uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

David Nadeau

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Semi-Supervised Named Entity Recognition:
Learning to Recognize 100 Entity Types with Little Supervision

TITRE DE LA THÈSE / TITLE OF THESIS

Stan Matwin

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Peter Turney

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

William Cohen

Diana Inkpen

Jean-Pierre Corriveau

Nathalie Japkowicz

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Semi-Supervised Named Entity Recognition:

Learning to Recognize 100 Entity Types with Little Supervision

David Nadeau

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the PhD degree in Computer Science

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-49385-4
Our file Notre référence
ISBN: 978-0-494-49385-4

Table of contents

List of tables

List of figures

Abstract

Named Entity Recognition (NER) aims to extract and to classify rigid designators in text such as proper names, biological species, and temporal expressions. There has been growing interest in this field of research since the early 1990s. In this thesis, we document a trend moving away from handcrafted rules, and towards machine learning approaches. Still, recent machine learning approaches have a problem with annotated data availability, which is a serious shortcoming in building and maintaining large-scale NER systems.

In this thesis, we present an NER system built with very little supervision. Human supervision is indeed limited to listing a few examples of each named entity (NE) type. First, we introduce a proof-of-concept semi-supervised system that can recognize four NE types. Then, we expand its capacities by improving key technologies, and we apply the system to an entire hierarchy comprised of 100 NE types.

Our work makes the following contributions: the creation of a proof-of-concept semi-supervised NER system; the demonstration of an innovative noise filtering technique for generating NE lists; the validation of a strategy for learning disambiguation rules using automatically identified, unambiguous NEs; and finally, the development of an acronym detection algorithm, thus solving a rare but very difficult problem in alias resolution.

We believe semi-supervised learning techniques are about to break new ground in the machine learning community. In this thesis, we show that limited supervision can build complete NER systems. On standard evaluation corpora, we report performances that compare to baseline supervised systems in the task of annotating NEs in texts.

Acknowledgements

Le rêve de bâtir un système autonome est partagé par la plupart des chercheurs du domaine et s'étend en fait probablement à tous les ingénieurs de systèmes intelligents. Lorsque j'ai mis au point un premier prototype de système de reconnaissance d'entités nommées, de 2001 à 2003, le problème de la maintenance est rapidement devenu manifeste. En plus, l'effort d'annotation de documents requis pour étendre le système était tel que le rêve de départ devenait un impératif. J'avais bien la motivation de créer ce système, mais je n'avais aucune idée de comment y arriver. C'était la conjoncture idéale pour entreprendre une thèse.

Je tiens à remercier Peter Turney et Stan Matwin qui ont supervisé et contribué à ce travail. En questionnant et en soupesant chaque idée et chaque phrase, ils m'ont fait comprendre beaucoup plus que des notions abstraites et des procédures informatiques.

Je tiens aussi à remercier Caroline Barrière, Cyril Goutte et Pierre Isabelle du Groupe de technologies langagières interactives du Conseil national de recherches Canada pour leur aide et leur commentaires sur les versions préliminaires de cette thèse.

Merci au Fonds québécois de la recherche sur la nature et les technologies ainsi qu'à l'Université d'Ottawa pour le support financier.

Chapter 1

Introduction

The term "Named Entity" (NE) is in current use in Information Extraction (IE) applications. It was coined at the sixth Message Understanding Conference (MUC-6) (Grishman & Sundheim 1996), which influenced IE research in the 1990s. At the time, MUC was focusing on IE tasks wherein structured information on company and defense-related activities are extracted from unstructured text, such as newspaper articles. In defining IE tasks, people noticed that it is essential to recognize information units such as names including person, organization, and location names, and numeric expressions including time, date, money, and percentages. Identifying references to these entities in text was acknowledged as one of IE's important sub-tasks and was called "Named Entity Recognition (NER)." Before the NER field was recognized in 1996, significant research was conducted by extracting proper names from texts. A paper published in 1991 by Lisa F. Rau (1991) is often cited as the root of the field.

For more than fifteen years, a dynamic research community advanced the fundamental knowledge and the engineered solutions to create an NER system. In its canonical form, the input of an NER system is a text and the output is information on boundaries and types of NEs found in the text. The vast majority of proposed systems fall in two categories: the handmade rule-based systems; and the supervised learning-based systems. In both approaches, large collections of documents are analyzed by hand to obtain sufficient knowledge for designing rules or for feeding machine learning algorithms. Expert linguists must execute this important amount of work, which in turn limits the building and maintenance of large-scale NER systems.

This thesis is about the creation of an autonomous NER system. It has the desirable property of requiring a small amount of work by an expert linguist. It falls in the new category of semi-supervised and unsupervised systems. Influential work in this category is relatively rare and recent, and we believe ours to be the first thesis devoted exclusively to the creation of an autonomous NER system.

This thesis is structured around the construction of an NER system and one of our goals is to create proof-of-concept software. System architecture is shown in Figure 1 and we'll refer to it throughout the thesis.

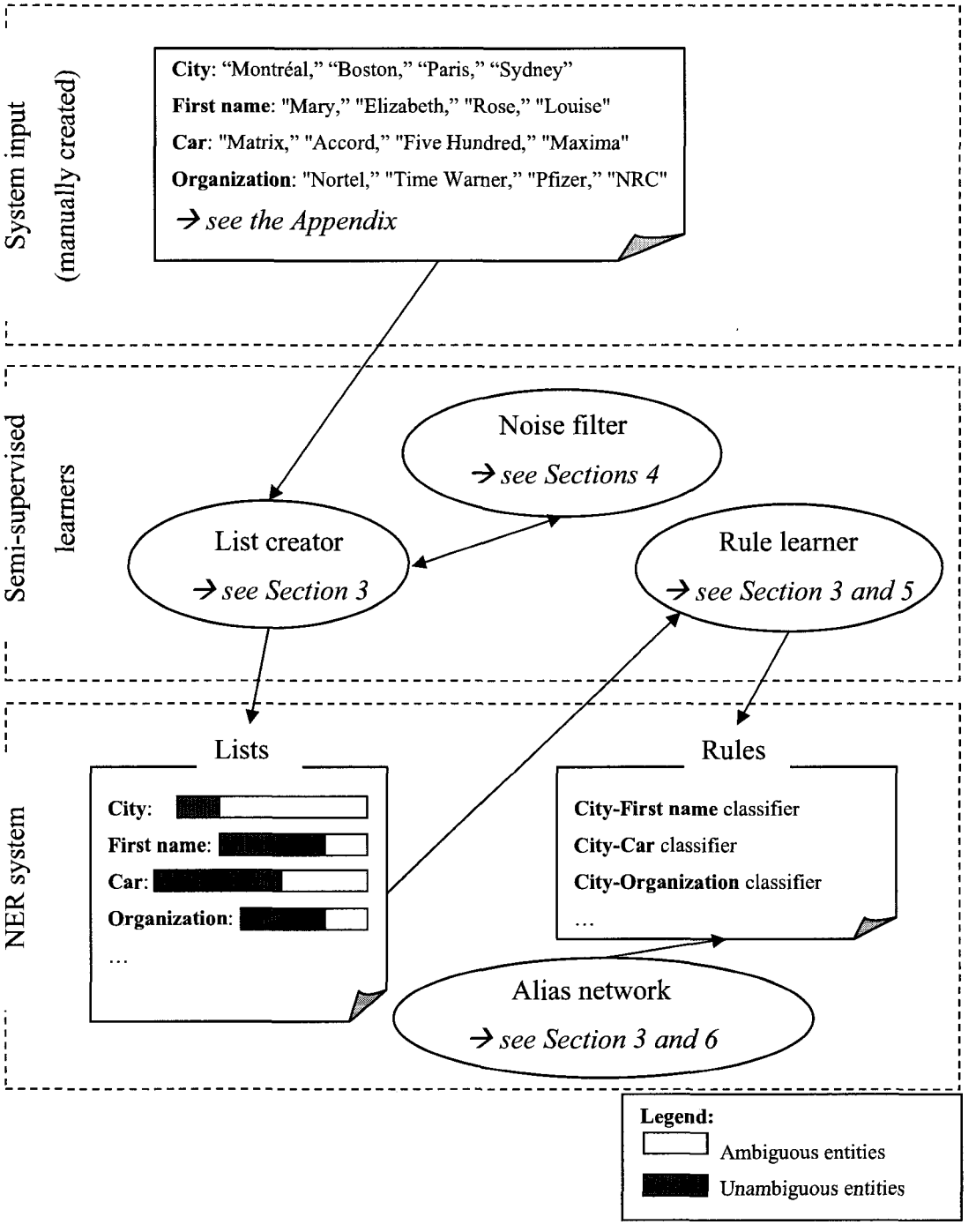

Figure 1: Overview of the semi-supervised NER system

Figure 1 has three main parts. The upper part is the system input that consists of a few examples for 100 entity types, as listed in the Appendix. This input constitutes very little supervision.

The middle part shows the semi-supervised modules. For instance, the "List Creator" module is explained in details in section 3.1 and it processes the system input, as illustrated by the arrow linking upper and middle parts of the Figure 1. The semi-supervised modules require no other manually created input. They however rely on very large corpora: the Web and a Terabyte-sized corpus of plain text (not shown in Figure 1).

The bottom part is the NER system, which is the program that can identify named entities in a given text. The modules of this system follow McDonald (1993) system division: Delimit, Classify, Record. "Lists" are use to delimit named entities, "Rules" are used to classify named entities, and an "Alias network" is used to record named entities.

The resulting semi-supervised system is in itself a significant contribution to and advance in the NER field. In addition, the proposed system implements state-of-the-art techniques from computational linguistics, semi-supervised machine learning, and statistical semantics. We claim four specific contributions to these fields:

1. The design of a baseline semi-supervised NER system (called BaLIE[1]) that performs at a level comparable to that of a simple supervised learning-based NER system (Chapter 3). The architecture of this system was published at Canadian AI 2006 (Nadeau et al. 2006).
2. The design of a noise filter for an NE list generation based on computational linguistic and statistical semantic techniques. The noise filter outperforms previous art (Chapter 4).
3. The demonstration of a simple strategy based on set intersections that enable the

[1] BaLIE is open source software released under GNU GPL: http://balie.sourceforge.net. A Web demo of BaLIE's NER is available at http://www.YooName.com.

identification of unambiguous examples for a given NE type (Chapter 5). Unambiguous NEs are a requirement for creating semi-supervised disambiguation rules.

4. An acronym detection algorithm—part of an alias resolution system—that outperforms previous art (Chapter 6), with experiments published at Canadian AI 2005 (Nadeau & Turney 2005).

These contributions are crucial components of a successful autonomous NER system, and they are best explained in the context of the whole system. We structured this thesis accordingly.

In Chapter 2, we introduce background work, related work, and NER applications. We give a formal definition of the NER task. Problems that are related to and may benefit from NER are discussed. Applications for both the research and industrial worlds are listed and presented. We also thoroughly survey fifteen years of research—from 1991 to 2006—in a systematic review published in a special issue of *Linguisticae Investigationes* (Nadeau & Sekine 2007).

In Chapter 3, we present BaLIE (Baseline Information Extraction), a system that learns to recognize NEs in an autonomous manner. BaLIE solves two limitations of rule-based and supervised NER systems. First, it requires no human intervention, such as manually labelling training data or creating gazetteers. Second, the system can handle more than the three classical named-entity types (person, location, and organization). System performances are shown to be comparable to that of a simple supervised learning-based system. Some significant details of the system were published in a regional French-language conference (Nadeau 2005b), and are translated and reported in this chapter.

Chapters 4, 5 and 6 are the core of this thesis. Here, we present extensions and improvements to BaLIE. Our contributions come from three hypotheses that were formulated to improve the baseline system of Chapter 3. First, we hypothesize that lexical features can improve noise-filtering techniques aimed at generating high-quality NE lists. We have included a

demonstration of this filter in Chapter 4. Then, we hypothesize that the differences between multiple NE lists are a set of unambiguous NE examples that are useful in learning disambiguation rules. We have included experiments that support this hypothesis in Chapter 5. Finally, we hypothesize that in the context of alias resolution, resolving acronyms would improve the system quality. An acronym detection algorithm is provided in Chapter 6.

Chapter 7 discusses the work that was accomplished in completing this thesis. It sheds light on the reasons behind BaLIE's design. More importantly, it discusses the limitations we faced at all stages of development, and the ideas we will retain in our future work. The thesis conclusion restates our contributions and summarizes the results of our experiments.

Chapter 2

Background and Related Work

The NER task consists of identifying the occurrences of some predefined phrase types in a text. Here is an example from Mikheev et al. (1999b), marked up with four entity types: <Date>, <Person>, <Organization>, and <Location>.

> On **<Date>**Jan 13th**</Date>**, **<Person>**John Briggs Jr**</Person>** contacted **<Organization>**Wonderful Stockbrockers Inc**</Organization>** in **<Location>**New York**</Location>** and instructed them to sell all his shares in **<Organization>**Acme**</Organization>**.

In the expression "Named Entity," the word "Named" aims to restrict the task to only those entities for which one or many rigid designators, as defined by Kripke (1982), stands for the referent. For instance, "the automotive company created by Henry Ford in 1903" is referred to as "Ford" or "Ford Motor Company." Rigid designators include proper names as well as certain natural terms, such as biological species and substances. There is a general agreement to include temporal expressions and some numerical expressions, such as money and measures in NEs. While some instances of these types are good examples of rigid designators (e.g. the year "2001"), there are also many invalid NEs (e.g., I take my vacations in "June"). In the first example, the year "2001" refers to the 2001st year of the Gregorian calendar. In the second example, "June" may refer to the month in an undefined year (past June, next June, June 2020, etc.). It can be argued that the NE definition is loosened in such cases for practical reasons.

The most common alternative formulation of the NER task is using speech as input (Favre et al. 2005). The task is considered more difficult since the capitalization of words, and generally the words themselves, are approximated by Automatic Speech Recognition (ASR) technologies. The same problem of degraded input arises when it comes from Optical Character Recognition (OCR) (Maynard et al. 2002). NER can also be done for semi-structured documents (e.g., HTML documents) (Kushmerick 1997). Supplemental

information is then available in a structure that may help recognize entity boundaries and/or entity types. However, textual context may be lost.

2.1 Related Work

In this section, we list some tasks related to NER. These tasks revolve around the notion of rigid designation, whereby the direct goal is not to recognize the named things from documents.

Personal name disambiguation (Mann & Yarowski 2003) is the task of identifying the correct referent of a given designator. In a given context, it may consist of identifying whether *Jim Clark* is the race driver, the film editor, or the Netscape founder. Corpus-wide disambiguation of personal names has applications in document clustering for information retrieval. In the work of Mann and Yarowski, it is used to create biographical summaries from corpora. This technology is about to be mainstream, with a new generation of people search engines, such as Zoominfo.com and Spock.com.

Identification of named entity descriptions (Radev 1998) is the identification of textual passages that describe a given NE. For instance, Bill Clinton is described as "the President of the U.S.," "the democratic presidential candidate" or "an Arkansas native," depending on the document. Description identification can be use as a cue in personal name disambiguation (see related work above). Radev's intention is to reuse these describers in the context of natural language generation.

Named entity translation (Fung 1995, Huang 2005) is the task of translating NEs from one language to another. For instance, the French translation of "National Research Council Canada" is "Conseil national de recherches Canada." NE translation is acknowledged as a major issue in machine translation as it may account for as much as 10% of translation errors (Vilar et al. 2006).

Analysis of name structure (Charniak 2001) is the identification of the parts in a person

name. For example, the name "Doctor Paul R. Smith" is composed of a person title, a first name, a middle name, and a surname. It is presented as a preprocessing step for NER and for the resolution of co-references to help determine, for instance, that "John F. Kennedy" and "President Kennedy" are the same person, while "John F. Kennedy" and "Caroline Kennedy" are two distinct persons.

Entity anaphora resolution (Dimitrov et al. 2002) mainly consists of resolving pronominal co-reference when the antecedent is an NE. For example, in the sentence "Rabi finished reading the book and he replaced it in the library," the pronoun "he" refers to "Rabi." Anaphora resolution can be useful in solving the NER problem itself by enabling the use of extended co-reference networks (see Section 3.2.3). Meanwhile it has many applications of its own, such as in "question answering" (e.g., answering "Who put the book in the library?").

Acronym identification (Nadeau & Turney 2005) is described as the identification of an acronym's definition (e.g., "IBM" stands for "International Business Machines") in a given document. The problem is related to NER because many organization names are acronyms (GE, NRC, etc.). Resolving acronyms is useful, again, to build co-reference networks aimed at solving NER (see Section 6.6). On its own, it can improve the recall of information retrieval by expanding queries containing an acronym with the corresponding definition.

Record linkage (Cohen & Richman 2001) is the task of matching named entities across databases. It involves the use of clustering and string matching techniques (Cohen & Sarawagi 2004) in order to map database entries having slight variations (e.g., *Frederick Mason* and *F. Mason*). It is used in database cleaning and in data mining on multiple databases.

Case restoration (Agbago et al. 2006) consists of restoring expected word casing in a sentence. Given a lower case sentence, the goal is to restore the capital letters usually appearing on the first word of the sentence and on NEs. This task is useful in machine translation, where a sentence is usually translated without capitalization information.

2.2 Applications

In this section, we list NER applications essentially built on having a textual document that identifies entities. We label these applications using three classifications: temporal (Temp) applications locate entities in time to analyze trends or calendar events; information retrieval (IR) applications are extensions of the search paradigm where the goal is access to relevant information in large corpora; and very large corpora (VLC) applications are based on annotating vast amounts of documents to allow information mining or to link information across documents, but not necessarily to access information.

[Temp] **Event detection** (e.g., Smith 2002) consists of detecting temporal entities in conjunction with other entities. For instance, conferences are usually made up of four parts: one conference name, one location, and two dates (e.g., name: "AAAI," location: "Boston," start date: "July 16th 2006," end date: "July 20th 2006").A person's birth or death is a person name and date pair (e.g., name: "John Lennon," date: "December 8th, 1980"). Smith uses event detection to draw maps where war locations and dates are identified.

[Temp, VLC] **Time varying entities analysis** (e.g., Swan & Allan 1999) is the analysis of key entities in a corpus at a given time or over a time period. It extends the event detection application in a significant manner either by intelligent aggregation or analysis. Swan and Allan extract events on multiple news, for a given topic, and they generate a story made of chosen textual passages. For instance, the story may relate that "France elects Jacques Chirac as president" on May 7th and "Jacques Chirac selects Alain Juppé as premier" on May 17th. Commercial "trend" or "buzz" analysis, that is, a simple analysis of entity frequencies over time, has already hit the market[2].

[IR] **Question answering** (e.g., Srihari & Li 1999) often involves NER at the core of the answering capabilities. A study by Srihari and Li shows that low-level information extraction like NER is often a necessary component in handling most types of questions. Out of 200

[2] BlogPulse, originally by Intelliseek, pioneered the idea: http://www.blogpulse.com (verified April 2007).

questions of TREC-8 competition, 80% asked for an NE as a response (e.g., who [person], when [time or date], where [location]).

[IR] **Semantic information retrieval** (e.g., Pasca 2004), unlike question answering, takes conventional Boolean queries, but returns something more than a list of Web documents. Pasca cites at least two "semantic" variants: returning a list of elements when the query is an entity category (e.g., "SAS," "SPSS," "Minitab," "BMDP" and "R" are returned for the query "statistical packages"); and returning a list of siblings when the query is an entity (e.g., returning "Morpheus," "Grokster" and "Napster" when the query is "Kazaa").

[IR] **Local search** (e.g., Wang et al. 2005) is the task of using location information expressed in a query (e.g., Ottawa restaurants) to return locally relevant results, such as a list of nearby restaurants. NER on queries, or any short text, is arguably more challenging than on long documents. Wang et al. proposes NER strategies for query strings. They mention that accurately and effectively detecting that a location is the true topic of a query has huge potential impact on increasing search relevance. Major commercial search engines are already offering local search prototypes.

[VLC] **Text/Web mining** (e.g., Sánchez & Moreno 2005) is the task of extracting implicit information from a large repository of documents. The goal is to extract knowledge from the mass of information that is unavailable in isolated documents. In the work of Sánchez and Moreno, NEs of the medical domains are extracted from a large corpus to build ontological knowledge. Those ontologies, in turn, may support collection browsing and classification.

2.3 Observations: 1991 to 2006

Computational research aiming at automatically identifying NEs in texts forms a vast and heterogeneous pool of strategies, methods, and representations. One of the first research papers in the field was presented by Lisa F. Rau (1991) at the 7th IEEE Conference on Artificial Intelligence Applications. Rau's paper describes a system to "extract and recognize [company] names." It relies on heuristics and handcrafted rules. From 1991 (1 publication) to

1995 (we found 8 publications in English), the publication rate remained relatively low. It accelerated in 1996, with the first major event dedicated to the task: MUC-6 (Grishman & Sundheim 1996). It has not decreased since, with steady research and numerous scientific events: HUB-4 (Chinchor et al. 1998); MUC-7 and MET-2 (Chinchor 1999); IREX (Sekine & Isahara 2000); CONLL (Tjong Kim Sang 2002, Tjong Kim Sang & De Meulder 2003); ACE (Doddington et al. 2004); and HAREM (Santos et al. 2006). The Language Resources and Evaluation Conference (LREC)[3] has also been staging workshops and main conference tracks on the topic since 2000.

2.3.1 Language Factor

A good proportion of work in NER research is devoted to the study of English, but a possibly larger proportion addresses language independence and multilingualism problems. German is well studied in CONLL-2003 and in earlier works. Similarly, Spanish and Dutch are strongly represented, and were boosted as the focus of a major conference: CONLL-2002. Japanese has been studied in the MUC-6 conference, the IREX conference, and other works. Chinese is studied in abundant literature (e.g., Wang et al. 1992, Chen & Lee 1996, Yu et al. 1998), and so are French (Petasis et al. 2001, Poibeau 2003), Greek (Boutsis et al. 2000), and Italian (Black et al. 1998, Cucchiarelli & Velardi 2001). Many other languages received some attention as well: Basque (Whitelaw & Patrick 2003), Bulgarian (Da Silva et al. 2004), Catalan (Carreras et al. 2003), Cebuano (May et al. 2003), Danish (Bick 2004), Hindi (Cucerzan & Yarowsky 1999, May et al. 2003), Korean (Whitelaw & Patrick 2003), Polish (Piskorski 2004), Romanian (Cucerzan & Yarowsky 1999), Russian (Popov et al. 2004), Swedish (Kokkinakis 1998), and Turkish (Cucerzan & Yarowsky 1999). Portuguese was examined (Palmer & Day 1997) and, at the time this survey was written, the HAREM conference was revisiting that language. Finally, Arabic (Huang 2005) has started to receive a lot of attention in large-scale projects such as Global Autonomous Language Exploitation (GALE)[4].

