed word
- 0,. . . ,n-2 following words after the second seed word

If n-grams are restricted to a maximal length of n, there can be at most

$$\max_{x \in 1,\ldots,n-1} x(n-x) = \max_{x \in \{\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil\}} x(n-x) = \left\lfloor \frac{1}{4} n^2 \right\rfloor$$

patterns per n-gram.

Simple calculations are required to derive the maximum patterns per n-gram.

$$\max_{x \in 0,\ldots,n-2} (x+1)(n-x-1)$$

$$\max_{x \in 1,\ldots,n-1} nx - x^2 - x + n - x - 1$$

optimal x:

$$n - 2x - 2 = 0 \Rightarrow n - 2 = 2x \Rightarrow x = (n-2)/2$$

for even n this results in:

$$max_x nx - x^2 - x + n - x - 1 = n(n-2)/2 - ((n-2)/2)^2 - 2(n-2)/2 + n - 1$$

$$n(n-2)/2 - (n-2)^2/4 - 2(n-2)/2 + n - 1$$

$$(n-2)(n-2)/2 - (n-2)^2/4 + n - 1$$

$$1/4 * (n-2)^2 + n - 1$$

$$1/4 * (n^2 - 4n + 4) + n - 1$$

$$1/4 * n^2$$

We support the following three types of pattern variations:

- seed at the extreme of the text: for example `<X> ... <Y>` is a very common template.
- tokens on either side of the seed: e.g. `preceding-token <X> ...<Y>` can be induced by applying the seeds `X, Y=<apple, sweet>` over the n-gram `very sweet apples`. In addition, `<X> ... <Y> following-token` is obtainable by useful for n-grams where the context is essential.
- patterns without any text between subject and object: for example, `very <ADJ> <NOUN>` is obtainable from `very green apple`.

Additionally, we use linguistic knowledge about the occurrences of seed in a pattern match to further improve the pattern match. We manually define linguistic constraints on patterns of relations based on the seeds part of speech candidate tags. For example, in a relation like `hasProperty`, if `x hasProperty y` then it is very likely that `y` is an adjective whereas `x` is a noun form. We consider such restrictions on the pattern matching. This does not involve deep linguistic knowledge and is not time consuming. In order to perform this, we need to provide part of speech tags to the input n-gram text. Tagging sentences is a well-solved problem, however performing part of speech tagging on a very small context such as an n-gram is not always accurate. We let the pattern ranker resolve the issue of noise.

3.5 Pattern Scoring

In order to remove the noise and less useful patterns, one relies on pattern scoring technique. A widely used measure for pattern scoring is PMI score. Many information extraction systems have relied on PMI scores alone to produce the final scoring [Pantel and Pennacchiotti, 2006].

PMI is a measure of association between two random variables and is widely used in information theory and statistics. The PMI score of a pattern is based on the association of the pattern with the seeds it matched. The variation of PMI that is widely used in pattern ranking is described next.

Given seeds $s \in S$ and a pattern p, one computes

$$\text{score}(p) = \frac{1}{|S|} \sum_{s \in S} \frac{\text{PMI}(s,p)}{\text{PMI}_{\max}} \cdot \text{score}(s)$$

where

$$\text{PMI}(s,p) = \log \frac{\text{freq}(s,p)}{\text{freq}(p) \cdot \text{freq}(s)}$$

$$\text{PMI}_{\max} = \max_{s \in S} \text{PMI}(s,p)$$

Seed scores, represented as $\text{score}(s)$ can be set to 1. We normalized scores in $[0,1]$ based on the seed frequency.

Thus, the PMI approach exploits the statistics about the patterns and seeds that we already possess.

3.5.1 Supervised Pattern Scoring

While PMI is widely used, we found upon investigation that many of the high PMI scoring patterns are not reliable, see Chapter 5. Therefore, instead of relying solely on PMI, we instead used it as one of the features in a supervised setting. We describe a supervised approach that learns the reliability of a pattern-based on a training set for each target relation. The supervised ranker considers some intuitive features for classifying if a given pattern is reliable. We believe that the classifier should be able to learn the importance of the features, including PMI. The features we consider are described below.

- PMI score between the seeds and pattern.
- The length of the pattern: It is observed that smaller patterns are more reliable.
- Percentage of stop words: It is observed that reliable patterns tend to contain stop words, e.g. "*which is always*", "*may be*", "*are not*".
- If part of speech tagged data is available, consider percentage of words that are noun, proper noun tokens.
- The frequency of the pattern: Higher frequency, i.e. higher number of matches in text, denotes significance.
- The number of seeds: How many seed tuples did the pattern occur with?

The supervised method delivers more useful patterns, but requires a small set of patterns be labeled for their supposed usefulness towards learning about common-sense tuples. For example, the unsupervised approach gave a high PMI score to the pattern "*and a*" because it matched seeds like `(rate, high)` in phrases like "*low unemployment rate and a high GDP*". The supervised approach removed many such patterns.

3.5.2 Statistics Based Pattern Scoring

In our approach, it is vital to score the patterns reliably. The supervised approach addresses some concerns of PMI, but more accurate pattern scoring is quintessential. Secondly, it is challenging to provide manually labeled training set for the supervised classifier. This poses more problems when one scales to large number of target relations. The challenge is thus to make judicious choices and keep only promising patterns. We base our decision on two important observations:

i) First of all, given the comparably large number of seeds that are available in ConceptNet, any pattern that only matches a single seed in the entire n-gram dataset is likely to be an unreliable pattern that does not really correspond to the target relation. It is interesting to note that many information extraction systems have relied on PMI scores to prune the pattern list [Pantel and Pennacchiotti, 2006]. Unfortunately, PMI considers raw frequencies only, without emphasizing the number of *distinct* seeds matched. In particular, it turns out that the PMI formula tends to give these rare patterns that only coincidentally match a single seed the highest scores. Hence keeping only the highest-ranked patterns in terms of PMI would leave us with only useless patterns.

ii) At the same time, there are patterns that match multiple seeds, but that simultaneously match seeds from other relations, distinct from the target relation. PMI does not consider whether a pattern also matches other relations than a given target relation. The more this occurs, the greater indication we have that the pattern is not a reliable indicator of a specific relation, but just a broad generic pattern that happens to match many different word combinations.

We devised a score that simultaneously addresses both of these issues. With respect to aspect i), we see as depicted in Figure 5.5 that the number of seeds $s(x)$ per pattern follows a power-law $s(x) \approx ax^k$, where the majority of the patterns are in the long tail. For different relations, the dominating patterns have different numbers of seeds, and we empirically found that it is not possible to choose a threshold that works well for all relations. Instead, we observe that the magnitude of the slope at a particular position x characterizes to what degree a pattern is in the long tail. We hence use least squares linear regression with respect to $\log s(x) = k \log x + \log a$ to estimate k and a from the data. For a given seed count $s(x)$, we have $x \approx \left(\frac{s(x)}{a}\right)^{\frac{1}{k}}$, and

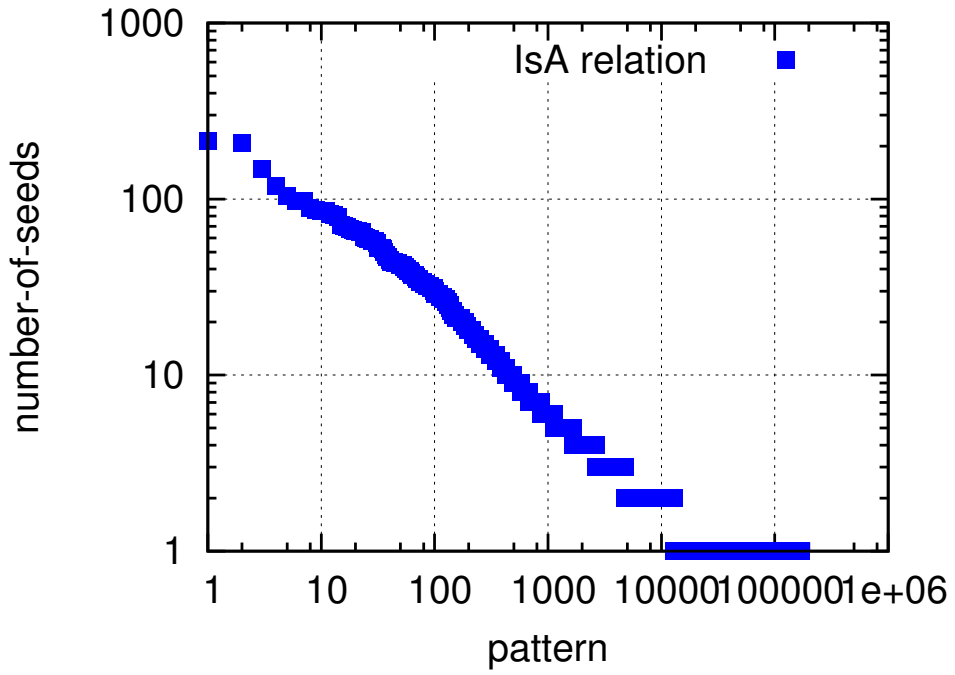

FIGURE 3.2: Power law distribution: number of seeds per pattern (log scale)

therefore,

$$\frac{d}{dx}s(x) \approx akx^{k-1} = ak\left(\frac{s(x)}{a}\right)^{\frac{k-1}{k}} \tag{3.1}$$

characterizes the slope at $s(x)$. The more negative this value is, the more dominating the pattern.

For aspect ii), we compute a score as follows

$$\phi(R_i, p) = \sum_{R_j, j \neq i} \frac{|s(R_i, p)|}{|s(R_i)|} - \frac{|s(R_j, p)|}{|s(R_j)|}. \tag{3.2}$$

This score considers the number of seeds $s(R_j, p)$ that the pattern matches from relations R_j (other than R_i) in comparison to the fraction of seeds it matches for R_i.

There is often a trade-off between the two aspects, as a score that matches many seeds will also be more likely to falsely match other relations. Given $\frac{d}{dx}s(x)$ from Equation 3.1 for a relation R_i and a pattern p, as well as $\phi(R_i, p)$ from Equation 3.2, we combine both scores conjunctively:

$$\theta(R_i, p) = \frac{e^{\phi(R_i,p)}}{1 + e^{\phi(R_i,p)}} \cdot \frac{|\frac{d}{dx}s(x)|}{1 + |\frac{d}{dx}s(x)|} \tag{3.3}$$

This corresponds to normalizing the two scores $\log|\frac{d}{dx}s(x)|$ and $\phi(R_i, p)$ to $[0, 1]$ using the logistic function and then multiplying them, which entails that only patterns with good characteristics with respect to both aspects will obtain high ranks. We can thus choose a set of

top-ranked patterns that are sufficiently significant to match a sufficient number of seeds, but at the same time do not over generate large numbers of irrelevant tuples.

Algorithm 1 Web-Scale Knowledge Acquisition

procedure HARVEST(n-gram dataset f, seeds S_i for relations $R_1, \ldots, R_m$, optional negative seeds S_i^-)
  $P_1, \ldots, P_m \leftarrow$ INDUCE_PATTERNS$(f, S_1, \ldots, S_m)$ ▷ **collect patterns** P_i for each relation
  $K_i \leftarrow \emptyset \quad \forall i$ ▷ candidate facts
  for all $s \in f(*)$ **do** ▷ for all n-grams
    for all i in $1, \ldots, m$ and $p \in P_i$ **do** ▷ for all patterns
      if $s \in f(p)$ **then** $K_i \leftarrow K_i \cup \{(\text{ARGX}(p, s), \text{ARGY}(p, s))\}$ ▷ if s matches p, the arguments form **new fact** candidates
  create labeled training sets T_i with labels $l_{R_i(x,y)} \in \{-1, +1\}$ for $(x, y) \in T_i$ ▷ using S_i and optionally S_i^-
  for all $i \in 1, \ldots, m$ **do**
    for all $(x, y) \in T_i$ **do** ▷ create **training vectors**
      create training vector $\mathbf{v}_{R_i(x,y)}$ using patterns in P_i and n-gram dataset f
    learn model M_i from $\{(\mathbf{v}_{R_i(x,y)}, l_{R_i(x,y)}) \mid (x, y) \in T_i\}$ ▷ use **learning algorithm**
  $K_i \leftarrow \{(x, y) \in K_i \mid M_i(\mathbf{v}_{R_i(x,y)}) > 0.5\} \quad \forall i$ ▷ vectors for candidates are created using P_i and f and **assessed** using M_i
  return accepted facts $K_1 \ldots, K_m$ ▷ accepted facts as final **output**
procedure INDUCE_PATTERNS(n-gram dataset f, seeds $S_1, \ldots, S_m$)
  $P_i \leftarrow \emptyset \quad \forall i$ ▷ sets of patterns
  for all $s \in f(*)$ **do** ▷ for all n-grams
    for all i in $1, \ldots, m$ and $(x, y) \in S_i$ **do**
      if s contains x and y **then** $P_i \leftarrow P_i \cup$ CREATEPATTERNS(s, x, y) ▷ replace x and y in s with **wildcards**, prune text
  $P_i \leftarrow \{p \in P_i \mid \theta(R_i, p) > \theta_{\min}\} \quad \forall i$ ▷ **prune** using Equation 3.3
  return $P_1, \ldots, P_m$

3.6 Tuple Ranking

Following the pattern ranking step, we have a set P of patterns that are likely to be at least somewhat relevant for a given relation. A given pattern $p \in P$ can be instantiated for specific candidate tuples (x, y) as $p(x, y)$ to yield an n-gram string. For instance, a pattern like "*<X> is located in <Y>*" can be instantiated with a tuple `(Paris,France)` to yield an n-gram "*Paris is located in France*". For such n-grams, we can then consult an n-gram dataset f to obtain (raw or derived) frequency information $f(p(x, y))$ that reveals how frequent the n-gram is on the Web.

