Relation Extraction using Distant Supervision, SVMs, and Probabilistic First Order Logic

Malcolm W. Greaves

CMU-CS-14-128

May 2014

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee
William Cohen, Chair
Tom M. Mitchell

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

This research was generously supported by Google Inc. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of Google Inc., Carnegie Mellon University, or any other entity.

Keywords:
Information extraction, Machine Learning, Natural Language Processing, Probabilistic First-Order Logic, Relation Extraction, Big Data, Large Scale Machine Learning, Support Vector Machines, Cost-Sensitive Learning

"Each new program that is built is an experiment. It poses a question to nature, and its behavior offers clues to an answer."

Allen Newell (1975)

CARNEGIE MELLON UNIVERSITY

Abstract

Computer Science Department
School of Computer Science

Master's of Science in Computer Science

Relation Extraction using Distant Supervision, SVMs, and Probabilistic First Order Logic

Malcolm W. Greaves

We are drowning in information and having difficulty finding knowledge: useful and actionable information. Recent studies estimate that humanity has stored in excess of 295 exabytes ($295*10^{18}$ bytes) of data. Much data is stored in the form of unstructured text, such as news articles, message boards and forums, texts, emails, status updates, tweets, and nearly a billion webpages.

In this thesis, we present a solution to extracting knowledge present in untold amounts of unstructured text. We define our problem as one of relation extraction: given a document, extract all instantiations of well-defined binary relations present in the text. To this end, we use distant supervision and a novel probabilistic first order logic system combined with co-reference resolution to identify candidate relation instances. These candidates are then classified by a series of cost augmented, soft-margin, binary Support Vector Machines to produce the final relation extractions. Results on a corpus of 5.7 million newswire articles over 27 different relations results in an across-relation, micro-averaged F1 of 42.02%. Results on a smaller, targeted search, consisting of 10 thousand documents, achieve F1 of 33.15%.

Acknowledgements

I stand of the shoulders of giants. For this, I am eternally grateful. I acknowledge all of those who have studied the relation extraction task and have disseminated their knowledge. Without the results compiled from a community of scientists, engineers, and researchers, this thesis work would not exist.

I want to give a special thanks to my thesis advisor, Professor William Cohen. William has been an absolutely excellent mentor and teacher. He taught me many lessons, ranging from extremely useful, practical algorithmic implementations to support on how to balance research and life and the mindset required for research. He has graciously given me his time through the years to help me become a more effective researcher and, ultimately, person. I also sincerely appreciate his financial generosity as a graduate student.

In addition to Professor Cohen, I want to thank Professor Tom Mitchell. Tom was the first person to give me a chance to learn about natural language processing, information extraction, and machine learning. As a first semester freshman, Tom invited me to sit-in on his research group meetings, where I became acquainted with the world of research. His seemingly care-free attitude mixed with his firm, directed, and intense focus have always inspired me. He gave me an excellent foothold to step into the world of research.

I would also like to thank both Bryan Kisiel and Kathryn Mazaitis. Through the years, both Bryan and Kathryn have given me their time and focus in teaching me about various research groups' software and processes. They also maintained and administered many servers that I used throughout this, and other, research. Without their support, my programs would have never computed a single bit.

Contents

List of Figures

List of Tables

Abbreviations

IE	**I**nformation **E**xtraction
RE	**R**elation **E**xtraction
ML	**M**achine **L**earning
NLP	**N**atural **L**angauge **P**rocessing
KBP	**K**nowledge **B**ase **P**opulation
LR	**L**ogistic **R**egression
SVM	**S**upport **V**ector **M**achine
NELL	**N**ever **E**nding **L**anguage **L**earner
ProPPR	**P**rogramming with **P**ersonalized **P**age**R**ank
POS	**P**art-**o**f-**S**peech
NER	**N**amed **E**ntity **R**ecognition
NE	**N**amed **E**ntity

Dedicated to the love of my life, Jamey Katherine Dixon. Through her unique blend of pragmatism, wisdom, and support, she has helped me grow into a more whole human being.

Chapter 1

Introduction

We are drowning in information. In a fairly recent study, Hilbert and López (2011) estimated that humanity has stored a total of 295 exabytes (EB) of information. The growth of digital information is staggering: from an estimated 2.6 EB in 1986, to 15.8 EB in 1993, to 54.5 EB in 2000, and 295 EB in 2007. And humanity is producing more data each and every year at an alarming rate. In the last two years we generated 90% of the total current sum of all electronic data. By not taking complete advantage of these tremendous amounts of data, Åse Dragland (2013) estimate that businesses and governments are missing the opportunity to save, and make, trillions of dollars. .

We are drowning in information and having difficulty finding *knowledge*: useful and actionable information. It is humanly impossible to find this knowledge on our own. Instead, we must use ideas and concepts from machine learning, information extraction, statistics, computer science, natural language processing, and computer systems as our aids.

This thesis work builds upon these ideas and charts a course to solve a specific big data problem. In this work, we study and present solutions to the problem of relation extraction in unstructured text. We start with a series of definitions on our task and problem scope. We then review the fields of information extraction (IE), machine learning (ML), and natural language processing (NLP) as they each pertain to our relation extraction task, including coverage of related work. Next, we discuss our relation extraction system from two perspectives. We first describe our system as a data processing pipeline. Second, we describe each component in-detail, discussing algorithmic and mathematical specifics. After describing our system, we present results from an empirical evaluation. We conclude with an analysis of results and discussion of our system in the larger context of the relation extraction task.

1.1 Defining the Relation Extraction Task

In our task, relations are semantic concepts that are true for a given set of entities.an **entity** is specific person, place, object, event, or abstract idea. Moreover, entities are unique: a piece of text may **mention** an entity and an entity may be referred to by many different literal strings. A **relation** r is a named tuple of the form $(R, e_1, e_2, \ldots, e_k)$. Each e_i is a distinct, but not necessarily unique, entity. A concrete example of a binary relation, where $k = 2$, is `per_title`, which represents the relationship between a person and her title, position, or responsibility. Titles range over `President, CEO, Congresswoman` etc. An example of a `per_title` **instance** is `per_title(Juanita Millender-McDonald, Congresswoman)`.

Entity and relation mentions exist in sentences and documents. In this work, a **sentence** is a sequence of one or more tokens that expresses an idea and ends with a period symbol. A **token** is a sequence of characters separated by whitespace or a period. And a **document** is a sequence of one or more sentences that all relate to a coherent topic or small set of topics. it is also useful to define a **corpus** as a collection of documents, all of which will have varying degree of similarity. In general, corpora are very large: on the order of hundreds of thousands or millions of documents.

To get a sense of the end output of a relation extraction system, consider the following two sentence document:

> Rep. Juanita Millender-McDonald, a seven-term congresswoman from southern California, died early Sunday of cancer. She was 68.

FIGURE 1.1: Example Sentence

Given these sentences, we would like to extract the following relation instances: `per_title(Juanita Millender-McDonald, congresswoman)`,`per_title(Juanita Millender-McDonald, Rep.)`, `per_place_of_residence(Juanita Millender-McDonald, southern California)` and `per_age(Juanita Millender-McDonald, 68)`.

In order to perform relation extraction, one needs a program that has an understanding of the semantics of language and of the relations in question. In this work, we use machine learning so that we can automatically learn how to perform this task without explicitly programming the solution. The **relation extraction learning task** is, given a set semantic relations with accompanying examples and a corpus with known instances of these relations, learn a model that performs the relation extraction task well. And finally, we define the **relation extraction task**:

Given a set of documents with an unknown number of mentions, a set of relations, and a learned extraction model over these relations, output all valid relation instances in the document set.

FIGURE 1.2: Definition of realtion extraction task

1.2 Contributions

The specific contributions of this thesis are:

- Review of information extraction, particularly focusing on distant labeling techniques for relation extraction.
- Creation of labeled training data from 5.7 million newswire documents across 27 different relations.
- Creation of distantly labeled training data from 10 thousand newswire and Internet documents across 37 different relations.
- Novel integration of deterministic co-reference resolution, named entity recognition, part of speech tagging, and probabilistic first order logic for relation candidate generation from unstructured text.
- Creation and presentation of a complete relation extraction system, released under a permissive open source license.
- Empirical results on newly constructed distantly labeled datasets using soft-margin cost-augmented binary Support Vector Machines with n-gram and k-skip n-gram features.

Chapter 2

Relation Extraction Review

We defined naturally the relation extraction (RE) task and the corresponding learning objective in section 1.1. In this chapter, we give a broad review of the many sub-tasks that are directly related to producing high-quality, web-scale relation extraction systems. We start with a history of the successes and advancements in IE. We summarize events from the 1970s until 2014. We then discuss the topic of evaluating IE systems. We focus on the motivation behind different evaluation metrics. Once we have a notion of how to compare systems, we describe the two opposing paradigms in IE: hand crafting extraction patterns using expert domain knowledge and learning extraction patterns using statistical machine learning. We then dive into the different methods used in modern work that are used to parse and featureize text, which is a critical component of any text based IE system. We conclude this chapter with a review of work related to this thesis. We cover distant supervision, web scale extraction, and successful relation extraction systems.

2.1 Brief History of Information and Relation Extraction

Research in information extraction dates back to the late 1970s (Andersen et al., 1992, Granger, 1977, Lehnert, 1977, Riesbeck and Schank, 1976, Schank, 1975). In 1975, Riesbeck and Schank (1976) published work on ELI, an English language interpreter. ELI was able to produce structured representations of the semantic information in stories. Granger's 1977 system "Foul-Up" was able to determine word meanings from context. Granger (1977) used a rudimentary parser and limited, hand-coded domain knowledge to build context specific definitions of unknown words. Lehnert's 1977 dissertation investigated question answering as a problem in natural language understanding. Lehnert (1977) proposed a theory of question answering that combined ideas from conceptual

information processing (Schank, 1975), human memory, and computation. Lehnert implemented this theory in two story understanding systems. In the mid-1980s, Andersen et al. (1992), working for the Carnegie Group, launched a commercial information extraction product, known as JASPER (Journalist's Assistant for Preparing Earnings Reports) JASPER was constructed for Reuters in order to provide real-time financial news to traders. It used a series of hand-crafted templates and heuristic procedures to produce facts from newswire text. JASPER was able to automate a tedious, mundane, routine task with an accuracy ranging from 61% to 96% on various labeled extraction tasks.