---

[3] http://www.lrec-conf.org/

[4] http://projects.ldc.upenn.edu/gale/

2.3.2 Textual Genre or Domain Factor

The impact of textual genre (journalistic, scientific, informal, etc.) and domain (gardening, sports, business, etc.) has been rather neglected in NER literature. Few studies are specifically devoted to diverse genres and domains. Maynard et al. (2001) designed a system for emails, scientific texts, and religious texts. Minkov et al. (2005) created a system specifically designed for email documents. Perhaps unsurprisingly, these experiments demonstrated that although any domain can be reasonably supported, porting a system to a new domain or textual genre remains a major challenge. For instance, Poibeau and Kosseim (2001) tested some systems on both the MUC-6 collection made up of newswire texts, and on a proprietary corpus made up of manual phone conversation translations and technical emails. They report a drop in performance for every system (some 20% to 40% of precision and recall).

2.3.3 Entity Type Factor

Early work formulates the NER problem as recognizing "proper names" in general (e.g., Coates-Stephens 1992, Thielen 1995). Overall, the most studied entity types are three specializations of "proper names": names of "persons," "locations," and "organizations." These types are collectively known as "enamex" since the MUC-6 competition. The "location" type can, in turn, be divided into multiple subtypes of "fine-grained locations" (Fleischman 2001, Lee & Geunbae Lee 2005). Similarly, "fine-grained person" sub-categories, like "politician" and "entertainer," appear in the work of Fleischman and Hovy (2002). The "person" type is quite common and used at least once in an original way by Bodenreider and Zweigenbaum (2000), who combine it with other cues for extracting medication and disease names (e.g., "Parkinson disease"). In the ACE program, the "facility" type subsumes entities of the "location" and "organization" types. The "GPE" type is used to represent a location that has a government, such as a city or a country.

The "miscellaneous" type is used in the CONLL-2002 and 2003 conferences, and includes proper names falling outside the classic "enamex." The class is also sometimes augmented

with the "product" type (e.g., Bick 2004). The "timex" (another term coined in MUC) "date" and "time" types, and the "numex" "money" and "percent" types are also quite predominant in the literature. Since 2003, a community named TIMEX2 (Ferro et al. 2005) has proposed an elaborated standard for annotating and normalizing temporal expressions. Finally, marginal types are sometime handled for specific needs: "film" and "scientist" (Etzioni et al. 2005); "email address" and "phone number" (Witten et al. 1999, Maynard et al. 2001); "research area" and "project name" (Zhu et al. 2005); "book title" (Brin 1998, Witten et al. 1999); "job title" (Cohen & Sarawagi 2004); and "brand" (Bick 2004).

A recent interest in bioinformatics, and the availability of the GENIA corpus (Ohta et al. 2002) led to many studies dedicated to types such as "protein," "DNA," "RNA," "cell line" and "cell type" (e.g., Shen et al. 2003, Settles 2004), as well as studies exclusively targeted at "protein" recognition (Tsuruoka & Tsujii 2003). Related works also include "drug" (Rindfleisch et al. 2000) and "chemical" (Narayanaswamy et al. 2003) names.

Some work does not limit the possible types to extract and is referred to as "open domain" NER (See Alfonseca & Manandhar 2002, Evans 2003). In this line of work, Sekine and Nobata (2004) defined a named entity hierarchy, which includes many fine grained subcategories, such as international organization, river, or airport, and adds a wide range of categories, such as product, event, substance, animal, religion, or colour. It tries to cover most frequent name types and rigid designators appearing in a newspaper. The number of categories is about 200, and they are now defining popular attributes for each category to make it ontological.

2.3.4 What's Next?

Recent researches in multimedia indexing, semi-supervised learning, complex linguistic phenomena, and machine translation suggest some new directions for the field. On one side, there is a growing interest in multimedia information processing (e.g., video, speech), particularly extracting NE from it (Basili et al. 2005). Much effort is also invested toward semi-supervised and unsupervised approaches to NER, motivated by the use of very large collections of texts (Etzioni et al. 2005) and the possibility of handling multiple NE types

(Nadeau et al. 2006). Complex linguistic phenomena (e.g., metonymy, acronym resolution, conjunction handling) that are common shortcomings of current systems are under investigation (e.g., Poibeau 2006). Finally, large-scale projects such as GALE, discussed in Section 2.3.1, pave the way for integrating NER and machine translation for mutual improvement, and more generally, multilingual NER (Steinberger and Pouliquen 2007).

2.4 Techniques and Algorithms to Resolve the NER Problem

The ability to recognize previously unknown entities is an essential part of NER systems. Such ability hinges upon recognition and classification rules triggered by distinctive modeling features associated with positive and negative examples. While early studies were mostly based on handcrafted rules, most recent ones use supervised machine learning (SL), as a way to automatically induce rule-based systems or sequence labelling algorithms, starting from a collection of training examples. In the research community, this is evidenced by the fact that five out of eight systems were rule-based in the MUC-7 competition, while the sixteen systems involved in CONLL-2003 were based on supervised learning techniques. When training examples are not available, handcrafted rules systems remain the preferred technique, as shown in Sekine and Nobata (2004), who developed an NER system for 200 entity types.

The idea of supervised learning is to study the features of positive and negative examples of NE over a large collection of annotated documents and design rules that capture instances of a given type. Section 2.4.1 explains SL approaches in more detail. The main shortcoming of SL is the requirement of a large annotated corpus. The unavailability of such resources and the prohibitive cost of creating them lead to two alternative learning methods: semi-supervised learning (SSL); and unsupervised learning (UL). These techniques are presented in Section 2.4.2 and Section 2.4.3, respectively.

2.4.1 Supervised Learning

The current dominant technique for addressing the NER problem is supervised learning. SL techniques include Hidden Markov Models (HMM) (Bikel et al. 1997), Decision Trees

(Sekine 1998), Maximum Entropy Models (ME) (Borthwick 1998), Support Vector Machines (SVM) (Asahara & Matsumoto 2003), and Conditional Random Fields (CRF) (McCallum & Li 2003). These are all variants of the SL approach, which typically feature a system that reads a large annotated corpus, memorizes lists of entities, and creates disambiguation rules based on discriminative features.

A baseline SL method that is often proposed consists of tagging test corpus words when they are annotated as entities in the training corpus. The performance of the baseline system depends on the vocabulary transfer, which is the proportion of words, without repetition, appearing in both training and testing corpus. Palmer and Day (1997) calculated the vocabulary transfer on the MUC-6 training data. They report a transfer of 21%, with the repetition of much as 42% of location names, but only 17% of organizations and 13% of person names. Vocabulary transfer is a good indicator of the recall (number of entities identified over the total number of entities) of the baseline system, but it is also a pessimistic measure since some entities are frequently repeated in documents. Mikheev et al. (1999) precisely calculated the baseline system recall on the MUC-7 corpus. They report a recall of 76% for locations, 49% for organizations, and 26% for persons, with precision ranging from 70% to 90%. Whitelaw and Patrick (2003) report consistent results on MUC-7 for the aggregated enamex class. For the three enamex types together, the recognition precision is 76% and the recall is 48%.

2.4.2 Semi-Supervised Learning

The term "semi-supervised" (or "weakly supervised") is relatively recent. The main technique for SSL is called "bootstrapping" and involves a small degree of supervision, such as a set of seeds, for starting the learning process. For example, a system aimed at "disease names" might ask the user to provide a small number of example names. Then, the system searches for sentences that contain these names and tries to identify some contextual clues common to the five examples. Then, the system tries to find other instances of disease names appearing in similar contexts. The learning process is then reapplied to the newly found examples, so as to discover new relevant contexts. By repeating this process, a large number of disease names and a large number of contexts will eventually be gathered. Recent

experiments in semi-supervised NER (Nadeau et al. 2006) report performances that rival baseline supervised approaches. Here are some examples of SSL approaches

Brin (1998) uses lexical features implemented by regular expressions in order to generate lists of book titles paired with book authors. It starts with seed examples such as {Isaac Asimov, The Robots of Dawn} and use some fixed lexical control rules such as the following regular expression, [A-Z][A-Za-z .,&]^{5,30}[A-Za-z.], used to describe a title. The main idea of his algorithm, however, is that many Web sites comply with a reasonably standardized format throughout the site. When a given Web site is found to contain seed examples, new pairs can often be identified using simple constraints, such as the presence of identical text before, between, or after the elements of an interesting pair. For example, the passage "The Robots of Dawn, by Isaac Asimov (Paperback)" would allow one to find, on the same Web site, "The Ants, by Bernard Werber (Paperback)."

Collins and Singer (1999) parse a complete corpus in search of NE pattern candidates. A pattern is, for example, a proper name (as identified by a part-of-speech tagger) followed by a noun phrase in apposition (e.g., "Maury Cooper, a vice president at S&P"). Patterns are kept in pairs {spelling, context} where "spelling" refers to the proper name and "context" refers to the noun phrase in its context. Starting with an initial seed of spelling rules (e.g., rule 1: if the spelling is "New York" then it is a Location; rule 2: if the spelling contains "Mr." then it is a Person; rule 3: if the spelling is all capitalized then it is an organization), the candidates are examined. Candidates that satisfy a "spelling" rule are classified accordingly, and their "contexts" are accumulated. The most frequent contexts found are turned into a set of contextual rules. Following the steps above, contextual rules can be used to find further spelling rules, and so on. Collins and Singer (1999) and Yangarber et al. (2002) demonstrate the idea that learning several types of NE simultaneously enables finding negative evidence (one type against all) and reduces over-generation. Cucerzan and Yarowsky (1999) also use a similar technique and apply it to many languages.

Riloff and Jones (1999) introduce mutual bootstrapping, which consists of growing a set of entities and a set of contexts in turn. It is a looser version of Collins and Singer's (1999) idea.

Instead of working with predefined NE candidates (found using a fixed syntactic construct), they start with a handful of seed entity examples of a given type (e.g., Bolivia, Guatemala, and Honduras are entities of the "country" type) and accumulate all patterns found around these seeds in a large corpus. Contexts (e.g., offices in X, facilities in X, etc.) are ranked and used to find new examples. Riloff and Jones note that the performance of that algorithm can deteriorate rapidly when noise penetrates the entity list or pattern list. While they report relatively low precision and recall in their experiments, their work proved to be highly influential.

Cucchiarelli and Velardi (2001) use syntactic relations (e.g., subject-object) to discover more accurate contextual evidence around the entities. Again, this is a variant of Riloff and Jones mutual bootstrapping (1999). Interestingly, instead of using human-generated seeds, they rely on existing NER systems (called "early NE classifier") for initial NE examples.

Pasca et al. (2006) also use techniques inspired by mutual bootstrapping. However, they innovate by using Lin's (1998) distributional similarity to generate synonyms—or, more generally, words belonging to the same semantic class—allowing pattern generalization. For instance, in the pattern "X was born in November," Lin's synonyms for "November" are {March, October, April, Mar, Aug., February, Jul, Nov., etc.}, thus allowing the induction of new patterns such as "X was born in March." One of Pasca et al.'s contributions is to apply this technique to very large corpora (100 million Web documents) and demonstrate that starting from a seed of 10 sample facts (defined as "person" type entities paired with "year" type entities, standing for the person's year of birth), it is possible to generate one million facts with a precision of about 88%.

Unlabelled data selection is a problem Heng and Grishman (2006) address. They demonstrate that an existing NE classifier can be improved using bootstrapping methods. The main lesson they report is that relying on large collections of documents is not sufficient on its own. Selecting documents using information retrieval-like relevance measures, as well as selecting specific contexts that are rich in proper names and co-references, bring the best results in their experiments.

2.4.3 Unsupervised Learning

The typical approach in unsupervised learning is clustering. For example, one can try to gather NEs from clustered groups based on context similarity. There are also other unsupervised methods. Basically, the techniques rely on lexical resources (e.g., WordNet), on lexical patterns, and on statistics computed on a large unannotated corpus. Here are some examples.

Alfonseca and Manandhar (2002) study the problem of labelling an input word with an appropriate NE type. NE types are taken from WordNet (e.g., location>country, animate>person, animate>animal, etc.). The approach is to assign a topic signature to each WordNet synset by merely listing words that frequently co-occur with it in a large corpus. Then, given an input word in a given document, the word context (words appearing in a fixed-size window around the input word) is compared to type signatures and classified under the most similar one.

In Evans (2003), the method for identification of hyponyms/hypernyms described in the work of Hearst (1992) is applied to identify potential hypernyms of capitalized word sequences appearing in a document. For instance, when X is a capitalized sequence, the query "such as X" is searched on the Web and, in the retrieved documents, the noun that immediately precedes the query can be chosen as the X hypernym. Similarly, in Cimiano and Völker (2005), Hearst patterns are used, but this time, the feature consists of counting the number of occurrences of passages like "city such as," "organization such as," etc.

Shinyama and Sekine (2004) observed that NEs often appear in several news articles synchronously, whereas common nouns do not. They found a strong correlation between being an NE, and appearing intermittently and simultaneously in multiple news sources. This technique allows for identifying rare NEs in an unsupervised manner, and it can be useful when combined with other NER methods.

In Etzioni et al. (2005), Pointwise Mutual Information and Information Retrieval (PMI-IR) is

used as a feature to assess that a named entity can be classified under a given type. PMI-IR, developed by Turney (2001), measures the dependence between two expressions using Web queries. A high PMI-IR means that expressions tend to co-occur. Etzioni et al. create features for each entity candidate (e.g., London) and a large number of automatically generated discriminator phrases, like "is a city," "nation of," etc.

2.5 Feature Space for NER

Features are describers or characteristic attributes of words designed for algorithmic consumption. An example of a feature is a Boolean variable with the "true" value if a word is capitalized and "false" if not. Feature vector representation is an abstraction of text where each word is typically represented by one or many Boolean, numeric, and nominal values. For example, a hypothetical NER system may represent each word in a text with 3 attributes:

1) a Boolean attribute with the "true" value if the word is capitalized and "false" if not;
2) a numeric attribute corresponding to the length of the word, in characters;
3) a nominal attribute corresponding to the lower case version of the word.

In this scenario, the sentence "The president of Apple eats an apple," excluding the punctuation, would be represented by the following feature vectors:

```
<true,  3, "the">,  <false, 9, "president">, <false, 2, "of">, <true,  5,
"apple">, <false, 4, "eats">, <false, 2, "an">, <false, 5, "apple">
```

Usually, the NER problem is resolved by applying a rule system over the features. For instance, a system might have two rules, a recognition rule ("capitalized words are entity candidates") and a classification rule ("the type of entity candidates of length greater than 3 words is organization"). These rules work well for the exemplar sentence above. However, real systems tend to be much more complex, and their rules are often created by automatic learning techniques.

In this section, we present the features most often used for the recognition and classification

of named entities. We organize them along three different axes: word-level features; list look-up features; and document and corpus features.

2.5.1 Word-Level Features

Word-level features are related to the character makeup of words. They specifically describe word case, punctuation, numerical value, and special characters. Table 1 lists subcategories of word-level features.

Table 1: Word-level features

Features	Examples
Case	- Starts with a capital letter - Word is all upper case - The word is mixed case (e.g., ProSys, eBay)
Punctuation	- Ends with period, has internal period (e.g., St., I.B.M.) - Internal apostrophe, hyphen or ampersand (e.g., O'Connor)
Digit	- Digit pattern (see below) - Cardinal and ordinal - Roman number - Word with digit (e.g., W3C, 3M)
Character	- Possessive mark, first person pronoun - Greek letters
Morphology	- Prefix, suffix, singular version, stem - Common ending (*see below*)
Part-of-speech	- proper name, verb, noun, foreign word
Function	- Alpha, non-alpha, n-gram (*see below*) - lower case, upper case version - pattern, summarized pattern (*see below*) - token length, phrase length

Digit pattern

Digits can express a wide range of useful information such as dates, percentages, intervals, identifiers, etc. Special attention must be given to some particular patterns of digits. For

example, two-digit and four-digit numbers can stand for years (Bikel et al. 1997), and when followed by an "s," they can stand for a decade; one and two digits may stand for a day or a month (Yu et al. 1998).

Common word ending

Morphological features are essentially related to a word's affixes and roots. For instance, a system may learn that a human profession often ends in "ist" (e.g., journalist, cyclist) or that nationality and languages often ends in "ish" and "an" (e.g., Spanish, Danish, Romanian). Other examples of common word endings are organization names that end in "ex," "tech," and "soft" (Bick 2004).

Functions over words

Features can be extracted by applying functions over words. An example is given by Collins and Singer (1999), who create a feature by isolating the non-alphabetic characters of a word (e.g., non-alpha [A.T.&T.] = ..&.). Another example is given by Patrick et al. (2002), who use character n-grams as features.

Patterns and summarized patterns

Pattern features were introduced by Collins (2002) and then used by others (Cohen & Sarawagi 2004 and Settles 2004). Their role is to map words onto a small set of patterns over character types. For instance, a pattern feature might map all upper-case letters to "A," all lower-case letters to "a," all digits to "0," and all punctuation to "-":

```
x = "G.M.": GetPattern(x) = "A-A-"
x = "Machine-223": GetPattern(x) = "Aaaaaaa-000"
```

The summarized pattern feature is a condensed form of the above, in which consecutive character types are not repeated in the mapped string. For instance, the preceding examples become:

```
x = "G.M.": GetSummarizedPattern(x) = "A-A-"
x = "Machine-223": GetSummarizedPattern(x) = "Aa-0"
```

2.5.2 List Look-Up Features

Lists are the privileged features in NER. The terms "gazetteer," "lexicon," and "dictionary" are often used interchangeably with the term "list." List inclusion is a way to express the relation "is a" (e.g., "Paris is a city"). It may appear obvious that if a word (Paris) is an element of a list of cities, then the probability that this word is a city, in a given text, is high. However, because of word polysemy, the probability is almost never 1 (e.g., the probability of "Fast" representing a company is low because "fast" as a common adjective is much more frequent).

Table 2: List look-up features

Features	Examples
General list	- General dictionary (see below)
	- Stop words (function words)
	- Capitalized nouns (e.g., January, Monday)
	- Common abbreviations
List of entities	- Organization, government, airline, educational
	- First name, last name, celebrity
	- Astral body, continent, country, state, city
List of entity cues	- Typical words in organization (see below)
	- Person title, name prefix, post-nominal letters
	- Location typical word, cardinal point

In Table 2, we present three significant list categories used in literature. We could enumerate many more examples of lists, but we decided to concentrate on those aimed at recognizing enamex types.

General dictionary

Common nouns listed in a dictionary are useful, for instance, in the disambiguation of capitalized words in ambiguous positions (e.g., sentence beginning). Mikheev (1999) reports that in a given corpus, from 2,677 words in ambiguous position, a general dictionary look-up can identify 1841 common nouns out of 1851 (99.4%), while discarding only 171 NEs out of 826 (20.7%). In other words, in that corpus, 20.7% of NEs are ambiguous as common nouns.

Words that are typical of organization names

Many authors propose to recognize organizations by identifying words that are frequently used in their names. For instance, knowing that "associates" is frequently used in organization names could lead to the recognition of "Computer Associates" and "BioMedia Associates" (McDonald 1993, Gaizauskas et al. 1995). The same rule applies to frequent first words ("American," "General") of an organization (Rau 1991). Some authors also exploit the fact that organizations often include a person's name (Wolinski et al. 1995, Ravin & Wacholder 1996), as in "Alfred P. Sloan Foundation." Similarly, geographic names can be good indicators of an organization name (Wolinski et al. 1995), as in "France Telecom." Organization designators such as "Inc." and "Corp" (Rau 1991) are also useful features.

On list look-up techniques

Most approaches implicitly require word candidates to match at least one element of a pre-existing list exactly. However, we may want to allow some flexibility in the match conditions. The NER field uses at least three alternate look-up strategies.

First, words can be stemmed (stripping off both inflectional and derivational suffixes) or lemmatized (normalizing for inflections only) before they are matched (Coates-Stephens 1992). For instance, if a list of cue words contains "technology," the inflected form "technologies" will be considered as a successful match. For some languages (Jansche 2002), diacritics can be replaced by their canonical equivalent (e.g., "é" replaced by "e").

Second, a word candidate can be "fuzzy-matched" against the reference list using some kind of thresholded edit-distance (Tsuruoka & Tsujii 2003) or Jaro-Winkler (Cohen & Sarawagi 2004). This captures small lexical variations in words that are not necessarily derivative or inflectional. For instance, "Frederick" could match "Frederik" because the edit-distance between the two words is very small (suppression of just one character, the "c"). Jaro-Winkler's metric was specifically designed to match proper names following the observation that the first letters tend to be correct, while name ending often varies.

Third, the reference list can be accessed using the Soundex algorithm (Raghavan & Allan 2004), which normalizes word candidates to their respective Soundex codes. This code is a combination of the first letter of a word plus a three-digit code that represents its phonetic sound. Hence, similar sounding names like "Lewinskey" (Soundex = l520) and "Lewinsky" (Soundex = l520) are equivalent with respect to their Soundex code.

2.5.3 Document and Corpus Features

Document features are defined by both document content and structure. Large collections of documents (corpora) are also excellent sources of features. In this section, we list features that go beyond the single-word and multi-word expressions, and include meta-information about documents and corpus statistics.

Table 3: Features from documents

Features	Examples
Multiple occurrences	- Other entities in the context - Upper-case and lower-case occurrences (see below) - Anaphora, co-reference (see below)
Local syntax	- Enumeration, apposition - Position in sentence, in paragraph, and in document
Meta-information	- Uri, email header, XML section, (see below) - Bulleted/numbered lists, tables, figures
Corpus frequency	- Word and phrase frequency - Co-occurrences - Multi-word unit permanency (see below)

Multiple occurrences and multiple casing

Thielen (1995), Ravin and Wacholder (1996), and Mikheev (1999) identify words that appear both in upper-case and lower-case form in a single document. These words are hypothesized as common nouns that appear both in ambiguous (e.g., sentence beginning) and unambiguous position.

Entity co-reference and alias

The task of recognizing the multiple occurrences of a unique entity in a document dates back to the earliest research in the field (McDonald 1993, Rau 1991). Co-references are the occurrences of a given word or word sequence referring to a given entity within a document. Deriving features from co-references is mainly done by exploiting the context of every occurrence (e.g., Macdonald was the first, Macdonald said, was signed by Macdonald, etc.). Aliases of an entity are the various ways in which the entity is written in a document. For instance, we may have the following aliases for a given entity: Sir John A. Macdonald, John A. Macdonald, John Alexander Macdonald, and Macdonald. Deriving features from aliases is mainly done by leveraging the union of alias words (Sir, John, A, Alexander, Macdonald).