Pattern	Pattern count
$pattern_1$	$\sum_i n_1 i * freq_1 i$
$pattern_2$	$\sum_i n_2 i * freq_2 i$
$pattern_k$	$\sum_i n_k i * freq_k i$
sum	$\sum_k pattern\#$

TABLE 3.1: Computing tuple frequency

3.6.1 Unsupervised Tuple Ranking

We use the modified form of PMI [Hagiwara et al., 2009] to measure the reliability of the tuples obtained.

$$PMI(tuple, pattern) = log \frac{frequency(tuple, pattern)}{frequency(pattern) * frequency(tuple)}$$

$$score(tuple) = \alpha(pattern) * \beta(pattern)$$

where,

$$\alpha(pattern) = \frac{1}{patternscount}$$

$$\beta(pattern) = \sum \frac{PMI(tuple, pattern)}{PMI_{max}} * score(pattern)$$

Here, $frequency(pattern)$ = Number of times the pattern appeared in the n-grams. It is computationally prohibitive to compute the pattern occurrence over entire n-grams corpus. Therefore, we propose an estimation method. Consider for example, the pattern `X ``are always`` Y`. In order to determine the frequency("are always"), we search the 2-gram corpus for "are always". The frequency returned gives an estimate of the number of times "are always" occurred. However, some unwanted occurrences of the pattern "are always" also play a part in this estimation, e.g. consider: "& are always ,". We inadvertently include such phrases that contain punctuations. We observed that there are only few such instances. The adverse affect is, however, smoothed in the overall computation because it is assumed that every PMI score is affected by such cases almost uniformly. Therefore, it does not affect the overall ranking scores.

Table 3.1 describes the computation of frequency of a tuple. Unfortunately, with several variations of a pattern i.e. with left, right and middle words in patterns, the frequency computation is not robust. Therefore, the unsupervised tuple ranking approach is not experimented extensively.

3.6.2 Supervised Tuple Ranking: Method-1

In a large corpus like the Web, a given tuple occurs frequently matching more than one pattern. Having said this, not all patterns are equally reliable. We observed that PMI scores alone are

not the best indicators of reliability. Hence, we opted for a supervised approach. The following features are employed:

- Frequency : The number of times the n-grams appears in the web corpus. This is relevant for evidence oftentimes, for instance, `flower, bright` has many occurrences while `flower, dull` has fewer occurrences. This correlates with the common sense knowledge that flowers are generally bright, thereby providing an evidence.
- Percentage Plural: Relatively, how many times has the concept occurred as a plural. e.g. {Flowers are red} has more prominence than {a flower was red} [Carlson, 1982].
- Percentage Named Entity: We observe that common sense knowledge is generic in nature and does not restrict to any particular named entity. Therefore, as a feature we count the number of times the concept contains a named entity. To see an example, consider the song titled "East is Red". This example fact is not common sense knowledge rather it is a fact. Here, 'East' is a proper noun, we are not looking for such concepts that are named entities. This feature adds a negative bias to the evidence of being a good common sense knowledge tuple.
- Part of speech tags: Consider a relation like `hasProperty`, it is relevant to note that the property occurs as an adjective. 'Green' could be used as a noun as well as adjective, e.g. the phrase 'Brown is Green' denotes Brown University is nature friendly: here Green is a proper noun. We do not consider such properties for the relation `hasProperty`. For each relation, we manually list the ideal part of speech tags that emulate the tuples of that relation. In order to rank the tuple of a given relation, the features are the relative count of the desired part of speech tag.
- Pattern scores: We observe that some patterns like `X are always Y` are more useful in extracting properties of a concept as compared to other patterns.

The Stanford Tagger [Toutanova and Manning, 2000] is used to tag the n-grams in order to get the part of speech required for some of the features described previously. We obviate the use of a more expensive parser by employing simple heuristics in order to estimate the feature 'percentage named entity'. Part of speech tags provide sufficient information in order to judge as named entity e.g. the tags NNP, NP were identified as being named entitites. The exact named entity type is not needed, hence a tagger is sufficient thereby avoiding a more expensive named entity recognizer.

Given a set of examples that are manually labeled as positive (the two items do stand in the respective relation to each other) or negative (they do not stand in the given relationship to each other). For example, for the `hasProperty` relationship, the pair `(apple,edible)` would be considered positive and `(apple,computer)` would be a negative pair.

3.6.3 Supervised Tuple Ranking: Method-2

For a set of m patterns $P = \{p_1, \ldots, p_m\}$, k n-gram data sources f_j ($j = 1 \ldots k$), and a given pair of words (x, y), we produce a $k(m+1)$-dimensional vector $\mathbf{v}_{(x,y)}$ to represent the corresponding pattern statistics.

$$v_{(x,y),i} = \begin{cases} \sum\limits_{p \in P} f_{\phi(i)}(p(x,y)) & \sigma(i) = 0 \\ f_{\phi(i)}\left(p_{\sigma(i)}(x,y)\right) & \text{otherwise} \end{cases} \tag{3.4}$$

where $\phi(i) = \left\lfloor \frac{i}{m+1} \right\rfloor$ and $\sigma(i) = i \mod (m+1)$. The idea is that for each n-gram dataset f_j, we have $|P|$ individual features $f_j(p(x,y))$ that capture pattern-specific frequency scores, as well as a single feature that captures the total frequency score $\sum_{p \in P} f_j(p(x,y))$.

Algorithm 1 gives the overall procedure for extracting information. Given the set of vectors $\mathbf{v}_{(x,y)}$ for the training examples (x, y) together with their respective labels $l_{(x,y)}$, a learning algorithm like support vector machines is used to derive a prediction model M for new pairs (x, y). The prediction model M provides values $M(\mathbf{v}_{(x,y)}) \in [0, 1]$ for the vectors of new pairs (x, y), where values over 0.5 mean that the pair is accepted. The specific learning algorithms used to derive the prediction model are listed in Chapter 5. It is assumed that the first n-gram dataset f_1 supports wildcards, so for a pattern like "*<X> is located in <Y>*" we can simply query for "*<?> is located in <?>*". The union T of all tuples matching patterns in P is the set of candidate tuples. Note that for the Google Web 1T dataset, we could simply iterate over all n-grams in the dataset, and see which n-grams match any of the patterns. Those candidate tuples predicted to be true by the classification model are the ones that constitute the final output.

3.6.4 Supervised Tuple Ranking: Method-3

Given a set P of patterns that characterize a given relation. A given pattern $p \in P$ can be instantiated for specific candidate facts (x, y) as $p(x, y)$ to yield an n-gram string. For instance, a pattern like "*<X> is located in <Y>*" can be instantiated with a fact `(Paris,France)` to yield an n-gram "*Paris is located in France*". For such n-grams, we can then consult an n-gram dataset f to obtain frequency information $f(p(x, y))$.

Certainly, one could use the union of all facts found as the final output. Fortunately, in a large corpus like the Web, a given fact will frequently occur with more than one pattern, and we can apply more sophisticated ranking measures, in order to obtain cleaner results. We proceed as follows. Given the set of seeds S_i for relations $R_1, \ldots, R_m$, we compute an $m \times m$ square similarity matrix as $M_{i,j} = \frac{|S_i \cap S_j|}{|S_i \cup S_j|}$. For each relation R_i, we then use S_i as a set of positive examples. If the database provides any negative seeds S_i^-, these are used as negative examples. If

too few negative seeds are available, we rely on $\bigcup_{j \neq i, M_{i,j} < 0.5\%} S_j$ as a pool of additional negative examples. We sample from this pool until we have an equal number of positive and negative examples, which can be used in conjunction with supervised learning algorithms.

For a set of l patterns $P = \{p_1, \ldots, p_l\}$ and a given pair of words (x, y) for some relation, we produce an $l + 1$-dimensional vector $\mathbf{v}_{(x,y)}$ with

$$v_{(x,y),0} = |\{f\,(p_i(x, y)) > 0 \mid i = 1, \ldots, l\}|,$$

$$v_{(x,y),i} = \begin{cases} 1 & f\,(p_i(x, y)) > 0 \\ 0 & \text{otherwise} \end{cases}$$

for $i > 0$. Algorithm 1 gives the overall procedure for extracting information. We induce patterns, prune the pattern list, and then iterate over the n-grams to find candidate facts K_i for each relation R_i. We then use a learning algorithm to derive prediction models M_i for each relation. The models provide values $M(\mathbf{v}_{R_i(x,y)}) \in [0, 1]$, where values over 0.5 mean that the pair is accepted.

3.7 Summary

In summary, this chapter reviews the core of this thesis. We rely on N-grams to overcome the limitations of scalability and yet avoid the challenges in procuring large-scale data. Secondly, this chapter discussed our model for pattern-based IE. We proposed different alternatives for pattern ranking, like supervised learning for pattern scoring. We introduced a novel pattern scoring strategy that carefully determines the patterns. We hope that this helps to overcome the robustness and scalability challenges of previous work. For tuple ranking, we proposed three different supervised models that could carefully judge the reliable tuples. The next chapters present the experimental evaluation of these approaches that measure the efficacy of the approaches.

Chapter 4

Experimental Setup

4.1 Dataset

We conducted several experiments using combinations of the following data sources:

Google Web 1T N-Gram Dataset Version 1 Google has published [Brants and Franz, 2006a] a dataset of raw frequencies for n-grams ($n = 1, \ldots, 5$) computed from over 1,024G word tokens of English text, taken from Google's Web page search index. In compressed form, the distributed data amounts to 24GB. While the dataset does not include n-grams with a frequency of less than 40, the fact that it is distributed as a complete dataset means that additional post-processing and indexing can be applied to support more sophisticated query syntax.

ConceptNet 4.0 (2010-02 database) A database of facts generated from user contributions [Havasi et al., 2007]. ConceptNet is developed at MIT Media Laboratory and is presently the largest common sense knowledge base. ConceptNet is a relational semantic network that is automatically generated from about 700,000 English sentences of the Open Mind Common Sense (OMCS) corpus. Nodes in ConceptNet are compound concepts in the form of natural language fragments (e.g. in order to mention a location, phrases like `grocery store`, and `at home` are used). Because the goal of developing ConceptNet is to cover pieces of common sense knowledge to describe the real world, there are nearly 40 relations categorized as causal, spatial, functional, and other categories. An exceptional feature of ConceptNet is that there are positive as well as negative assertions. For example, `hasPropertyflower, soft[always]` and `hasPropertyflower, ugly[not]`. These assertions can be exploited in constructing negative instances in order to train a classifier as well as to impose constraints. ConceptNet has been adopted in many interactive applications [Havasi et al., 2007].

Upon closer inspection, we discovered that ConceptNet is not as reliable as it could be. There are facts like `UsedFor(see, cut wood)` and `IsA(this, chair)`, resulting from misinterpretations of the user-provided sentences. Further, some other types of misinterpretations in ConceptNet we discovered are as follows.

- Normalization of tuples: Several tuples are not normalized, e.g. `HasProperty(apple, sweet very green yellow red)[usually]` would confuse a machine, as the properties are not decompounded. This incorrect tuple must be splitted into four tuples as: `HasProperty(apple, sweet)`, `HasProperty(apple, very green)`, `HasProperty(apple, yellow)`, and `HasProperty(apple, red)`.
- Zero score or negative labels to correct tuples: Correct tuples are oftentimes provided a score of zero, e.g. `HasProperty(wallet, black)` has a score of zero, even though in reality, majority of the wallets are black! On the other hand, some tuples are incorrectly labeled as negative, though the appropriate label is positive, e.g., a common fact is that jeans are generally blue. However, `HasProperty(jeans, blue)[not]` implies jeans are not blue.
- Contradictory polarity of tuples: Some tuples occur simultaneously with the label [not] and [always], this is contradictory , e.g., `HasProperty(jeans, blue)[not]` that implies jeans are not blue. However, there is `HasProperty(jeans, blue)[often]` which implies that jeans are often blue.
- Mismatching relations: The relation is incorrectly labeled, e.g. `HasProperty(literature, book)` does not belong to the relation `hasProperty`.
- Encyclopedic information instead of common sense, e.g. `HasProperty(high point gibraltar, rock gibraltar 426 m)` is a fact rather than common sense.