2.1.1 Message Understanding Conferences: 1987 - 1998

Information extraction and Natural Language Processing saw rapid growth starting in the late 1980s and early 1990s. Refinement of ideas, methods, tasks, and evaluation strategies allowed researchers to focus their collective effort. From 1987 until 1998, a series of competition based, conferences, known as the Message Understanding Conferences (MUC), assisted and led this advance (Grishman and Sundheim, 1996). The goal of each conference was to perform a well-defined information extraction task and invent solutions to push the state-of-the-art. MUC was supported heavily by the U.S. Defense Advanced Research Projects Agency (DARPA). As such, initial conferences had a heavy emphasis on information extraction that would be of use to the military. As the conferences evolved, they incorporated more civilian themes. The MUC topics are as follows:

- MUC-1 & MUC-2 (1987, 1989) : US Navy fleet operations messages
- MUC-3 & MUC-4 (1991, 1992): Reports on Terrorism in Latin America
- MUC-5 (1993): International joint ventures and circuit fabrication
- MUC-6 (1995): Articles on management changes
- MUC-7 (1998): Reports of satellite launches

FIGURE 2.1: List of MUC conferences with dates and topics

As Grishman and Sundheim (1996) report, the first MUC more more exploratory in nature. Competing teams used different output formats and there was no formal evaluation. Two years later, in the same domain, the conference organizers had formalized the task with a set of pre-defined templates. MUC-2 set the standard for information extraction tasks in this regard. Participants were given definitions of different classes of events. Each individual event had a corresponding template. Each template had several

slots that defined different types of information. Examples of slots included the type of event, the agent participating in the event, the time and location of the event. Combining a template with answers extracted from text created a piece of structured, factual information. Templates in the MUC-2 challenge had 10 different slots. An example template (from a later MUC) is as follows:

NAME :	"Coca − Cola"
ALIAS :	"Coke"
TYPE :	"Company"
LOCALE :	"Atlanta CITY"
COUNTRY :	"United States"

FIGURE 2.2: Standardized slot-filling template for Organization from MUC-6. Reproduced from Grishman and Sundheim (1996)

The successive MUCs followed this same template filling approach. In MUC-3 and MUC-4 participants had to construct template filling programs that parsed and extracted information from Foreign Broadcast Information Service reports of terrorist events in Latin America. In keeping with MUC's goal of advancing the state-of-the-art in information extraction, the conference organizers complicated the task and increased the number of slots from 10 to 18, and then to 24 for MUC-4.

MUC-5 introduced a substantially more difficult and complex task. The tasks involved extracting information about international joint ventures and electronic circuit fabrication from both English and Japanese language sources. In total, there were 11 templates using 47 different slots. Moreover, MUC-5 was the first conference to use nested templates to describe events in terms of sub-events. This recursive structure laid the groundwork for more advanced structured information extraction.

The emphasis for the next MUC was on developing portable, task independent systems. This was a significant change of pace for the Message Understanding Conference. Preparing data, research, design, and development of systems, and manual evaluation in previous MUC tasks had been labor intensive and expensive. In addition, previous MUCs resulted in systems with only "shallow" understanding of the text. To accomplish these goals, MUC-6 sought to (1) identify domain-independent algorithms and technology that had enough accuracy and reliability to be ready for immediate practical use; (2) develop highly portable systems; and (3) challenge participants to construct systems that demonstrated a thorough, "deeper" understanding of the textual information.

In following with (1), MUC-6 incorporated the task of Named Entity Recognition (NER). NER consists of identifying the names of all locations, people, organizations, and other

specialized entity types, such as times, dates, and currency, as they occur in text. To tackle (2), MUC-6 introduced standardized templates for several common event types, including people and organizations. See figure 2.2 for an example. The deeper understanding challenges in (3) were collectively known as SemEval ("Semantic Evaluation"). SemEval consisted of 3 subtasks: co-reference resolution, word sense disambiguation, and uncovering predicate-argument sentence structure.

The co-reference task is to determine which pairs (or if more, chains) of nounphrases in a text refer to one another. MUC-6 focused on the part-whole and set-subset co-reference relations. The specific word sense disambiguation task was to link each mention of a adjective, adverb, noun, and verb in a text with a specific meaning of the word in WordNet (Miller, 1995). Lastly, the objective of the predicate-argument structure recovery subtask was to construct a tree that syntacticly links the constituent members of a sentence using grammatical rules.

MUC-7 continued with these tasks while adding complexity and reinforcing the commitment to developing portable systems. MUC-7 included Chinese, Japanese, and English texts on the NER task (Chinchor, 2001). The other tasks remained relatively unchanged from their MUC-6 definitions.

2.2 Learning Relation Extractors

In section 2.1, we glossed over the details of the relation extraction systems. Roughly before the 90s, most RE systems used hand crafted extraction patterns and explicitly programmed rules that that used manually evaluated knowledge (Andersen et al., 1992, Granger, 1977, Grishman and Sundheim, 1996, Riesbeck and Schank, 1976, Schank, 1975). Even early successful commercial systems, such as JASPER (Andersen et al., 1992), were based upon this extremely time consuming, tedious, error prone, and ultimately fragile approach. As noted in section 2.1.1, one of the main emphases of MUC was on developing potable RE systems. So much so that the NER task was introduced in MUC-6 (Grishman and Sundheim, 1996).

In an effort to improve portability and flexibility of automated RE systems, there was an intriguing, fundamental shift in relation extraction research and development. In the late 80s and early 90s, applications of statistics to problems in computer sciences gave birth to a new field, machine learning. The fundamental insight underlying machine learning is to not explicitly program solutions, but rather, as Mitchell (1997) put it, to "construct computer programs that automatically improve with experience." The often-cited definition of learning algorithm is, "a computer program is said to learn from

experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." (Mitchell, 1997) In application to relation extraction, the definition is: T, the task, is to identify and extract structured facts that express a binary relation and P, the performance metric, is F1 on a held-out dataset.

Defining experience, E, for relation extractors is hotly contested. And the disagreement in the field stems from larger differences in the classes of machine learning algorithms. Each class has a different method for learning from data (i.e. gaining experience). In general, there are three different types of learning algorithms. *Unsupervised* learning algorithms build models from data that has not been annotated, i.e. the label information is not explicit in the data (Chapelle et al., 2006). On the other hand, *supervised* learning algorithms operate on data that has been explicitly annotated and labeled (Chapelle et al., 2006, Mitchell, 1997). This setting is usually considered easier as there is a a well-defined metric for performance: the model's agreement with labeled data.

Many real-world data, such as the overwhelming majority, if not all of, of Internet text, is unlabeled (Hilbert and López, 2011). Automatically evaluating performance on unlabeled data is difficult. Most often, unsupervised algorithms are evaluated on labeled data. Supervised algorithms usually perform better than unsupervised algorithms because, during learning, they can directly optimize their internal modeling through automatic evaluation. Supervised learning algorithms have the ability to compare performance to some "ground-truth" labeling and thus have the opportunity to be consistent with said labeling. However, as noted earlier, many data that is relevant to relation extraction is not explicitly labeled, rendering the advantages of completely supervised algorithms useless.

The third kind of learning algorithm is seen as a combination of the former two. *Semi-supervised* learning (SSL) algorithms construct models from a combination of labeled and unlabeled data. In SSL, algorithms are supplied with a small amount of labeled examples and a large number of unlabeled examples. Because human annotation of text for binary relations is a costly process, there is little labeled data applicable to the relation extraction task (defined in figure 1.1). In SSL, the objective is to learn a hypothesis that is consistent with the few labeled examples while using prior knowledge to incorporate the information locked away in the multitude of unlabeled examples (Chapelle et al., 2006). In a sense, semi-supervised algorithms attempt to combine the best of both worlds.

2.2.1 Distant Supervision

There are a multitude of semi-supervised learning algorithms. An extremely effective and scalable SSL for relation extraction is known as *distant supervision* (Banko et al., 2007, 2008, Carlson et al., 2010, Etzioni et al., 2004, 2005, Go et al., 2009, Hoffmann et al., 2011, 2010, Krause et al., 2012, McDonald, 2005, Min, 2013, Mintz et al., 2009, Mitchell et al., 2009, Nguyen and Moschitti, 2011a,b, Nigam et al., 1998, Pantel and Pennacchiotti, 2006, Purver and Battersby, 2012, Riedel et al., 2010, Rosenfeld and Feldman, 2007, Roth et al., 2014, Surdeanu et al., 2010, Takamatsu et al., 2012, Xu et al., 2013). Craven et al. (1999) are credited with the first application of distant supervision when they applied the technique to relation extraction in biomedical texts.

Distant supervision occurs in domains where unlabeled data is plentiful and there exists a source of structured labeled data. In relation extraction, the most common form of distant supervision is to heuristically align a knowledge base – a database of relation instances between entities – to entity mentions in text (Banko et al., 2007, Hoffmann et al., 2011, 2010, Mintz et al., 2009, Nigam et al., 1998, Riedel et al., 2010, Schmitz et al., 2012, Surdeanu et al., 2010, Takamatsu et al., 2012, Xu et al., 2013). Commonly used sources of publicly available knowledge bases include Freebase (Bollacker et al., 2008), YAGO (Suchanek et al., 2007), and DBPedia (Auer et al., 2007), which is derived from Wikipedia infoboxes.

An active area of research is in these alignment heuristics. The most common heuristics involve finding all sentences that have a matching relation and entity pair mention, then using these matches as positive examples for the relation (Mintz et al., 2009, Roth et al., 2014, Schmitz et al., 2012, Surdeanu et al., 2010). For each record $(r, (Q, A))$ in the knowledge base, these distant supervision systems query the corpus for sentences containing Q `AND` A. The returned sentences are then featurized and turned into positive examples for relation r. As a concrete example, consider the triple (`per_title`, (`Jimmy Carter`, `President`)). When querying the corpus, we might retrieve the following sentence:

President Jimmy Carter pardoned Jefferson Davis in 1978.

FIGURE 2.3: Example of matched sentence in distant supervision

Under the widely held assumption that a sentence that contains an entity pair (Q, A) that is listed in the knowledge base (KB) under relation r is expressing the relation r, we would use the sentence in figure 2.3 as a positive example for the relation `per_title`. It is also possible to consider this sentence as a positive example for another relation under the same entity pair. For example, with the relation `per_political_office` as in (`per_political_office`, (`Jimmy Carter`, `President`)). This is an example of

overlapping labels, which, sometimes, can pose problems for distant supervision. Given two mutually exclusive relations, r_1 and r_2, it is not true that a single sentence s that has the entity pair (Q, A) is a positive example for both r_1 and r_2. Moreover, it is not even guaranteed that (Q, A) in s express either r_1 or r_2. Despite these theoretical problems which could lead to noisy training data, this straightforward knowledge base alignment heuristic is extremely effective in practice, resulting in state-of-the-art relation extraction systems (Mintz et al., 2009, Roth et al., 2014).

Riedel et al. (2010) and Hoffmann et al. (2011) flip a key assumption of the KB alignment distant supervision approach on its head. While they still do a heuristic KB alignment, they assume that a relation and entity pair is expressed *at least once* in a corpus. In contrast, the works described above assume that every sentence with that has (Q, A) for the KB tuple (r, (Q, A)) express relation r. From this different assumption, both sets of authors develop a factor graph model that is able to effectively capture this assumption on a per relation and entity pair basis. In their models, each entity pair embedded in a sentence is modeled by a relation variable for all possible relations that the entity pair could participate in. This graphical model also includes variables for different lexical and syntactic features. Finally, they define a conditional probability distribution over this factor graph's variables, which yields a distribution over these sentences with embedded entity pair mentions. This distribution allows one to determine which sentences are most likely to express a given relation. In turn, this yields training data that is more internally consistent and less noisy. Hoffman et al. extend the model proposed by Ridel et al. to handle overlapping relations. They accomplish this through a different factorization and probability distribution.