Finding co-references and aliases in a text can be reduced to the same problem of finding all occurrences of an entity in a document. This problem is of great complexity. Gaizauskas et al. (1995) use 31 heuristic rules to match multiple occurrences of company names. For instance, two multi-word expressions match if one is the initial subsequence of the other. An even more complex task is recognizing the mention of an entity documents. Li et al. (2004) propose and compare a supervised and unsupervised model for this task. They propose the use of word-level features engineered to handle equivalences (e.g., prof. is equivalent to professor), and relational features to encode the relative order of tokens between two occurrences.

For complex problems such as metonymy—the use of different words to refer to the same entity (e.g., "Hexagon" stands for "France") —word-level features are often insufficient. Poibeau (2006) demonstrates that in such cases, semantic tagging is a key issue.

Document meta-information

Most meta-information about documents can be used directly: email headers are good indicators of person names; news often starts with a location name; etc. Some authors make original use of meta-information. Zhu et al. (2005) uses document URL to bias entity probabilities. For instance, many names (e.g., bird names) have a high probability of being a

"project name" if the URL is from a computer science department domain.

Statistics for multi-word units

Da Silva et al. (2004) propose some interesting feature functions for multi-word units that can be thresholded using corpus statistics. For example, they establish a threshold on the presence of rare and long lower-case words in entities. Only multi-word units that do not contain rare lower-case words (rarity calculated as relative frequency in the corpus) of a relatively long size (meaning size calculated from the corpus) are considered as NE candidates. They also present a feature called permanency, which consists of calculating the a word's frequency (e.g., Life) within a corpus, divided by its frequency in case insensitive form (e.g., life, Life, LIFE, etc.)

2.6 Evaluation of NER

Thorough evaluation of NER systems is essential to their progress. Many techniques were proposed to rank systems based on their capacity to annotate a text like an expert linguist. In the following section, we take a look at three main scoring techniques used for the MUC, IREX, CONLL, and ACE conferences. First, let's summarize the task from an evaluation point of view.

In NER, systems are usually evaluated based on how their output compares with the output of human linguists. For instance, here's an annotated text marked up according to MUC guidelines. Let's call it the "solution."

```
Unlike <ENAMEX TYPE="PERSON">Robert</ENAMEX>, <ENAMEX TYPE="PERSON">John
Briggs Jr</ENAMEX> contacted <ENAMEX TYPE="ORGANIZATION">Wonderful
Stockbrockers Inc</ENAMEX> in <ENAMEX TYPE="LOCATION">New York</ENAMEX> and
instructed them to sell all his shares in <ENAMEX
TYPE="ORGANIZATION">Acme</ENAMEX>.
```

Now, let's hypothesize a system producing the following output:

```
<ENAMEX TYPE="LOCATION">Unlike</ENAMEX> Robert, <ENAMEX
TYPE="ORGANIZATION">John Briggs Jr</ENAMEX> contacted Wonderful <ENAMEX
TYPE="ORGANIZATION">Stockbrockers</ENAMEX> Inc <TIMEX TYPE="DATE">in New
York</TIMEX> and instructed them to sell all his shares in <ENAMEX
TYPE="ORGANIZATION">Acme</ENAMEX>.
```

The system produced five different errors[5], explained in Table 4. In this example, the system gives one correct answer: (`<Organization> Acme </Organization>`). Ultimately, the question is "What score should we give this system?" In the following sections, we survey how the question was answered in various evaluation forums.

Table 4: NER error types

Correct solution	System output	Error
On	<Location> On </Location>	The system hypothesized an entity where there is none.
<Person> Robert </Person>	Robert	An entity was completely missed by the system.
<Person> John Briggs Jr </Person>	<Organization> John Briggs Jr </Organization>	The system noticed an entity but gave it the wrong label.
<Organization> Wonderful Stockbrockers Inc </Organization>	<Organization> Stockbrockers </Organization>	A system noticed there is an entity but got its boundaries wrong.
<Location> New York </Location>	<Date> in New York </Date>	The system gave the wrong label to the entity and got its boundary wrong.

[5] Type of errors are from an informal publication http://nlpers.blogspot.com/2006/08/doing-named-entity-recognition-dont.html

2.6.1 MUC Evaluations

In MUC events (Grishman & Sundheim 1996, Chinchor 1999), a system is scored on two axes: its ability to find the correct type (TYPE); and its ability to find exact text (TEXT). A correct TYPE is credited if an entity is assigned the correct type, regardless of boundaries as long as there is an overlap. A correct TEXT is credited if entity boundaries are correct, regardless of the type. For both TYPE and TEXT, three measures are kept: the number of correct answers (COR); the number of actual system guesses (ACT); and the number of possible entities in the solution (POS).

The final MUC score is the micro-averaged f-measure (MAF), which is the harmonic mean of precision and recall calculated over all entity slots on both axes. A micro-averaged measure is performed on all entity types without distinction (errors and successes for all entity types are summed together). The harmonic mean of two numbers is never higher than the geometric mean. It also tends toward the lesser number, minimizing the impact of large outliers and maximizing the impact of small ones. The f-measure therefore tends to favour balanced systems.

In MUC, precision is calculated as COR/ACT and the recall is COR/POS. For the previous example, COR = 4 (2 TYPE + 2 TEXT), ACT = 10 (5 TYPE + 5 TEXT), and POS = 10 (5 TYPE + 5 TEXT). The precision is therefore 40%, the recall is 40%, and the MAF is 40%.

This measure has the advantage of taking into account all possible types of errors of Table 4. It also gives partial credit for errors occurring on only one axis. Since there are two evaluation axes, each complete success is worth two points. The worst errors cost these same two points (missing both TYPE and TEXT), while other errors cost only one point.

2.6.2 Exact-Match Evaluations

IREX and CONLL share a simple scoring protocol. We can call it "exact-match evaluation." Systems are compared based on the micro-averaged f-measure (MAF), with the precision being the percentage of NEs found by the system that are correct, and the recall being the

percentage of NEs in the solution that are found by the system. An NE is correct only if it is an exact match with the corresponding entity in the solution.

For the previous example, there are 5 true entities, 5 system guesses, and only one guess that exactly matches the solution. The precision is therefore 20%, the recall is 20%, and the MAF is 20%.

For some applications, the constraint of an exact match is unnecessarily stringent. For instance, in some bioinformatics work, the goal is to determine whether or not a particular sentence mentions a specific gene and its function. Exact NE boundaries are not required: all the information needed to determine if the sentence refers to the gene is there (Tzong-Han Tsai et al. 2006).

2.6.3 ACE Evaluation

ACE has a complex evaluation procedure. It includes mechanisms for dealing with various evaluation issues (partial match, wrong type, etc.). The ACE task definition is also more elaborate than previous tasks at the NE "subtypes" and "class" levels, as well as entity mentions (co-references), and more, but these supplemental elements will be ignored here.

Basically, each entity type has weight parameters and contributes up to a maximum proportion (MAXVAL) of the final score (e.g., if each person is worth 1 point and each organization is worth 0.5 point, then it takes two organizations to counterbalance one person in the final score). According to ACE parameters, some entity types such as "facility" may account for as little as 0.05 points. In addition, customizable costs (COST) are used for false alarms, missed entities, and type errors. Partial matches of textual spans are only allowed if the NE head matches on at least a given proportion of characters. Temporal expressions are not treated in ACE since they are evaluated by the TIMEX2 community (Ferro et al. 2005).

The final score called Entity Detection and Recognition Value (EDR) is 100% minus the

penalties. In the Table 4 examples, the EDR score is 31.3%. It is computed as follows, using ACE parameters from 2004[6]. Each of the five entities contributes up to a maximum value to the final score. Using default ACE parameters, the maximum values (MAXVAL) for person entities is 61.54% of the final score, the two organizations worth 30.77%, and the location worth 7.69%. These values sum up to 100%. At the individual type level, one person span was recognized (John Briggs Jr) but with the wrong type (organization); one person entity was missed (Robert); the two organization spans (Wonderful Stockbrockers Inc and Acme) were considered correct, even with the former partial matches; one geopolitical span was recognized (in New York) but with the wrong type; and there was one false alarm (On). Globally, the person entities error (function of COST and MAXVAL) accounts for 55.31% of the final EDR loss (30.77 for the miss and 24.54 for the type error), the false alarm account for 5.77% of loss, and the location type error accounts for 7.58%. The final EDR of 31.3% is 100% minus these losses.

ACE evaluation may be the most powerful evaluation scheme because of its customizable error cost and its wide coverage of the problem. However, it is problematic because the final scores are only comparable within fixed parameters. In addition, complex methods are not intuitive and make error analysis difficult.

2.7 Conclusion

The NER field has been thriving for more than fifteen years. It aims to extract from text and to classify rigid designators mentions such as proper names, biological species, and temporal expressions. In this chapter, we presented related works and applications of NER. We have also shown the diversity of languages, domains, textual genres, and entity types covered in the literature. More than twenty languages and a wide range of named entity types are studied. However, most of the work has concentrated on limited domains and textual genres, such as news articles and Web pages.

[6] http://www.nist.gov/speech/tests/ace/ace04/index.htm

We have also provided an overview of the techniques employed to develop NER systems, documenting the recent trend away from hand-crafted rules towards machine learning approaches. Handcrafted systems provide good performance at a relatively high system engineering cost. When supervised learning is used, the availability of a large collection of annotated data is a prerequisite. Such collections are available from the evaluation forums but remain rather rare and limited in domain and language coverage. Recent studies in the field have explored semi-supervised and unsupervised learning techniques that promise fast deployment for many entity types without the prerequisite of an annotated corpus. We have listed and categorized the features that are used in recognition and classification algorithms. The use of an expressive and varied set of features turns out to be just as important as the choice of machine learning algorithms. Finally we have also provided an overview of the evaluation methods that are in use in the major forums of the NER research community. We saw that in a simple example made up of only five NEs, the score of three different evaluation techniques varies from 20% to 40%.

NER will have a profound impact on our society. Early commercial initiatives are already modifying the way we use yellow pages by providing local search engines (search your neighborhood for organizations, product and services, people, etc.). NER systems also enable monitoring trends in the huge space of textual media produced every day by organizations, governments, and individuals. It is also at the basis of a major advance in biology and genetics, allowing researchers to search an abundance of literature for interactions between named genes and cells.

Chapter 3

Creating a Baseline Semi-Supervised NER System

In this chapter, we demonstrate BaLIE, a system that learns to recognize named entities in an autonomous manner. To gain NER capabilities, BaLIE also features a tokenizer, a sentence boundary detector, a language guesser, and a part-of-speech tagger. These modules were developed using well-known techniques, and we do not consider them major contributions. More details are available in a technical report (Nadeau 2005a). Frunza et al. (2005) made a significant contribution by adding Romanian language support to BaLIE. However, in this chapter we exclusively cover BaLIE's NER module.

BaLIE solves two common limitations of rule-based and supervised NER systems. First, it requires no human intervention such as manually labelling training data or creating gazetteers. Second, the system can handle more than the three classical named-entity types (person, location, and organization). The chapter is structured around one contribution:

- The design of a baseline semi-supervised NER system that performs at a level comparable to that of a simple supervised learning-based NER system.

This chapter covers the "List Creator" module along with primitive version of the "Rule learner" and the "Alias Network" of Figure 1. The resulting baseline system has therefore all the required functionalities defined in the McDonald's (1993) "Delimit, Classify, Record" paradigm. Figure 2 expands these modules and details the process of training and evaluating them.

BaLIE builds on past work in unsupervised NER by Collins and Singer (1999) and Etzioni et al. (2005). Our goal is to create a system that can recognize NEs in a given document without prior training (supervised learning) or manually constructed gazetteers. (For our purposes, the terms "gazetteer" and "named-entity list" are interchangeable.)

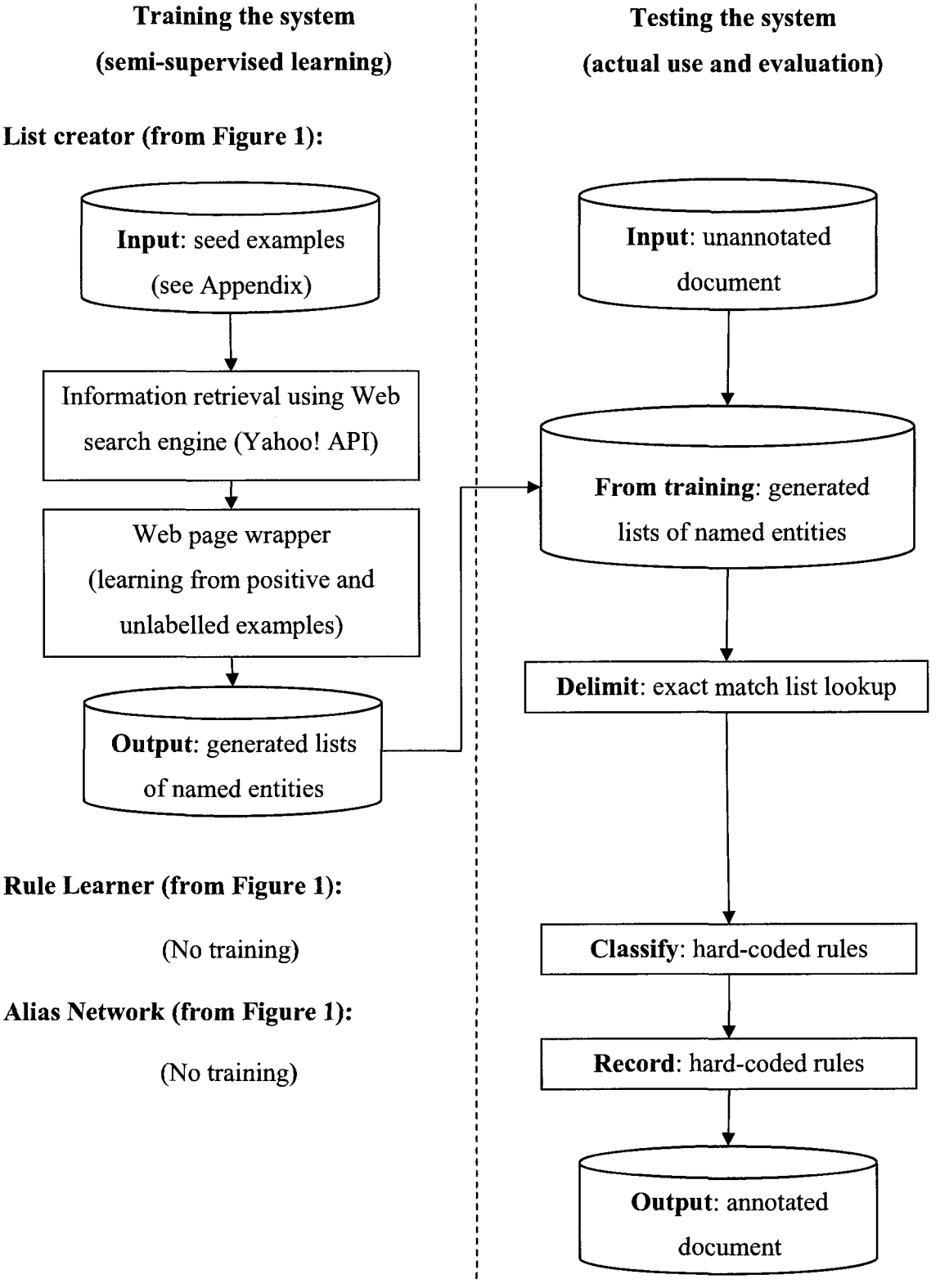

Figure 2: Details of the baseline named entity recognition system

Collins and Singer's system exploits a large corpus to create a generic list of proper names (NEs of arbitrary and unknown types). Proper names are gathered by looking for syntactic patterns with specific properties. For instance, within a noun phrase, a proper name is a sequence of consecutive words that are tagged as NNP or NNPS by a part-of-speech tagger, and in which the last word is identified as the head of the noun phrase. Like Collins and Singer, we use a large corpus to create NE lists, but we present a technique that can exploit diverse types of text, including text without proper grammatical sentences, such as tables and lists (marked up with HTML).

Etzioni et al. refer to their algorithm as an NE "extraction" system. It is not intended for NE "recognition." In other words, it is used to create large lists of NEs, but it is not designed for resolving ambiguity in a given document. The distinction between these tasks is important. It might seem that having a list of entities on hand makes NER trivial. One can extract city names from a given document merely by searching it for each city name in a city list. Still, this strategy often fails because of ambiguity. For example, consider the words "It" (a city in the state of Mississippi and a pronoun) and "Jobs" (a person's surname and a common noun). The task addressed by Etzioni et al. could be called "automatic gazetteer generation." Without ambiguity resolution, a system cannot perform robust, accurate NER. This claim is supported by the experiments we present in Section 3.2

In this chapter, we propose an NER system that combines NE extraction with a simple form of NE disambiguation. We use some simple yet highly effective heuristics, based on the work of Mikheev (1999), Petasis et al. (2001), and Palmer and Day (1997), to perform NE disambiguation. Using the MUC-7 NER corpus (Chinchor 1999), we compare the performance of our unsupervised system with that of a basic supervised system. We also show that our technique is general enough to be applied to other NE types, such as car brands or bridge names. To support this claim, we include an experiment with car brands.

The chapter is divided as follows. First, we present the system architecture. The system is made up of two modules. The first one, presented in Section 3.1, is used to create large gazetteers of entities, such as a list of cities (the "List creator" of Figure 1). The second

module, presented in Section 3.2, uses simple heuristics to identify and classify and record entities in the context of a given document (primitive versions of the "Rule Learner" and "Alias Network" of Figure 1). We compare BaLIE's performance with a supervised baseline system on the MUC-7 corpus in Section 3.3. Next, in Section 3.4, we show that the system can handle other type of entities in addition to the classic three (person, location, and organization). We discuss the degree of supervision in Section 3.5. We conclude in Section 3.6 by arguing that our system advances the state-of-the-art of NER by avoiding the need for supervision and by handling novel types of NEs. The system's source code is available under the GPL license at http://balie.sourceforge.net. A Web demo of BaLIE's NER is available at http://www.YooName.com.

3.1 Generating Gazetteers

The task of automatically generating lists of entities has been investigated by several researchers. In Hearst (1999), the studied lexical patterns can be used to identify nouns from the same semantic class. For instance, a noun phrase that follows the pattern "the city of" is usually a city. In Riloff and Jones (1999), a small set of lexical patterns and entities are developed using mutual bootstrapping. Finally, Lin and Pantel (2001) show how to create large clusters of semantically related words using an unsupervised technique. Their idea is based on examining words with similar syntactic dependency relationships. They show they can generate semantic classes such as car brands, drugs, and provinces. However, their technique does not discover the labels of the semantic classes, which is a common limitation of clustering techniques.

The algorithm of Etzioni et al. (2005) outperforms all previous methods for creating a large list of a given type of entity or semantic class: the task of automatic gazetteer generation. In the remainder of this section, we explain how to generate a list of thousands of cities from only a few seed examples, in two steps (Section 3.1.1 and 3.1.2) repeated if necessary.

3.1.1 Retrieve Pages with a Seed

The first step is information retrieval from the Web. We used the Yahoo! Web search engine

(through the developer API). A query is composed of k manually chosen entities (e.g., "Montreal" AND "Boston" AND "Paris" AND "Mexico City"). In our experience, when k is set to 4 (as suggested by Etzioni et al. 2005) and the seed entities are common city names, the query generally retrieves Web pages that contain many names of cities, in addition to the seed names. The basic idea of the algorithm is to extract these additional city names from each retrieved Web page. In the query, less than four entities results in lower precision, and more than four entities results in lower recall.

The same strategy can be applied to person names, company names, car brands, and many other types of entities. In Chapter 5 of this thesis, we present the resulting list generation for 100 entity types.

3.1.2 Apply Web Page Wrapper

The second step is to apply a Web page wrapper that acts as an abstract layer over HTML whose goal is to isolate desired information. Given it is provided with the location of a subset of the desired information within a page, the wrapper isolates the entire set of desired information and hides the remainder of the page. The goal of the wrapper is therefore to hide everything in the page but the named entities that are likely to be in HTML structures similar to that of the seed names. This step is explained in greater details in section 3.1.4.

3.1.3 Repeat

The two steps above (3.2.1, 3.2.2) are repeated as needed. Each iteration brings new entities that are added to the final gazetteer. At each iteration, k new randomly chosen entities are used to refresh the seed for the system. Entities are chosen from the gazetteer under construction. Preference is given to seed entities that are less likely to be noise, such as those appearing in multiple Web pages.

3.1.4 Detailed algorithm for Web page wrapping

Learning to isolate desired information on a Web page starting with a few seed examples is

an instance of learning from positive and unlabelled data. A Web page is encoded in a tree structure, where the top node <html> contains the entire page. The HTML nodes containing the desired information are labelled "positive," and other nodes are unlabelled. For instance, in the following HTML code, the <a> node that contains the city name "Ottawa" is the desired information and is labelled "positive" for the purpose of training the wrapper.

```
<tr>
<td>Day5</td>
<td><img src="bullet.gif">
<a href="vacation.htm" label="positive">Ottawa</a> </td>
<td>Ottawa, Museum of Civilization: Morning drive to Canada's capital city,
Ottawa. This afternoon visit the Canadian Museum of Civilization…</td>
</tr>
```

Identifying all the relevant nodes in a Web page is a classification problem (to show or to hide a node). Eighteen features are used to describe a node within the HTML tree. In Cohen and Fan (1999), the learning algorithm in use is Ripper (Cohen, 1995). In comparison with the original Cohen and Fan set of features, we dropped three features that seem redundant or less pertinent and we added two novel features.

An important improvement on the original approach is the addition of two new features with a significant predictive power. These features describe the nodes by row and column number of the innermost table to which they belong.

3.1.4.1 Web page wrapper attributes

We describe all attributes and assert their type as either "numeric" (real value) or "nominal" (set of predefined values).

```
Tag name: nominal {div, td, img, p, a, …}[7]
Text length: numeric
Non-white text length: numeric
Recursive text length: numeric
```

[7] This enumeration of nominal values should contain every valid HTML tag.

```
Recursive non-white text length: numeric
Depth: numeric
Normalized[8] depth: numeric
Number of children: numeric
Normalized number of children: numeric
Number of siblings: numeric
Normalized number of siblings: numeric
Parent tag name: nominal {div, td, img, p, a, …}
Node prefix count: numeric
Normalized node prefix count: numeric
Node suffix count: numeric
Normalized node suffix count: numeric
Cell row in innermost table: numeric
Cell column in innermost table: numeric
Class: {Positive, Negative}
```

Here is the description of a typical HTML node using this representation:

```
a,6,0,0,4002,26,0.684211,8,0.222222,1,0.027778,td,104,0.514851,0,0,2,1,
Positive
```

This instance describes a positive node. The tag name is "a" and parent is "td." The node wraps around six immediate characters, but there are zero characters embedded in children nodes. Among other features, the value of the "node prefix count" means that 104 other nodes in the page share the same "prefix" (e.g., html>body>table>tr>td) . Other noteworthy features indicate that the cell containing this node is in row 2, column 1, of the innermost table.