Despite these limitations, ConceptNet is a useful resource primarily due to the sufficiently large-scale, including several relations. The top ranking commonsense facts in ConceptNet are generally of very high quality, both in the positive (e.g. [always]) as well as the negative category (e.g. [not]).

WordNet WordNet is a manually constructed lexical system developed at Princeton University. The project originated from a vision to produce a dictionary that could be searched conceptually instead of only alphabetically. The basic object in WordNet is a set of strict synonyms called a synset. By definition, each synset in which a word appears is a different sense of that word. There are four main divisions in WordNet, one each for nouns, verbs, adjectives, and

adverbs. The IsA relation is the dominant relation, and organizes the synsets into a set of approximately ten hierarchies.

Seeds from WordNet are extracted with scores using a simple ranking formula based on corpus frequencies. Note that for antonymy, we get each tuple twice (once in each direction). Some sample WordNet sample seeds are `Hypernym(friend,person)` and `Holonym(water,ice)`.

WordNet and ConceptNet have several similarities: (1) their structures are both relational semantic networks; (2) both of them are general-purpose (that is, not domain specific) knowledge bases; and (3) concepts in both resources are in the form of natural language. On the other hand, WordNet and ConceptNet differ from each other in some aspects: (1) as their processes of development differ (manually handcrafted vs. automatically generated), intuitively WordNet has higher quality and robustness; (2) while WordNet focuses on formal taxonomies of words, ConceptNet focuses on a richer set of semantic relations between compound concepts; and (3) WordNet differentiates ambiguous meanings of a word as synsets, however, ConceptNet bears ambiguity of common sense knowledge in its concepts and relations.

However, upon further investigation we found that WordNet is not suited to common sense knowledge for two reasons. Firstly, ConceptNet contains many more common sense relations, and the size of the relations is very large. Secondly, although WordNet seeds are more accurate, they tend to be found very rarely in text. As we discussed in Section 3.3, it is important to maintain a balance between rarity and typicality of seeds. The problem is that it is not easy to obtain typical seeds from Wordnet. ConceptNet alleviates this problem and provides a good balance. Therefore, we choose ConceptNet as the seed source.

The Semantically Annotated Snapshot of the English Wikipedia [Atserias et al., 2008] provides annotations of a 2007 version of the English Wikipedia. However, the small size of Wikipedia and encyclopedic nature of information limits to insufficient pattern induction for common sense knowledge relations or even to generate seeds automatically for many relations. In our experiments, we only used it for `hasProperty` relation.

The Microsoft Web N-gram Corpus Microsoft's Web N-gram Corpus [Wang et al., 2010] is based on the complete collection of documents indexed for the English US version of the Bing search engine. In addition to offering n-gram language models computed from the document bodies (1.4T tokens), it also offers separate n-gram statistics computed from document titles (12.5G tokens) and document anchor texts (357G tokens). Experiments have shown that the latter two have rather different properties [Wang et al., 2010]. The data set is not distributed as such, but made accessible by means of a Web service, which is described using the WSDL standard. The service provides smoothed n-gram language models rather than raw frequencies,

AtLocation	CapableOf	Causes	CausesDesire
ConceptuallyRelatedTo	CreatedBy	DefinedAs	Desires
HasA	HasFirstSubevent	HasLastSubevent	HasPainCharacter
HasPainIntensity	HasPrerequisite	HasProperty	HasSubevent
InheritsFrom	InstanceOf	IsA	LocatedNear
MadeOf	MotivatedByGoal	PartOf	ReceivesAction
SimilarSize	SymbolOf	UsedFor	WordNetAntonymAdj
WordNetHypernym	WordnetHasCategory	WordnetAntonymNoun	WordnetAntonymVerb
WordnetCause	WordnetEntailment	WordnetHolonymMember	WordnetHolonymPart
WordnetHolonymSubstance	WordnetInstance		

TABLE 4.1: The set of relations considered.

generated using an algorithm that dynamically incorporates new n-grams as they are discovered [Huang et al., 2010]. We used this corpus to conduct experiments in which we combined the scores from different n-gram statistics, including Google n-gram, in order to produce results that are more accurate.

The ClueWeb09 collection We also evaluated our approach on raw 5-gram frequencies computed from the complete English subset of the ClueWeb collection[1], which consists of over 500 million web pages. The ClueWeb09 collection is also used in several TREC tasks.

4.2 Relations

We consider around 40 relations for common sense knowledge extraction. Most of these relations overlap with the relations present in ConceptNet. We consider these relations because they are a good representative of common sense relations. Secondly, ConceptNet can be leveraged to get a rich set of seeds for these relations without any additional efforts. Few relations used WordNet as a source of seeds, however as discussed earlier, these seeds gave rise to a very limited set of patterns due to the rarity of the seeds. Therefore, in the large-scale experiments that we conducted, we do not focus on the WordNet inspired relations. On the other hand, some of these relations are already covered by the ConceptNet relations. For example, `wordnet-hypernym` which is the hypernym relation is already covered by the relation `isA`.

A list of the relations used in our system is presented in Table 4.1. It is noteworthy that most of these relations can be obtained with only very short sentences (e.g. 5-grams in our system) as empirically demonstrated in Chapter 5.

[1] `http://boston.lti.cs.cmu.edu/Data/clueweb09/`

4.3 System Architecture

This section presents the architecture and implementation details of our system, which is the first automated large-scale common sense information extraction system. We used the Java programming language to develop a platform-independent knowledge base processing framework. The knowledge base consists of facts in English language. The system is a fully implemented and extracts common sense relational tuples from trillions of words. The system has three primary modules as described in Chapter 3. We discuss the three core modules of the system with a focus on architecture and implementation.

- Seed processor and Pattern Inducer: Given an existing fact knowledge base having a mix accuracy of tuples, the Seed processor inspects seeds that can lead to potentially good patterns. We start with processing seeds from ConceptNet. The tuples in ConceptNet is available in more than one language. We consider the most widely used edition i.e. the English language knowledge base. ConceptNet is distributed in two forms. The first form is a text file of about 180 MB in size that contains all tuples (all relations included) in a standardized format. The format of the text file looks like this: X/assertion/assertion-ID conceptnet:LeftConcept X/concept/1005523; conceptnet:RelationType http://X/reltype/PartOf; conceptnet: RightConcept X/concept/1000370; conceptnet:LeftText "*a door*"; conceptnet:RightText "*a house*"; conceptnet:FrameId X /frame/1387; conceptnet:Language X/language/en; conceptnet:Creator X/user/14494; conceptnet:Score 6; conceptnet:Sentence "*a house has a door*". Here, X can be substituted with `http://conceptnet.media.mit.edu`.

  The second way to procure ConceptNet is through a Python wrapper that allows us to obtain all facts in a particular relation and query in many different ways, for example, facts of a certain polarity (negative or positive) and with parameters of score. We used this wrapper and constructed a format suited to our needs. The format looks like this: `HasProperty(apple, red)[] 10`. The next step is to normalize the seeds by pluralizing the seeds in order to capture four combinations obtained by pluralizing the subject and object respectively. We consider pluralizing instead of the stemming because stemming leads to several incorrect forms. This leads to errors. On the other hand, pluralizing can at times create some rare seeds, however this ensures that we capture all different variations of the seed pair.

  In order to consider only the high scoring seeds, we settled on the ConceptNet tuples with a score greater than 3.0. We obtained this score by empirical observations over several relations such that this gives rise to a sufficient number of seeds, and at the same time is globally accepted in terms of reliability (for all relations). This ensured that we do not hard code different threshold for each relation, however some tuples were a bit unreliable in this process since we took a large number of seeds of about few hundred for

different relations. Some relations have a large number of seeds above the threshold score of 3.0, whereas others have only limited seeds. Overall, there were sufficiently many seeds available for each relation. The next step is the pattern induction step.

The next step of pattern induction requires applying the seeds to some text, in order to derive patterns. As text corpus, we use Google n-grams. The system provides the flexibility to work seamlessly with other data sources. These n-grams are distributed in about 120 text files of nearly 300 MB each. These files are not compressed, however they are sorted by the first token. We used 5-grams in our system because they provide the longest context available with this n-gram dataset. These files contain n-grams that contain frequency statistics e.g. `apple is considered very delicious 300`, where 300 is a frequency of the number of times this 5-gram is seen in the Web corpus from which these n-grams were derived.

Our pattern matching sub-module could match the seeds in the text in three contexts: middle , extremes, and no-word in between. Suppose we have N n-grams and M seeds, then a naive implementation for matching, requires N*M steps. This implementation would take a seed pair and scan over the entire n-gram set. However, the amount of time taken is very large in such a case, therefore we consider some optimizations in order to be efficient e.g. avoiding regular expressions, and identifying meaningless n-grams (containing all numbers, say).

Additionally, for matching patterns with the seeds, our model asserts the part of speech that is generally observed for the seeds of this particular relation. A relation type of `hasProperty(x,y)` is likely to have x as a noun form and y as an adjective form. We use the Stanford tagger in order to tag the n-grams. The tagger was not always accurate, however we observed that even with short context, the part of speech were generally correctly predicted.

It must be noted that we distributed the task of pattern induction over several machines. Distribution of task can drastically reduce the run time of the system. In our case, the runtime was reduced from a few hours to a few minutes just by using distributing using Hadoop framework [Borthakur, 2007]. Hadoop is a framework that makes the distributed computed robust to failures of participating machines.

- Pattern ranker: The patterns obtained from the pattern inducer are noisy. The pattern ranker adopts a statistical approach that is language independent as well as relation independent. In order to rank the patterns, we introduced a novel pattern ranking method that we introduced in Section 3.5.2. The computations required for this system were easy to implement, as those statistics were easily available.

- Fact extractor and ranker: The fact extractor applies the clean patterns on text (5-grams) and collects facts with certain statistics. These statistics include the record of pattern that

match the tuple, along with the frequency count. Primarily, this data is passed on to the fact ranker, see Figure 4.1 for an example. An easy way to rank the tuples is to count the support of patterns. However, this approach is error-prone as illustrated in the Figure 4.1. Those tuples that match several patterns (shown in blue) are generally correct, but the task becomes completely ambiguous with few matching patterns. We learn a decision tree instead. The features for the decision tree are the logarithm of the frequencies of pattern matches. The decision tree provides as output the confidence score of a tuple to be in the positive class.

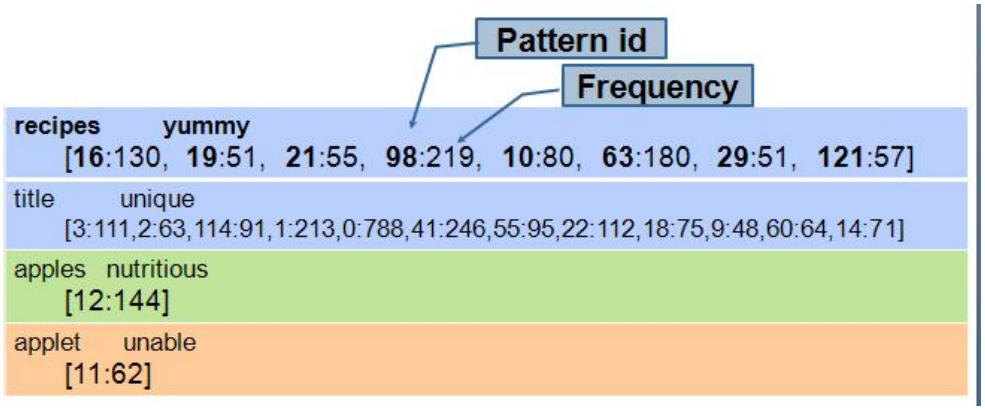

FIGURE 4.1: Sample extractions to rank

Such a supervised classification task requires labeled data, but we do not employ any human for the supervised task, instead we rely on ConceptNet. Thus, one can obtain many positive instances and relatively lesser negative instances from ConceptNet. We synthetically generate additional negative labeled data in case the negative labeled data is insufficient. In order to synthetically generate the additional negative labeled data, we leverage on the similarity between relations. We rely on the following intuition: relations that are more similar will share common seeds and least similar relations will barely share common seeds. We construct an adjacency metric based on a standard metric like Jaccard similarity. Consider the seeds of two unrelated relations $R1$ and $R2$: $seed_R1(x, y)$ can be combinatorially combined with the $seed_R2(x', y')$ to synthetically generate a seed that is not correct i.e. a negative seed : $seed_R1(x, y')$ and $seed_R2(x', y)$. This compensates for the inadequate negative seeds when necessary. Thus, the system is able to judge reliable tuples.

4.3.1 Distributed Framework of Execution

Our system is a distributed system and uses the Hadoop framework [Borthakur, 2007] for distributing the tasks across different nodes. The pattern inducer and fact extractor require distributed computing. Note that the whole process (pattern inducer and fact extractor) took several

hours without distribution. However, the distributed framework significantly reduced the running time of the system to 30 minutes from 30 hours, achieving a increase of several hundred percent. This has an additional advantage of scaling up. If more machines are available, one can significantly speed up the processing. Therefore, the distributed system ensured efficiency in our system.