A different distant supervision approach takes a few cues from the discipline of information retrieval (IR). In IR, the objective is to find a set of documents that are *relevant* to a query (Manning et al., 2008). The common approach in IR systems is to learn this query to document set mapping by learning a document ranking function. Then, given a query, this function can produce a list of documents in decreasing relevance to the query. The common approach to learning this function is through relevance feedback, which can be explicit (a list of queries with known relevant documents) or implicit (noting which documents are pursued after a user submits a query). In practice, a medium between these two is used; known as pseudo-relevance feedback (Manning et al., 2008). Pseudo-relevance feedback involves assuming that the top k (for small values of k, such as 10 - 50) documents returned from the function are relevant, using the words of the documents to expand the query's scope (known as *query expansion*), and then re-performing the search with the expanded query and returning the most relevant k documents.

Xu et al. (2013) use a modified notion of pseudo-relevance feedback in order to distantly label and extract novel relation mentions. In their work, the authors assume that sentences that are relevant to a individual relation and entity pair contain entity pairs that are relevant to the relation in question. In turn, these entity pairs are used in the query expansion step to find more sentences. These sentences are then positively labeled and used as examples for the extraction model.

Another effective and popular distant supervision scheme is known as bootstrap learning (Banko et al., 2008, Brin, 1999, Carlson et al., 2010, Etzioni et al., 2004, 2005, Jones et al., 1999, Krause et al., 2012, Mitchell et al., 2009, Pantel and Pennacchiotti, 2006, Schmitz et al., 2012). Instead of a heuristic knowledge base alignment, the *bootstrap learning* paradigm iterates between learning extraction patterns from a set of labeled examples and adding new examples to this set using the previous iteration's extraction patterns. Bootstrap learning begins using a small set of labeled examples, known as *seeds*. These seed examples are manually annotated by humans and usually on the order of 10 per relation. These examples usually consist of an entire annotated sentence or document. That is, the example has all of the information that the relation extractor would have available to it at test time, with the addition of a human annotation specifying the specific relation present between a specific entity pair.

The first iteration consists of learning relation extraction models from these seed instances. This provides the models with a set of extraction patterns, i.e. features that are able to recover the relation and entity pair mention. The updated model is then run over the entire corpus. This step produces many extractions, which are most nearly guaranteed to contain false positives. A critical decision is in how to manage these newly extracted instances. Most methods take the approach of passing these extractions into a knowledge aggregator that is able to use constraints between relations and other information in order to eliminate spurious extractions (Carlson et al., 2010, Mitchell et al., 2009). Other approaches use grammars and probabilistic grammar models to rank extractions and filter incorrect extractions (Etzioni et al., 2004, Schmitz et al., 2012). Simpler approaches involve static thresholds, such as "accept the top k newly extracted instances for each relation." These thresholds often work well enough in practice (Carlson et al., 2010).

A common critique of bootstrap learning systems is that they are prone to semantic drift (Curran et al., 2007, McIntosh and Curran, 2009). *Semantic drift* is the condition of extracting new relation examples that deviate so greatly from the seed examples that they are no longer related. This causes the semantics of the set of extracted instances to become diluted and noisy. And due to the iterative nature of bootstrapping methods,

using incorrect examples as justification for extracting more examples leads to an ever-increasing amount of semantic drift. In practice, semantic drift can be devastating. Often the solution is to employ human annotators to prune incorrect extraction patterns and examples from the set of learned knowledge. However, Curran et al. (2007) and McIntosh and Curran (2009) provide algorithms that aim to automatically detect and negate semantic drift. In general, these algorithmic solutions exploit domain knowledge about the relations in order to constrain their definitions and thus the types of extraction patterns that are used during the bootstrapping process. For example, Curran et al. rely on mutual exclusion between relations in order to constrain the process. McIntosh and Curran present a distributional similarity filter which rejects extractions if the extraction is more similar to other recently extracted examples than it is to the seed examples. In practice, McIntosh and Curran find that their method can improve precision on later iterations by as much as 10%.

Chapter 3

Data

This thesis work germinated from an entry in the 2012 KBP Slot Filling task (Ji et al., 2010). In this competition, we had access to a dataset of many news articles and message boards. In addition, we had a well-defined list of semantic relations with labeled examples that were known to occur in the dataset. It was natural to use these data to further our own research.

However, before embarking on this thesis, we thought it crucial to understand this data. Before making assumptions about how the data is distributed and the patterns that are present within it, we sought to get a sense of the signals and information present in the text. As such, we decided to perform manual inspection of many document samples.

We read through hundreds of documents from the different text sources of this corpus. Immediately, we observed disparaging differences in quality between these sources. The documents sampled from the news article sources consisted of significantly higher-quality writing than the documents sampled from the message board sources. On the whole, we noted that the news articles had many desirable properties. The articles had few, if any, grammatical or spelling errors. Their sentences were well-constructed and effectively communicated facts. The documents followed, to some extent, over-arching patterns, which gave us hope that we might be able to successfully extract meaningful and useful information.An example of such a sentence is in figure 3.1.

"A. Sukrisno, the former ambassador to Vietnam and Romania, and Jawoto, a former ambassador to Beijing, are both living in exile in Amsterdam."

FIGURE 3.1: Example of a desirable sentence. Importantly, it is factual; it conveys useful information about two people. It is a relation mention and thus useful for relation extraction.

In contrast, the articles fetched from message boards were unsatisfactory. An example is in figure 3.2. The informal nature of message boards leads to many undesirables.

"Boy, the Democrats sure did try to shoot the messenger on this one huh?"

FIGURE 3.2: Example of an undesirable sentence. Not factual with high emotional content. It is not a relation mention and thus not useful to our task.

Responses were usually not as well written as news articles. Sentences were littered with spelling and grammatical mistakes, incoherent ramblings, and unknowable references. We often observed that responses would refer to other information that was not on the message board. We thought it would add an additional, unnecessary layer of complexity to our system if we had to perform extensive dereferencing from other data sources in order to use the message board data. In the end, we concluded that the entirety of the message board data was not to be trusted and used within our relation extraction system. In retrospect, this seems like a natural conclusion: it is difficult to extract valuable information from a comment thread crafted by an angry mob!

Before this course-grained document filtering, there were 218,223,253 sentences in 32,25,591 documents. After, there were 7,733,089 sentences in 5,726,579 documents. Interestingly, after filtering, 78% of these documents contain only a single sentence.

3.1 Relations and Labeled Data

In table 3.1, we list all of the relations from the TAC-KBP 2012 slot filling challenge (Ji et al., 2010). We use these relations in experiments in chapter 5. Due to some problems arising from label sparsity and distant labeling, we do not use all of these relations. Rather we use two subsets of these relations, which we describe in our experiments chapter (chapter 5).

Although the semantics of the relations used in TAC-KBP challenges stayed the same between years, some of the relation names did not. Therefore, we manually mapped every relation name from 2009-2011 to the 2012 TAC-KBP names.

We use the manually annotated assessments from the 2009-2012 TAC-KBP results as our knowledge base. There are 3,862 positive examples across all relations

Relation	**Example**
org_alternate_names	(American Bar Association, ABA)
org_city_of_headquarters	(International Crisis Group, Brussels)
org_country_of_headquarters	(Awami League, Bangladesh)
org_date_dissolved	(American Basketball Association, 1976)
org_date_founded	(Lashkar-e-Taiba, 1989)
org_founded_by	(United National Congress, Basdeo Panday)
org_member_of	(Blackburn Rovers, Premier League)
org_members	(American Beverage Association, Susan Neely)
org_parents	(USA Network, NBC Universal)
org_political_religious_affiliation	(Focus On The Family, Evangelical)
org_shareholders	(Arsenal Football Club, Daniel Fiszman)
org_subsidiaries	(Carnival Corporation, Princess Cruises)
org_top_members_employees	(Pentax Corporation, Fumio Urano)
per_age	(Ellen Degeneres, 56)
per_children	(Juanita Millender-McDonald, Keith McDonald)
per_cities_of_residence	(Kelly Cutrone, New York City)
per_city_of_death	(Irene Kirkaldy, Gloucester)
per_countries_of_residence	(Jo Ann Davis, United States)
per_country_of_birth	(Susan Boyle, Britain)
per_date_of_birth	(Jane Bolin, 1908-April-11)
per_date_of_death	(Theodor Kollek, 2007-January-02)
per_employee_of	(Jennifer Dunn, IBM)
per_member_of	(Gilbert Gude, Army Medical Corps)
per_origin	(Steven Derounian, Armenian)
per_parents	(Beverly Sills, Shirley Silverman)
per_siblings	(Spencer Pratt, Stephanie Pratt)
per_stateorprovinces_of_residence	(Jake Pavelka, Texas)
per_title	(Jefferson DeBlanc, Colonel)

Table 3.1: TAC-KBP 2012 Relations, each with one real example. We use subsets of these relations in our experiments, detailed in chapter 5.

Chapter 4

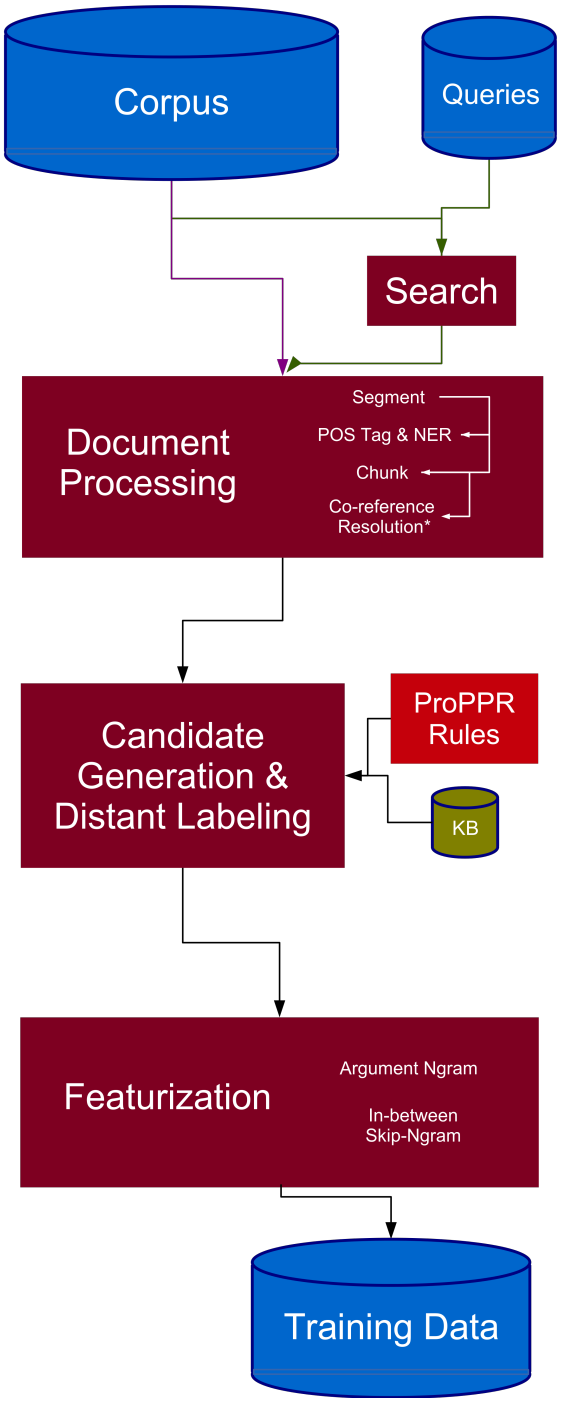
Methods

Below, we detail our relation extraction methods. We describe our system from two perspectives. We give a "bird's eye view" and illustrate the relation extraction machinery as a data processing pipeline. We detail the inputs and outputs of each component and map the links between each component. We also describe the algorithms behind each component in detail.