3.1.4.2 Web page wrapper as classification rules

We designed a Web page wrapper as a rule-based system that identifies the location of specific types of information within the Web page. For example, a wrapper for isolating the location of city names on craigslist.org Web site might contain the following rule: "A city name is contained in an HTML node of type <a>, with text length between 4 and 20 characters, in the first or second column of the a table of depth 2, and with at least 20 other

[8] Refer to W. Cohen and Fan (1999) for normalization issues and information on each attribute.

nodes in the page that satisfy the same rule."

The gazetteer generation algorithm proceeds by learning rules that identify the locations of named entity examples. The Web page wrapper is trained on the *k*-positive examples (from Section 3.1.1) that are known to appear in the page, but only if they are strictly contained in an HTML node (e.g., <td> Boston </td>). The advantage of this constraint is that HTML tags act as named entity boundary delimiters. It allows identifying complex named entities such as "<td> Saint-Pierre and Miquelon </td>" without additional parsing.

It is also possible train the wrapper on nodes containing a small amount of text around a named entity within an HTML node (e.g., <td> Boston hotel </td>). A technique, that we called "node cleaning" is described in Nadeau (2005b) and is presented in section 3.1.4.6.

The remaining HTML nodes are unlabelled: some are positive, some are negative but we can't separate them at this point. Our strategy is to treat the unlabelled nodes in the page as if they were negative examples, but we only include in the negative set the nodes with the same HTML tags as the positive examples. For instance, if the positive *k* nodes are tagged as bold (i.e., "<b>"), then the negative examples will be restricted to the Web page's remaining bold text. All other nodes are hidden by default.

As described above, Web page wrapping is a classification problem. A supervised learning algorithm is used to classify unknown entities in the current Web page. In this application, the training and testing sets are the same. The learning algorithm is trained on the given Web page, then the learned model is applied to reclassify the text in the same Web page. The idea is to learn rules, during training, that identify the locations of the known entities (the seed entities) and can be applied, during testing, to identify entities appearing in similar contexts, which may provide more positive examples.

Three main problems make this task difficult. First, there is noise in the training data class labels, since everything but the seed words are initially labelled as negative. If the page contains more than *k* entities of the desired type, the very nodes we want to extract were

labelled as negative.

The second problem is the class imbalance. Along with the positive *k* examples, there are usually hundreds or thousands of negative examples. These two problems are handled by noise-filtering and wise data sampling, respectively. At this point, our technique goes beyond the system of Etzioni et al. (2005), which uses a simple Web page wrapper consisting of handcrafted rules.

Interestingly, the first and second problems are typical of learning that uses only positive examples. In the Web page wrapper, the positive examples are the initial seeds. We solved the first and second problems by under-sampling and then over-sampling the data set. The notion of using these types of sampling to force focused learning is described by Chawla et al. (2002) in an algorithm called SMOTE. The Web page wrapper uses a SMOTE-like algorithm. In Section 4.2.2, we'll use the exact same strategy to guide learning in another one-class problem.

The third problem is the residual noise, that is, invalid entity candidates that pass through the Web page wrapper and are added to the final lexicon. We discuss the three problems in much detail in the following subsections.

3.1.4.3 Class noise problem

To handle the problem of noise in the class labels, we use a filtering approach inspired by Zhu et al. (2003). The noise-filtering strategy is to simply remove any instance similar to a positive instance. We say that two nodes are similar when their feature vectors are identical, except for the text length feature. Removing class noise is a kind of "wise" under-sampling of negative examples.

The noise filter is not used on the testing set. When the trained model is applied to the testing set, some of the examples that were absent in training may be classified as positive, while others may be classified as negative.

We evaluated this technique on 40 Web pages retrieved from 9 "k-city queries" (i.e., queries composed of k names of city). These pages were found by manually verifying the 100 first hits for each query and keeping all pages in which all the queried city names are exactly contained in an HTML node. Using the class noise filter, a mean of 42% of the HTML nodes that are initially labelled as negative are removed from the training set; thus significantly under sampling the initial dataset. When testing the Web page wrapper on the 40 manually annotated Web pages with class noise filtering, the performance of finding city names improves by 30% (from 65% accuracy to 84.8%).

3.1.4.4 Class imbalance

To handle the problem of class imbalance, we use over-sampling of positive examples. Using the original unbalanced data set, the wrapper is almost incapable of extracting new entities. It mainly guesses the majority class (negative) and only extracts the initial seed from Web pages. To discourage the learning algorithm from using the trivial solution of always guessing the majority class, the positive examples are over-sampled to rebalance the data set. This rebalancing must be done for each individual Web page, to take into account the imbalance ratio of each wrapper. Rebalancing is performed automatically by randomly choosing HTML nodes to add to the data set, up to the desired ratio of positive to negative examples.

Past research suggests that supervised learning algorithms work best when the ratio is near 1:1 (Ling & Li, 1998). We hypothesized that the wrapper would work best when we rebalanced the data set by duplicating positive instances until the ratio reached 1:1.

On the dataset presented in the previous section, positive example over-sampling provides an additional 2% gain in accuracy. When used alone, that is without class noise filtering, over-sampling accounts for up to 8% of improvement in classification accuracy.

3.1.4.5 Residual noise problem

Web page wrapper frequently extracts invalid candidates from pages. For instance, it may extract table headers, wrong lists, or simply extract elements of a heterogeneous list of valid

and invalid entities (e.g., drug names mixed with symptoms and disease names).

In the baseline system, in order to filter noise, we used hard-coded rules. For each entity types, we defined a minimum and maximum length, a valid set of characters, and an absolute minimum redundancy (number of times the entity is extracted from distinct Web pages). In Chapter 4, we demonstrate advanced noise filtering based on semi-supervised techniques.

3.1.4.6 HTML node cleaning

In section 3.1.4.2, we set the constraint that HTML nodes must exactly embed named entity examples so we don't need additional boundary delimitation inside the node. However, this is not always the case. A significant amount of web pages presents the desired information with extra words inside the node (e.g., <td> New York Hotels </td>).

If these extra words are present in all positive nodes, we apply the wrapper algorithm and post-process newly found named entity by removing the constant noise. In our experiments, we found that this simple technique augments the number of named entity found significantly. It allows finding 76% more city names, for instance, in the list of Table 6, and 17% more car brand names in the list of Section 3.4.

3.2 Resolving Ambiguity

The "list look-up strategy" is the method of performing NER by scanning through a given input document to look for terms that match a list entry. The list look-up strategy has three main problems: entity-noun ambiguity errors (Section 3.2.1); entity boundary detection errors (Section 3.2.2); and entity-entity ambiguity errors (Section 3.2.3). Due to these three problems, the gazetteer-generating module presented in Section 3.1 is not in itself adequate for reliable NER. We found heuristics in the literature to tackle each of these problems.

3.2.1 Entity-Noun Ambiguity

Entity-noun ambiguity occurs when an entity is the homograph of a noun. The plural word "jobs" and the surname "Jobs" is an example of this occurrence. To avoid this problem,

Mikheev (1999) proposes the following heuristic: in a given document, assume that a capitalized word or phrase (e.g., "Jobs") is a named-entity, unless it sometimes appears in the document without capitals (e.g., "jobs"); it only appears at the start of a sentence or at the start of a quotation (e.g., "Jobs that pay well are often boring."); or it only appears inside a sentence in which all words with more than three characters start with a capital letter (e.g., a title or section heading). This heuristic is called H1 in the remainder of this chapter.

3.2.2 Entity Boundary Detection

A common problem with the list look-up strategy involves errors in recognizing where an NE begins and ends in a document (e.g., finding only "Boston" in "Boston White Sox"). This can happen when an NE is composed of two or more words (e.g., "Jean Smith") that are each listed separately (e.g., "Jean" as a first name and "Smith" as a last name). It can also happen when an entity is surrounded by unknown capitalized words (e.g., "New York Times" as an organization followed by "News Service" as an unlisted string). Palmer and Day (1997) propose the longest match strategy for these cases. Accordingly, we merge all consecutive entities of the same type and every entity with any adjacent capitalized words. We did not, however, merge consecutive entities of different types, since we would not have known the resulting type. This heuristic is called H2 in the remainder of this chapter.

The rule above is general enough to be applied independently of the entity type. We found that other merging rules could improve the precision of our system, such as "create a new 'organization' type entity by merging a location followed by an organization." However, we avoided rules like this because we believe that this kind of manual rule engineering results in brittle, fragile systems that do not adapt well to new data. Our goal is to make a robust, portable, general-purpose NER system, with minimally embedded domain knowledge.

3.2.3 Entity-Entity Ambiguity

Entity-entity ambiguity occurs when the string standing for an NE belongs to more than one type. For instance, if a document contains the "France" NE, it could be either the name of a person or the name of a country. For this problem, Petasis et al. (2001) and others propose

that at least one occurrence of the NE should appear in a context where the correct type is clearly evident. For example, in the context "Dr. France," it is clear that "France" is the name of a person.

We could have used cues, such as professional titles (e.g., farmer), organizational designators (e.g., Corp.), personal prefixes (e.g., Mr.) and personal suffixes (e.g., Jr.), but as discussed in the preceding section, we avoided this kind of manual rule engineering.

Definitions:

D = a given input document.

$A = \{a_1,...,a_n\}$ = the set of all sets of aliases in the document D.

$a_i = \{e_1,...,e_m\}$ = a set of aliases = a set of different entity instances, referring to the same actual entity in the world.

$e = \langle D, s, p \rangle$ = a unique instance of an NE, consisting of a string s in document D at position p.

overlap(e_i, e_j) = a Boolean function; returns **true** when $e_i = \langle D, s_i, p_i \rangle$ and $e_j = \langle D, s_j, p_j \rangle$ and the strings s_i and s_j share at least one word with more than three characters; returns **false** otherwise.

Algorithm:

Let $A = \{\}$.

For each instance of an NE e in document D:

If there is exactly one alias set a_i with a member e_j such that overlap(e, e_j), then modify A by adding e to a_i.

If there are two or more alias sets a_i, a_j with members e_k, e_l such that overlap(e, e_k) and overlap(e, e_l), then modify A by creating a new alias group a_p that is the union of a_i, a_j, and $\{e\}$, add a_p to A, and remove a_i and a_j from A.

Otherwise, create a new alias set a_q, consisting of $\{e\}$, and add a_q to A.

Figure 3: Simple alias resolution algorithm

Instead, we applied a simple alias resolution algorithm, presented in Figure 3. When an

ambiguous entity is found, its aliases are used in two ways. First, if a member of an alias set is unambiguous, it can be used to resolve the whole set. For instance, "Atlantic ocean" is clearly a location, but "Atlantic" can be either a location or an organization. If both belong to the same alias set, then we assume that the whole set is a "location" type. Another way of using the alias resolution is to include unknown words in the model. Typical unknown words are introduced by the heuristics in Section 3.2.2. If an entity (e.g., "Steve Hill") is formed from a known entity (e.g., "Steve") and an unknown word (e.g., "Hill"), we allow occurrences of this unknown word to be added in the alias group. This heuristic is called H3 in the remainder of this chapter.

3.3 Evaluation with the MUC-7 Enamex Corpus

In the Message Understanding Conferences (MUC), the NER track focuses on three classic NE types: person, location, and organization. These three NE types are collectively called "enamex." In this section, we compare the performance of our system with a baseline supervised system using the corpus from MUC-7. For this experiment, a portion of the corpus is given to the supervised system in order to train it. Our unsupervised system simply ignores this portion of the corpus.

The same baseline experiment was conducted on the MUC-6 and MUC-7 corpora by Palmer and Day (1997) and Mikheev et al. (1999), respectively. Their systems work as follows. A training corpus is read, and the tagged entities are extracted and listed. Given a testing corpus, the lists are used in a simple look-up strategy, so that any string that matches a list entry is classified accordingly.

Table 5 presents Mikheev et al.'s results on the MUC-7 corpus (in the "Learned lists" columns). There is also a comparison with a system that uses handmade lists of common entities (in the "Common lists" columns). The "Combined lists" columns are based on a combination of both approaches. These results are also published experiments by Mikheev et al. In the following tables, "re" is the recall, "pr" is the precision, and "f" is the f-measure (the harmonic mean of precision and recall), all expressed in percentages.

Table 5: Results of a supervised system for MUC-7

	Learned lists			Common lists			Combined lists		
	re	Pr	f	re	pr	f	re	pr	f
organization	49	75	59	3	51	6	50	72	59
person	26	92	41	31	81	45	47	85	61
location	76	93	84	74	94	83	86	90	88

For the purpose of comparison, we ran our system on the MUC-7 corpus using the gazetteers we generated, as described in Section 3.1. We generated gazetteers for some of the NE subtypes given by Sekine and Nobata (2004). The generated gazetteers are described in Table 6. We also used a special list of the months of the year because we noticed they were an abnormally important source of noise on the development (dry run) set[9]. Many months are also valid as personal first names.

Table 6: Type and size of gazetteers built using Web page wrapper

Gazetteer	Size
Location: city	14,977
Location: state/province	1,587
Location: continent/country/island	781
Location: waterform	541
Location: astral body	85
Organization: private companies	20,498
Organization: public services	364
Organization: schools	3,387
Person: first names	35,102
Person: last names	3,175
Person: full names	3,791
Counter-examples: months	12

List size depends on how efficiently the Web page wrapper extracts entities. Section 3.3.1

[9] It can be argued that the month list is a form of manual rule engineering, contrary to the principles discussed in Section 3.2.2. We decided to use it because most of the noise was clearly corpus-dependent, since each article contains a date header. For results without the month list, subtract 5% from the "person" type precision.

puts forth an experiment suggesting that these lists have a precision of at least 90%. We did not restrict Web mining to a specific geographic region, and we did not enforce strict conditions for list elements. As a result, the "state/province" list contains elements from around the world (not just Canada and the U.S.), and the "first name" list contains a multitude of compound first names although, as explained in Section 3.2.2, our algorithm is designed to capture them by merging sequences of first names.

Table 7 shows the result of a pure list look-up strategy, based on our generated gazetteers (in the "Generated lists" columns). For the sake of comparison, the table also shows the best supervised results from Table 5 (in the "Mikheev combined lists" columns). The results we report in previous tables are all based on the MUC-7 held-out formal corpus.

Table 7: Supervised list creation vs. unsupervised list creation techniques

	Mikheev combined lists			Generated lists		
	re	pr	F	re	pr	f
organization	50	72	59	70	52	60
person	47	85	61	59	20	30
location	86	90	88	83	31	45

We believe this comparison gives a good sense of the characteristics of both approaches. The supervised approach is quite precise but its recall is lower, since it cannot handle rare entities. The unsupervised approach benefits from large gazetteers, which make for higher recall at the cost of lower precision.

The case of locations is interesting. There is evidence of a substantial vocabulary transfer between the training data and the testing data, which allows the supervised method to have an excellent recall on the unseen texts. Mikheev's lists get a high recall with a list of only 770 locations. The supervised method benefits from highly repetitive location names in the MUC corpus.

These results are slightly misleading. The MUC scoring software that produces these

measures allows partial matching. This means that if a system tags the expression "Virgin Atlantic" when the official annotated key is "Virgin Atlantic Group," it will be interpreted as a success. In Table 8, we provide another view of the system's performance, which may be less misleading. For our system, it gives the precision and recall of all entity types at the "text" level; that is, how well it finds exact string matches.

Table 8: Generated list performance on text matching

	Generated lists		
	re	Pr	f
text	61	29	39

The next step in our evaluation consists of adding the heuristics presented in sections 3.2.1 to 3.2.3. These heuristics are designed to be unsupervised; that is, they require no training (unlike n-gram contexts, for example), and they are not deduced from our domain knowledge about a specific entity type. Table 9 demonstrates the contribution of each heuristic. The "Generated lists" columns are copied from Table 7 and Table 8, to show the performance of the list look-up strategy without disambiguation.

Table 9: Performance of heuristics to resolve NE ambiguity

	Generated lists			H1 (Entity-noun ambiguity)			H1 + H2 (Entity boundary)			H1 + H2 + H3 (Entity-entity ambiguity)		
	re	pr	f	re	pr	f	Re	pr	f	re	pr	f
org.	70	52	60	69	73	71	69	74	71	71	75	73
per.	59	20	30	58	53	55	66	63	64	83	71	77
loc.	83	31	45	82	69	75	81	77	79	80	77	78
text	61	29	39	61	57	59	72	72	72	74	72	73

The contribution of each heuristic is additive. H1 (Section 3.2.1) procures a dramatic improvement in precision with negligible loss of recall. The main source of ambiguity is entity-noun homographs such as "jobs," "gates," and "bush."

Heuristic H2 (Section 3.2.2) provides small gains in precision and recall of individual entity types (the first three rows in Table 9). As explained, these scores are misleading because they count partial matches, thus these scores are not sensitive to the boundary detection errors corrected by H2. However, the text matching performance is greatly improved (last row in Table 9). We noticed that most corrected boundaries are attributed to person entities composed of a known first name and an unlisted capitalized string, presumably standing for the surname.

H3 (Section 3.2.3) mainly increases precision and recall for "person" type NEs, due to the alias resolution algorithm. An occurrence of a full person name is usually unambiguous, so it can help to annotate isolated surnames, which are often either ambiguous (confused with organization names) or simply unlisted strings.

3.3.1 List Precision

In Nadeau (2005b), we evaluated the precision of a list of 17,065 automatically generated city names. We sampled 100 names randomly and counted a precision penalty for each noisy entry. The list precision is 97% (by the Binomial Exact Test, the 95% confidence interval is 91.4% to 99.4%).

We did this again with lists created for Chapter 5 experiments: city, first name, clothing brand, and song title. Again, we sampled 100 examples randomly and calculated the precision from there. Table 10 reports population sizes and estimated precisions.

Table 10: Estimated precision of automatically generated lists

List	Population size	Precision	95% confidence interval
City	15,500	97.0%	91.4% - 99.4%
First name	40,000	99.0%	94.6% - 99.9%
Clothing brand	799	98.0%	93.0% - 99.8%
Song title	5,900	99.0%	94.6% - 99.9%

3.4 Evaluation with Car Brands

There are many more NE types than those three classic enamex types. Sekine and Nobata (2004) propose a hierarchy of 200 NE types. Evans (2003) proposes a framework to handle such a wide variety. His approach is based on lexical patterns, inspired by Hearst (1992). He paired this technique with a heuristic for handling ambiguity in capitalized words. Our system is similar, but it is based on a method proven to give better entity-finding recall.

In this section, we show how well the system recognizes car brands. Intuitively, it seems that this type would be easier to handle than something like "person," which has an almost infinite extension. Yet recognizing car brands poses many difficulties. Car brands can be confused with common nouns (e.g., Focus, Rendez-Vous, Matrix, Aviator) and with company names (e.g., "Ford" versus "Ford Motor Company"). Another difficulty is the fact that new car brands are created every year, so it is challenging to keep a gazetteer up-to-date.

We created a small pilot corpus composed of news specifically about cars from some popular news feeds (CanWest, National Post, and The Associated Press). We use eight documents, for a total of 5,570 words and 196 occurrences of car brands.

The Web page wrapper technique was used to generate a list of 5,701 car brands, and the heuristics of sections 3.2.1 to 3.2.3 were applied without any modifications. Table 11 reports the results.

Table 11: System performance for car brand recognition

	Generated list			H1, H2 and H3		
	Re	pr	f	re	pr	f
cars	86	42	56	85	88	86
text	71	34	46	79	83	81

The performance on this task is comparable to that of the enamex. Without ambiguity resolution (in the "Generated list" columns), the precision is low, usually under 50%. This is the impact of frequent and ambiguous words like "will" (Toyota Will), and noise in our list (e.g., new, car, fuel). The ambiguity resolution algorithms (in the "H1, H2, and H3"

columns) increase the precision to above 80%. The remaining recall errors are due to rare car brands (e.g., "BMW X5 4.8is" or "Ford Edge"). The remaining precision errors are due to organization-car ambiguity (e.g., "National" as in "National Post" versus "Chevrolet National") and noise in the list (e.g., Other, SUV). We believe that the good performance of gazetteer generation, combined with ambiguity resolution on an entirely new domain, emphasizes their domain-independent quality and shows the strength of the unsupervised approach.

3.5 Supervised versus Unsupervised

We describe our system as unsupervised, but the distinction between supervised and unsupervised systems is not always clear. In some systems that are apparently unsupervised, it could be argued that the human labour involved in generating labelled training data has merely been shifted to embedding clever rules and heuristics in the system.

In our gazetteer generator (Section 3.1), the supervision is limited to a seed of four entities per list, a primitive noise filter (Section 3.1.4.5), the knowledge that month-person ambiguity is particularly problematic in MUC-7 (Section 3.3, Table 6) and three heuristics (Section 3.2) for handling entity ambiguity and adjusting entity boundaries. In our ambiguity resolver (Section 3.2), we attempt to minimize the use of domain knowledge of specific entity types. Our system exploits human-generated HTML mark-up in Web pages to generate gazetteers. However, because Web pages are available in such a quantity, and because the creation of Web pages is now intrinsic to the work-flow of most organization and individuals, we believe this annotated data comes at a negligible cost. For these reasons, we believe it is reasonable to describe our system as unsupervised.

3.6 Conclusion

In this chapter, we presented a named-entity recognition system that advances the NER state-of-the-art by avoiding the need for supervision and by handling novel NE types. In a comparison on the MUC corpus, our system outperforms a baseline supervised system, but it is still not competitive with more complex supervised systems. There are fortunately many

ways to improve our model. One interesting way would be to generate gazetteers for a multitude of NE types (e.g., all 200 of Sekine's types), and use list intersection as an indicator of ambiguity. This idea would not resolve the ambiguity itself, but it would clearly identify where to invest further efforts.

Chapter 4

Noise-Filtering Techniques for Generating NE Lists

In this chapter, we present a first improvement to BaLIE. It comes from the observation that entities of a given type tend to be lexically similar, in that they are comparable in length, they are made up of characters from a given character set, they often have common prefixes and suffixes, and so forth. We therefore formulated the hypothesis that lexical features are useful in identifying valid instances of an NE type. Our contributions are the following:

- The design of a noise filter for NE list generation based on lexical features;
- First experiments in using statistical semantics as noise filter.

This chapter covers the "Noise filter" that is an improvement to the "List creator" module of Figure 1. The noise filter works on the output of the Web page wrapper module of BaLIE in order to generate NE lists of greater quality, as shown in Figure 4. Both the noise filter we present in this chapter and the Web page wrapper presented in the previous chapter are instances of the problem of learning from positive and unlabelled examples. In both case, we use an algorithm inspired by SMOTE (Chawla et al. 2002) to solve the problem. SMOTE is reviewed in Section 4.2.2.