4.4 Evaluation Metric

For evaluating the results, we rely on the standard metrics precision and recall. These two measures have their roots in document retrieval. Recall measures what fraction of all desired items is in the set of items selected by the system. In contrast, precision measures fraction of all items selected by the system that are actually correct. In the field of information (document) retrieval, the items are documents and the desired set consists of documents that the users consider relevant for a given query. In our context, the items to evaluate are extracted tuples/facts of a given relation e.g. hasProperty(flower, soft) and the desired items are the tuples that indeed belong to the relation specified. In the literature, the desired (correct) items are sometimes called positives, and incorrect items are called negatives. Usually, the system provides a measure of confidence that the system believes the tuple is positive/correct.

Neither precision nor recall is necessarily very useful on its own. A classifier that accepts all items has a perfect recall of 1, while a classifier that does not accept any items obtains a precision of 1. In order to compare different evaluation results one uses F-measure, which is defined as the harmonic mean of precision and recall.

Measuring the actual precision and actual recall is infeasible in a setting like a search engine. It requires computing the entire set of relevant documents. Therefore, one resorts to a sampled set for measuring precision and recall. Similarly, in our context it is unrealistic to assume there is a complete set of tuples belonging to a relation. One can obtain a random sample of correct tuples to compute precision and recall of the system over this reduced set.

Chapter 5

Evaluation Results

The aim of this chapter is to evaluate how close do we reach to the goal we started with, in the beginning (Chapter 1) of the thesis by employing the different approaches that are introduced and explained in Chapter 3. Reiterating the original desiderata of the common sense knowledge base that we envision- simple automatic construction, machine-comprehensible representation, extendible, accurate and high coverage. We formulate these desiderata in form of three questions.

Throughout this chapter, for every approach, we try to answer the following three questions (referred as `G3`:Gold standard-three questions).

1. How effective is the pattern induction, in terms of correctness of the patterns and the number, quality of patterns induced.
2. How effective is the tuple extraction and ranking, in terms of correctness of the tuples and the number of facts extracted.
3. In terms of the overall message of the thesis, how effectively do we address the two underlying themes of the thesis, (a) n-grams as a text corpus for information extraction, (b) robustness of the system.

The chapter is organized on the three approaches described in Chapter 3. We recap the three approaches below.

1. Supervised pattern induction, supervised tuple ranking, referred as approach-1, see 3.4: This approach classifies patterns as useful or not useful based on a defined set of features. The clean set of patterns obtained after the supervised pattern induction are used to extract tuples. The tuples are scored using a supervised classifier. The features of the

tuple scoring classifier are matching pattern frequencies and some other heuristic based features.

2. PMI based pattern induction, supervised tuple ranking, referred as approach-2, see 3.5: This approach ranks patterns based on PMI. The clean set of patterns obtained after the supervised pattern induction are used to extract tuples. The tuples are scored using a supervised classifier. The features of the classifier include matching pattern frequencies and some diverse features including N-gram models.

3. A new method based pattern induction, supervised tuple ranking, referred as approach-3, see 3.5.2: This approach ranks patterns based on the proposed method. The high-scoring patterns extract tuples that are scored using a supervised classifier. The features of the classifier include matching pattern frequencies.

Section 5.1 evaluates approach-1 followed by a summary of how well the three questions `G3` are addressed. Section 5.2 evaluates approach-2 followed by a summary of how well the three questions `G3` are addressed. Section 5.3 evaluates approach-3 followed by a summary of how well the three questions `G3` are addressed. Finally, in section 5.4, we identify the best approach and discuss the insights gained from the experiments.

5.1 Supervised Pattern Induction, Supervised Tuple Ranking: (approach-1)

5.1.1 Supervised Pattern Induction

In this setting, we constructed seeds and induced patterns over Wikipedia for the relation `hasProperty` for the other desired relations there were inadequate seeds in Wikipedia. However, this was among the first experiments and therefore it was important to measure the effectiveness of using Wikipedia for pattern induction. Google N-grams were used as a corpus for extracting facts using the clean patterns.

From the Wikipedia corpus, we used 410MB of data to derive the seeds and seed patterns for the relation `hasProperty`. The number of seed tuples was restricted to 175 by ranking them based on their frequency and applying a threshold. For example, we find the following positive seeds (shown with example patterns as predicates):

```
are_always(rose,red),was(review,bad),are(creatures,dangerous),
was(performance,good)
```

Algorithm	Precision	Recall	F-measure	Kappa	Parameter Settings
C4.5	73.3	73.5	73.4	0.42	min. 12 instances per leaf, pruning
AdaBoost M1	74.4	74.5	74.4	0.45	50 iterations, Decision Stumps
Random Forests	70.5	71.0	70.7	0.38	26 features per tree, 100 trees
SVM	62.8	60.3	61.4		RBF kernel with $\gamma = 0.0001$

TABLE 5.1: Cross-Validation results

These seeds allowed us to discover 8,702 patterns, out of which 309 were ranked with a perfect score of 1.0. Another 2,401 patterns were given a near-zero score. This was attributed to the quality of the corpus, which retained and even parsed non-textual, Wikipedia-specific information like lists of categories. For instance, a pattern obtained was "*Oklahoma & texas*", as found in "*the .. artists oklahoma & texas local ..*" due to the seed `(artists,local)`. The classifier gave it a low score because the percentage of stop words was zero, the frequency of the seed was low, etc. Several of such incorrect patterns had high frequency because many pages contain category information. In such cases, the most helpful feature was the PMI between seed and pattern.

Other examples of good patterns extracted are: "*is possibly*", "*is practically*", "*is preferably a*", "*is presumed to be*", "*is primarily*", "*is probably*", "*is quite*", "*is rather*", "*is regarded as*", "*is relatively*", "*is roughly*", "*is said to be*", "*is seen as a*". It is apparent that many different variations are found in free text, more than one could anticipate by manually specifying patterns.

5.1.2 Supervised Tuple Ranking

To obtain the output tuples, we leveraged the newly found patterns over the Google N-Gram corpus. Learning algorithms were used to distinguish tuples likely to be correct from those likely to be incorrect. For training, we annotated 500 tuples (for each of 50 concept words, 10 random properties occurring in the n-gram data with any of the seed patterns). 66% of the training set received a positive label, the rest was negative. For each tuple, 314 features were computed, as specified in Section 3.6.2.

The feature vectors were used to obtain models with several different learning algorithms. We experimented with Adaptive Boosting M1 [Freund and Schapire, 1996], C4.5 Decision Trees [Quinlan, 1993], Random Forests [Breiman, 2001], and Support Vector Machines using LIBSVM [Chang and Lin, 2001]. Parameter tuning was performed using 10-fold leave-one-out cross-validation. Table 5.1 provides details on the parameter settings as well as the cross-validation results. Apparently, there are regularities in the features that rule-based systems capture more easily than other types of algorithms.

Algorithm	Precision	Recall	F-measure	Kappa
C4.5	91.3	90.7	91.0	0.82
AdaBoost M1	91.3	90.7	91.0	0.82
Random Forests	78.5	78.3	78.4	0.56
SVM	75.2	45.6	56.8	

TABLE 5.2: Classifier Scores on Test Set

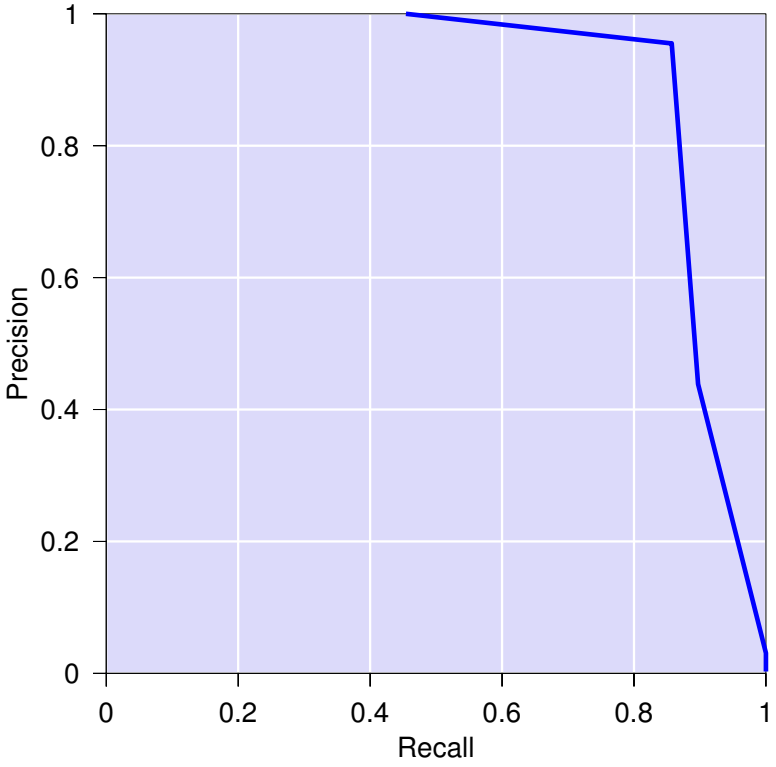

FIGURE 5.1: Precision-Recall Curve

For the final output, we decided to use C4.5, which gave the best results in our cross-validation setup, together with AdaBoost. Problems we encountered included multi-word expressions like "*pink panther*" leading to the tuple `(panther,pink)`. An additional evaluation was performed using a test set generated by annotating all properties found for the word "*flowers*". The experimental results are given in Table 5.2. Surprisingly, the results are even better than in the cross-validation, so perhaps attributes of flowers are easier to handle than for more abstract concepts. Figure 5.1 shows a precision-recall curve for this classifier, describing the natural trade off between the two. We see that we can still get a very high precision at a substantial level of recall.

Run over a 1GB (uncompressed) subset of the Google 5-grams, applying the 309 seed patterns, our system extracted 119,393 tuples in total.

Threshold	Precision	Recall
0.6	0.392	0.02
0.5	0.399	0.03
0.4	0.397	0.04
0.3	0.397	0.06
0.2	0.410	0.09

TABLE 5.3: Test Set results for WordNet with Lesk similarity

FIGURE 5.2: Tag cloud showing properties of flowers in ConceptNet

5.1.3 Comparison with other resources

We compare our system with other possible sources of CSK. WordNet 3.0 includes 639 instances of an `attribute` relation, however this relation does not convey information about the common sense attributes of concepts. Instead, it is used to specify what an adjective describes, e.g. "*unimportant*" describes "*importance*" and "*rich*" describes "*financial condition*". Apart from that, WordNet contains glosses and relations that provide rich information about relatedness between words, e.g. "*ball*" has a high relatedness with "*round*", so we investigated whether relatedness can be used to obtain properties. We relied on a version of the Lesk similarity measure [Banerjee and Pedersen, 2002] to compute the maximum normalized gloss overlap for all pairs of senses of the two words in each CSK tuple. The results in Table 5.3 show that gloss overlap, too, gives only a low recall and precision. Moreover, increasing the threshold does not lead to a higher precision, so correct properties do not necessarily have particularly high similarities.

The second resource we considered was ConceptNet[1]. For the concept `flowers`, we extracted all the `hasProperty` relations. The Figure 5.2 shows the CSK tag cloud generated for the concept `Flowers`. With respect to our test set, ConceptNet had a precision of 90.9% and a recall level of 9.8%. We noticed that properties in ConceptNet are fairly precise, but the number of attributes for many words is not that high compared to what can be obtained from existing text the Web. In addition, the scores do not necessarily represent the salience of properties.

Folksonomies and tagging systems can be regarded as naturally evolving, common sense organization structures. The Flickr API[2] provides a clustering of tags used for photographs on Flickr. Concepts like `flower` have related tags[3] such as `nature`, `garden`, `yellow`, `purple`,

[1] `http://conceptnet.media.mit.edu`
[2] `http://www.flickr.com/`
[3] `http://www.flickr.com/photos/tags/flowers/clusters`

FIGURE 5.3: Tag cloud showing properties of `flower` found by our system

`white`, `green`, `orange`. Being derived from manually assigned tags, this data is fairly precise. However, there are two problems. The first is that the clustering is based on co-occurrence and hence does not distinguish different types of relationships. Therefore, for our purposes, it is not always clear which adjectives apply to which nouns. For example, "*garden*" in `(flower, garden)` refers to a location where flowers are found and "*bokeh*" in `(flower, bokeh)` is a technical term related to photography, but not the concept flowers. Since Flickr has no other supporting text, it is expensive to compute the predicates. In our experiments, we found that out of 788 true tuples in our test set, only 40 tuples co-occurred in a Flickr cluster.

We also experimented with Delicious[4] for the concept of flowers. There were 22 related tags extracted from the delicious website, but none of them was relevant as a property. Examples include "*wedding*", "*tutorial*", "*photography*". Apparently, Flickr tags are more relevant as a source of attributes, especially for concepts with salient visual properties.