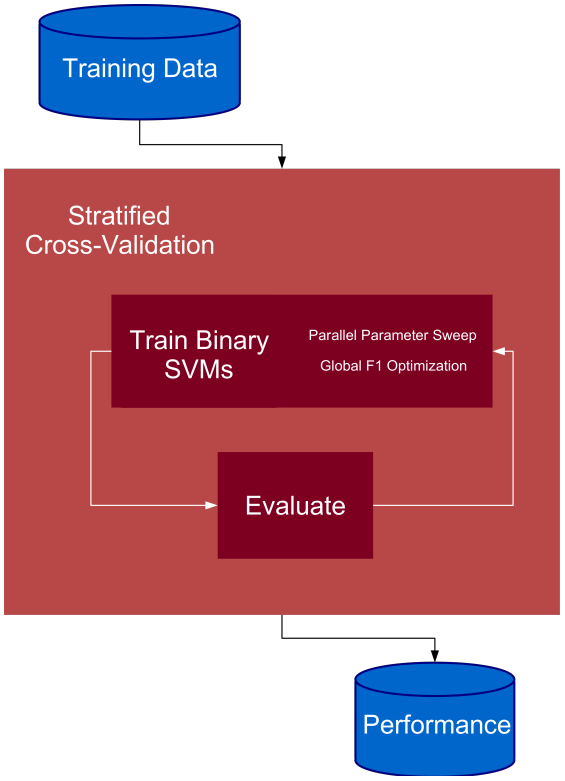
4.1 Relation Extraction Pipeline

Our system is logically split into two pipelines. The first pipeline takes as input a set of documents and outputs training data for our relation extraction models. This pipeline executes standard text processing algorithms (section 4.1.1), generates candidates using a probabilistic first-order logic and distant supervision (section 4.1.2), and constructs features for these candidates to make training data for the evaluation pipeline (section 4.1.3).

We discuss the second pipeline in section 4.2. This pipeline trains and evaluates our relation extraction models, optimizing for F1 on held out testing sets.



(A) Distant Labeling Pipeline.



(B) Train and Evaluation Pipeline. We train a separate binary Support Vector Machine for each relation. We sweep through positive misclassification costs and output the model that achieves the highest average F1. See section 4.2 for detail.

FIGURE 4.1: Relation extraction pipeline. Figure 4.1a shows the process of creating training data from a corpus of unstructured text documents. Our pipeline is able to generate, distantly label, and featurize candidates from either an entire corpus or from a targeted search. However, due to computational constraints, we are only able to perform co-reference resolution and use the ProPPR candidate generation rules on the smaller document set that is output from the targeted search.

4.1.1 Document and Sentence Processing

The first stage of the distant labeling pipeline performs a series of text processing steps. These steps serve to add structure to the document's text. Different sub-components of this system annotate the individual tokens in the document. These annotations transform unstructured text into a representation that is suitable for candidate generation.

Although the document processing component only accepts a set of documents as input, we have two distinct methods for feeding this component's input. The first method is to completely process an entire corpus. This mode is suitable for large-scale relation extraction. Due to our modular architecture, we are able to process documents in an embarrassingly parallel fashion. This nearly linear speedup drastically reduces the overall running time of document processing.

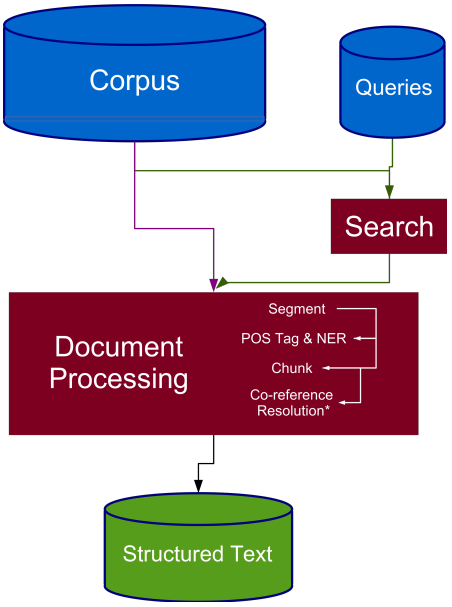


FIGURE 4.2: The document processing component annotates a set of unstructured text documents.

The second method is to use a list of queries to perform a targeted search over the corpus. By only returning the k most relevant documents for each query, we are able to focus our document processing efforts on a drastically smaller set of documents, which is crucial for practical document-wide co-reference resolution.

As stated earlier, this component performs a sequence of algorithms on unstructured text, each of which adds additional structure to the original text. The first such algorithm is sentence segmentation. We identify and index each sentence present in the document. Each sentence is assigned a unique identifier, which is the concatenation of the document's ID and the sentence's order within the document (e.g. sentence 0, 1, 2, ...). After this step, we tokenize each sentence and assign Part-of-Speech (POS) and Named Entity (NE) tags to each token. Next, we *chunk* tokens into phrases. We use a simple and efficient chunking algorithm that does a single pass through the tokens, chunking any two successive tokens if and only if the tokens have the same NE tag. See figure 4.3 for an example tagged and chunked sentence. Note that in the example we are interested in finding *nounphrases*: contiguous sequences of tokens that act as a single noun. Assigning POS tags and performing NER allows us to find these nounphrases in text.

The final text processing step is to perform document-wide co-reference resolution. This resolution gives us the ability to generate candidates that span multiple sentences. See section 4.1.2 for a detailed explanation of how we incorporate co-reference information in our candidate generation.

```
General/NNP Paul Eaton/NNP/PERSON said/VBD/ George Bush/NNP/PERSON is/VBZ
  holding/VBG our/PRP soldiers/NNS hostage/NN to/TO his/PRP ego/NN ./.
```

FIGURE 4.3: Example of a text-processed sentence. Note the POS tags (e.g. proper noun `NNP` and plural noun `NNS`). Also note that the `PERSON` NE tag allows us to form distinct chunks for both "Paul Eaton" and "George Bush."

We use the robust NLP tools provided by the Stanford CoreNLP library in order to perform segmentation, tokenization, POS tagging, NER, and co-reference resolution (Finkel et al., 2005, Lee et al., 2011, Toutanova and Manning, 2000).

The CoreNLP sentence segmenter and tokenizer is a finite state automaton. The POS tagger used in the library is a log-linear model with local and non-local features. The model is estimated using the maximum entropy method. The NER is a Conditional Random Field (CRF) that incorporates local and non-local features as well. The co-reference resolution engine is a sieve-based approach. The engine successively applies deterministic models, in order of increasing precision, on the document in order to determine co-reference.

To perform our search over the documents, we use the Apache Lucene search engine (`http://lucene.apache.org/`).

4.1.2 Candidate Generation and Distant Labeling using Probabilistic First-Order Logic and Co-reference Resolution

In the data, we often observed sentence constructions where the query of interest is mentioned in the sentence preceding the answer. Usually a pronoun (or sometimes another noun, such as an acronym or other alias) in the answer sentence would refer to the query in the preceding sentence. We immediately noted that this could be a potential recall problem: a per-sentence based relation extraction system would be completely unable to answer this type of query.

To circumvent this issue, we adopt a novel candidate generation approach that uses a highly scalable probabilistic first-order logic system known as ProPPR (Wang et al., 2013). ProPPR uses prolog rules, atomic facts, and a novel graphical representation to perform probabilistic inference. The system represents the logic program as a directed

graph. Nodes in the graph are logic program states, i.e. a rule with some number of satisfied variables and a partial mapping of variables to values. Edges between logic program states correspond to invoking a specific inference rule. A complete proof is sum of all paths through the graph from a rule consisting of all variables (the *query*) to a rule consisting of no free variables. ProPPR uses an approximate of PageRank algorithm in order to probabilistically prove rules. Moreover, ProPPR uses a local grounding technique in order to construct this graph. The particular algorithm that ProPPR uses can construct such a graph in time that is not proportional to the size of its input. This fact makes ProPPR an ideal choice for relation extraction, where our input data is naturally large.

We construct inference rules in ProPPR that allow us to generate candidates from both within-sentence entity pairs and from entity pairs that share a link to a common referent. We describe all of these rules in detail below. We also present the exact rule formulations in figure 4.4, with supporting rule definitions in figure 4.5, and definitions of atomic statements in figure 4.6.

For the within-sentence case, the `candidateSent`(Q, S, A) rule will propose that two entities (Q, A) are a relation extraction candidate iff Q and A are from the same sentence S and A has a noun-type POS tag.

```
candidateSent(Q, S, A)          :- sentlink(Q, S, A), constraint(S, A) .
candidateDoc(Q, Sq, Sa, Ref, A) :- doclink(Q, Sq, Sa, Ref), sentlink(Ref, Sa, A),
                                   coref(Q, Sa, Ref, Sa), constraint(S, A) .
candidateDoc(Q, Sq, Sa, Ref, A) :- sentlink(Q, Sq, Ref), doclink(Ref, Sq, Sa, A),
                                   coref(Ref, Sq, A, Sa), constraint(S, A) .
```

FIGURE 4.4: Candidate generation rules in the ProPPR probabilistic first-order logic system. Note that the words that begin with capital letters (e.g. Q) are variables.

The other rule has two different definitions, both of which generate candidates from the same document using co-reference information. The first definition of `candidateDoc` (Q, Sq, Sa, Ref, A) will propose the candidate (Q, A) iff:

- Entity Q is in sentence Sq, entity Ref is in sentence Sa, and Sq and Sa are sentences from the same document.
- Entity Ref in sentence Sa refers to entity Q in sentence Sq.
- Entity Ref and entity A are in same sentence Sa.s
- Entity A has a noun-type POS tag.

The second definition for `candidateDoc` proposes the candidate (Q, A) iff:

- Entity Q and entity Ref are in sentence Sq together.
- Sentence Sq and sentence Sa are in the same document.
- Entity Ref in sentence Sq refers to entity A in sentence Sa.
- Entity A is in sentence Sa and A has a noun-type POS tag.

```
sentlink(Q, S, A)      :- entInSent(Q, S), sentHasEnt(S, A) .
doclink(Q, Sq, Sa, A)  :- entInSent(Q, Sq), sentInDoc(Sq, D)
                          docHasSent(D, Sa), sentHasEnt(Sa, A) .
constraint(S, A)       :- sentHasEntPOS(S, A, NN) .
constraint(S, A)       :- sentHasEntPOS(S, A, NNS) .
constraint(S, A)       :- sentHasEntPOS(S, A, NNP) .
constraint(S, A)       :- sentHasEntPOS(S, A, NNPS) .
```

FIGURE 4.5: Supporting candidate generation rules. The four definitions for `constraint` mean that this rule can be satisfied in four different ways.