NE lists—also called dictionaries, lexicons, or gazetteers—are a typical component of NER systems. Lists are either an explicit system component (e.g., Cunningham et al. 2002), or they are derived from an annotated training data set (e.g., Bikel et al. 1999). For instance, a typical NER system that recognizes city names will refer to a list of cities and apply a mechanism to resolve entity boundary and type ambiguity. However, lists are rarely exhaustive and they require ongoing maintenance to stay up-to-date. This is particularly true with NE types such as "company," which are very volatile. Moreover, the initial cost of creating a list of NEs is usually high because it either requires manual NE harvesting, or manually annotating a large collection of documents.

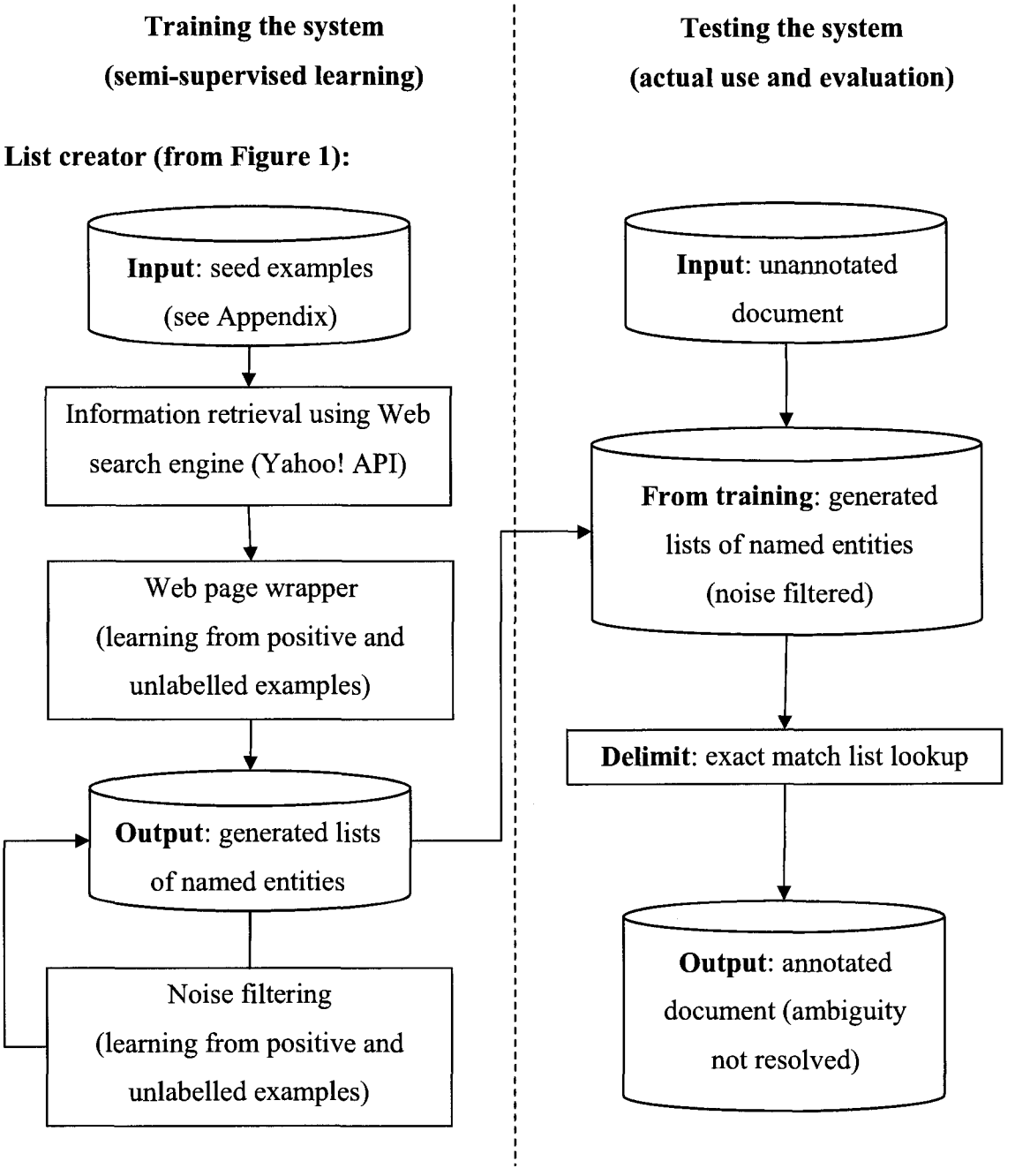

Figure 4: Details of noise filtering as a post-process for the Web page wrapper

Recently, many techniques have been proposed to generate large NE lists starting from an initial seed of a few examples (e.g., Etzioni et al. 2005). Techniques have also been proposed to autonomously maintain an existing NER system (e.g., Heng and Grishman 2006) by increasing its underlying training data set. These semi-supervised learning techniques are

based on bootstrapping lexical knowledge from a large collection of unannotated documents (e.g., the Web). An early example of a bootstrapping algorithm is provided by Riloff and Jones (1999).

In Section 3.2 of the previous chapter, we proposed our own technique for NE list generation based on a bootstrapping algorithm. For efficiency, we kept this algorithm simple. The penalty for simplicity is noise in the generated NE list, but even the most sophisticated algorithm will generate noise. Most of our research focuses on the problem of noise. In Section 4.1, we summarize our NE list generation technique and explain the role of noise filtering. Our main contribution, detailed in Section 4.2, is a new noise-filtering technique, based on lexical NE features. In Section 4.3, we compare our technique to an existing noise-filtering technique, based on information redundancy, and we also examine the combination of our lexical filter with the information redundancy filter. In Section 4.4, we show that the combination of the two noise filters is better than either filter taken individually. In Section 4.5, we demonstrate the use of a third noise filter, based on statistical semantics techniques. Because of the computational complexity of this filter, we report the results of its use as a post-processing step, after the list generation process. Section 4.6 summarizes and concludes.

4.1 Generating NE Lists from the Web

The technique described in this section is inspired by Etzioni et al.'s (2005) "List Extraction" technique combined with "Wrapper Induction" (Section 3.1.2). The algorithm requires, as input, a seed list of a few examples of a given NE type (e.g., cities). Some seed examples are conjoined in a query sent to a Web search engine (e.g., "Boston" AND "New Delhi" AND "Louvain-la-Neuve" AND "Tokyo"). A query composed of four seeds seems to be optimal (Etzioni et al. 2005). A smaller query returns many irrelevant documents, and a longer query returns too few documents.

The returned documents necessarily contain occurrences of all of the seed examples in the given query. The list extraction technique consists of detecting whether or not the seed examples appear inside an HTML list structure in the returned documents (e.g., a table or a

bulleted list). If a list structure is detected, then the entire list is extracted from the document and the elements of the list (except the seed NEs) are considered to be new NEs. In order to detect a list structure and extract elements from it, a wrapper induction algorithm is used. The new examples of NEs found are kept in quarantine until a noise filter has been applied. Entities that pass the noise filter are promoted to a list of "accepted" entities and the remainders are held in quarantine.

The algorithm is iterative, and an iteration consists of:

```
Let S be a list of seed elements.
Let A be a list of accepted named entities.
Let Q be a quarantine list of candidate named entities.

At the first iteration, let initialize A = S and Q = {}.

INPUT: A, Q

1. Sample 4 elements from A and conjoin them to create a query;
2. Send the query to a Web search engine and get top documents;
3. Detect documents with a list structure;
4. Apply list extraction technique to gather all elements of the list;
5. Accumulate new NE examples in the quarantine list Q;
6. Apply the noise filter test to all elements of Q:
      6.1. A' := Add elements that pass the test to A;
      6.1. Q' := Only keep elements that fail the test in Q.

OUTPUT: A', Q'
```

Figure 5: Algorithm for one iteration of the NE list generation process

The quarantine Q is persistent so that an element that is not promoted at a given iteration may be promoted in a subsequent iteration.

The notion of NE frequency is very important here. If an NE is seen on *n* Web pages, then its frequency is *n*. The frequency of the input examples is initialized to one. We use the following parameter settings, which were experimentally found by manually adjusting the

parameter values and observing their qualitative effect. Making small changes to the parameters usually has a minor impact. We manually generate a seed list S containing three times the number of elements required at step 1 ($3 \cdot 4 = 12$ seeds). This lets the algorithm run for three iterations without having to promote elements from Q. At each iteration, four examples are sampled from the list A of accepted NEs. Preference is determined first by NE frequency, and second, by order of appearance. A query is formed by quoting the elements (for exact phrase matching) and joining them with AND. We used the Yahoo! Web search engine (through the developer API). We retrieved the top 200 results for each query. We performed three initial iterations before applying step 6 (i.e., all new NEs are kept in Q). The goal of these preliminary iterations is to gather sufficient data for the filter to be effective (the filter benefits from a larger sample size). After the third iteration, and for all subsequent iterations, step 6 is applied (i.e., the best new NEs are moved from quarantine to the list of accepted NEs). We stopped the bootstrapping process after 10 iterations.

The NE list generation process involves three lists: the seed list (S), the list of accepted NEs (A) and the quarantine list (Q). The seed list is only use at the first iteration to initialize the list of accepted NEs. When a new NE is retrieved from the Web, it is put in the quarantine list. It is promoted to the list of accepted NEs only if it successfully passes a noise-filtering test.

A noise filter based on information redundancy performed well in the task of generating a list of cities and a list of mayors (Downey et al. 2005). We present this filter in Section 4.3. It uses the frequency of an extraction as its filtering criterion. In our experiments, we also noted this noise filter's good performance.

However, this filter's weakness is that it does not take into account lexical information such as capitalization, punctuation, and the length of the NE candidate. Our hypothesis is that lexical information is useful to filter NEs.

In the following section, we present our novel noise-filtering technique based on lexical features. Then, in Section 4.1.2, we compare and combine it with Downey et al.'s (2005)

information redundancy filter.

4.2 Lexical Noise Filter

Our experience with people's names suggests that a person's first name may contain a hyphen but probably not an ampersand, and that the name may often be less than six characters long. Conversely, a company name may contain an ampersand, and it will often be more than six characters long. This experience, indicative of distinctive lexical NE characteristics of a given type, drives the design of our noise filter.

The role of a noise filter is to distinguish valid NEs despite the noise involved in the process of generation lists. The hypothesis that lexical features may be used for filtering noise comes from the observation that entities of a given type often appear similar at the character string level. To calculate entity string similarity, we defined more than fifty features. Table 12 presents a list of our features and their data types. All these features can be found in NER literature, and in various NER systems. An explanation for each feature can be found in the Section 2.5.

As explained earlier, NE candidates come from lists and tables on Web pages. The wrapper induction algorithm is designed so that extracted the tables and lists are made up of HTML nodes that wrap around NEs exactly (e.g., <td>Tokyo</td>). When the seeds are wrapped in HTML nodes exactly, NE candidates (the table's remaining nodes) are usually wrapped accordingly. NE candidates are not full sentences. The NE boundary is usually resolved by the HTML mark-up (e.g., <td>), but in some cases, there is additional context (e.g., <td>city of Ottawa</td>, <td>Ottawa, Canada</td>, etc.). These examples are considered noise since the list extraction algorithm does not implement contextual patterns or parsing of any kind.

Table 12: NE lexical features

Type	Feature
Boolean	HasCapitalLetter
Boolean	StartWithCapitalLetter
Boolean	IsAllCapitalized
Boolean	IsMixedCase
Boolean	HasPunctuation
Boolean	HasDigit
Boolean	HasDigitsOnly
Boolean	EndsInPeriod
Boolean	ApostropheIsSecondChar
Boolean	HasSpecificPunctuation[10]
Boolean	IsRomanDigit
Numeric	Length
Numeric	NumSpace
Numeric	NumericValue
Numeric	NumLeadingDigits
Numeric	NumTrailingDigits
Nominal	Pattern
Nominal	SummarizedPattern
Nominal	Prefix (of length 1,2,3)
Nominal	Suffix (of length 1,2,3)
Nominal	Alphabetical
Nominal	NonAlphabetical

4.2.1 Learning to Filter Noise with Lexical Features

In the context of the current task, learning to filter noise illustrates the general problem of learning from positive and unlabelled examples. At a given iteration, the list of accepted NEs (A) is a pool of presumably positive examples (examples that were in S, and examples that passed through the noise filter in previous iterations). The quarantine list Q presumably contains both positive (valid NEs) and negative (noise) examples, but their actual classes are unknown. The quarantine list Q is therefore a pool of unlabelled data.

[10] There is one feature for each of the following punctuations: apostrophe, slash, backslash, open and close bracket and parenthesis, open and end quote, colon, semi-colon, comma, period, question and exclamation mark, "at," copyright and currency symbols, hyphen, and underscore.

Positive examples are usually learned from using standard machine learning techniques, while negative examples are selected from the unlabelled data, either directly or through a gradual, iterative process (Liu 2003).

In Schölkopf et al. (2001), a Support Vector Machines (SVM) is used to learn from positive and unlabelled examples (also called "one-class SVM"). The technique is implemented in LibSVM[11]. In our experiments, we had no success with this technique. The classifier was overly conservative—classifying the vast majority of instances as noise—and the resulting filter performed below our baseline (Figure 6).

In Schwab and Pohl (1999), a kind of instance-based learning is used to learn from positive examples exclusively. First, a threshold distance (α) is selected in the n-dimension space, where n is the number of features (e.g., from Table 12). Examples within this distance of any positive example are classified as positive, while examples that are too distant are classified as noise. A variant of the idea involves using the centroid of positive examples as the singular reference point. Schwab and Pohl calculate the Hamming distance between two points, though the Euclidian distance can also be used in the n-dimensional space. They also assign variable weights to features according to their relative importance. We tested many configurations of this classifier and we obtained the best results using the centroid of positive examples as a reference point, calculating Euclidian distances between points, setting α as the positives' mean distance from the centroid, and setting equal weight to every feature. The performance of the resulting classifier, called IB (instance-based), is presented in Figure 3.

We tested many techniques in addition to those mentioned above. One technique, inspired by the SMOTE algorithm (Chawla et al. 2002), gave a superior performance. We coined the technique "SMOTE One-Class Learner" to note our original application of the SMOTE algorithm to the problem of learning from positive and unlabelled examples (also called one-class learning). SMOTE's main novelty is the dual use of data under-sampling and data over-

[11] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

sampling.

4.2.2 SMOTE One-Class Learner

The prerequisite for this learning algorithm is for positive examples to be in a minority class with respect to unlabelled data. When this condition is not met, we simply choose a random subset of positive examples so that there are fewer in our data set than unlabelled examples.

A SMOTE One-Class Learner can be created by combining any standard supervised learning classifier (e.g., Naïve Bayes, Decision Trees, etc.) with a pre-processor that under-samples the unlabelled examples, and over-samples the positive examples to balance the data set prior to applying the learning algorithm.

The original SMOTE algorithm was effective in forcing focused learning and introducing a bias towards the minority class (Chawla et al. 2002). Our SMOTE One-Class Learner forces focused learning on the positive examples that we deliberately assign to the minority class.

The original SMOTE algorithm performs a "wise" minority class over-sampling and a random majority class under-sampling (Chawla et al. 2002). In the one-class scenario, we perform an "even" over-sampling of positive examples and a "wise" under-sampling of unlabelled data. The key to focused learning is the synergy of both types of sampling. We explain the various data-sampling techniques in the following paragraphs.

Preliminary sampling (if necessary): First, sample a few positive examples and all the unlabelled examples from the data set. Positive examples are chosen randomly, if necessary, so that the data set has at least an imbalance ratio of 1:2 positive to unlabelled ratio. Usually, the data is already highly imbalanced, as with the case of the list generation algorithm's initial iteration, wherein we have four positive examples (the initial seed) and thousands of unlabelled examples (the candidates in quarantine).

Majority class undersampling: The problem with the majority class (unlabelled data) is that it contains both positive and negative NE examples. The under-sampling strategy

consists of trying to remove positive examples from this pool of examples. We excluded unlabelled examples corresponding to class noise, which are examples with a feature vector exactly equal to that of a positive example. We also exclude unlabelled examples with a Hamming distance of 1 from any positive examples (in other words, they are one feature away from being class noise).

Minority class over-sampling: Usually, the previous sampling results in an imbalanced data set. We therefore duplicate the positive examples evenly up to a positive to unlabelled 1:1 ratio.

Reclassify: A standard classification model can be learned using the resulting set of positive examples and the resulting set of unlabelled examples, acting as negative examples for classification purpose. In our experiments, we use the RIPPER algorithm (Cohen 1995) that performs well in the original SMOTE algorithm. The classification model is used to reclassify the unlabelled examples (the NE candidates). Examples with a positive outcome are promoted to the list A, while examples with negative outcome are kept in Q.

4.2.3 Evaluation of Lexical Filter

We evaluated the list generation algorithm's precision and recall with and without the noise filter. This evaluation is performed on the final list of NEs obtained after 10 iterations.

To evaluate precision and recall automatically, we built NE reference lists by merging lists from existing NER systems and resources: Gate (Cunningham et al. 2002), MinorThird (Cohen 2004), Oak (Sekine and Nabota 2004), and MUC-7 reference data[12]. However, we only used NE list subsets that were available in a minimum of three systems out of four. We believe it removes system bias and guarantees more complete references. Indeed, a single resource is very often biased or incomplete. For instance, only Oak's list of provinces contains Japanese province names; only Gate's list of countries contains the French names of

[12] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2001T02

countries; and MUC-7's list of cities contains 150,000 city names, while other resources contains less than 5,000 city names. In using lists from three resources, we aim to minimize bias and maximize completeness in our references. Table 13 presents our reference lists.

Table 13: Reference lists for noise filter evaluation

Type	Sources	Mean size
First name	Gate, Minorthird, Oak	~6,800
City	Gate, MUC-7, Oak	~50,000
State/Prov.	Gate, MUC-7, Oak	~2,600
Country	Gate, MUC-7, Oak	~400

We generated NE lists for the types listed in Table 13, and we calculated standard precision and recall by looking for exact NE matches between generated lists and reference lists. The final metric quality is f-measure, which is the harmonic mean between precision and recall.

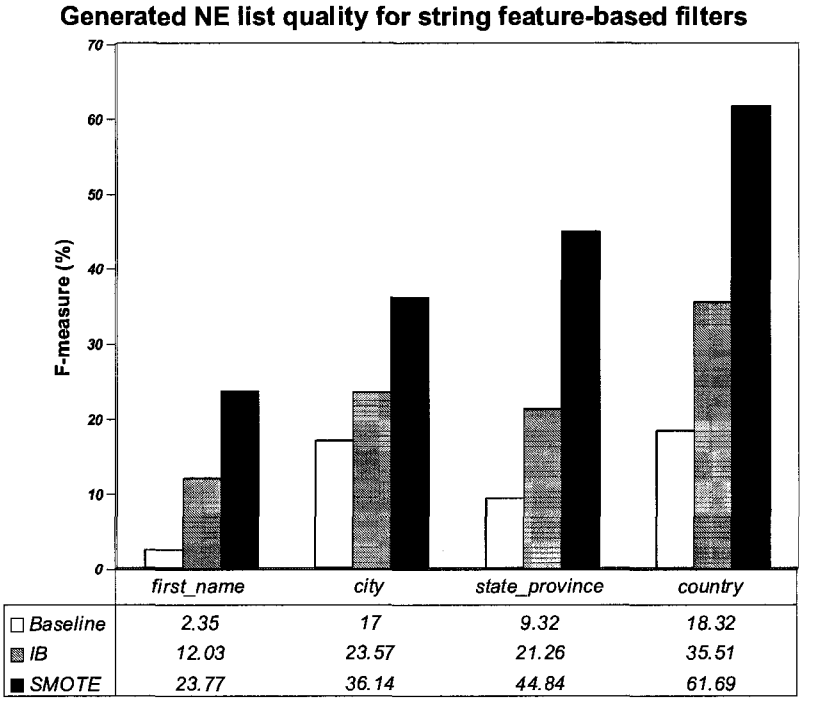

Figure 6: Comparing lexical filters

In Figure 6, we report the mean f-measure on the three reference lists. Results are given for

the following classification strategies: Baseline (no noise filter), IB (instance-based, Schwab and Pohl 1999), SMOTE (SMOTE one-class classifier, Section 4.2.2). An SVM strategy (Schölkopf et al. 2001) returns very poor results, omitted here. In most cases of our experiments, an SVM-based filter allows a tiny 0.1% of NE to be promoted from Q to A, resulting in poor recall.

This experiment shows that the SMOTE One-Class Learner outperforms other techniques for learning to filter noise based on positive and unlabelled NE examples.

4.3 Information Redundancy Filter

When a noise filter is based on information redundancy (Downey et al. 2005), the intuition is such that an extraction obtained from multiple, distinct documents is likely more valid than that obtained from only one.

The information redundancy filter is based on the "balls-and-urns" model from combinatorics, in which extracting an NE candidate from the Web corresponds to a draw from an "urn." Given background knowledge about the content of the "urn," it assigns the candidate the probability of being valid given that it has a frequency of k (the number of times this particular extraction was drawn from the "urn") and a sample of size n (the overall number of draws from the "urn"). Candidates with a high probability (e.g., higher than 90%) are promoted to the NE list.

An urn is characterized by C, the set of valid NEs ($|C|$ is the number of single NEs in the urn) and E, the set of errors ($|E|$ is the number of single errors in the urn). Background knowledge required to use the model is the size of the NE population $|C|$, the size of error population $|E|$, and the accuracy of the p extraction process. As in Downey et al. (2005), the number of errors $|E|$ is approximated to $1x10^6$, and the extraction process is said to be accurate at $p = 90\%$. The high accuracy of the extraction process means that the valid information in the urn is far more redundant than the noise (even if noise is more frequent).

We approximate the size of an NE population $|C|$ using the size of its mean in Table 13. For instance, the number of valid first names is around 6,800. Under some simplifying assumptions to approximate the distribution of C and E, Equation 1 estimates the probability of an NE candidate's validity, given it is seen k times in n draws.

$$\mathrm{P}(x \in \mathrm{C} \mid x \text{ was seen } k \text{ times in } n \text{ draws}) \approx \frac{1}{1+\frac{|E|}{|C|}\left(\frac{p_E}{p_C}\right)^k e^{n(p_C-p_E)}} \qquad (1)$$

P_C, the probability that a particular element of C will appear in a draw, is $\frac{p}{|C|}$, and P_E is $\frac{(1-p)}{|E|}$. Figure 7 reports the quality of information redundancy compared to the best lexical filter of Section 4.2. The filters perform at similar levels.