Figure 5.3 shows the tag cloud generated by our system for the concept of `flowers`. Due to a lack of space, we randomly chose 26 properties with scores ≥ 0.80 or ≤ 0.30 (the latter displayed with a smaller font). Our system extracts significantly more properties of the concept than currently provided by other sources. We also see that synonyms and variations are extracted as well, e.g. `attractive`, `beautiful`, `amazing`. The scores define the rank of the properties, e.g. `(flower,beautiful)` , `(flower,red)` are scored high, whereas `(flower,delicately)` is ranked low.

5.1.4 Summary

We summarize the approach by answering the G3 questions.

1. How effective is the pattern induction, in terms of correctness of the patterns and the number, quality patterns induced: The pattern induction was robust, but limited to only one relation `hasProperty`. Wikipedia proved to be a good resource for this relation, however, it is restricted in terms of seeds for other common sense relations mentioned in

[4] `http://www.delicious.com/`

Section 4.2. The number of patterns obtained were moderately large. To summarize, the technique for constructing seeds was restricted. The accuracy of the patterns was good, however, it relied on supervised learning. Therefore, scalability to more relations is an issue with this approach.

2. How effective is the tuple extraction and ranking, in terms of correctness of the tuples and the number of facts extracted: N-grams corpus proved to be very effective in terms of number of facts extracted, even though only a subset of the Google N-grams was used. This is an encouraging result coming out of the approach. The supervised learning strategy also proved effective, however we were restricted in terms of labeled data especially negative examples to learn from. In addition, comparison with existing resources like WordNet and ConceptNet shows the scale of our system is already better on the relation `hasProperty`.

3. In terms of the overall message of the thesis, how effectively do we address the two underlying themes of the thesis, (a) n-grams as a text corpus for information extraction: On one relation `hasProperty`, we demonstrate the effectiveness of our approach, (b) robustness of the system: the system is robust, but limited to only one relation `hasProperty`. Therefore, there is a need for scaling to more relations.

5.2 PMI Based Pattern Induction, Supervised Tuple Ranking: (approach-2)

5.2.1 PMI Based Pattern Induction

We chose 5 relations that we believe fulfill the conditions mentioned earlier in Section 3.1. These are listed in Table 5.4. We used the ConceptNet [Havasi et al., 2007] as a source of 100 tuples per relation, selecting the top-ranked 100 as seeds. Upon further inspection, we noticed that some of the seeds are not correct, despite having high scores in ConceptNet, e.g. we found tuples like `partOf(children,parents)` and `isA(winning,everything)` in ConceptNet. Such incorrect tuples stem from the crowd sourcing approach that ConceptNet has adopted for gathering knowledge, where just a few input sentences like "*winning is everything*" suffice to give the `isA`-tuple a high score. Fortunately, our approach is robust with respect to inaccurate seed tuples for pattern induction. Our approach later relies on the pattern statistics for hundreds or thousands of induced patterns (rather than just "<*X*> *is* <*Y*>") to assess the validity of tuples. The learning algorithm should automatically learn that a pattern like "<*X*> *is* <*Y*>" alone is too unreliable.

TABLE 5.4: Relations

Relation	Seeds	Patterns discovered	Labeled examples All	Labeled examples Positive
`isA`	100	2991	530	530 (18%)
`partOf`	100	3883	516	516 (40%)
`hasProperty`	100	3175	400	297 (74%)

TABLE 5.5: Patterns for `isA`

Pattern	PMI range
<X> and almost any <Y>	high
<X> and some other <Y>	high
<X> betting basketball betting <Y>	high
<X> online <Y>	high
<X> is my favorite <Y>	high
<X> shoes online shoes <Y>	high
<X> wager online <Y>	high
<X> is a <Y>	medium
<X> is a precious <Y>	medium
<X> is the best <Y>	medium
<X> or any other <Y>	medium
<X> , and <Y>	medium
<X> and other smart <Y>	medium
<X> and small <Y>	medium
<X> and grammar <Y>	low
<X> content of the <Y>	low
<X> or storage <Y>	low
<X> when it changes <Y>	low

Table 5.5 shows examples of top, middle, and bottom-ranked patterns for the `isA` relation. We see that PMI alone is not a reliable estimator of pattern goodness, as some top-ranked patterns are wrong. In particular, we notice the effects of web spam. Many of the best patterns actually had mid-range PMI scores. Patterns with very low PMI scores, are indeed either incorrect or useless (too rare). This shows why it makes sense to mistrust the PMI values when working with large-scale web data. Supervised approaches have a much better chance of determining which patterns are most reliable. Tables 5.6 shows similar results for the `partOf` relation. Overall, it is apparent that the patterns are very diverse. Using web-scale data, even fairly rare patterns can give us significant amounts of correct output tuples.

TABLE 5.6: Patterns for `partOf`

Pattern	PMI range
<X> with the other <Y>	high
<X> on your right <Y>	high
<X> of the top <Y>	high
<X> online <Y>	high
<X> and whole <Y>	high
<X> shoes online shoes <Y>	high
<X> wager online <Y>	high
<X> from the <Y>	medium
<X> or even entire <Y>	medium
<X> of host <Y>	medium
<X> from <Y>	medium
<X> on the <Y>	medium
<X> appearing on <Y>	medium
<X> of a different <Y>	medium
<X> entertainment and <Y>	low
<X> Download for thou <Y>	low
<X> affects your daily <Y>	low
<X> company home in <Y>	low

5.2.2 Labeled Data

For training and testing, we randomly chose tuples for the relations among all word pairs matching the patterns for the respective relation in the Google Web1T corpus. These tuples were then labeled manually as either correct or incorrect. The number of labeled examples is again given in Table 5.4. Note that these tuples are distinct from the initial seed tuples.

As the main learning algorithm, we used RBF-kernel support vector machines as implemented in LIBSVM [Chang and Lin, 2001]. In Section 5.2.4, we additionally report and evaluate results obtained with alternative algorithms.

For evaluation, we rely on 10-fold leave-one-out cross-validation, where the set of labeled examples is randomly partitioned into 10 equal-size parts, and then an average score is computed over 10 runs. In each run, a different part is reserved for testing, and the remaining 9 parts are used as the training set.

5.2.3 Accuracy and Coverage

Table 5.11 gives the overall results for using RBF-kernel support vector machines simultaneously relying on all available n-gram datasets. The table lists the average precision, average recall, and average F_1 score, i.e. the harmonic mean of precision and recall. Since, the recall

TABLE 5.7: Overall Results

Relation	Precision	Recall	F_1	Output per million n-grams[1]
isA	88.9%	8.1%	14.8%	983
partOf	80.5%	34.0%	47.8%	7897
hasProperty	75.3%	99.3%	85.6%	26180

1: the expected number of distinct accepted tuples per million input n-grams (the total number of 5-grams in the Google Web 1T dataset is ∼1,176 million)

was computed with respect to the union of all tuples matching any of the patterns in the Web-scale Google Web 1T dataset, even very low recall values mean that we can get hundreds of thousands of high-quality output tuples if we run over the entire dataset. We ran an additional experiment on a sample of 1 million input 5-grams to find the number of unique candidate tuples. The final column of Table 5.11 lists the number of expected number of distinct tuples accepted when running over a million n-grams. These figures were computed by counting the number of unique candidate tuples in the sample of one million n-grams, extrapolating the number of *true* tuples among those candidate tuples using the positive to negative ratio in the training set distributions, and then figuring in the recall values obtained by the classification model. The total number of 5-grams in the Google Web 1T dataset is over 1,176 million and additionally there are also 1,313 million 4-grams and 977 million trigrams, but a linear extrapolation to the entire dataset is not possible due to duplicates.

The overall results show that large numbers of high-accuracy output tuples can be obtained using n-gram data. Even greater numbers of tuples can be obtained by extending the range of patterns considered. We mentioned that an n-gram approach has less access to linguistic processing. For example, Hearst's study of extracting isA tuples from text used NLP techniques to filter the results. Using n-gram datasets, we can obtain similar results simply by relying on a greater range of patterns. Many linguistic features are implicitly captured via combinations of patterns.

Studying the results in further detail, we observe that there are great differences between the relations considered. The isA relation seems to be the hardest relation to extract with high coverage. It seems that many tuples are found that match a rather generic pattern like "<*X*> , *and* <*Y*>" and happen to be true isA pairs, but which do not occur frequently enough using more reliable patterns like the Heart patterns [Hearst, 1992]. The partOf relation is somewhat easier, and finally for the hasProperty relation, both precision and recall are very high.

TABLE 5.8: Differences between n-gram datasets (`partOf` relation)

Dataset	N-gram length	Source	Precision	Recall	F_1
Microsoft	3-grams	Document Body	58.5%	33.2%	42.3%
Microsoft	3-grams	Document Title	51.7%	29.8%	37.8%
Microsoft	3-grams	Anchor Text	57.3%	36.1%	44.2%
Microsoft	3-grams	Body / Title / Anchor	40.4%	100.0%	57.5%
Google Web 1T	3-grams	Document Body	55.9%	38.5%	45.6%
Microsoft	4-grams	Document Title	55.8%	34.6%	42.7%
Microsoft	4-grams	Anchor Text	49.6%	27.4%	35.3%
Microsoft	4-grams	Title / Anchor	40.3%	100.0%	57.3%
Google Web 1T	4-grams	Document Body	52.6%	43.3%	47.5%
Google Web 1T	5-grams	Document Body	48.1%	42.8%	45.3%
ClueWeb	5-grams	Document Body	51.7%	35.6%	42.2%
Microsoft	3-/4-grams	Body (3-grams only) / Title / Anchor	40.5%	98.1%	57.3%
Google Web 1T	3-/4-grams	Document Body	53.9%	42.8%	47.7%
Google Web 1T	3-/4-/5-grams	Document Body	58.7%	43.8%	50.1%

5.2.4 Detailed Analysis

We conducted more specific experiments to evaluate the role of the size of the n-grams and differences between datasets. Table 5.8 provides results for the `partOf` relation. The results are reported for the random forest algorithm [Breiman, 2001], as it balanced precision and recall more and thus produced results that are easier to compare than RBF-kernel SVMs. Overall, SVMs showed similar results and trends, however they sometimes preferred obtaining a high accuracy by classifying all examples as negative examples, leading to a precision and recall of 0.

Several interesting observations can be made. First of all, Microsoft and Google achieve similar results on the document body, but by combining information from document bodies, titles, and anchor texts, Microsoft outperforms both its own individual 3-gram datasets as well as Google's 3-grams. This shows that the titles and anchor texts offer complementary information that may differ from what is found in the document body for certain patterns. Essentially, this means that in the future we may see a more diverse range of n-gram datasets being used in different applications rather than a simple one-size-fits-all solution. For example, we may find specific n-gram datasets being offered, based only on news text, only on blogs, or only on documents found in the `.edu` domain. We may also see n-gram datasets derived from medical journals or even other modalities like speech transcripts.

While Microsoft's diversity is beneficial for the model accuracy, distributable datasets like the Google Web 1T dataset also have a number of significant advantages for information extraction, as they enable iterating over the data to induce patterns and again to determine the large set of candidate tuples. While equivalent results could also be obtained by using service-based n-gram datasets that support regular expression or wild card searches, the API for the Microsoft n-gram dataset does not currently have these features. One advantage of web services is that they can be easier to use than distributable datasets, as not much infrastructure is required to access a web service, while it is non-trivial to handle the large volumes of data provided in distributable n-gram datasets.

In Figure 5.4, we plotted the $\sum_{p \in P} f_{\phi(i)}(p(x, y))$ score from Equation 3.4 for the labeled `isA` examples for Microsoft document body 3-grams against the corresponding scores with Microsoft anchor text 3-grams and Google Web 1T document body 3-grams. We see that the scores from anchor text are quite different from the ones from the Microsoft body texts, while the correlation with titles and with the Google n-gram dataset is higher.

We further observe in Table 5.8 that different n-gram sizes show somewhat similar results, but that by combining different n-gram sizes improved results are possible. Nevertheless, it should be pointed that longer n-grams still are generally preferred when wildcard queries are possible, because an n-gram dataset can then also be used to find patterns with lengths $n' < n$. For instance, for a pattern of length 3 like "<*X*> *are* <*Y*>" we could use 5-gram queries like "*apples are green ? ?*" and "*? ? apples are green*". In our experiments, to enable a fair comparison with datasets that do not offer wildcard support, for each n-gram dataset we only considered patterns with a respective corresponding length of exactly n.

In an additional experiment, we evaluated different learning algorithms. Table 5.9 compares RBF-kernel SVMs [Chang and Lin, 2001], random forests [Breiman, 2001], Adaptive Boosting M1 [Freund and Schapire, 1996], and C4.5 Decision Trees [Quinlan, 1993] using all n-gram datasets together. The algorithms perform similarly for `hasProperty`, while for the other two relations LibSVM delivers a higher precision at the expense of a lower recall. Overall, there is no clear winner.

5.2.5 Summary

Notably, this approach was able to scale more number of relations because ConceptNet was used as a seed resource as opposed to approach-1 that does not employ ConceptNet. We summarize the approach by answering the G3 questions.