`entInSent`(Q, S)	:=	Entity Q is in sentence S
`sentHasEnt`(S, A)	:=	Sentence S has entity A
`sentInDoc`(S, D)	:=	Sentence S in in document D
`docHasSent`$(D,)$	:=	Document D has sentence S
`sentHasEntPOS`$(S, A,$ `T`$)$	:=	Entity A in sentence S has POS tag `T`
`coref`(Ref, Sq, A, Sa)	:=	Ref in sentence Sq is a referent to A in sentence Sa

FIGURE 4.6: Definitions of atomic statements.

In our candidate generation, we accept all inferences of a candidate generation rule that have non-zero probability.

We follow the widespread distant labeling assumption as detailed in our review in section 2.2.1. As applied to our candidate generation approach, we label a candidate (Q, A) with relation r if and only if the record $(r, (Q, A))$ appears in our knowledge base (KB). Figure 4.7 gives a schematic representation of the candidate generation and distant labeling component.

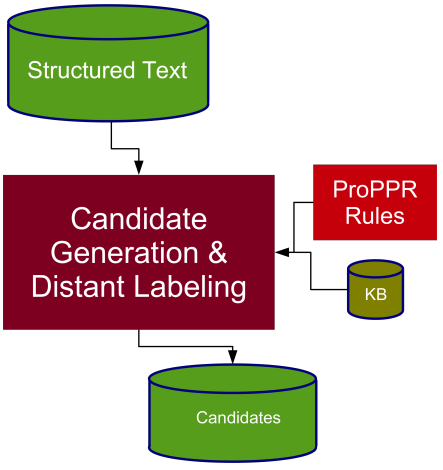


FIGURE 4.7: The candidate generation and distant labeling component.

4.1.3 Featurization: k-skip n-grams

Our choice of features are inspired by Roth et al.
(2014) and Mintz et al. (2009).

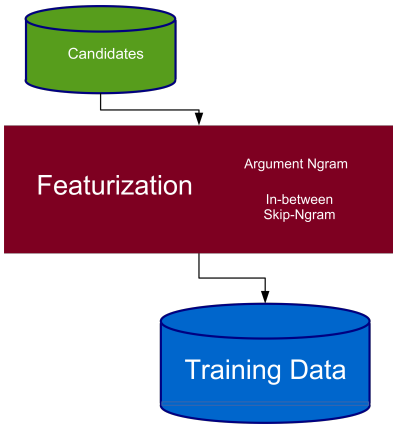


FIGURE 4.8: The featurization component.

We use two different types of features: n-grams and k-skip n-grams. An n-gram is a contiguous sequence of n tokens. A k-skip n-gram is a generalization of this idea. They are like n-grams, as they have exactly n successive tokens, but they allow at most k skips in-between any pair of tokens. Note that a 0-skip n-gram is an n-gram. Thus, k-skip n-grams subsume all n-grams as well as all $k-1$, $k-2$, etc.-skip n-grams.

We provide examples of n-grams and k-skip n-grams produced from the sentence, "Insurgents killed in ongoing fighting." in figures 4.9 and 4.10, respectively. These examples are reproduced from Guthrie et al. (2006).

Uni-gram	:=	Insurgents, killed, in, ongoing, fighting
Bi-gram	:=	Insurgents killed, killed in, in ongoing, ongoing fighting
Tri-gram	:=	Insurgents killed in, killed in ongoing, in ongoing fighting

FIGURE 4.9: Examples of n-grams, comma-separated.

1-skip bi-grams	:=	Insurgents killed, killed in, in ongoing, ongoing fighting, Insurgents in, killed ongoing, in fighting
2-skip bi-grams	:=	Insurgents killed, killed in, in ongoing, ongoing fighting, Insurgents in, killed ongoing, in fighting, Insurgents ongoing, killed fighting

FIGURE 4.10: Examples of k-skip n-grams, comma-separated.

Formally, we follow the definition of a k-skip n-gram as it is presented in Guthrie et al. (2006). This definition is listed in figure 4.11, where we define elements of the set of k-skip n-grams.

$$\{w_{i_1}, w_{i_2}, \ldots, w_{i_n} | \sum_{j=1}^{n} i_j - i_{j-1} \leq k\}$$

FIGURE 4.11: Definition of a k-skip n-gram.

For the query and answer nounhprases, we take the bi-grams and uni-grams that are adjacent to the nounphrase. We also construct 2-k-skip bi-grams from the tokens in-between the query and answer nounphrases. In the event that the candidate was generated from two sentences, we take the 2-k-skip n-grams between the answer or query (depending on the specific candidate generation rule) and the referent. Note that we must construct these k-skip n-gram features from two nounphrases in the same sentence.

4.2 Learning Extraction Features

Our choice of learning algorithm is also inspired by Roth et al. (2014). The authors make an important empirical insight in relation extraction. They note that relation extraction systems must constantly handle negative examples. The overwhelming majority of sentences and pairs of sentences in documents do not express any known relation. Therefore, the overwhelming majority of generated candidates will not express any known relation. In general, the majority of learning algorithms assume that the training data is uniformly balanced between positive and negative examples. In the relation extraction task, this assumption is violated.

Roth et al.'s solution is to use a cost-augmented learning algorithm. If the learner is able to penalize misclassifications on positive examples more seriously than misclassifications on negative examples, then the learner is significantly more likely to learn a hypothesis that is able to adequately discriminate between unobserved positive and negative examples in a class-unbalanced setting. Specifically, for each relation, we use a cost-augmented binary soft-margin Support Vector Machine (SVM) to learn the function from adjacent n-gram

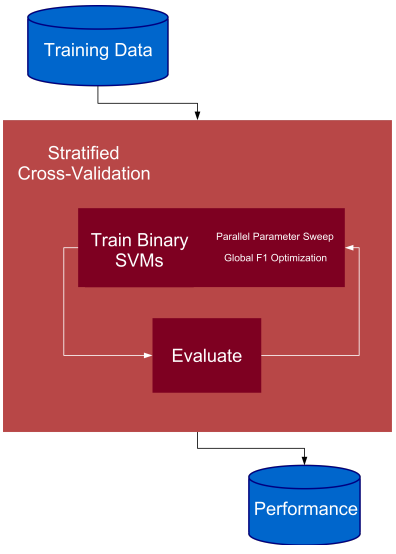


FIGURE 4.12: The evaluation pipeline.

and inner k-skip n-gram features of candidates to the relation.

4.2.1 Hard vs. Soft Margin SVMs

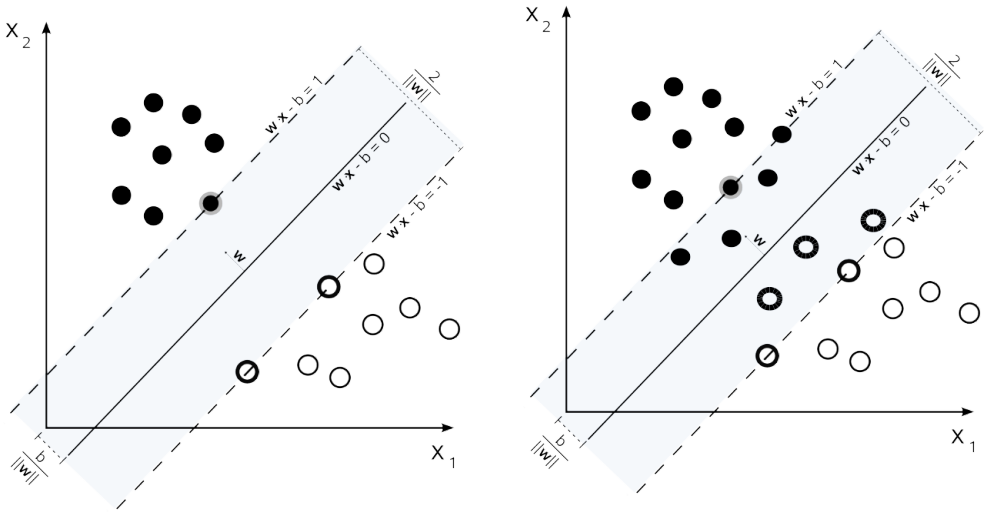
All SVMs learn a hyperplane that separates the data into distinct classes. Moreover, all SVMs find the hyperplane that maximizes the margin between the hyperplane and each class.

The original hard SVM finds a linearly separable hyperplane. That is, all of the data points are perfectly separated into their respective classes and the hyperplane that accomplishes this separation is linear. In the easy to visualize two-dimensional case, this hyperplane is a straight line. See figure 4.13a[1] for an example.

Hard SVMs have an Achilles' heel: they do not work on non-linearly separable datasets. This is a frustrating limitation as most (if not all) real-world uses of classifiers are in non-linearly separable domains. In our domain, where we have tens of millions of noisy features and hundreds of thousands of labeled training examples, hard SVMs are completely impractical. Add to this distantly labeled data, which almost always has some small labeling errors, and any justification for using hard-margin SVMs evaporates.

However, there is a very useful re-formulation of the traditional SVM to deal with non-linearly separable datasets. So called soft-margin SVMs introduce *slack variables*, ξ, which account for each data points' transgression into the margin. This change in the objective function introduces new support vectors. While in both the hard and soft case the vectors that are on the margin are the support vectors, the soft SVM case adds all vectors that have non-zero ξ's. The vectors that have non-zero values for their slack variables are exactly the ones that have passed into the margin. See figure 4.14a for the soft-SVM objective function and see figure 4.13b for a visual example of a soft-SVM's hyperplane and margin.

[1] Public Domain. http://commons.wikimedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png



(A) Hard SVM formulation. Note how the support vectors are only the data points that are on the margin.

(B) Soft SVM formulation. Critical difference is that support vectors include data points that are in-between the separating hyperplane w and the class margins.

FIGURE 4.13: Hard and soft Support Vector Machine formulations on synthetic dataset. [1]

4.2.2 Cost-Augmented Support Vector Machines

Training SVMs is equivalent to optimizing an objective function (see figure 4.14). The margin-maximizing idea is equivalent to minimizing the squared L_1 norm of the hyperplane w plus the sum of each slack variable ξ_i weighted by a cost C, subject to the constraint that each data point must reside on the class-appropriate side of the hyperplane.

This formulation has been empirically shown to work well on a variety of classification tasks. However, it is sensitive to the distribution of labeled examples between the positive and negative classes. For example, if the ratio of negative to positive examples is 1 million to 1, then the SVM will learn a hyperplane w and associated margin that will be extremely likely to classify any data point as negative merely because the negative examples greatly outnumber the positive examples seen during training.