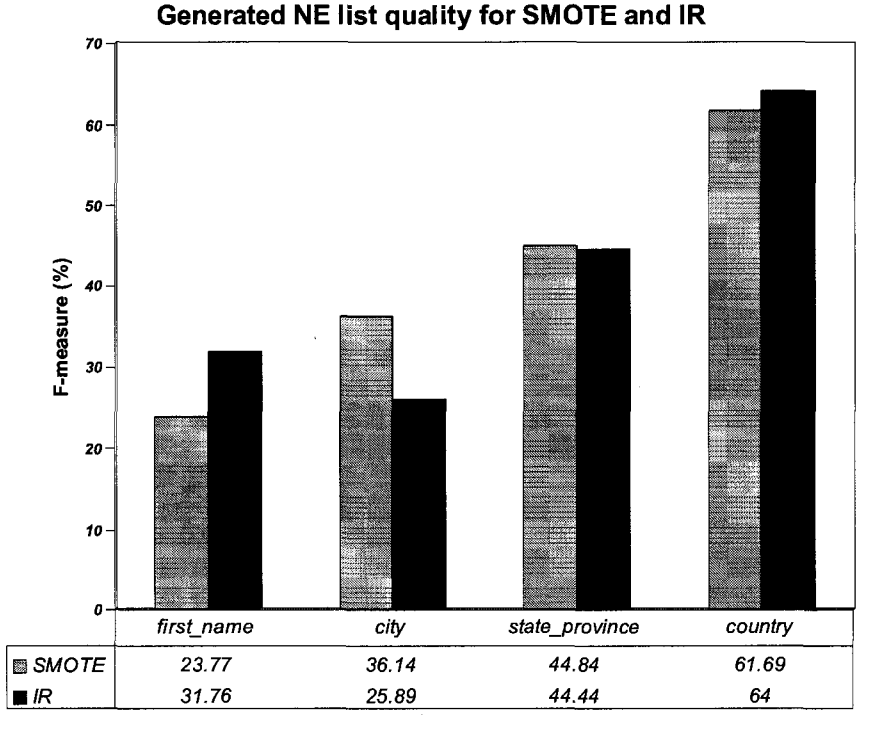

Figure 7: Comparison of lexical filter and information redundancy filter

In preliminary experiments, we verified that the information redundancy model was superior to using an absolute frequency threshold. One strategy was to use a very simple noise filter that promotes every candidate with a higher frequency than 2 or 3. However, this strategy gives poor results, often under the baseline of Figure 6. This is explained by the fact that the frequency threshold is dependent on the number of valid entities $|C|$ and the current number of n draws. For instance, knowing that there are only 200 valid country names, if the extraction process returns 100,000 candidates, then the frequency threshold must be very high to filter that amount of noise. Even with highly precise information extraction techniques (e.g., $p = 90\%$), there would be 10,000 noisy entries and 90,000 repetitions of the 200 valid countries. A candidate repeated four or five time would likely be noise, since we expect a valid country to be repeated 450 times. In this scenario, the information redundancy model seems perfectly suited to filter noise.

4.4 Noise Filter Combination

Both the lexical filter and the information redundancy filter can output a probability estimate. Moreover, they use different sources of information: one is based on the internal properties of NEs, considered character strings; and the other is based on the external properties of NEs, derived from their statistical distributions in lists on the Web. We can combine the probability estimates of the two filters by taking their average. Figure 8 compares the independent components and their combination.

The combined noise filter probability function (SMOTE+IR) is the weighted sum of the SMOTE and IR components' probability functions. Since both components give comparable performances, we give them equal weight. The combination of both noise filters brings interesting improvements. For instance, the "country" type performance increases by almost 10%.

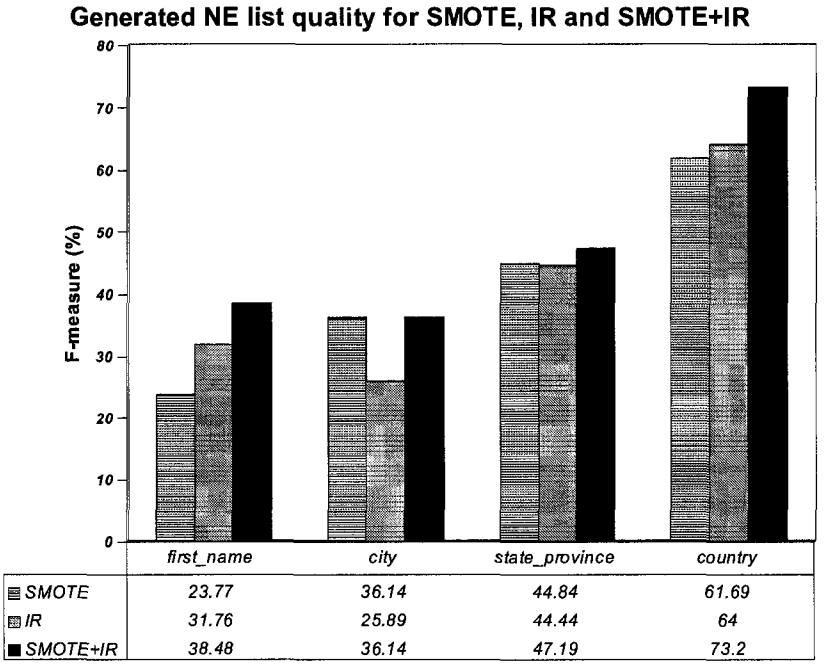

Figure 8: Comparison of individual filters and their combination

The impact of filtering noise from lists can be measured on the NER task. In Table 14, we compare performance of BaLIE with unfiltered lists (taken from Table 9) and with filtered lists (using the combination of noise filters).

Table 14: BaLIE performance on MUC-7 corpus with and without noise filtering

	Without noise filtering			With noise filtering		
Type	Precision	Recall	F-measure	Precision	Recall	F-measure
Organization	75	71	73	75	78	76
Person	71	83	77	71	79	75
Location	77	80	78	76	81	78
Text	72	74	73	74	79	76

The immediate impact of noise filtering is not to improve precision on the NER task. Most noise is well handled, when annotating NEs, by rules such as looking at word capitalization (heuristic H1, Section 3.2.1). However, the noise filter enables the gazetteer generation algorithm to run for more iterations and creates larger and cleaner lists. It raises the NER recall, particularly for the 'organization' type, which has a very large set of possibilities. We

believe that precision errors cannot be addressed by generating larger lists of named entities. There is rather a need for better disambiguation rules, particularly for resolving entity-entity ambiguity. This is the aim of experiments described in Chapter 5.

4.5 Statistical Semantics Filter

Statistical Semantics is the study of how the statistical patterns of word usage can be used to solve problems requiring semantic information. One technique in this field is called "Latent Relational Analysis" (LRA), and was designed for the classification of semantic relations between word pairs (Turney 2005). LRA measures relational similarity between two pairs of words. When two pairs have a high degree of relational similarity, they are analogous. For example, the pair "cat:meow" is analogous to the pair "dog:bark."

In this section, we show that LRA can be used as a noise filter for generating NE lists. We use LRA to measure the relational similarity of pairs made up of an NE candidate NE and its type (e.g., "London:city," "John:first_name," "Canada:country"). LRA lets us measure the similarity between known valid pairs (e.g., "Boston:city") and candidate pairs (e.g., "Kolkata:city," "Click Here:city"). For instance, a high relational similarity means that *"Kolkata* is to *city* as *Boston* is to *city."* Conversely, a low relational similarity means that *"Click Here* is not to *city* as *Boston* is to *city."*

The previous filters were based on lexical features and on redundancy information, whereas a statistical semantics filter uses information on the relation between an NE and its type by looking at word usage patterns in a large collection of documents. The LRA algorithm we used in this experiment is similar to that of Turney (2005), using a corpus of one terabyte of textual data (Terra & Clarke, 2003).

LRA requires much more time to compute than SMOTE (Section 4.2.2) or IR (Section 4.3). For a lexicon of one hundred entries, LRA usually takes up to five minutes to run. Our lexicons will often exceed 100,000 NE candidates. For practical reasons, we apply LRA outside the list generation process of Figure 1. Instead of integrating LRA into the iterative

algorithm, we use it as a post-processing filter applied to the final NE list, obtained after all iterations.

Our goal is to model the relations between known valid word pairs (the seed words), and to measure the similarity of the modeled relation with that of the NE candidate. We use it in two approaches: one for demoting NE that were added to the NE list; and one for promoting NE that were kept in the quarantine. These approaches follow the steps described in the next paragraphs.

First, a passage-retrieval search engine is used on a 1Tb textual data corpus (Terra & Clarke, 2003) to find passages where a word and its type (e.g., "Prague" and "city") appear with a maximum of three intervening words.

Generalizations of all passage are generated and listed. Passages (e.g., "Prague is a city") and generalizations of passages (e.g., "Prague * a city") are called "patterns." A pattern is constructed by replacing any or all or none of the intervening words with wild cards (one wild card can only replace one word).

Total pattern frequencies for all NEs under examination are smoothed using entropy and log transformations (Landauer and Dumais 1997).

Singular value decomposition (SVD) is performed on a matrix made up of word pairs and associated patterns. SVD compensates for the sparseness of the matrix.

Resulting matrix rows associated with word pairs are used as vectors to compute a similarity value based on vector cosine. Given two word pairs, the similarity of their associated rows is therefore computed using the cosine of the angle between the rows. Each candidate pair is compared to the seed pairs. The relational similarity is the mean of similarity between a candidate pair and each seed pair. The idea is to keep word pairs analogous to NE seeds by setting a threshold on the relational similarity. In our experiment, the threshold we use is the minimal similarity found by comparing each of the seeds against one another.

During a first approach, all NEs from the list are compared to the seed, and those that are not analogous are demoted and returned to the quarantine. In a second approach, all candidates from the quarantine are compared to the seed and the analogous candidates are promoted to the final NE list.

We applied the statistical semantics filter on the output of the list generation process that uses the SMOTE+IR filter. The resulting list quality improves for two NE types out of four. For the "state/province" and "city" types, there is no statistically significant change. The improvement for "first name" and "country" types is mainly attributed to the second approach (promotions) and brings a recall gain. The respective f-measures rise to 39.23% and 73.51%. The improvement is slight yet statistically significant.

We believe the statistical semantics filter is able to capture very difficult cases of noise such as concept drift (e.g., a continent name appearing in list of countries; a full name appearing in a list of first names), as well as highly redundant noise (table headers such as "country," "population," etc.). However, we require further investigation and experiments to better understand LRA's contribution to noise filtering.

4.6 Conclusion

Generating NE lists using a semi-supervised learning technique applied on large collections like the Web is a noisy process. Filtering noise early on is essential since bootstrapping algorithms use knowledge of a given iteration to extract new knowledge at the next iteration.

In this research, we look at three noise-filtering techniques. Our main contribution is the development of a lexical filter that classifies NE candidates according to surface cues like capitalization, punctuation, etc. We compare and combine this to a noise-filtering technique based on information redundancy. We show that combining both filters performs better than using any of them in isolation. In the final experiment, we demonstrate the use of a statistical semantics filter making use of the LRA algorithm. This last experiment had a slightly

positive outcome, and most of our future works will aim at better integrating and understanding the use of statistical semantics in the NE list generation algorithm. Successfully generating large NE lists is a key component in semi-supervised NER. Such technology will enable autonomous deployment of NER systems, as well as automatic maintenance of existing systems

Chapter 5

Discovering Unambiguous NEs for Disambiguation Rule Generation

In this chapter, we put forth a second BaLIE improvement. In the baseline system (Chapter 3), ambiguity between two entity types (e.g., "France" is either a country or a person) is resolved only if there is a very strong cue for one entity type in its alias network (e.g., the unambiguous passage "Ms. France Beaudoin" appears in the text). This is rarely the case. We improved the resolution of this ambiguity by learning disambiguation rules that are applicable to any textual passage. This way, an entity-entity ambiguity can be resolved even without an alias network (Section 3.2.3). If such a network exists, each instance of a resolution can contribute to the final decision.

NE ambiguity resolution is not novel. Many techniques were proposed in supervised learning settings using classifiers (e.g., Sekine 1998 decision tree) and sequence-labelling techniques (e.g., Bikel et al. 1997 HMM). However, in a semi-supervised learning setting, this presents more challenges. Cucerzan and Yarowsky (1999) proposed an algorithm, and highlight that the general precondition for building a semi-supervised NE ambiguity resolution system is the need for unambiguous NE examples. Manually listing unambiguous NEs is a bottleneck in this kind of system. Our contribution is the following:

- The demonstration of a simple strategy based on set intersection, which helps identify unambiguous NE examples.

This chapter presents an improvement for the "Rule learner" module of Figure 1 that serves the purpose of the "Classify" step of McDonald's (1993) paradigm. Figure 9 details the process of training and evaluating disambiguation rules.

An NE is unambiguous if its label refers to only one object. Examples of ambiguous NEs are common: "Apple" refers to a company as well as a fruit; "Chicago" refers to a city and a musical, etc. Finding unambiguous NEs is a difficult task because examples that may seem unambiguous at first (e.g., Nevada, Vancouver, etc.) often turn out to be ambiguous due to

the broad range of certain entity types, such as brands (e.g., "Nevada" is a Sears Canada clothing brand), and linguistic phenomena, such as metonymy (e.g., "Vancouver" also stands for the "Vancouver Canucks" hockey team).

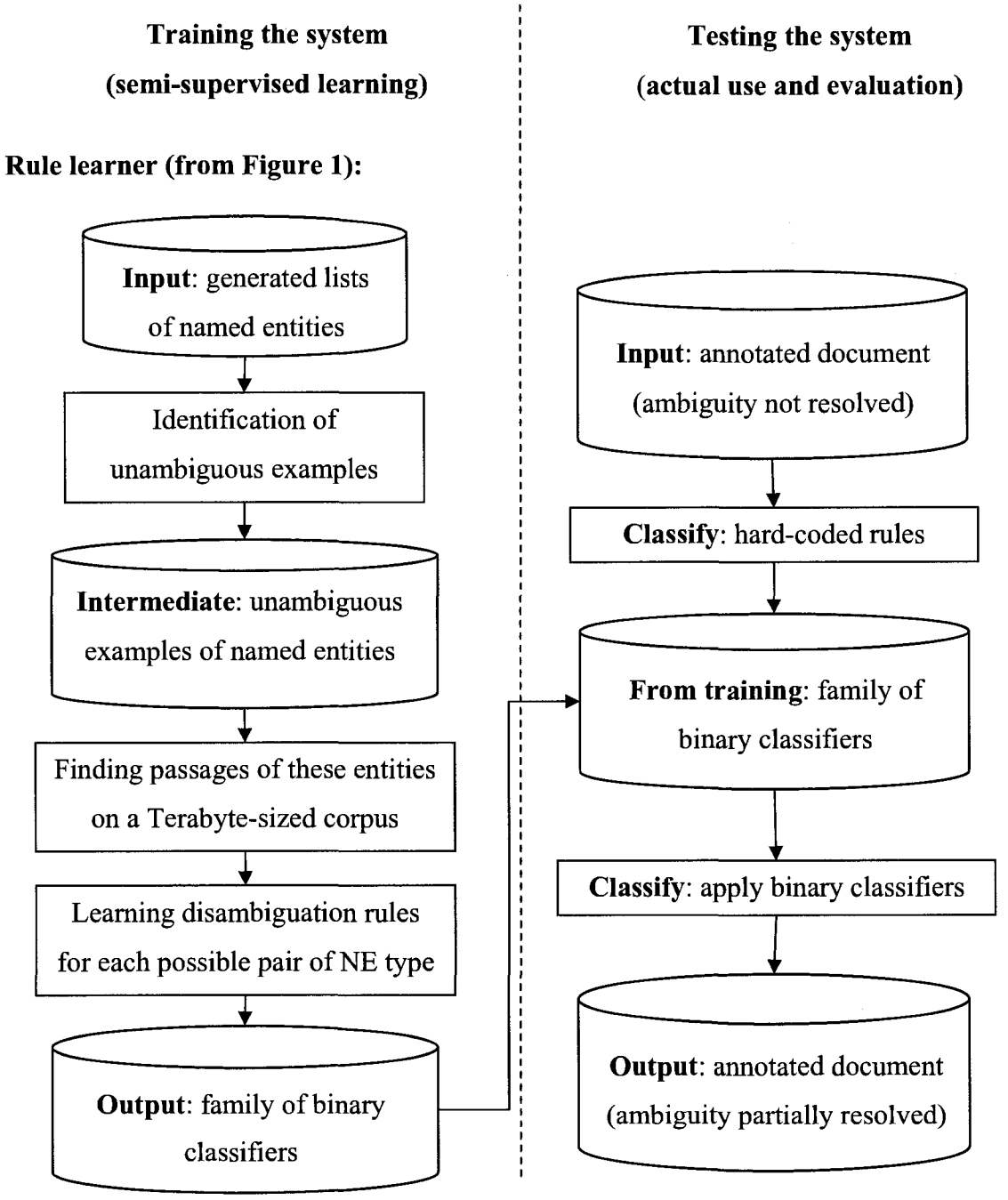

Figure 9: Details of training disambiguation rules in a semi-supervised manner

We present a technique that can identify the unambiguous NE in a list. The idea is that

unambiguous NEs can be used to automatically generate training data for NE disambiguation (Cucerzan & Yarowsky 1999).

This chapter is built on the hypothesis that the set differences of automatically generated multiple gazetteers are a set of unambiguous NEs. That is, if an NE appears in exactly one gazetteer, given a large set of gazetteers, then we assume that the NE is unambiguous. It may sound simple, but testing this requires extensive linguistic resources.

In Section 5.1, we present related work in the NER field wherein unambiguous NEs are collected and used for various tasks. In Section 5.2, we describe the result of a massive gazetteer generation for 100 NE types. In Section 5.3, we measure and qualify the ambiguity between NEs. In Section 5.4, we explain how to create disambiguation rules from unambiguous NE examples. Then, in Section 5.5, we put forth a framework for evaluation that supports our hypothesis, which states that the set differences of automatically generated multiple gazetteers are a set of unambiguous NEs. Section 5.6 presents the conclusion.

5.1 Related Work

Unambiguous NEs are discovered and used to develop baseline NER systems for benchmarking (Mikheev et al. 1999), as well as features in standard NER systems (Szarvas 2006). A baseline system can be created by tagging all NEs in a training data set, and by removing the ambiguous NEs that appear under more than one type. The remainder is used to search for an exact match in the unambiguous list, then to tag a test corpus. This technique is known as "supervised learning," since the NE list is derived from annotated data. In our work, we do not use annotated data.

Unambiguous NEs are also used in related work by Cucerzan and Yarowsky (1999). It begins with a small seed of unambiguous NE examples to bootstrap a larger set of NEs paired with sense-disambiguation rules. This technique falls into the semi-supervised systems category, but it requires manually feeding the system with unambiguous examples listed by an expert linguist. Our technique identifies unambiguous NEs automatically.

A popular technique to identity ambiguity in NE lists consists of applying the set intersection operator between an NE list and a general dictionary (e.g., Mikheev 1999). It is mainly used to disambiguate ambiguous common-noun NEs (e.g., "Apple"). In our work, we address both entity-noun ambiguity and entity-entity ambiguity (e.g., "Chicago").

Identifying unambiguous NE using set differences of automatically generated multiple gazetteers is novel for three reasons:

1. Our technique is not based on analysis of annotated data;
2. We eliminate the constraint of manually finding unambiguous NE examples;
3. We address entity-entity ambiguity.

5.2 Massive Generation of NE Lists

We generated NE lists for the 100 types specified in the Appendix. Our choice of type is influenced by Sekine's hierarchy (Sekine and Nobata 2004) and the BBN corpus[13]. The following table demonstrates statistics for all these types. We included the overlap measurement between BaLIE and Oak lexicons (Sekine and Nabota 2004) calculated as the number of named entities belonging to both lexicons. Oak is a handmade NE system of lexicons and rules.

Table 15: BaLIE and Oak lexicon comparison

Type	BaLIE lexicon size	Oak lexicon size	Overlap size
first_name	40,000	7,000	4,852
last_name	6,700	82,000	3,334
person_title	15	7	6
celebrity	6,600	1,400	226
title	915	121	25
character	1,600	3	2
company	27,200	13,900	3,125

[13] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2005T33

military	32	502	1
association	1,700	568	65
government	1,300	1,700	406
political_party	430	987	23
nationality	285	161	99
market	243	56	43
sports_team	163	288	123
city	15,500	1,300	1,144
state_province	1,600	393	188
country	1,000	777	610
county	867	1,700	867
region	548	1034	44
landform	74	2,500	0
river	1,100	2,800	506
sea	138	8	6
planet	17	10	10
star	310	84	35
cathedral	25	2	1
school	3,600	4,800	2,375
museum	2,800	4,300	1,153
airport	580	7,500	256
port	155	2	2
library	159	14	2
road	313	2	1
bridge	78	2	1
station	64	5	2
railroad	404	2	0
amusement_park	300	3	2
monument	67	2	1
car	961	33	6
ship	1,300	736	72
train	20	5	1
aircraft	53	731	2
spaceship	179	47	24
opera_musical	238	259	78
song	5,900	1	1
sculpture	57	1	1
broadcast	2,400	10	4
movie	327	654	42
book	2,000	52	12
newspaper	1,300	1,700	419
magazine	125	107	35

weapon	259	9	2
drug	5,300	14,800	437
food	130	3	2
game	243	42	2
war	145	57	21
crime	351	5	1
conference	41	23	4
mammal	99	4	4
mineral	139	30	23
disease	1,400	1,200	547
religion	166	3	3
colour	25	4	4
language	127	2	1
award	226	317	12
sport	189	3	1
academic	80	3	3
rule	213	869	10
theory	121	161	2
total	141,000	157,000	21,312

Overlap in Table 15 should not be interpreted as an evaluation of the BaLIE lists' quality. Rather, it gives an idea of the intersection between the BaLIE and Oak lexicons. In addition, BaLIE handles 29 NE types that are not implemented in Oak. Table 16 reports these types and the size of each list.

Table 16: Additional BaLIE lexicons

Type	Lexicon size
vocation	1,700
political_line	19
religious_group	300
lake	600
ocean_bay	20
continent	8
amphitheatre	271
castle	16
skyscraper	142
sport_place	251
hotel	13
hospital	25

park	55
painting	73
food_brand	663
clothing_brand	799
holiday	41
hurricane	138
insect	73
sea_animal	170
fish	45
reptile	15
bird	202
vegetal	20
measure	67
currency	83
month	22
weekday	14
god	15
Total	5860

5.3 NE Ambiguity

Here, we observe automatically generated NE lists. In Section 5.3.1, we manually qualify ambiguity by finding its source in four important entity types. In Section 5.3.2, we quantify ambiguity levels in generated lists to highlight the proportion of ambiguous entities, as well as the most and least ambiguous entity type.

5.3.1 Qualifying Ambiguity

We looked at ambiguity in four important NE types: first name, city, clothing brand, and song. We chose these entity types because they have high cardinality, and also because they intuitively exhibit different kinds of ambiguity. We randomly sampled 100 elements from these lists and queried a passage-retrieval search engine, which leveraged 1Tb of data (Terra & Clarke 2003). For each entity, we retrieved up to 50 textual passages, for a total of up to 5,000 passages per entity type. When 50 passages of an entity refer to the correct type (manually verified), the entity is considered unambiguous. If one or more passages refer to the wrong type or to something else, the entity considered ambiguous. This criterion is more

rigorous than that of Szarvas et al. (2006), which consider an entity unambiguous if it has the same sense 90% of the time in a training corpus. In the following table, we qualify the ambiguity between entity types.