1. How effective is the pattern induction, in terms of correctness of the patterns and the number, quality of patterns induced: The pattern induction was robust for the five relations

TABLE 5.9: Alternative learning algorithms

	SVM	Random Forests	AdaBoost	C4.5
`isA`				
Precision	88.9%	25.5%	18.4%	26.8%
Recall	8.1%	49.5%	94.9%	49.5%
F_1	14.8%	33.7%	30.8%	34.8%
`partOf`				
Precision	80.5%	45.0%	39.4%	48.0%
Recall	34.0%	67.8%	80.3%	46.2%
F_1	47.8%	54.1%	52.8%	47.1%
`hasProperty`				
Precision	75.3%	74.4%	74.3%	74.6%
Recall	99.3%	98.7%	100.0%	100.0%
F_1	85.6%	84.8%	85.2%	85.5%

considered. ConceptNet proved to be a good resource for these relations. The number of reliable patterns obtained was large. However, the pattern ranking approach (PMI) was not very effective, and there is a need for a more robust approach.

2. How effective is the tuple extraction and ranking, in terms of correctness of the tuples and the number of facts extracted: These experiments provide much more insights that N-grams corpus is very effective in terms of number of facts extracted, even though only a subset of the Google N-grams was used. Further experiments must experiment with the entire Google N-gram corpus. The supervised learning strategy also proved effective; however, we were marginally restricted in terms of labeled data especially negative examples to learn from.

3. In terms of the overall message of the thesis, how effectively do we address the two underlying themes of the thesis, (a) n-grams as a text corpus for information extraction: The approach effectively demonstrates large-scale information extraction, additionally, by scaling to even more relations, the message can be further corroborated. (b) robustness of the system: the system is not as robust, especially with the pattern ranking approach, PMI was not effective.

TABLE 5.10: Patterns for IsA (not showing POS tags), with <S>, </S> as sentence begin/end markers, respectively

Top-Ranked Patterns (PMI)	Top-Ranked Patterns (θ)
Varsity <Y> <X> Men	*<Y> / <X>*
<Y> MLB <X>	*<Y> : <X> </S>*
<Y> <X> Boys	*<Y> <X> </S>*
<Y> Posters <X> Basketball	*<Y> - <Y> </S>*
<Y> - <X> Basketball	*<Y> such as <X>*
<Y> MLB <X> NBA	*<S> <X> <Y>*
<Y> Badminton <X>	*<X> and other <Y>*

TABLE 5.11: Overall Results

Relation	Prec˙	Recall	Final #Facts
AtLocation	57%	67%	13,273,408
CapableOf	77%	45%	907,173
Causes	88%	49%	3,218,388
CausesDesire	58%	61%	3,964,677
ConceptuallyRelatedTo	63%	60%	10,850,413
CreatedBy	44%	57%	274,422
DefinedAs	N/A	N/A	4,249,382
Desires	58%	65%	4,386,685
HasA	61%	64%	13,196,575
HasFirstSubevent	92%	86%	761,677
HasLastSubevent	96%	85%	797,245
HasPainCharacter	N/A	N/A	0
HasPainIntensity	N/A	N/A	0
HasPrerequisite	82%	55%	5,336,630
HasProperty	62%	48%	2,976,028
HasSubevent	54%	45%	2,720,891
InheritsFrom	N/A	N/A	106,647
InstanceOf	N/A	N/A	0
IsA	62%	27%	11,694,235
LocatedNear	71%	61%	13,930,656
MadeOf	52%	79%	13,412,950
MotivatedByGoal	53%	69%	76,212
PartOf	71%	58%	11,175,349
ReceivesAction	69%	70%	663,698
SimilarSize	74%	49%	8,640,737
SymbolOf	91%	64%	8,781,437
UsedFor	58%	49%	6,559,620

5.3 A New Method Based Pattern Induction, Supervised Tuple Ranking: (approach-3)

5.3.1 Extraction

We used up to 200 seeds with a score of at least 3.0 in ConceptNet for each relation, with automatic addition of plural variants of words. The algorithm was implemented using Hadoop for distributed processing. Table 5.10 shows examples of top-ranked patterns for the `IsA` relation in terms of PMI and our θ. Our analysis revealed several reasons why PMI is inadequate for Web-scale n-gram data:

- Influence of spam and boilerplate text: Large portions of the Web consist of automatically generated text, often replicated millions of times. PMI is misled by the high frequencies, whereas θ takes into account that such patterns only match few *distinct* seeds.
- Less control over the selection of input documents: In n-gram datasets, there are large numbers of rare patterns that only coincidentally match some of the seeds.
- Less linguistically deep information, less context information: For instance, word boundaries and parts of speech may not be clear. Our own approach thus combines evidence from multiple patterns for scoring rather than trusting occurrences of any individual pattern.

Figure 5.5 shows the power law behavior of the number of seeds per pattern, which leads to Equation 3.1. We empirically settled on a threshold of $\theta_{\min} = 0.6$, because lower thresholds were leading to overwhelming volumes of extraction data. This reduced the number of patterns to below 1000 per relation. In the fact extraction phase, we used the Stanford tagger to check part-of-speech tags when matching patterns.

5.3.2 Accuracy and Coverage

For the fact assessment, we used C4.5 Decision Trees with Adaptive Boosting (M1) to prune the output. Table 5.11 provides the final results. Precision and recall were computed using 10-fold leave-one-out cross-validation on the labeled sets T_i, which contained several thousand of human-supplied positive and negative examples from ConceptNet. For a small number of relations, there were insufficient seeds to find patterns or perform 10-fold cross-validation, but for most relations in ConceptNet, very encouraging results are obtained. Since the labeled sets are balanced, a random baseline would have only 50% precision. Additionally, we can opt to generate output of higher quality by trading off precision and recall and still obtain very large numbers

of output facts. Figure 5.6 provides the precision-recall curve for three relations. We additionally verified the quality by manually assessing 100 random samples each for `CapableOf` (64% accuracy), `HasProperty` (78%), and IsA (67% accuracy).

The last column provides the final output results after classification retaining only those facts with decision tree leaf probabilities greater than 50%. We see that our resource is orders of magnitude larger than ConceptNet. Note that with a little additional supervision, the quality can be improved even further, e.g. in an active learning setting.

5.3.3 Summary

This approach was able to address all the concerns expressed with the previous approaches mentioned in Section 5.1 and Section 5.2. In addition, it met the expectations of the system we wanted to achieve. Let us summarize the approach by answering the G3 questions.

1. How effective is the pattern induction, in terms of correctness of the patterns and the number, quality of patterns induced: Starting out with a large number of seeds from Conceptnet made the pattern induction effective and robust. The number of reliable patterns obtained was also large. Introduction of the novel pattern ranking strategy proved to overcome the limitations of PMI based pattern ranking approach. The pattern ranking was robust to all relations.

2. How effective is the tuple extraction and ranking, in terms of correctness of the tuples and the number of facts extracted: These experiments provide the massive potential of using N-grams. We were able to obtain tuples in excess of 200 million for the relations considered. Gaining from the insight of the previous experiments about the effectiveness of a Decision tree on our dataset, the supervised learning strategy proved effective. Our approach to synthetically generate negative example from Conceptnet, addressed the problem of inadequate negative examples to learn from.

3. In terms of the overall message of the thesis, how effectively do we address the two underlying themes of the thesis, (a) n-grams as a text corpus for information extraction: The approach effectively demonstrates large-scale information extraction on a large set of relations. (b) robustness of the system: the system is robust, especially with the pattern ranking approach where we overcome the limitations of pattern ranking based on PMI .

5.4 Conclusion of Evaluation

In summary, we provide some key observations from the experiments. The clear winner is approach-3 that utilizes a large number of seeds and provides a robust pattern ranking scheme. The number of tuples extracted is an impressive number exceeding 200 million.

The results from experimental approach-1 demonstrate that existing resources like WordNet are largely inadequate, and one can leverage the knowledge base for several interesting applications like similarity on flickr tags. This can help in semantic search in the future. In the subsequent experiments, we address these concerns, including, leveraging a large number of facts from an existing knowledge base.

The results from experimental approach-2 suggest that even at a very large scale more data is better data. We can obtain results that are more precise by moving towards even larger document collections. The ClueWeb collection, which is already based on over 500 million documents, does not give the same results as the larger Google Web 1T collection. The Microsoft dataset gives even better results. In related experiments, we noted that the cut-off in the Google n-gram corpus was limiting. For example, we investigated obtaining patterns for the `imports` and `exports` relations from YAGO [Suchanek et al., 2007],e.g. `imports(China,Coal)`. However, due to the cut-off we found only very few patterns. Another reason, of course, is that such relations are typically expressed using longer patterns, e.g. "*<X> is the main exporter of <Y>*".

With approach-3, we have introduced a framework for deriving a common sense fact database from ConceptNet in conjunction with Web-scale n-gram datasets based on tera-scale numbers of words. The results from experiments with approach-3 suggest that our pattern ranking approach is robust and outperforms PMI on all relations. Further, our pattern ranking approach is generic and applicable to other pattern-based IE systems operating on full text not necessarily n-grams. In terms of precision, with a little additional supervision, the quality can be improved even further, e.g. in an active learning setting. Our resource is orders of magnitude larger than ConceptNet, more than 200 times larger.

Our large-scale common sense knowledge-harvesting framework meets the requirements envisioned at the start of this thesis i.e. simple automatic construction, machine-comprehensible representation, extendible, accurate, and high coverage.

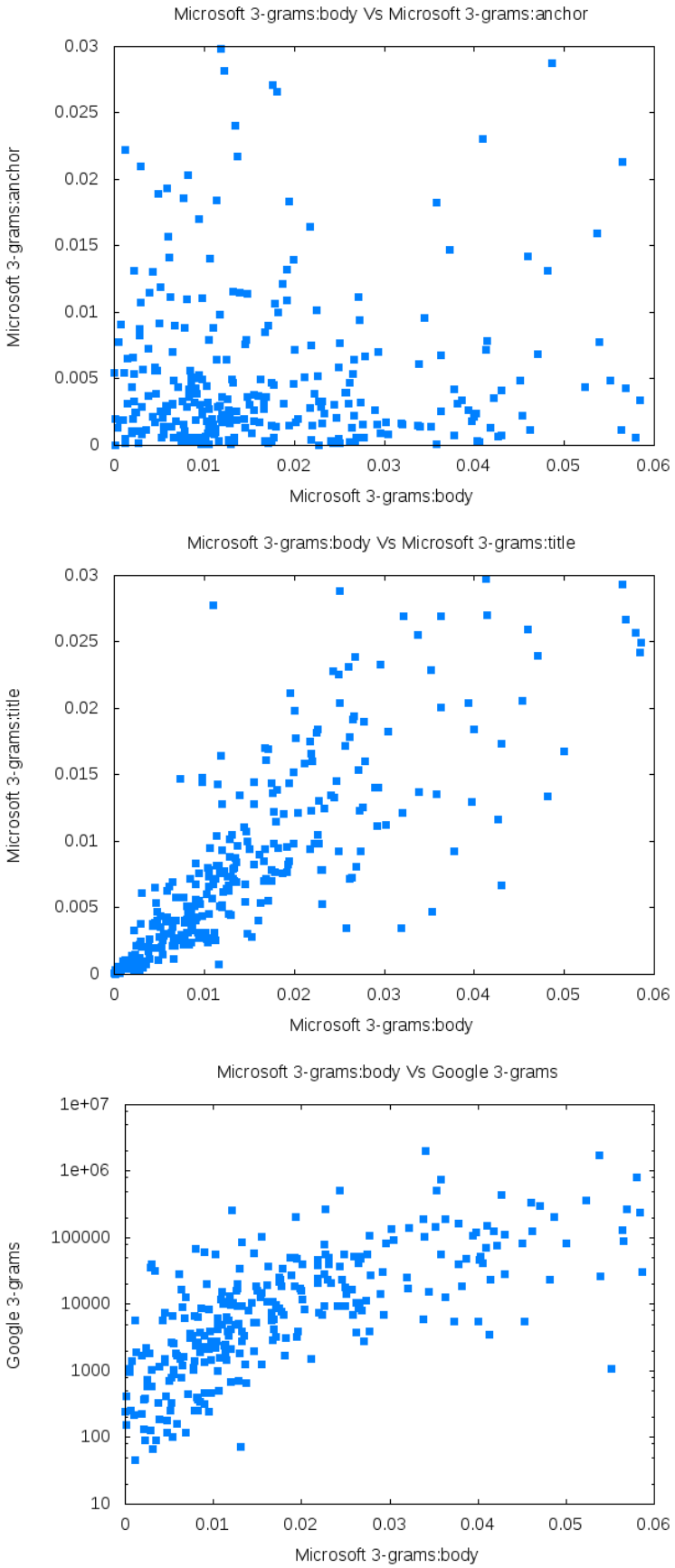

FIGURE 5.4: Plots comparing Microsoft Document Body 3-grams with Anchor 3-grams (above), Title 3-grams (middle), and Google Body 3-grams (bottom)