In our relation extraction task, as well as many other NLP applications, we suffer from a class imbalance problem. Namely, the negative distantly labeled relation candidates (section 4.1.2) greatly outnumber the positively distant labeled candidates. To overcome this problem, we adopt a cost-augmented approach to training SVMs.

The cost-augmented approach we follow is the same that is presented by Morik et al. (1999). The critical idea is a relatively simple change: separate the cost C into costs specific to misclassifications on positive and negative examples seen during training.

These two new costs, C_+ and C_-, are only applied to examples that belong to either the positive class (label $y_i = +1$) or the negative class (label $y_i = -1$), respectively. See figure 4.14b[2] for the complete objective function formulation.

In our experiments in chapter 5, we use the cost-augmented binary SVM implementation known as svm-light (Morik et al., 1999).

Minimize:	$\frac{1}{2}\|\|\vec{w}\|\|^2 + C \sum \xi_i$
subject to:	$\forall k : y_k[\vec{w} \cdot \vec{x_k} + b] \geq 1 - \xi_k$

(A) Normal soft SVM objective function. Note the single cost penalty C.

Minimize:	$\frac{1}{2}\|\|\vec{w}\|\|^2 + C_+ \sum_{i:y_i=1} \xi_i + C_- \sum_{j:y_j=-1} \xi_j$
subject to:	$\forall k : y_k[\vec{w} \cdot \vec{x_k} + b] \geq 1 - \xi_k$

(B) Cost-augmented soft SVM objective function. Note that the cost is split between the positively labeled examples C_+ and the negative examples C_-.

FIGURE 4.14: Objective functions for soft and cost-augmented soft SVMs.[2]

4.2.3 Stratified Cross Validation

Cross validation is a useful technique to thoroughly evaluate the performance of any learning algorithm. The idea behind cross validation is to shuffle all of the training examples and then partition this data into k folds, where $k \geq 2$. For each fold, we select the fold to be held out and train a model on the other $k - 1$ folds. Once we have a model, we evaluate its performance on the held out fold. This procedure produces k performance estimates.

While in general cross validation is a useful evaluation paradigm, this method can produce skewed performance estimates when the training data is unbalanced. In our case, we do not have an approximately even distribution of relations. To counteract this class imbalance problem while yielding a minimum of unnecessary influence on our empirical evaluation, we use a technique known as stratified cross validation. In stratified cross validation, we ensure that each fold has the same distribution of labeled examples. I.e. for each relation, we ensure that each fold has $\frac{1}{k}$ of the relation's positively labeled examples. Moreover, we ensure that each fold has $\frac{1}{k}$ of the negative examples.

We use stratified cross validation in order to evaluate our learned relation extractors.

[2]Reproduced from Morik et al. (1999)

Chapter 5

Experiments

In this chapter, we describe our experiments to evaluate our relation extraction system. We specify our setup and describe the methods we use to generate our evaluations, including hyperparameter values. We present relation classification performance on two separate experiments.

5.1 Experimental Setup

We train a an independent binary classifier for each relation using a cost-adjusted Support Vector Machine (SVM). We justify our use of a cost-augmented SVM in section 4.2.2. The ability to penalize the SVM more for misclassifying positive examples is critical to the overall success of our relation extraction system. Without this cost-augmentation, our system is not able to effectively learn discriminative features, resulting in near zero F1 scores. In all experiments, we perform a parameter sweep over the three cost values: 6, 10, and 14, which correspond to the SVM penalizing misclassifications on positive examples 6, 10, and 14 times as much as it does misclassifications of negative examples. We note that a cost value of 1 is equivalent to a non-cost-augmented SVM.

We generate training data using our pipeline described in 4.1. Specifically, for each relation, we generate relation candidates and distantly label them using our knowledge base (described in section 4.1.2), extract adjacent n-gram and inner k-skip n-gram features (described in section 4.1.3), and output these training examples to disk. In addition to generating labeled examples, our candidate generation step also generates negative labeled examples. A candidate (Q, A) is considered negative if it does not align to any relation in our knowledge base, i.e. $\nexists r$ s.t. $(r, (Q, A)) \in \text{KB}$. These unmatched candidates are assigned the virtual label `NOT_RELATED`. They are critical in our training and

evaluation as they help the learner understand how to reject candidates, which heavily influences precision. For each relation, we uniformly at random evenly split the distantly labeled examples into 3 folds. We also uniformly at random evenly distribute the `NOT_RELATED` examples to each relation's 3 folds.

5.2 Evaluation

As described in section 4.2.3, we use 3-fold stratified cross-validation. We use standard metrics for relation extraction evaluation. Namely precision, recall, and F1. We provide precise definitions of these metrics in figure 5.1. Intuitively, precision can be thought of as how often the extracted relations were correct. Recall can be thought of as how many of the known relation mentions in the text were extracted by the system. F1 is the harmonic average between precision and recall.

$$\begin{aligned} \text{Precision} &= \frac{|\{\text{all correct mentions}\}| \cap |\{\text{extracted mentions}\}|}{|\{\text{extracted mentions}\}|} \\ \text{Recall} &= \frac{|\{\text{all correct mentions}\}| \cap |\{\text{extracted mentions}\}|}{|\{\text{all correct mentions}\}|} \\ \text{F1} &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \end{aligned}$$

FIGURE 5.1: Definitions of evaluation metrics: precision, recall, and F1.

F1 is a useful metric as it balances performance between providing correct extractions and finding all relation mentions. It would be quite easy to achieve 100% recall: a system could simply output absolutely everything. This behavior is, of course, highly undesirable, which would be captured by this hypothetical system having near-zero precision. On the flip side, a system could only output its single, most confident extraction. This would most likely be a correct extraction, and thus the system would have perfect precision. However, it would have near-zero recall as it only output a single extraction. F1 puts emphasis on doing well in both of these areas, without unduly sacrificing either one.

In order to make better sense of the results, we limit our presentation to a few key views. We provide aggregate precision, recall, and F1 measures for each experiment. Each one of these metrics is averaged across each relation in a technique known as micro-averaging. In addition, we provide a F1 on a per-relation basis for each dependent variable in each experiment. Finally, we manually aggregate all of the relations into 10 functional types. This mapping is listed in table 5.1. This mapping allows us to visualize performance and better understand classification performance across the relations.

Relation	**Functional Type**
org_alternate_names	org to misc
org_city_of_headquarters	org to location
org_country_of_headquarters	org to location
org_date_dissolved	org to date
org_date_founded	org to date
org_founded_by	org to person
org_member_of	org to org
org_members	org to person
org_parents	org to org
org_political_religious_affiliation	org to misc
org_shareholders	org to person
org_stateorprovince_of_headquarters	org to location
org_subsidiaries	org to org
org_top_members_employees	org to person
per_age	person to misc
per_alternate_names	person to person
per_cause_of_death	person to date
per_charges	person to misc
per_children	person to person
per_cities_of_residence	person to location
per_city_of_birth	person to location
per_city_of_death	person to location
per_countries_of_residence	person to location
per_country_of_birth	person to location
per_country_of_death	person to location
per_date_of_birth	person to date
per_date_of_death	person to date
per_employee_of	person to org
per_member_of	person to org
per_origin	person to misc
per_other_family	person to person
per_parents	person to person
per_religion	person to misc
per_schools_attended	person to misc
per_siblings	person to person
per_spouse	person to person
per_stateorprovince_of_birth	person to location
per_stateorprovince_of_death	person to location
per_stateorprovinces_of_residence	person to location
per_statesorprovinces_of_residence	person to location
per_title	person to misc

TABLE 5.1: Complete relation to functional types. The 10 functional types are org (organization) to date, org to location, org to misc, org to org, org to person, person to date, person to location, person to misc, person to org, and person to person.

5.3 Empirical Results

We empirically evaluate our relation extraction system on a per-relation basis. Using the motivation from section 4.2.3 and metrics described in figure 5.1, we evaluate performance using precision, recall, and F1 on a held out test sets using 3 fold stratified cross validation.

5.3.1 Large Scale Relation Extraction

Our first experiment is a comparison with many other distantly labeled relation extractions systems found in the literature (Mintz et al., 2009, Roth et al., 2014, Schmitz et al., 2012, Surdeanu et al., 2010). In this experiment, we limit our candidate generation stage to candidates that come from the same sentence. We perform candidate generation using the entire newswire corpus (chapter 3).

This candidate generation step exactly follows the common distant supervision assumption: a sentence expresses a relation r is the sentence has entities Q, A and the triple (r, Q, A) is in the knowledge base (see section 2.2.1 for a review on distant supervision). Since both of the candidate's entities come from the same sentence, we do not use coreference resolution. Using this *within-sentence* candidate generation method, we find 22,539,854 within-sentence candidate pairs.

We provide micro-averaged results for precision, recall, and F1 in table 5.2. We also provide a per-relation breakdown of F1 in table 5.3. In figure 5.2, we plot the micro-averaged F1 for each functional relation type.

Precision	**Recall**	**F1**
50.89%	38.70%	42.27%

TABLE 5.2: Micro-averaged (across all relations and folds) results of relation extraction system using within-sentence candidate generation on entire newswire corpus.

Relation	**Precision**	**Recall**	**F1**
org_city_of_headquarters	44.05	35.71	39.05
org_country_of_headquarters	47.62	28.71	35.59
org_date_dissolved	0.00	0.00	0.00
org_date_founded	100.00	50.00	66.67
org_founded_by	46.11	27.16	33.55
org_member_of	0.00	0.00	0.00
org_members	17.81	11.11	13.55
org_parents	38.27	36.11	34.67
org_political_religious_affiliation	68.20	68.52	68.05
org_shareholders	45.83	20.00	26.19
org_subsidiaries	0.00	0.00	0.00
org_top_members_employees	53.77	33.73	41.16
per_age	82.07	26.32	38.56
per_children	0.00	0.00	0.00
per_cities_of_residence	75.63	52.85	62.07
per_city_of_death	83.33	50.00	61.11
per_countries_of_residence	80.07	68.88	74.01
per_country_of_birth	55.59	45.85	50.19
per_date_of_birth	0.00	0.00	0.00
per_date_of_death	31.94	50.00	38.33
per_employee_of	73.95	70.20	71.99
per_member_of	67.83	46.95	55.23
per_origin	91.05	80.95	85.69
per_parents	78.12	33.75	44.65
per_siblings	31.19	27.86	28.11
per_stateorprovinces_of_residence	95.93	57.54	71.00
per_title	96.28	94.22	95.24

TABLE 5.3: Per-relation results only using within sentence distant labeling on the entire newswire corpus. Metrics are micro-averaged across 3 folds using stratified cross validation.