Table 17: Source of ambiguity between entity types

Source of ambiguity	First name	City	Clothing brand	Song
No ambiguity	38%	48%	23%	39%
Common noun/phrase	3%	4%	11%	47%
First name	N/A	6%	2%	1%
Last name	34%	13%	16%	1%
Full name	5%	1%	38%	3%
City	6%	N/A	1%	1%
State	-	3%	-	-
County	2%	4%	-	-
Country	-	6$	-	-
Company	4%	2%	2%	2%
Street	5%	2%	-	-
Prayer	1%	-	-	-
Product	1%	1%	-	-
Tree	-	2%	-	-
Car	-	1%	-	-
Sports team	-	3%	-	-
Nationality	-	1%	-	-
Scientific journal	-	-	1%	-
Lake	-	-	1%	-
Dance	-	-	1%	-
National Park	-	-	1%	1%
Award	-	-	1%	-
Hotel	-	-	-	2%
Book	-	-	-	1%
Movie	-	-	-	1%

The surveyed entity types show high levels of ambiguity: 52% (city) to 77% (clothing brand) of all NE instances were ambiguous. First names are predominantly ambiguous with last names (e.g., Frank, Isabel, Matthews, Robert), full names (e.g., Robert William, Sarah Jane, etc.), cities (e.g., Carlton, Clarinda, Orlando, etc.), companies (e.g., Nielsen, Sierra), and

street names. Streets are often named after people, and while they often contain street markers such as "Drive" or "Avenue," metonymic references render them completely ambiguous (e.g., "...go to the Carol Sue intersection (third stoplight). Turn left...").

Approximately half of cities are non-ambiguous. Therefore, in NER, one city out of two can be recognized in a simple lexicon look-up. Ambiguity is mainly identified in last names (e.g., Branson, Laval, Nurnberg), common nouns (e.g., cork, little rock), and countries (e.g., Texan city "Italy," Mexico's capital "Mexico city").

Clothing brands are highly ambiguous because of the tendency to name brands after the designer's full name (e.g., Christian Dior, Tommy Hilfiger, Ralph Lauren) or last name (e.g., Armani, Puma, Gant). Clothing brands also often use common nouns (e.g., Fossil, Iceberg, Polo). There are also some interesting ambiguities such as "Joop" (clothes and perfumes) and "JOOP" (Journal of Object-Oriented Programming).

Finally, songs are unique due to their broad intersection with common nouns and phrases (e.g., "Black velvet," "Crazy," "Don't be cruel," "On the road again," "Satisfaction," "You really got me"). Songs can also be named after people (e.g., "Billie Jean," "Gloria") and, interestingly, we identified ambiguity with hotel names ("Heartbreak Hotel," an Elvis Presley song that is also the name of numerous hotels worldwide, in Graceland, Florida, and more). However, we found no ambiguity with the song "Hotel California" in our sample.

An interesting conclusion can be drawn from these observations: most ambiguities can be identified by intersecting NE lists. In the Table 17, only 3 entities out of 400 are ambiguous with an NE type outside of BaLIE: a first name that also describes a kind of prayer (Marian: "...of the most popular Marian prayers of the Western..."); a clothing brand that is also the name of a scientific journal (JOOP: "...Dr. Dobb's Journal, JOOP..."); and a clothing brand that also describes a kind of dance (Samba: "...just heed a Samba rhythm, carried north on...").

5.3.2 Quantifying Entity-Entity Ambiguity

In this section, we demonstrate the proportion of entity-entity ambiguity per type. Table 18 presents the entity types sorted from the most to the least ambiguous.

Table 18: Percentage of entity-entity ambiguity per type

Type	Ambiguity (%)	Type	Ambiguity (%)	Type	Ambiguity (%)
language	93.60	ship	22.32	celebrity	8.21
nationality	74.81	religion	21.85	railroad	8.10
country	70.16	food_brand	21.68	song	8.03
last_name	61.58	city	21.20	drug	7.44
state_province	60.09	first_name	19.76	sports_team	6.45
god	60.00	monument	19.70	road	6.42
planet	58.82	food	18.60	bridge	6.15
mammal	56.57	mineral	17.99	spaceship	5.75
region	54.28	political_line	17.65	crime	5.42
religious_group	51.28	broadcast	16.43	train	5.00
weekday	50.00	bird	16.34	award	4.95
colour	50.00	star	16.12	association	4.54
fish	48.33	aircraft	15.69	cathedral	4.00
currency	47.06	ocean_bay	15.38	rule	3.57
month	45.71	government	15.37	military	3.33
sea_animal	44.31	game	15.00	disease	2.97
measure	42.42	war	14.50	market	2.88
reptile	42.11	station	14.29	theory	2.67
magazine	41.53	book	14.11	car	2.09
movie	38.54	castle	12.50	park	1.75
clothes	37.89	title	12.28	museum	1.68
opera_musical	36.91	sculpture	12.28	sports_place	1.65
vegetable	35.00	lake	11.53	skyscraper	1.49
person_title	33.33	weapon	11.22	port	0.67
sea	32.56	river	10.97	school	0.22
character	30.78	painting	9.59	county	0.00
academic	28.85	library	9.47	continent	0.00
amphitheatre	27.71	company	9.31	airport	0.00
holiday	27.50	newspaper	8.73	hotel	0.00
insect	24.66	landform	8.70	hospital	0.00
sport	24.54	amusement_park	8.70	hurricane	0.00
vocation	23.35	political_party	8.29	conference	0.00

Ambiguity is measured by the intersection of a type with all other types. Intersecting names must exactly match so that the airport "Toronto Lester B Pearson International Airport" is not ambiguous with the city "Toronto" or the person "Lester B Pearson". An ambiguity of 93.6% means that 93.6% of the type's instances intersect with an instance of another type.

Two problems arise from analyzing this table. First, some types intersect heavily. This is the case for the "language" and "nationality" types, as well as "planet" and "god," which share lot of elements naturally (there's a fuzzy line between languages and nationality; planets are named after Roman deities). This is also the case where slight concept drifts bring lot of ambiguity. For instance, the "region" and "country" types can share lot of island names.

The second problem is the inconsistency between the level of ambiguity in Table 18 and that which is calculated in Table 17 of the previous section. For instance, the manual ambiguity analysis revealed that 59% of first names, 48% of cities, 66% of clothing brands, and 14% of songs are ambiguous with elements of another entity type (excluding ambiguity with common nouns and phrases). The intersection of lists allows us to identify, respectively, 19.76%, 21.2%, 37.89%, and 8% of ambiguity. This is approximately half of the manually assessed ambiguity. We believe the main reason for this discrepancy is the recall of BaLIE lists versus the true extension of entity lists.

5.4 From Unambiguous NE to Disambiguation Rules

Cucerzan and Yarowsky (1999) demonstrate that disambiguation rules can be learned from a set of unambiguous NEs. Their semi-supervised learning technique illustrates the problem of learning from positive examples. In this section, we show that we can greatly disentangle the problem by using heuristics and classical binary classification exclusively. On the one hand, we can resolve a great deal of noun-entity ambiguity with simple capitalization constraints as outlined by the Mikheev (1999) technique (see Section 3.2.1). On the other hand, given an ambiguity between two or more types, we can create one or many Boolean classifiers made of positive examples of one type against those of another type. In the following sections, we

present the experimental set-up for entity-entity disambiguation.

5.4.1 Entity-Entity Disambiguation Rules

Disambiguation is required when an entity is described by two or more NE types. Let's examine a case with two ambiguous types. A scenario with more than two types is covered in the next section.

A Boolean classifier is built using positive examples of two entity types under examination. Positive examples of a given type are found by querying a passage-retrieval search engine (Terra & Clarke, 2003) with unambiguous NE instances. These unambiguous NEs are found by removing entities that intersect with any other type. According to our hypothesis, the retrieved passages are therefore positive examples of a correct NE type in its context.

We chose to build classifier similar to the baseline system aimed at word sense disambiguation (WSD) called "Duluth 6" (Pedersen 2002). Duluth 6 is an ensemble of naive Bayes classifiers trained on different sets of features. A first classifier uses word unigrams in the context of the ambiguous word. A second classifier uses word bigrams in the context of the ambiguous word. A third classifier uses word unigrams adjacent to the ambiguous word. The context of an ambiguous word is made of ten words on its left and ten words on its right.

Above and beyond Duluth 6, we added a fourth classifier that uses features of the present NEs in the context of the ambiguous words. For example, "Dell" is ambiguous between a person and a company name. In the phrase "Michael Dell", the fourth classifier would use the information that "Dell" is preceded by a known first name. This additional classifier is based on the common assumption (e.g., Carreras et al. 2003) that contextualized NEs can predict other NEs (e.g., a last name usually follows a first name, city and state names are commonly co-occurring; the enumeration of entities is a strong indicator of other entities, etc.). Carreras et al. use predicted entities in "left context," which are entities that the system has already identified and classified. Conversely, we opted to use every entity type candidate from both sides by searching the lexicon, and by not resolving potential ambiguity.

Problem with prior probabilities

The training data that we develop from unambiguous NE is not representative of the real distribution and importance of entities. For instance, we have very few examples for the language type because the vast majority of it is considered ambiguous. In a naive Bayes classifier, the impact on prior probabilities can be severe. To work around the problem, we created perfectly balanced data sets in accordance with the entity under examination with the fewest examples. For instance, since we only have 200 examples of sentences with an unambiguous language, we create classifiers for ambiguous languages (e.g., language-nationality, language-country) by sampling 200 examples from the other NE type.

There are some semi-supervised strategies that could be used to approximate the real prior of each class, but we report on this in the section of Chapter 7 on future work.

5.4.2 Entity Ambiguity for More Than Two Types

When there are more than two possible types for a given entity (e.g., "Murray" can be a first name, a last name, a city, and a river), we apply a round-robin technique (Fürnkranz 2002). This consists of evaluating the outcome of all possible pairwise classifiers, summing up victories, and guessing the class with the most victories. A random decision is used to break ties.

5.4.3 Entity-Entity Classifier Cross-Validation

We proceed to the classifier evaluation using 10-fold cross-validation on the training data. This is not equivalent to the classifier evaluation in the extrinsic NER task, a topic covered in Section 5.5. In the following table, we demonstrate the accuracy of the classifiers compared to random guesses. The result of random guessing always tends toward 50% accuracy because of our balanced data set.

Table 19: Accuracy of entity-entity classifiers

	Binary classifier	Accuracy (%)
Worst classifier	currency-measure	65.05
Median classifier	state_province-amusement park	82.69
Best classifier	drug-person title	93.38

The mean accuracy is 82.40% and the standard deviation is 4.79%. We can identify three groups of classifiers: weak, medium, and strong. Weak classifiers give a range of 65 to 70% accuracy. Examples include currency vs. measure, sculpture vs. painting, movie vs. book, fish vs. sea animal, and movie vs. broadcast. The low accuracy is understandable and almost pardonable because of the examples' close conceptual proximity. Some weak classifiers also occur when the training data is insufficient, particularly when the "language," "god," "planet," or "mammal" types are in paired. Being highly ambiguous (see Table 18), these types have small lexicons (see Table 15 and Table 16), thus resulting in very small training data.

The majority of classifiers perform at a level between 78 and 87%. There is no significant property that distinguishes these classifiers, so we hypothesize that the size of the data set and the difficulty of the task account for the variation in accuracy level.

Finally, there is the group of strong classifiers with accuracy ranging between 90 and 94%. Examples include drug vs. person title, association vs. celebrity, clothes vs. spaceship, car vs. city, railroad vs. government, and song vs. museum. Interestingly, most of these classifiers involve types that are considered among the least ambiguous (see Table 18). We believe this aptly supports the conclusion of Cucerzan and Yarowsky (1999), stating that unambiguous NEs can create accurate classifiers.

5.5 Experiments on the NER Task

In this section, we proceed with the NER task's extrinsic evaluation of disambiguation rules. The BaLIE system design is unchanged with respect to Chapter 3, except for the addition of the noise filter of Chapter 4. However, instead of four entity types, 100 are supported. We

believe that one direct impact of this capability scaling is a performance drop for individual entity types. More entity types mean more ambiguity and, therefore, more disambiguation decisions. Where "Chicago" was unambiguously classified as a location, it must now be checked against the "musical" type. Where "David" was unambiguously classified as a person, it must now be checked against the "sculpture" type. Same for "Layla," which could be a song, or "Queen Elizabeth," which could be a ship, and so forth.

In the following sections, we compare the system with and without disambiguation rules. The version without disambiguation rules is equivalent to the system described in Chapter 3. In case of ambiguity, the heuristics of Section 3.2.2 are applied. The version with disambiguation rules is the same system plus the ambiguity classifiers.

We first evaluate the system on the MUC-7 corpus. This is directly comparable to results obtained in Section 3.3. Then, we evaluate BaLIE on the CONLL-2003 English corpus. This data set contains the classic MUC enamex (person, location, organization) as well as a "miscellaneous" type used for most NE types outside enamex. In a third evaluation, we evaluate the BBN data set, which is much more fine-grained than other corpora.

5.5.1 Evaluation on MUC-7 Corpus with the MUC Scorer

Let's first report the results of Section 3.3. The MUC scorer has the particularity of allowing partial matches (e.g., the organization "New York Times News Service" is considered correctly identified even if the system tags "New York Times," for instance). We present results for the three enamex types as well as the special "text" row, reporting the proportions of exact matches for all types. These results are extracted from Table 9 of Chapter 3.

Table 20: Three-type BaLIE performance on MUC-7 corpus

Type	Precision	Recall	F-measure
Organization	75	71	73
Person	71	83	77
Location	77	80	78
Text	72	74	73

Now, here are the results for 100-type BaLIE run with and without disambiguation rules. The portion without disambiguation rules is extracted from Table 14 of Chapter 4.

Table 21: 100-type BaLIE performance on MUC-7 corpus with and without rules

	Without rules			With rules		
Type	Precision	Recall	F-measure	Precision	Recall	F-measure
Organization	75	78	76	75	80	77
Person	71	79	75	76	82	79
Location	76	81	78	80	84	82
Text	74	79	76	74	79	76

The left-hand side of the table shows results that are consistent with experiments in Chapter 3. There are two main performance variations: a rise in recall for the organization type and a drop in recall for the person type. We believe the rise for the organization type is caused by having many more organization subtypes than in Chapter 3 (e.g., associations, government, military, sports team, political party, market, etc.). Conversely, the person type has fewer new subtypes and may be prone to higher ambiguity potential, as discussed in the introduction.

On the right-hand side, we report improvements when using disambiguation rules. All types benefit from a better recall, which means ambiguous entities that were misclassified by Chapter 3 heuristics are now recovered. Moreover, precision for the person and location types is significantly improved, which means a lot of ambiguous entities that were classified by default are now discarded.

5.5.2 Evaluation on CONLL-2003 English Corpus with the CONLL Scorer

CONLL-2003 is a difficult data set in comparison to MUC-7. It particularly consists of much sports news, where results and standings are given in batches, as in Figure 10. In the CONLL corpus, city names in a sports context are annotated as organizations (e.g., "Hartford" stands for the "Hartford Whalers" organization).

```
League games on Thursday  ( home team in CAPS ) :
Hartford          4  BOSTON           2
FLORIDA           4  Ny Islanders     2
```

Figure 10: CONLL corpus metonymic references

These metonymic references were all incorrectly annotated by our system. They account for as many as 700 occurrences and roughly 20% of precision errors for the location type and 20% of recall errors for the organization type. Complete results are provided in Table 22.

Table 22: BaLIE's performance on the CONLL corpus

Type	Precision	Recall	F-measure
Person	49.50	52.10	50.77
Location	65.49	72.71	68.91
Organization	43.26	51.27	46.93
Miscellaneous	61.37	52.35	56.50

These are very consistent with MUC-7 results, since the CONLL scorer only considers exact matches to be successful (see Section 2.6.2). Therefore, the absolute scores are pessimistic compared to MUC-7.

The following table presents a comparison of BaLIE macro-averaged measures with the CONLL baseline, and with the best supervised system. We report BaLIE results without disambiguation rules in the first row. Even with the rules, our system performs slightly below the baseline. This could be due to the very large proportion of metonymic references in CONLL. A supervised system could easily learn this problem since we noted that all city

names followed by a number are classified as organizations.

Table 23: System comparison on CONLL corpus

System	Precision	Recall	F-measure
BaLIE (without rules)	51.23	54.25	52.70
BaLIE (with rules)	54.90	57.11	55.98
Baseline	71.91	50.90	59.60
Best supervised	88.99	88.54	88.76

The disambiguation rules improve both precision and recall, which compares to observed rises on the MUC corpus. BaLIE performs slightly under the baseline, but would outperform it if metonymic references were disregarded. The best supervised system (Florian et al. 2003) is far better on the CONLL task, but may over-fit the corpus. For instance, in the online demo of a system trained on CONLL, city names followed by a small digit are often recognized as organizations.

5.5.3 Evaluation on the BBN Corpus with the CONLL Scorer

We evaluated BaLIE on the BBN corpus, which was designed for the task of question answering, but annotated with NEs. However, we found no published NER experiment or baseline for this corpus in the literature. As well, we decided not to create a baseline system because no training and split-testing are defined.

The BBN corpus contains NE annotations for 64 types and subtypes. By design, we do not handle numex and timex types, so we excluded them from evaluation. In this corpus, most entity types are paired with a "description" type such as "ORGANIZATION: DESCRIPTION" (e.g., the firm, the newspaper, a library, the hospital, etc.) In the passage, "The Citizen is a newspaper", "Citizen" is annotated as an "ORGANIZATION" while "newspaper" is a "ORGANIZATION: DESCRIPTION". We excluded "description" types that may be useful for co-refence resolution, but that are not real named entities. Finally, we do not report results for types with zero or very few instances in the corpus (e.g., ORGANIZATION: HOTEL, WORK_OF_ART: PAINTING). In all, we report results for 30 types.

Table 24: BaLIE's performance on BBN corpus

	Without rules			With rules		
Type	**Precision**	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
PERSON	50.81	60.71	55.32	55.50	63.20	59.10
LANGUAGE	23.46	22.62	23.03	26.98	20.24	23.13
NORP: NATIONALITY	76.75	69.64	73.02	76.43	70.60	73.40
NORP: RELIGION	46.07	46.59	46.33	69.31	48.89	57.33
NORP: POLITICAL	83.58	33.09	47.41	85.23	33.23	47.82
FAC: BUILDING	57.69	9.74	16.67	57.69	9.74	16.67
FAC: AIRPORT	84.21	47.06	60.38	84.21	47.06	60.38
FAC: HIGHWAY_STREET	2.89	4.31	3.46	3.51	5.17	4.18
ORGANIZATION: GOVERN	73.74	33.12	45.71	74.65	33.14	45.90
ORGANIZATION: CORPOR	55.71	49.47	52.41	57.73	51.59	54.49
ORGANIZATION: EDUCATI	77.06	35.79	48.88	77.46	36.61	49.72
ORGANIZATION: MUSEUM	6.98	42.86	12.00	7.06	42.86	12.12
ORGANIZATION: POLITIC	25.30	10.17	14.51	25.30	10.17	14.51
ORGANIZATION: HOSPITAL	50.00	4.35	8.00	50.00	4.35	8.00
ORGANIZATION: OTHER	17.31	6.06	8.97	18.18	6.22	9.26
GPE: COUNTRY	79.50	78.05	78.77	79.98	76.12	78.00
GPE: CITY	52.29	64.32	57.68	55.41	65.26	59.93
GPE: STATE_PROVINCE	59.37	56.46	57.88	59.55	59.50	59.52
LOCATION: RIVER	17.07	35.90	23.14	16.45	64.10	26.18
LOCATION: LAKE_SEA_OC	27.54	23.75	25.50	27.78	25.00	26.32
LOCATION: REGION	15.88	10.84	12.88	38.10	15.21	21.74
LOCATION: CONTINENT	56.46	83.59	67.40	56.46	83.59	67.40
LOCATION: OTHER	31.31	17.13	22.14	42.67	17.68	25.00
PRODUCT: WEAPON	13.33	9.52	11.11	11.11	9.52	10.26
PRODUCT: VEHICLE	19.35	9.42	12.68	26.06	9.69	14.12
EVENT: WAR	55.81	51.06	53.33	55.81	51.06	53.33
EVENT: HURRICANE	98.28	54.81	70.37	98.28	54.81	70.37
EVENT: OTHER	38.46	20.45	26.71	37.19	20.45	26.39
SUBSTANCE: DRUG	43.15	18.22	25.62	45.45	18.22	26.02
DISEASE	41.61	17.98	25.11	41.61	17.98	25.11

Most evaluated types provide a good performance estimate for the NER task. For some types, however, the BBN corpus contains annotations for both NEs and entity descriptions. This is the case with the "SUBSTANCE: DRUG" type, for which references to unnamed drugs (e.g., drug, pill, medicine, narcotic, vaccine, hormone) are annotated. It accounts for approximately 312 occurrences out of 439 (71%), and it clearly explains our system's low

recall in this case.

5.6 Conclusion

In this chapter, we demonstrated how to learn disambiguation rules in a semi-supervised manner. The technique is based on identifying unambiguous NE examples using a textual corpus to constitute a data set of unambiguous passages. Our hypothesis is that the set differences of a very large number of NE types are sets of unambiguous examples. This is interesting because it is not based on annotated data analysis, it eliminates the constraint of manually finding unambiguous NE examples, and it addresses entity-entity ambiguity.

We demonstrated the validity of the hypothesis using two means. First, we manually verified the source of ambiguity for four important NE types. We observed that most ambiguities can be identified by intersecting NE lists. From the 400 examples we checked, only three fell outside BaLIE's 100 NE types. Second, we tested the system with and without disambiguation rules on three standard NER data sets. We showed that using disambiguation rules learned in a semi-supervised manner always significantly improves the system's performance. We claim that only rules learned from unambiguous examples can provide this improvement.

The disambiguation rules we created take the form of pairwise classifiers for each possible entity-entity ambiguity. We implemented well-known baseline strategies from the word-sense disambiguation field. A classifier relies on contextual evidence, such as preceding and following words, and the presence of other entities. We calculated that the majority of entity-entity classifiers perform within a range of 78 to 87% accuracy.

A problem with our technique is the lack of prior probability for NE types. In artificially creating a data set of textual passages, the distribution of examples is arbitrary. For instance, most "languages" are ambiguous with most "nationalities." For these two entity types, we can retrieve a limited number of passages since unambiguous NEs are rare. We discuss this issue in the thesis conclusion in greater detail.