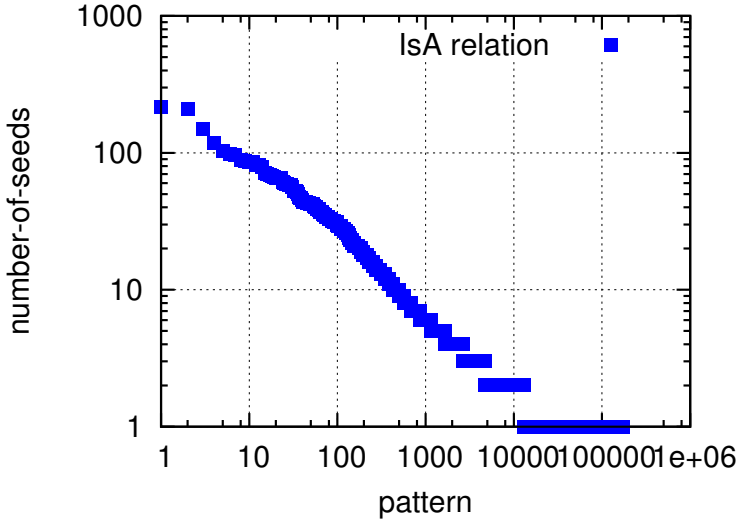

FIGURE 5.5: Number of seeds per pattern (log scale)

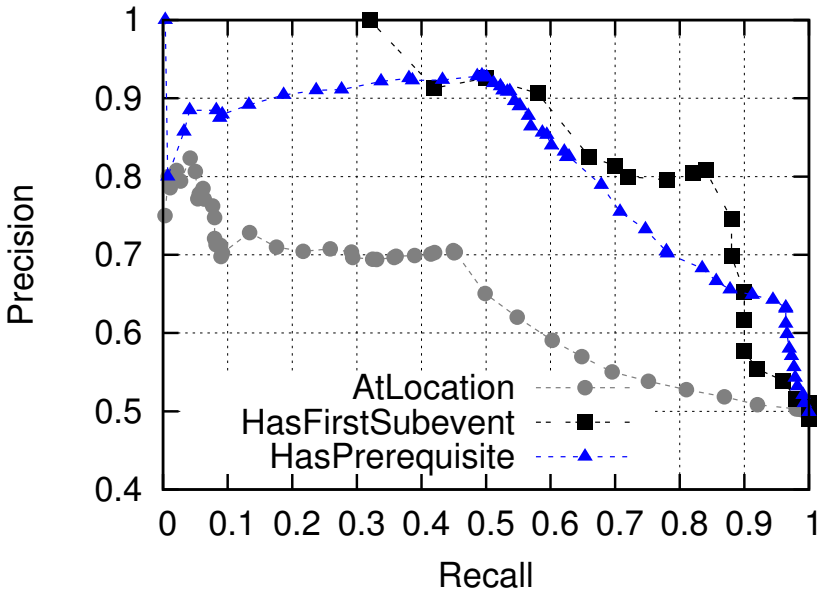

FIGURE 5.6: Precision-Recall Curve

Chapter 6

Conclusions

6.1 Conclusion

This thesis introduces a framework for automatically deriving a large-scale common sense fact database. The two key messages of the thesis are, 1) N-grams simulate a larger corpus and can be used for common sense knowledge harvesting, and, 2) Existing approaches must be modified in order to be robust to this setting, e.g. PMI performed sub-optimally. We introduce a novel pattern scoring strategy that carefully determines the patterns likely to lead to good extractions. This helps to overcome the robustness and scalability challenges of previous work.

The thesis successfully leverages an existing knowledge base to bootstrap with a large number of seeds and avoids drift during iterations of pattern based IE approaches. Additionally, we rely on a kind of weakly supervised approach for scoring the output facts unlike previous unsupervised outputs. The model is obtained from the input data, without any need for additional manual labeling.

Our study highlights the enormous possibilities as well as the limitations of following an approach of information extraction from N-grams. Not all types of information can be harvested from n-gram data; however, fortunately, for majority of the common sense relations only short context suffices between words and short entities. N-gram datasets allow us to go beyond the small corpora used in current information extraction. Although the overall recall is low relative to what is theoretically available on the Web, we are able to extend ConceptNet by many orders of magnitude (more than 200 times).

6.2 Future Work

We identify several directions that can leverage from our common sense knowledge bases as well as further advance the research in common sense knowledge harvesting.

Our knowledge base can be used in a large number of common sense based applications including query expansion, video annotation, faceted search, and distance learning, among other things. Moreover, the database can be leveraged to provide more fine-granular relations by viewing our knowledge base as a large matrix of tuples and patters. We performed Modularity Based Clustering [Clauset et al., 2004] using the bipartite graph between patterns and the extracted facts for the `HasProperty` relation and obtained a modularity value of 0.61. Values of at least 0.3 indicate significant community structure. Thus, the clustering result further encourages the usage of our database to provide fine-grained relations. For example, the `hasProperty` relation can be further divided into sub-relations: `hasColour`, `hasSmell`, `hasTaste`. Automatic grouping into sub-relations helps the machine to precisely understand the tuples and is better suited in applications that require such precise information.

Further, we can extension our framework to construct a Multilingual CSK Database because our system is mostly language independent, it may be feasible to extend to multiple languages and thereby increasing coverage even further. Another major step will be studying how the output of our system can be integrated into and supplement existing knowledge bases like DBpedia, YAGO, and Cyc. Integration of knowledge bases is recently gaining increased attention in the research community.

Our database will be made freely available under an open source license. We hope that it can pave the way for an entire ecosystem of novel intelligent applications.

List of Figures

1.1 Some desirable common sense knowledge about a flower 3

2.1 The duality that DIPRE harnesses 13
2.2 Snowball flow diagram 14

3.1 System Flow Diagram 25
3.2 Power law distribution: number of seeds per pattern (log scale) 32

4.1 Sample extractions to rank 45

5.1 Precision-Recall Curve 50
5.2 Tag cloud showing properties of flowers in ConceptNet 51
5.3 Tag cloud showing properties of `flower` found by our system 52
5.4 Plots comparing Microsoft Document Body 3-grams with Anchor 3-grams (above), Title 3-grams (middle), and Google Body 3-grams (bottom) 64
5.5 Number of seeds per pattern (log scale) 65
5.6 Precision-Recall Curve 65

List of Tables

1.1 Desired properties 4
1.2 Sample CSK Relations 5

3.1 Computing tuple frequency 34

4.1 The set of relations considered. 42

5.1 Cross-Validation results 49
5.2 Classifier Scores on Test Set 50
5.3 Test Set results for WordNet with Lesk similarity 51
5.4 Relations 54
5.5 Patterns for `isA` 54
5.6 Patterns for `partOf` 55
5.7 Overall Results 56
5.8 Differences between n-gram datasets (`partOf` relation) 57
5.9 Alternative learning algorithms 59
5.10 Patterns for `IsA` (not showing POS tags), with <S>, </S> as sentence begin/end markers, respectively 60
5.11 Overall Results 60

Bibliography

[Agichtein, 2005] Agichtein, E. (2005). Scaling information extraction to large document collections. *IEEE Data Eng. Bull*, 28:3–10.

[Agichtein and Gabrilovich, 2011] Agichtein, E. and Gabrilovich, E. (2011). Information organization and retrieval with collaboratively generated content. *AAAI tutorial*.

[Agichtein and Gravano, 2000] Agichtein, E. and Gravano, L. (2000). Snowball: extracting relations from large plain-text collections. In *Proc. 5th ACM Conference on Digital Libraries (DL '00)*, pages 85–94, New York, NY, USA. ACM.

[Altadmri and Ahmed, 2009] Altadmri, A. A. and Ahmed, A. A. (2009). VisualNet: Commonsense knowledgebase for video and image indexing and retrieval application. In *Proc. IEEE ICICS 2009, China*.

[Anacleto et al., 2006] Anacleto, J. C., de Carvalho, A. F. P., de Almeida Néris, V. P., de Souza Godoi, M., Zem-Mascarenhas, S., and Neto, A. T. (2006). How can common sense support instructors with distance education? In *Proc. SBIE 2006*.

[Atserias et al., 2008] Atserias, J., Zaragoza, H., Ciaramita, M., and Attardi, G. (2008). Semantically annotated snapshot of the English Wikipedia. In *Proc. LREC 2008*.

[Auer et al., 2007] Auer, S., Bizer, C., Lehmann, J., Kobilarov, G., Cyganiak, R., and Ives, Z. (2007). DBpedia: A nucleus for a web of open data. In *Proc. ISWC/ASWC*, LNCS 4825. Springer.

[Auer and Lehmann, 2007] Auer, S. and Lehmann, J. (2007). What have Innsbruck and Leipzig in common? extracting semantics from wiki content. *Lecture Notes in Computer Science*, 4519:503.

[Banerjee and Pedersen, 2002] Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proc. CICLing '02*, pages 136–145, London, UK. Springer-Verlag.

[Banko, 2009] Banko, M. (2009). Open information extraction from the web.

[Banko et al., 2007] Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web. In *Proc. IJCAI 2007*, pages 2670–2676.

[Banko et al., 2008] Banko, M., Etzioni, O., and Center, T. (2008). The tradeoffs between open and traditional relation extraction. In *Proc. ACL 2008)*, pages 28–36.

[Bast et al., 2007] Bast, H., Chitea, A., Suchanek, F., and Weber, I. (2007). Ester: Efficient search in text, entities, and relations. In *Proc. SIGIR*, Amsterdam, Netherlands. ACM.

[Borthakur, 2007] Borthakur, D. (2007). The hadoop distributed file system: Architecture and design. *Hadoop Project Website*.

[Brants and Franz, 2006a] Brants, T. and Franz, A. (2006a). Web 1T 5-gram Version 1. *Linguistic Data Consortium, Philadelphia*.

[Brants and Franz, 2006b] Brants, T. and Franz, A. (2006b). Web 1T 5-gram Version 1. *Linguistic Data Consortium, Philadelphia*.

[Brants et al., 2007] Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proc. EMNLP-CoNLL 2007*, pages 858–867.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[Brin, 1999] Brin, S. (1999). Extracting patterns and relations from the world wide web. *The World Wide Web and Databases*, pages 172–183.

[Cafarella, 2009a] Cafarella, M. (2009a). *Extracting and Managing Structured Web Data*. PhD thesis, University of Washington.

[Cafarella, 2009b] Cafarella, M. (2009b). Extracting and Querying a Comprehensive Web Database. In *Proc. 4th Biennial Conference on Innovative Data Systems Research (CIDR 2009), Asilomar, CA, USA*.

[Cafarella et al., 2005] Cafarella, M., Downey, D., Soderland, S., and Etzioni, O. (2005). Knowitnow: fast, scalable information extraction from the web. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 563–570. Association for Computational Linguistics.

[Cai and Lu, 2010] Cai, S. and Lu, Z. (2010). An improved semantic similarity measure for word pairs. In *2010 International Conference on e-Education, e-Business, e-Management and e-Learning*, pages 212–216. IEEE.

[Cao et al., 2008] Cao, Y., Cao, C., Zang, L., Zhu, Y., Wang, S., and Wang, D. (2008). Acquiring commonsense knowledge about properties of concepts from text. In *Proc. Fuzzy Systems and Knowledge Discovery 2008 (FSKD'08)*, volume 4.

[Carlson et al., 2010] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E. R. H., and Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In *Proc. AAAI 2010*.

[Carlson, 1982] Carlson, G. (1982). Generic terms and generic sentences. *Journal of Philosophical Logic*, 11(2):145–181.

[Chang and Lin, 2001] Chang, C.-C. and Lin, C.-J. (2001). LIBSVM: a library for support vector machines.

[Chklovski and Pantel, 2004a] Chklovski, T. and Pantel, P. (2004a). Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP*, volume 4, pages 33–40.

[Chklovski and Pantel, 2004b] Chklovski, T. and Pantel, P. (2004b). Verbocean: Mining the web for fine-grained semantic verb relations. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 33–40, Barcelona, Spain. Association for Computational Linguistics.

[Choi and Choi, 2008] Choi, D. and Choi, K. (2008). Automatic relation triple extraction by dependency parse tree traversing. page 23.

[Clark and Harrison, 2009] Clark, P. and Harrison, P. (2009). Large-scale extraction and use of knowledge from text. In *Proc. K-CAP 2009*. ACM.

[Clauset et al., 2004] Clauset, A., Newman, M., and Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(6):66111.

[Cruse, 1986] Cruse, D. A. (1986). *Lexical Semantics*. Cambridge University Press,, Cambridge, UK.

[Culotta and Sorensen, 2004] Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proc. ACL 2004*, volume 4.

[Downey et al., 2007] Downey, D., Schoenmackers, S., and Etzioni, O. (2007). Sparse information extraction: Unsupervised language models to the rescue. In *In Proc. of ACL*, pages 696–703.

[Etzioni et al., 2004a] Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D., and Yates, A. (2004a). Web-scale information extraction in KnowItAll: Preliminary results. In *Proc. WWW 2004*, pages 100–110.

[Etzioni et al., 2004b] Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D., and Yates, A. (2004b). Web-scale information extraction in knowitall:(preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, pages 100–110. ACM.