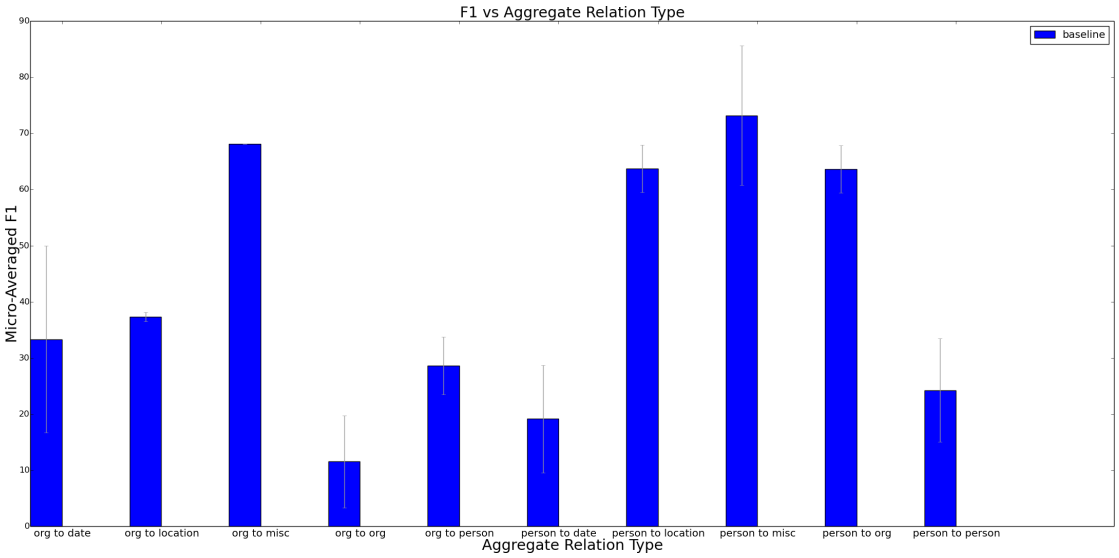


FIGURE 5.2: Micro-averaged F1 for each functional relation type used in the large scale experiment, using within-sentence candidate generation. Note the high variance across all functional relation types. F1 scores range from the low teens to the low 70s.

5.3.2 Searching and ProPPR Candidate Generation with Co-reference Resolution

Our second experiment is compares different candidate generation methods. In this experiment, we use the within-sentence candidate generation, the ProPPR inference rule based candidate generation (section 4.1.2), and a method that combines both within-sentence and ProPPR to generate relation extraction candidates.

Due to computational constraints imposed by our text processing, namely the co-reference resolution engine, we are unable to process the entire corpus. Instead, we take the 80 queries used in the TAC-KBP 2012 slot filling task and locate all documents that these queries occur in. This targeted search returns 193,571 sentences in 10,254 documents. Of these documents, only 4% consist of a single sentence.

Table 5.4 contains the micro-averaged (across all folds and relations) precision, recall, and F1 for each candidate generation method on this dataset. We provide micro-averaged F1 results on a per-relation basis for each candidate generation method in table 5.5. Figure 5.3 plots the micro-averaged F1 across all folds for each functional relation type. Finally, we provide the per-relation precision, recall, and F1 for each separate candidate generation method in tables 5.6, 5.7, and 5.8.

	Precision	**Recall**	**F1**
Within-Sentence	34.45%	27.13%	29.29%
ProPPR Rules	36.35%	28.37%	29.54%
Both	**38.66%**	**31.86%**	**33.15%**

TABLE 5.4: Micro-averaged (across all relations and folds) precision, recall, and F1 for each candidate generation method on the search results. Note that the best aggregate performance occurred when we combined the within-sentence and ProPPR rule-based candidate generation methods.

Relation	**Within-Sentence**	**ProPPR Rules**	**Both**
org_alternate_names	30.88	30.88	30.88
org_city_of_headquarters	12.83	19.65	12.83
org_country_of_headquarters	43.10	56.21	43.10
org_founded_by	40.92	37.00	40.92
org_member_of	33.61	30.34	33.61
org_members	23.23	39.04	23.23
org_parents	28.84	37.56	28.84
org_political_religious_affiliation	64.01	70.23	64.01
org_shareholders	14.06	30.26	14.06
org_stateorprovince_of_headquarters	22.69	17.67	22.69
org_subsidiaries	16.07	24.87	16.07
org_top_members_employees	38.12	40.33	38.12
per_alternate_names	86.67	47.04	86.67
per_cause_of_death	100.00	83.33	100.00
per_charges	0.00	0.00	0.00
per_children	0.00	48.35	0.00
per_cities_of_residence	13.89	17.71	13.89
per_city_of_birth	0.00	0.00	0.00
per_city_of_death	24.54	29.04	24.54
per_countries_of_residence	22.36	30.70	22.36
per_country_of_birth	13.95	0.00	13.95
per_country_of_death	0.00	0.00	0.00
per_date_of_death	22.00	27.76	22.00
per_employee_of	32.23	40.71	32.23
per_member_of	16.46	14.49	16.46
per_origin	12.03	21.90	12.03
per_other_family	0.00	0.00	0.00
per_parents	40.04	43.27	40.04
per_religion	0.00	50.00	0.00
per_schools_attended	0.00	0.00	0.00
per_siblings	13.25	17.38	13.25
per_spouse	0.00	0.00	0.00
per_stateorprovince_of_birth	0.00	0.00	0.00
per_stateorprovince_of_death	12.41	15.09	12.41
per_statesorprovinces_of_residence	6.21	17.95	6.21
per_title	66.24	66.00	66.24

TABLE 5.5: Micro-averaged F1 (across 3 folds) per-relation results using queried document set and with all candidate generation methods. Note that while some relations have incredible performance (e.g. per_cause_of_death), there are many relations that have an F1 of 0. These zeros are attributable to not having enough labeled data for the relation.

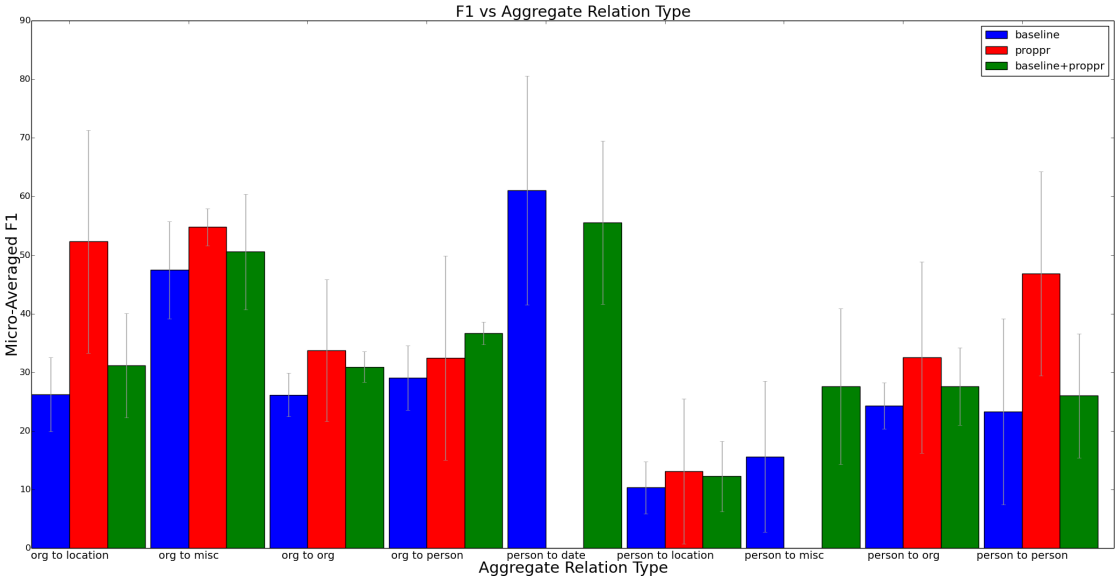


FIGURE 5.3: Micro-averaged F1 for each functional relation type used in the search-results experiment, using within-sentence, ProPPR rules, and both candidate generation methods. Note the high variance presented across all of the measurements. Critically, note that for two complete functional relation types (person to date and person to misc), the ProPPR candidate generation method was not able to extract any relations. F1s reange from the low teens to the high 50s.

Relation	**Precision**	**Recall**	**F1**
org_alternate_names	48.03	23.74	30.88
org_city_of_headquarters	14.29	11.67	12.83
org_country_of_headquarters	47.20	39.97	43.10
org_founded_by	48.72	35.86	40.92
org_member_of	61.11	24.60	33.61
org_members	30.66	18.83	23.23
org_parents	29.29	28.77	28.84
org_political_religious_affiliation	78.77	57.96	64.01
org_shareholders	18.18	11.71	14.06
org_stateorprovince_of_headquarters	22.14	23.92	22.69
org_subsidiaries	19.42	14.60	16.07
org_top_members_employees	41.19	36.78	38.12
per_alternate_names	100.00	80.00	86.67
per_cause_of_death	100.00	100.00	100.00
per_charges	0.00	0.00	0.00
per_children	0.00	0.00	0.00
per_cities_of_residence	14.61	13.81	13.89
per_city_of_birth	0.00	0.00	0.00
per_city_of_death	25.08	24.14	24.54
per_countries_of_residence	27.54	20.59	22.36
per_country_of_birth	18.89	11.80	13.95
per_country_of_death	0.00	0.00	0.00
per_date_of_death	26.92	21.03	22.00
per_employee_of	34.28	31.70	32.23
per_member_of	18.77	15.56	16.46
per_origin	15.01	13.35	12.03
per_other_family	0.00	0.00	0.00
per_parents	48.98	34.72	40.04
per_religion	0.00	0.00	0.00
per_schools_attended	0.00	0.00	0.00
per_siblings	12.88	14.29	13.25
per_spouse	0.00	0.00	0.00
per_stateorprovince_of_birth	0.00	0.00	0.00
per_stateorprovince_of_death	12.88	12.50	12.41
per_statesorprovinces_of_residence	8.67	4.95	6.21
per_title	77.72	57.82	66.24

TABLE 5.6: Per-relation results using the within-sentence candidate generation method only on the searched corpus. Precision, recall, and F1 are averaged across 3-fold stratified cross validation.

Relation	**Precision**	**Recall**	**F1**
org_alternate_names	32.73	94.82	48.42
org_city_of_headquarters	72.09	66.67	67.45
org_country_of_headquarters	93.98	85.43	89.47
org_founded_by	0.00	0.00	0.00
org_member_of	0.00	0.00	0.00
org_members	92.59	75.15	82.81
org_parents	68.11	36.35	46.49
org_political_religious_affiliation	100.00	44.44	61.11
org_shareholders	0.00	0.00	0.00
org_stateorprovince_of_headquarters	0.00	0.00	0.00
org_subsidiaries	74.45	46.67	54.75
org_top_members_employees	54.51	41.48	47.06
per_alternate_names	90.74	37.03	50.74
per_cause_of_death	0.00	0.00	0.00
per_charges	0.00	0.00	0.00
per_children	100.00	50.00	66.67
per_cities_of_residence	0.00	0.00	0.00
per_city_of_birth	100.00	51.67	64.68
per_city_of_death	0.00	0.00	0.00
per_countries_of_residence	0.00	0.00	0.00
per_country_of_birth	0.00	0.00	0.00
per_country_of_death	0.00	0.00	0.00
per_date_of_death	0.00	0.00	0.00
per_employee_of	74.73	58.34	65.15
per_member_of	0.00	0.00	0.00
per_origin	0.00	0.00	0.00
per_other_family	0.00	0.00	0.00
per_parents	91.67	75.00	82.14
per_religion	0.00	0.00	0.00
per_schools_attended	0.00	0.00	0.00
per_siblings	100.00	70.37	81.48
per_spouse	0.00	0.00	0.00
per_stateorprovince_of_birth	0.00	0.00	0.00
per_stateorprovince_of_death	0.00	0.00	0.00
per_statesorprovinces_of_residence	80.56	40.48	53.74
per_title	0.00	0.00	0.00

TABLE 5.7: Per-relation results using the ProPPR inference candidate generation method only on the searched corpus. Precision, recall, and F1 are averaged across 3-fold stratified cross validation. Note the large degree of variance across all relations. For instance, precision ranges from 0% to 100%.

Relation	**Precision**	**Recall**	**F1**
org_alternate_names	48.03	23.74	30.88
org_city_of_headquarters	23.61	16.97	19.65
org_country_of_headquarters	56.65	56.05	56.21
org_founded_by	42.08	33.81	37.00
org_member_of	49.84	23.92	30.34
org_members	41.87	36.70	39.04
org_parents	39.80	36.15	37.56
org_political_religious_affiliation	85.86	61.15	70.23
org_shareholders	34.84	27.24	30.26
org_stateorprovince_of_headquarters	16.06	20.21	17.67
org_subsidiaries	25.20	24.90	24.87
org_top_members_employees	42.79	39.29	40.33
per_alternate_names	83.33	37.22	47.04
per_cause_of_death	75.00	100.00	83.33
per_charges	0.00	0.00	0.00
per_children	60.74	50.00	48.35
per_cities_of_residence	16.87	18.69	17.71
per_city_of_birth	0.00	0.00	0.00
per_city_of_death	29.61	29.30	29.04
per_countries_of_residence	31.12	31.36	30.70
per_country_of_birth	0.00	0.00	0.00
per_country_of_death	0.00	0.00	0.00
per_date_of_death	26.18	29.86	27.76
per_employee_of	43.24	39.28	40.71
per_member_of	17.77	12.46	14.49
per_origin	36.23	18.04	21.90
per_other_family	0.00	0.00	0.00
per_parents	41.03	47.22	43.27
per_religion	100.00	33.33	50.00
per_schools_attended	0.00	0.00	0.00
per_siblings	15.88	20.00	17.38
per_spouse	0.00	0.00	0.00
per_stateorprovince_of_birth	0.00	0.00	0.00
per_stateorprovince_of_death	14.20	16.48	15.09
per_statesorprovinces_of_residence	19.29	17.28	17.95
per_title	80.38	56.21	66.00

TABLE 5.8: Per-relation results using the within-sentence and ProPPR inference candidate generation methods on the searched corpus. Precision, recall, and F1 are averaged across 3-fold stratified cross validation.

Chapter 6

Conclusion

From our empirical results in chapter 5, we conclude the sum of all of our algorithms and techniques described in chapter 4 has culminated in an relation extraction system that is competitive with state-of-the-art systems. Our distant labeling pipeline is efficient and scales to millions of documents. While not as scalable, our novel candidate generation method using ProPPR, a locally groundable probabilistic first-order logic, proved itself in a typical relation extraction setting.

Our large-scale extraction experiment in in section 5.3.1 showed that our relation extraction system is efficient and suitable for corpus-sized datasets. Despite not having access to more advanced candidate generation using ProPPR, this large-scale system was still able accurately classify relation candidates, with a micro-averaged F1 of 42.27%. We note that the entire newswire corpus had few multi-sentence documents. In fact, only 22% of the documents had more than one sentence. A main advantage of using ProPPR inference is to propose candidates that come from different sentences within the same document. Therefore, we conclude that our inference rules for ProPPR would not be as effective in this corpus.

In our targeted search experiment, detailed in section 5.3.2, we empirically validated our novel approach to candidate generation using ProPPR. Although candidate generation using ProPPR suffered from sparsity, leading to several relations that had 0 performance scores, this technique achieved stellar performance on other relations. In the presentation of our results, especially in figure 5.3, we note that this led to very high variance in our reported metrics.

In contrast, the within-sentence method did not suffer from the same data sparsity issues. Upon inspection, we noted many mismatches between co-reference candidates

and candidates produced from our nounphrase chunking using NE tags. Unfortunately, this meant that we were not able to exploit the full power behind ProPPR inference.

From a practical standpoint, we were able to improve upon the typical baseline, within-sentence candidate generation method. We were able to combine ProPPR inference with the within-sentence method to achieve a synergistic effect in relation classification performance. Our overall micro-averaged F1 of 33.15% using this method is competitive with state-of-the-art relation extraction systems (Roth et al., 2014).

Outside of candidate generation, we note the effectiveness of our feature representation and our choice of learning algorithm. n-grams and k-skip n-grams have been used in a variety of NLP tasks, ranging from information extraction to language modeling. In our task, we found that these features were effective and provided a good space in which to learn to discriminate between relations.

We make a crucial note about using a cost-augmented Support Vector Machine as a learning algorithm. Early prototypes of our system that used a non-cost-augmented SVM had terrible performance. Using within-sentence candidate generation and n-gram and k-skip n-gram features, a non-cost augmented SVM had a near zero micro-averaged F1.

In closing, we conclude that our approach, including our novel candidate generation method using ProPPR, was highly effective and resulted in a highly competitive and performant relation extraction system.

Bibliography

Andersen, P. M., Hayes, P. J., Huettner, A. K., Schmandt, L. M., Nirenburg, I. B., and Weinstein, S. P. (1992). Automatic extraction of facts from press releases to generate news stories. In *Proceedings of the third conference on Applied natural language processing*, pages 170–177. Association for Computational Linguistics.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction for the web. In *IJCAI*, volume 7, pages 2670–2676.

Banko, M., Etzioni, O., and Center, T. (2008). The tradeoffs between open and traditional relation extraction. In *ACL*, volume 8, pages 28–36.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

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

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

Chapelle, O., Schölkopf, B., Zien, A., et al. (2006). *Semi-supervised learning*, volume 2. MIT press Cambridge.

Chinchor, N. A. (2001). Overview of muc-7 / met-2.

Craven, M., Kumlien, J., et al. (1999). Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86.

Curran, J. R., Murphy, T., and Scholz, B. (2007). Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 172–180.

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

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. *Artificial Intelligence*, 165(1):91–134.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Go, A., Bhayani, R., and Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12.

Granger, R. H. (1977). Foul-up: A program that figures out meanings of words from context. In *In Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 172–178.

Grishman, R. and Sundheim, B. (1996). Message understanding conference-6: A brief history. In *COLING*, volume 96, pages 466–471.

Guthrie, D., Allison, B., Liu, W., Guthrie, L., and Wilks, Y. (2006). A closer look at skip-gram modelling. In *Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC-2006)*, pages 1–4.

Hilbert, M. and López, P. (2011). The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65.

Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., and Weld, D. S. (2011). Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Hoffmann, R., Zhang, C., and Weld, D. S. (2010). Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 286–295. Association for Computational Linguistics.

Ji, H., Grishman, R., Dang, H. T., Griffitt, K., and Ellis, J. (2010). Overview of the tac 2010 knowledge base population track. In *Third Text Analysis Conference (TAC 2010)*.

Jones, R., McCallum, A., Nigam, K., and Riloff, E. (1999). Bootstrapping for text learning tasks. In *IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, volume 1. Citeseer.

Krause, S., Li, H., Uszkoreit, H., and Xu, F. (2012). Large-scale learning of relation-extraction rules with distant supervision from the web. In *The Semantic Web–ISWC 2012*, pages 263–278. Springer.

Lee, H., Peirsman, Y., Chang, A., Chambers, N., Surdeanu, M., and Jurafsky, D. (2011). Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34. Association for Computational Linguistics.

Lehnert, W. G. (1977). The process of question answering. Technical report, DTIC Document.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

McDonald, R. (2005). Extracting relations from unstructured text. *Rapport technique, Department of Computer and Information Science-University of Pennsylvania*.

McIntosh, T. and Curran, J. R. (2009). Reducing semantic drift with bagging and distributional similarity. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 396–404. Association for Computational Linguistics.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Min, B. (2013). *Relation Extraction with Weak Supervision and Distributional Semantics*. PhD thesis, New York University.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Mitchell, T. M. (1997). Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45.

Mitchell, T. M., Betteridge, J., Carlson, A., Hruschka, E., and Wang, R. (2009). Populating the semantic web by macro-reading internet text. In *The Semantic Web-ISWC 2009*, pages 998–1002. Springer.

Morik, K., Brockhausen, P., and Joachims, T. (1999). Combining statistical learning with a knowledge-based approach: a case study in intensive care monitoring. Technical report, Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund.

Nguyen, T.-V. T. and Moschitti, A. (2011a). End-to-end relation extraction using distantf supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 277–282. Association for Computational Linguistics.

Nguyen, T.-V. T. and Moschitti, A. (2011b). Joint distant and direct supervision for relation extraction. In *IJCNLP*, pages 732–740.

Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (1998). Learning to classify text from labeled and unlabeled documents. *AAAI/IAAI*, 792.

Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 113–120. Association for Computational Linguistics.

Purver, M. and Battersby, S. (2012). Experimenting with distant supervision for emotion classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–491. Association for Computational Linguistics.

Riedel, S., Yao, L., and McCallum, A. (2010). Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Riesbeck, C. K. and Schank, R. C. (1976). Comprehension by computer: Expectation-based analysis of sentences in context. Technical report, DTIC Document.

Rosenfeld, B. and Feldman, R. (2007). Using corpus statistics on entities to improve semi-supervised relation extraction from the web. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 600.

Roth, B., Barth, T., Wiegand, M., Singh, M., and Klakow, D. (2014). Effective slot filling based on shallow distant supervision methods. *arXiv preprint arXiv:1401.1158*.

Schank, R. C. (1975). *Conceptual information processing*. Elsevier Science Inc.

Schmitz, M., Bart, R., Soderland, S., Etzioni, O., et al. (2012). Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Surdeanu, M., McClosky, D., Tibshirani, J., Bauer, J., Chang, A. X., Spitkovsky, V. I., and Manning, C. D. (2010). A simple distant supervision approach for the tac-kbp slot filling task. In *Proc. TAC 2010 Workshop*. Citeseer.

Takamatsu, S., Sato, I., and Nakagawa, H. (2012). Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 721–729. Association for Computational Linguistics.

Toutanova, K. and Manning, C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics.

Wang, W. Y., Mazaitis, K., and Cohen, W. W. (2013). Programming with personalized pagerank: a locally groundable first-order probabilistic logic. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2129–2138. ACM.

Xu, W., Le Zhao, R. H., and Grishman, R. (2013). Filling knowledge base gaps for distant supervision of relation extraction. In *Proceedings of ACL*.

Åse Dragland (2013). Big data, for better or worse: 90last two years. *ScienceDaily*, (3):885–887.