[Etzioni et al., 2005] Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165.

[Fellbaum, 1998] Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. The MIT Press.

[Fernau, 2005] Fernau, H. (2005). Algorithms for learning regular expressions. *LNCS*, 3734:297.

[Ferrucci et al., 2010] Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J., Nyberg, E., Prager, J., et al. (2010). Building watson: An overview of the deepqa project. *AI Magazine*, 31(3):59–79.

[Firth, 1957] Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis*, 1952-59:1–32.

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proc. ICML 1996*, pages 148–156, San Francisco. Morgan Kaufmann.

[Fundel et al., 2007] Fundel, K., Kuffner, R., and Zimmer, R. (2007). RelEx–relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365.

[Girju et al., 2006] Girju, R., Badulescu, A., and Moldovan, D. (2006). Automatic discovery of part-whole relations. *Comput. Linguist.*, 32(1):83–135.

[Gong et al., 2005] Gong, Z., Cheang, C. W., and U, L. H. (2005). Web query expansion by WordNet. In *Proc. DEXA 2005*, volume 3588 of *LNCS*, pages 166–175. Springer.

[Graff and Cieri, 2003] Graff, D. and Cieri, C. (2003). English Gigaword. *Linguistic Data Consortium, Philadelphia*.

[Hagiwara et al., 2009] Hagiwara, M., Ogawa, Y., and Toyama, K. (2009). Bootstrapping-Based Extraction of Dictionary Terms from Unsegmented Legal Text. *New Frontiers in Artificial Intelligence*, pages 213–227.

[Halevy et al., 2009] Halevy, A., Norvig, P., and Pereira, F. (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24:8–12.

[Harris, 1954] Harris, Z. (1954). Distributional structure. *Word*, 10(23):146–162.

[Havasi et al., 2007] Havasi, C., Speer, R., and Alonso, J. (2007). Conceptnet 3: a flexible, multilingual semantic network for common sense knowledge. In *Recent Advances in Natural Language Processing*, Borovets, Bulgaria.

[Havasi et al., 2009a] Havasi, C., Speer, R., and Alonso, J. (2009a). Conceptnet: A lexical resource for common sense knowledge. *Recent advances in natural language processing V: selected papers from RANLP 2007*, 309:269.

[Havasi et al., 2009b] Havasi, C., Speer, R., Pustejovsky, J., and Lieberman, H. (2009b). Digital intuition: Applying common sense using dimensionality reduction. *IEEE Intelligent Systems*, pages 24–35.

[Hearst, 1992] Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545.

[Horvath et al., 2009] Horvath, T., Paass, G., Reichartz, F., and Wrobel, S. (2009). A logic-based approach to relation extraction from texts. In *Proc. ILP 2009, Leuven, Belgium*.

[Hsu et al., 2006] Hsu, M.-H., Tsai, M.-F., and Chen, H.-H. (2006). Query expansion with ConceptNet and WordNet: An intrinsic comparison. In *Information Retrieval Technology*.

[Huang et al., 2010] Huang, J., Gao, J., Miao, J., Li, X., Wang, K., Behr, F., and Giles, C. L. (2010). Exploring web scale language models for search query processing. In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 451–460, New York, NY, USA. ACM.

[Islam and Inkpen, 2009] Islam, A. and Inkpen, D. (2009). Real-word spelling correction using google web 1tn-gram data set. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1689–1692, New York, NY, USA. ACM.

[Klein and Manning, 2003] Klein, D. and Manning, C. (2003). Fast exact inference with a factored model for natural language parsing. *Advances in Neural Information Processing Systems*, pages 3–10.

[Kushmerick et al., 1997] Kushmerick, N., Weld, D., and Doorenbos, B. (1997). Wrapper induction for information extraction. In *Proc. Int. Joint Conf. Artificial Intelligence*.

[Lapata and Keller, 2004] Lapata, M. and Keller, F. (2004). The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of nlp tasks. In *HLT-NAACL*, pages 121–128.

[Lenat, 1995] Lenat, D. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

[Li et al., 2008] Li, Y., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., and Jagadish, H. (2008). Regular expression learning for information extraction. In *Proc. EMNLP 2008*, pages 21–30. ACL.

[Lin et al., 2010] Lin, D., Church, K., Ji, H., Sekine, S., Yarowsky, D., Bergsma, S., Patil, K., Pitler, E., Lathbury, R., Rao, V., Dalwani, K., and Narsale, S. (2010). New tools for web-scale n-grams. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

[Lin et al., 2009] Lin, T., Etzioni, O., and Fogarty, J. (2009). Identifying interesting assertions from the web. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1787–1790. ACM.

[Liu and Singh, 2004] Liu, H. and Singh, P. (2004). ConceptNet: a practical commonsense reasoning toolkit. *BT Technology Journal*.

[Lyons, 1977] Lyons, J. (1977). *Semantics, vol. 1*. Cambridge University Press, Cambridge, UK.

[Mandhani and Soderland, 2008a] Mandhani, B. and Soderland, S. (2008a). Exploiting hyponymy in extracting relations and enhancing ontologies. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 3, pages 325–329. IEEE.

[Mandhani and Soderland, 2008b] Mandhani, B. and Soderland, S. (2008b). Exploiting Hyponymy in Extracting Relations and Enhancing Ontologies. In *Proc. WI-IAT'08*, volume 3.

[Matuszek et al., 2005] Matuszek, C., Witbrock, M., Kahlert, R., Cabral, J., Schneider, D., Shah, P., and Lenat, D. (2005). Searching for common sense: Populating Cyc from the Web. In *Proc. AAAI 1999*.

[Miller, 2009] Miller, D. (2009). A system for natural language unmarked clausal transformations in text-to-text applications.

[Milne et al., 2007] Milne, D. N., Witten, I. H., and Nichols, D. M. (2007). A knowledge-based search engine powered by Wikipedia. In *Proc. CIKM 2007*, New York, NY, USA. ACM.

[Niles and Pease, 2001] Niles, I. and Pease, A. (2001). Toward a Standard Upper Ontology. In *Proc. FOIS 2001*.

[Pantel and Pennacchiotti, 2006] Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proc. ACL 2006*. ACL.

[Pantel et al., 2004] Pantel, P., Ravichandran, D., and Hovy, E. (2004). Towards terascale knowledge acquisition. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, page 771, Morristown, NJ, USA. Association for Computational Linguistics.

[Paparizos et al., 2009] Paparizos, S., Ntoulas, A., Shafer, J., and Agrawal, R. (2009). Answering web queries using structured data sources. In *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, pages 1127–1130, New York, NY, USA. ACM.

[Partee, 1985] Partee, B. (1985). Dependent plurals are distinct from bare plurals. *U. Mass Ms*.

[Qin et al., 2009] Qin, P., Lu, Z., Yan, Y., and Wu, F. (2009). A new measure of word semantic similarity based on wordnet hierarchy and dag theory. In *Web Information Systems and Mining, 2009. WISM 2009. International Conference on*, pages 181–185. IEEE.

[Quinlan, 1993] Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.

[Reichartz et al., 2009] Reichartz, F., Korte, H., and Paass, G. (2009). Dependency tree kernels for relation extraction from natural language text. In *Proc. ECML/PKDD 2009: Part II*, page 285. Springer.

[Riloff, 1993] Riloff, E. (1993). Automatically constructing a dictionary for information extraction tasks. In *AAAI*, pages 811–816.

[Ritter et al., 2009] Ritter, A., Soderland, S., and Etzioni, O. (2009). What is this, anyway: Automatic hypernym discovery. In *Proceedings of AAAI-09 Spring Symposium on Learning by Reading and Learning to Read*, pages 88–93.

[Sarawagi and Cohen, 2004] Sarawagi, S. and Cohen, W. W. (2004). Semi-markov conditional random fields for information extraction. In *NIPS*.

[Schubert, 2002] Schubert, L. (2002). Can we derive general world knowledge from texts? In *Proc. HLT '02*.

[Schwartz and Gomez, 2009a] Schwartz, H. and Gomez, F. (2009a). Acquiring applicable common sense knowledge from the web. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, pages 1–9. Association for Computational Linguistics.

[Schwartz and Gomez, 2009b] Schwartz, H. and Gomez, F. (2009b). Acquiring applicable common sense knowledge from the Web. In *NAACL HLT Worksh. Un-/Minimally Superv. Learning of Lex. Sem.* ACL.

[Shi, 2007] Shi, H. (2007). Best-first decision tree learning. Master's thesis, University of Waikato, Hamilton, NZ. COMP594.

[Singh, 2002] Singh, P. (2002). The public acquisition of commonsense knowledge. In *Proc. AAAI Spring Symposium on Acquiring (and Using) Linguistic (and World) Knowledge for Information Access*.

[Sleator and Temperley, 1995] Sleator, D. and Temperley, D. (1995). Parsing english with a link grammar. *Arxiv preprint cmp-lg/9508004*.

[Snow et al., 2005] Snow, R., Jurafsky, D., and Ng, A. (2005). Learning syntactic patterns for automatic hypernym discovery. In *Proc. NIPS 2005*, volume 17, pages 1297–1304. Citeseer.

[Speer et al., 2009a] Speer, R., Krishnamurthy, J., Havasi, C., Smith, D., Lieberman, H., and Arnold, K. (2009a). An interface for targeted collection of common sense knowledge using a mixture model. In *Proc. IUI 2009*, pages 137–146. ACM.

[Speer et al., 2009b] Speer, R., Krishnamurthy, J., Havasi, C., Smith, D., Lieberman, H., and Arnold, K. (2009b). An interface for targeted collection of common sense knowledge using a mixture model. In *Proceedings of the 13th international conference on Intelligent user interfaces*, pages 137–146. ACM.

[Suchanek et al., 2006a] Suchanek, F., Ifrim, G., and Weikum, G. (2006a). Combining linguistic and statistical analysis to extract relations from web documents. In *Proc. KDD 2006*, pages 712–717. ACM New York, NY, USA.

[Suchanek et al., 2006b] Suchanek, F., Ifrim, G., and Weikum, G. (2006b). Leila: Learning to extract information by linguistic analysis. In *Proceedings of the ACL-06 Workshop on Ontology Learning and Population*, pages 18–25.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A core of semantic knowledge. In *Proc. WWW 2007*, New York, NY, USA. ACM Press.

[Suchanek et al., 2009] Suchanek, F. M., Sozio, M., and Weikum, G. (2009). Sofie: a self-organizing framework for information extraction. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 631–640, New York, NY, USA. ACM.

[Suh et al., 2006a] Suh, S., Halpin, H., and Klein, E. (2006a). Extracting common sense knowledge from wikipedia. In *Proceedings of the Workshop on Web Content Mining with Human Language Technologies at ISWC*, volume 6. Citeseer.

[Suh et al., 2006b] Suh, S., Halpin, H., and Klein, E. (2006b). Extracting common sense knowledge from Wikipedia. In *Proc. Workshop on Web Content Mining with HLT at ISWC*, volume 6.

[Tandon and de Melo, 2010] Tandon, N. and de Melo, G. (2010). Information extraction from web-scale n-gram data. In *Web N-gram Workshop*, page 7.

[Tandon et al., 2011] Tandon, N., de Melo, G., and Weikum, G. (2011). Deriving a web-scale common sense fact database. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

[Toutanova and Manning, 2000] Toutanova, K. and Manning, C. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. EMNLP/VLC-2000*, pages 63–70.

[von Ahn, 2011] von Ahn, Law, E. (2011). Human computation: Core research questions and state of the art. *AAAI tutorial*, pages 1–38.

[von Ahn et al., 2006] von Ahn, L., Kedia, M., and Blum, M. (2006). Verbosity: a game for collecting common-sense facts. In *Proc. CHI 2006*, pages 75–78, New York, NY, USA. ACM.

[Vyas et al., 2009] Vyas, V., Pantel, P., and Crestan, E. (2009). Helping editors choose better seed sets for entity set expansion. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 225–234. ACM.

[Wang et al., 2010] Wang, K., Thrasher, C., Viegas, E., Li, X., and Hsu, B.-j. P. (2010). An overview of microsoft web n-gram corpus and applications. In *Proceedings of the NAACL HLT 2010 Demonstration Session*, pages 45–48, Los Angeles, California. Association for Computational Linguistics.

[Weikum and Theobald, 2010] Weikum, G. and Theobald, M. (2010). From information to knowledge: harvesting entities and relationships from web sources. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems of data*, pages 65–76. ACM.

[Yap and Baldwin, 2009] Yap, W. and Baldwin, T. (2009). Experiments on pattern-based relation learning. In *Proc. CIKM 2009*, pages 1657–1660. ACM.

[Yates, 2007a] Yates, A. (2007a). *Information extraction from the web: Techniques and applications*. PhD thesis, Citeseer.

[Yates, 2007b] Yates, A. (2007b). *Information Extraction from the Web: Techniques and Applications*. PhD thesis, Citeseer.

[Yu and Chen, 2010] Yu, C. and Chen, H. (2010). Commonsense knowledge mining from the web. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*.