Chapter 6

Detecting Acronyms for Better Alias Resolution

In this chapter, we present the third improvement to BaLIE. This improvement falls into the category of the "less common and very difficult problems," a sign that the NER field is maturing. Recently, much attention has been given to problems such as metonymy (e.g., when "New York" stands for the "New York Yankees" organization), which represents a fraction of the errors committed by NER systems, but requires advanced algorithms to be resolved. Another problem is acronym detection. Indeed, it is necessary to identify acronyms and their expansions (e.g., "NY" and "New York") in text to fully benefit from alias resolution. Our contribution is the following:

- The development of an acronym-detection algorithm that outperforms previous art.

This chapter presents an improvement for the "Alias network" module of Figure 1 that serves the purpose of the "Record" step of McDonald's (1993) paradigm. An acronym and its expansion are indeed aliases of a given named entity. The use of acronyms within BaLIE is illustrated in Figure 11. We present a supervised learning solution to acronym detection. It is however trained once for every acronym, and the model is not dependant on a specific named entity type or the named entity recognition task itself. The trained system can then be added to the BaLIE alias-resolution algorithm, as explained in Section 6.6.

Acronym identification is the task of processing text to extract pairs consisting of a word (the acronym) and an expansion (the definition), where the word is the short form of, or stands for, the expansion. For instance, in the sentence, "The two nucleic acids, deoxyribonucleic acid (DNA) and ribonucleic acid (RNA), are the informational molecules of all living organisms," there are two acronyms, "DNA" and "RNA," along with their respective definitions, "deoxyribonucleic acid" and "ribonucleic acid." In this work, we do not discriminate between acronyms (short forms of multi-word expressions) and abbreviations (contractions of single words). We use the term "acronym" to signify both.

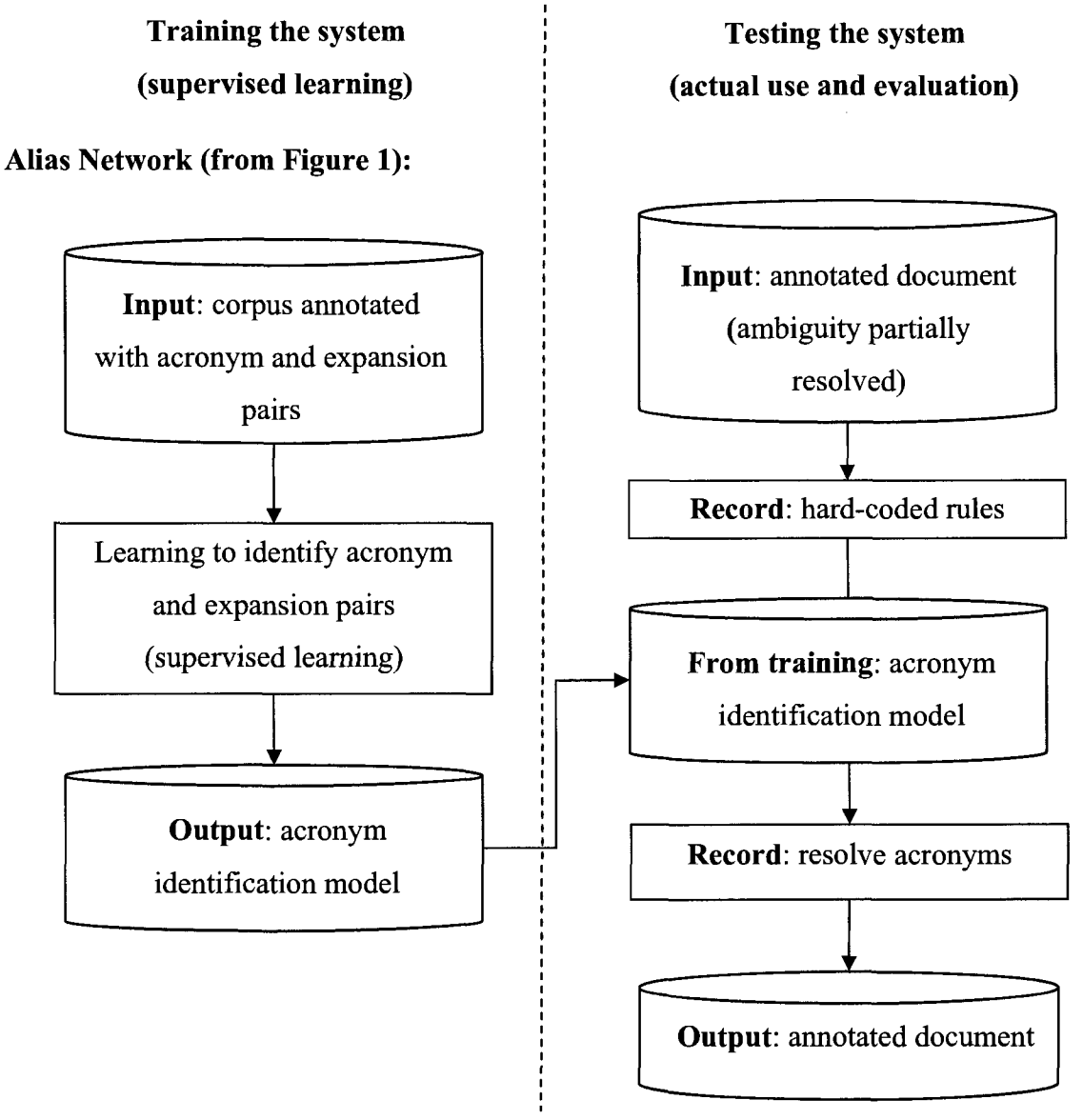

Figure 11: Details of acronym identification as a component of the alias network

The task of identifying acronyms can be extended in many ways. It is possible to try to resolve acronyms even when there are no explicit definitions in the text. For instance, the familiar acronym "HIV" will often appear without being defined. Another extension is to try to disambiguate polysemous acronyms (e.g., "CMU" means "Carnegie Mellon University" and also "Central Michigan University"). The task requires to identify the intended sense of the acronym, even when its definition is absent. Ambiguous acronyms are particularly

problematic for information retrieval.

In this section, we only tackle the core task. That is, given an input text, our algorithm will attempt to extract all explicit acronym-definition pairs. Our goal is to create a dictionary of acronym-definition pairs specific to a single text. For example, an algorithm that addresses the core task can be used to enhance a list of key phrases by resolving acronyms. More importantly, such an algorithm is a key component in systems that handle the various extended tasks, like co-reference resolution for NER or automatic query expansion for information retrieval. The literature on automatic acronym identification details many attempts to solve the core task. Our contribution is to demonstrate a supervised learning approach with fewer constraints on the forms of acronyms and definitions that can be identified. Our results compare with what has been achieved on the same testing data by human-engineered rule systems with more constraints.

The next section presents a detailed summary of related work. Section 6.2 presents our supervised learning approach to identifying acronyms, and Section 6.3 discusses the training and testing corpus we used. At least three other papers use the same corpus for evaluating their systems (Pustejovsky et al. 2001; Chang et al. 2002; Schwartz and Hearst 2003). The remaining sections discuss our experiments' results, and conclude.

6.1 Related Work

In this section, we present previous work on the task of identifying acronyms. We focus on the constraints that these systems use to extract valid acronym–definition pairs.

One of the earliest acronym identification systems (Taghva and Gilbreth, 1999) is the Acronym Finding Program (AFP). The AFP system first identifies acronym candidate, which the authors define as upper-case words of three to ten letters. It then tries to find a definition for each acronym by scanning a $2n$-word window, where n is the number of letters in the acronym. The algorithm tries to match acronym letters to initial letters in the definition words. Some types of words receive special treatment: stopwords can be skipped,

hyphenated words can provide letters from each of their constituent words, and finally, acronyms themselves can be part of a definition. Given these special cases, the longest common sequence (LCS) between acronym letters and initial letters in definitions is computed.

Yeates (1999) proposes the automatic extraction of acronym-definition pairs in a program called Three-Letter Acronyms (TLA). Although the name suggests that acronyms must have three letters, the system can find *n*-letter acronyms as well. The algorithm divides text into chunks using commas, periods, and parentheses as delimiters. It then checks whether adjacent chunks have acronym letters matching one or more of the three initial letters of the definition words. Further heuristics are then applied to each candidate, ensuring that the acronym is upper case, is shorter than the definition, contains the initial letters of most of the definition words, and has a certain words to stopwords ratio.

Larkey et al. (2000) developed Acrophile. They compared various strategies and found their "canonical/contextual" method to be the most accurate. First, they force acronym candidates to be upper-cased, allowing only embedded lower case letters (internal or final), periods (possibly followed by spaces), hyphens or slashes, and digits (at most one, non-final digit). They allow a maximum of nine alphanumeric characters in acronyms. They search for expansions in a 20-word window adjacent to the given acronym. Stopwords can contribute to an inner letter, but only once for the entire acronym. Furthermore, an expansion is only valid if it fits a given pattern, such as being surrounded by parentheses or preceded by a cue phrase (e.g.,"also known as").

Recently the fields of genetics and medicine have become especially interested in acronym detection (Pustejovsky et al., 2001, Yu et al. 2002). Pustejovsky et al. present an approach with few constraints, designed to capture a wide range of acronyms that are abundant in medical literature. For example, "PMA" stands for "phorbol ester 12-myristate-13-acetate" and "E2" stands for "estradiol-17 beta." Pustejovsky et al.'s acronym detection technique searches for acronym definitions within noun phrases. Acronym-definition pair candidates must match a given set of regular expressions, designed to be very general. The final

decision about whether a pair is valid relies on counting the number of acronym characters and definition words that match.

Another strategy, also developed for the medical field, comes from Schwartz and Hearst (2003)[14]. Their approach is similar to Pustejovsky et al.'s (2001) strategy, and the emphasis is again on complicated acronym-definition patterns for cases in which only a few letters match (e.g., "Gen-5 Related N-acetyltransferase" [GNAT]). First, they identify acronym-definition pair candidates by looking for patterns, particularly "acronym (definition)" and "definition (acronym)." The length of the definition must be at most $\min(|A|+5, |A|\cdot 2)$, where |A| is the number of letters in the acronym[15]. Then, they count the number of overlapping letters in the acronym and its definition, and compare the sum to a given threshold. They force the first letter of the acronym to match the first letter of a definition word. They also handle various cases where an acronym is entirely contained in a single definition word.

Park and Byrd (2001) combine mechanisms such as text-markers and linguistic cues with pattern-based recognition. Larkey (2000) uses the same combination. This eliminates some constraints on identifiable acronyms. The reason for these mechanisms is to cope with the growing popularity of acronyms that deviate from the tradition of using only the first letter of each word of the definition. They use cue expressions (e.g., "or," "short," "acronym," "stand") to reinforce the confidence in acronym-definition pairs. They also allow acronyms to include a digit at the beginning or the end. Thus, "5GL (Fifth Generation Language)" would be a valid candidate.

Adar (2002) presents a technique that requires only four scoring rules for evaluating acronym-definition pairs: add one to the score if an acronym letter begins a definition word; subtract one for each extra word that does not match acronym letters; add one if the definition is next to a parenthesis; and finally, as the number of definition words should be

[14] The Java source code for their system is available at http://biotext.berkeley.edu/software.html.

[15] This formula is borrowed from Y. Park and Byrd (2001).

less than or equal to the number of acronym letters, subtract one for each extra word.

Chang et al. (2002) present a supervised learning approach to identifying acronyms. In order to circumscribe the learning, they impose a strongly restrictive condition on acronym-definition pair candidates by searching only for "definition (acronym)" patterns. Interestingly, this pattern accounts for the majority of positive cases in their evaluation corpus. Chang et al.'s learning algorithm uses eight features that describe the mapping between acronym letters and definition letters (e.g., percentage of letters aligned with the beginning of a word, number of definition words that are not aligned with the acronym, etc.). The learning algorithm they use is logistic regression.

Zahariev (2004) dedicates a thesis on a complete review of acronym identification literature. He also extends the task to the multilingual perspective and offers in-depth analysis of the acronym phenomenon. However, the proposed system uses the same constraint patterns as Larkey et al. (2000).

Table 25 summarizes related work on acronym identification. In this table and in the forthcoming sections, "participation" means that an acronym letter is found in a definition word. Generally, either there are many constraints on the acronym (e.g., "all acronym letters must be capitals" or "the number of letters must exceed some minimum") or the definition pattern is fixed (e.g., "the definition must be in parentheses"). Such strong constraints ensure reasonable precision but, generally (for heterogeneous text from unrestricted domains), they necessarily limit recall. In our work, we try to use few constraints on both the acronym and the definition.

Table 25: Summary of constraints on acronyms and definitions

Author (Year)	Strongest constraints on acronym candidates	Strongest constraints on definition candidates
Taghva and Gilbreth (1999)	upper-case word of 3 to 10 characters	must be adjacent only first letters of definition words can

		participate
Yeates (1999)	upper-case word	must be adjacent first 3 letters of definition words can participate
Larkey et al. (2000)	need some upper-case letters maximum size of 9 characters	pattern "acronym (definition)" or "definition (acronym)" cue (e.g.: "also known as")
Pustejovsky et al. (2001)	a word between parentheses or adjacent to parentheses	pattern "acronym (definition)" or "definition (acronym)"
Schwartz and Hearst (2003)	a word between parentheses or adjacent to parentheses	pattern "acronym (definition)" or "definition (acronym)"
Park and Byrd (2001)	at least 1 capital from 2 to 10 characters	parentheses pattern or linguistic cue (also known as, short for, etc.)
Adar (2002)	1 word between parentheses	adjacent on the left of parenthesis
Chang et al. (2002)	1 word between parentheses	adjacent on the left of parenthesis
Zahariev (2004)	1 word between parentheses or adjacent to parentheses	pattern "acronym (definition)" or "definition (acronym)"

6.2 Supervised Learning Approach

The acronym identification task can be framed in terms of supervised learning. The concept we want to learn is a pair $\langle A, D \rangle$ made of an acronym A (a single token) and a definition D (a sequence of one or more consecutives tokens). Given a sequence T of n tokens, $T = \langle t_1, ..., t_n \rangle$, from which we wish to extract a pair $\langle A, D \rangle$, there are n possible choices for $A = t_i$. Each possible acronym ($A = t_i$) can be defined (D) by any combination of one or more consecutive tokens taken from the left context $\{t_1, ..., t_{i-1}\}$ or from the right context $\{t_{i+1}, ..., t_n\}$. The number of possible pairs, in the worst case, is $O(n^3)$ (n choices for $A = t_i$ multiplied by n choices for the first token in D multiplied by n choices for the last token in D). Therefore, before applying supervised learning, we reduce the space of possible $\langle A, D \rangle$ pairs with some heuristics.

We describe our heuristics for reducing the search space for acronym candidates, and then we discuss the constraints for definition candidates. Together, these sections explain how we reduce the space of $\langle A, D \rangle$ pairs that the supervised learning algorithm must consider. After the space has been reduced, the remaining pair candidates must be represented as feature vectors to apply standard supervised learning algorithms (Witten and Frank, 2000).

The constraints that follow are relatively weak, compared to most past work on acronym identification, but they still exclude some possible acronym-definition pairs from consideration by the supervised learning algorithm. The resulting decrease in recall is discussed in Section 6.5.3.

6.2.1 Space-Reduction Heuristics for Acronym Candidates

The acronym space (the set of choices for $A = t_i$) is reduced using syntactic constraints on the tokens, $T = \langle t_1, ..., t_n \rangle$, expressed by the conjunction of the following statements:

- $A = t_i$, where $1 \leq i \leq n$.

- $\text{Size}(t_i) \geq 2$, where $\text{Size}(t_i)$ is the number of characters in the token t_i (including numbers and internal punctuation).
- $\text{NumLetter}(t_i) \geq 1$, where $\text{NumLetter}(t_i)$ is the number of alphabetic letters in the token t_i (excluding numbers and punctuation).
- $(\text{Cap}(t_i) \wedge \text{UnknownPOS}(t_i)) \vee \text{Cue}(t_i)$, where $\text{Cap}(t_i)$ means that the token starts with a capital letter, $\text{UnknownPOS}(t_i)$ means that the token's part-of-speech is neither conjunction, determiner, particle, preposition, pronoun, nor verb, and $\text{Cue}(t_i)$ means that the token contains a digit, punctuation, or a capital letter.

The rationale behind $\text{Size}(t_i) \geq 2$ is that, in most cases, isolated letters like "H" will not be acronyms (although "H" can stand for "Hydrogen"). Statement 4 says that the t_i token should have some capitalization or special characters, but in the former case, the token should not have a known part-of-speech. The calculation of $\text{UnknownPOS}(t_i)$ requires applying a part-of-speech tagger to the text. We used qTag (Tufis and Mason, 1998) as our part-of-speech tagger.

The above heuristic constraints are less restrictive than previous approaches (compare with Table 25).

6.2.2 Space-Reduction Heuristics for Definition Candidates

Once an acronym candidate $A = t_i$ is found in the text, we search its definition D on both sides of t_i. First, we require that both acronym and definition appear in the same sentence. This considerably reduces the search space for $\langle A, D \rangle$ by reducing the n of T size, although the space is still $O(n^3)$. We then need stronger criteria to define a reasonable set of definition candidates. We impose the following additional constraints:

- The first word of a definition must use the first letter of the acronym (Pustejovsky et al. 2001).

- A definition can skip one letter of the acronym, unless the acronym is only two letters long.
- The definition can skip any number of digits and punctuation characters inside the acronym.
- The maximum length for a definition is $\min(acronymlen+5, acronymlen\times 2)$ (Park and Byrd 2001).
- A definition cannot contain a bracket, colon, semi-colon, question mark, or exclamation mark. (We found counter-examples for other punctuation. For instance, the acronym "MAM" expands to "meprin, A5, mu," where the comma is used.)

Usually, these constraints will dramatically reduce the number of definition candidates. thus increasing precision, while including the vast majority of true positive cases, thus preserving recall.

To illustrate the remaining search space, consider the following sentence:

"Microbial control of mosquitoes with special emphasis on bacterial control (Citation)."

The word "Citation" is not an acronym, but it fits our constraints since it is a capitalized noun. Even with the above constraints, there are 92 definition candidates in this example. Note that according to the second rule above, the definition can skip one letter of the acronym, except the leading "C." Here is one of the definition candidates (acronym letters are marked with square brackets):

[c]ontrol of mosqu[i]toes wi[t]h speci[a]l emphas[i]s [o]n bacterial co[n]trol

6.2.3 Acronym-Definition Features for Supervised Learning

The above heuristics reduce the search space significantly, so that the number of ways to extract a pair $\langle A, D\rangle$ from a token sequence $T=\langle t_1,...,t_n\rangle$ is now much less than $O(n^3)$. The

next step is to apply supervised learning to select the best $\langle A, D \rangle$ pairs from the remaining candidates. Standard supervised learning algorithms require input in the form of feature vectors. We defined seventeen features to describe an instance of an acronym-definition candidate. The handcrafted rules described in previous work inspired the design of many of the following features. Our features mainly describe the mapping of acronym letters to definition letters, and syntactic properties of the definition.

1. The number of participating letters matching the first letter of a definition word;
2. (1) normalized by the acronym length;
3. the number of participating definition letters that are capitalized;
4. (3) normalized by the acronym length;
5. the length (in words) of the definition;
6. the distance (in words) between the acronym and the definition;
7. the number of definition words that do not participate;
8. (7) normalized by the definition length;
9. the mean size of the words in the definition that do not participate;
10. whether the first definition word is a preposition, a conjunction, or a determiner (inspired by Park and Byrd, 2001);
11. whether the last definition word is a preposition, a conjunction, or a determiner (inspired by Park and Byrd, 2001);
12. the number of prepositions, conjunctions, and determiners in the definition;
13. the maximum number of letters that participate in a single definition word;
14. the number of acronym letters that do not participate;
15. the number of acronym digits and punctuations that do not participate;
16. whether the acronym or the definition is between parentheses;
17. the number of verbs in the definition.

If the heuristics propose an acronym-definition pair candidate $\langle A_1, D_1 \rangle$, then there are three possibilities:

(1) In the manual annotation of the corpus, there is an officially correct acronym-definition

pair $\langle A_2, D_2 \rangle$ such that $A_1 = A_2$ and $D_1 = D_2$. In this case, $\langle A_1, D_1 \rangle$ is labelled as positive for both training and testing the algorithm.

(2) In the manual annotation of the corpus, there is an officially correct acronym-definition pair $\langle A_2, D_2 \rangle$ such that $A_1 = A_2$ but $D_1 \neq D_2$. In this case, $\langle A_1, D_1 \rangle$ is ignored during training (see Section 6.3 for details).

(3) In the manual annotation of the corpus, there is no officially correct acronym-definition pair $\langle A_2, D_2 \rangle$ such that $A_1 = A_2$. In this case, $\langle A_1, D_1 \rangle$ is labelled as negative for both training and testing.

6.3 Evaluation Corpus

We use the Medstract Gold Standard Evaluation Corpus[16] (Pustejovsky et al., 2001) to train and test our algorithm. The corpus is made of Medline abstracts in which each acronym-definition pair is annotated. The training set is composed of 126 pairs and the testing set is composed of 168 pairs. What is most interesting about this corpus is that it was annotated by a biologist using an informal definition of a valid pair. Therefore the corpus reflects human interpretation of acronym-definition pairs, and acronym identification is rendered challenging for an automated process.

Past results with this corpus are reported in Table 26. All of the results are based on modified versions of the Medstract Gold Standard Evaluation Corpus, and, unfortunately, they all use different modifications. Here are some remarks on each of the modifications:

1. Chang et al. (2002) do not describe their modifications.
2. Pustejovsky et al. (2001) note that they removed eleven elements, which they judged as non-acronyms.

[16] http://medstract.org/gold-standards.html

3. Schwartz and Hearst (2003) mention that they made modifications, but do not describe what modifications they made.
4. We attempted to replicate the results of Schwartz and Hearst (2003), while making only minimal modifications to the original corpus. Our modifications were aimed at creating a valid XML file and a consistent set of tags. We had to remove embedded acronyms and remove or correct obvious errors.

Since Schwartz and Hearst's system is available online, we were able to repeat their experiment on our modified version of the corpus. This is the version of the corpus that we use in the following experiments, detailed in Section 6.4.

Table 26: Acronym detection performance reported various teams

Team	Pr	Re	F1	Corpus Modification
Chang et al. (2002)	80%	83%	81.5%	See (1)
Pustejovsky et al. (2001)	98%	72%	83.0%	See (2)
Schwartz and Hearst (2003)	96%	82%	88.4%	See (3)
Schwartz and Hearst (our replication)	89%	88%	88.4%	See (4)

6.4 Experiment Results

We use the Weka Machine Learning tool kit to test various supervised learning algorithms. The results are reported in Table 27. We found that performance varies greatly depending on the chosen algorithm. Good classifiers were PART rules (rules obtained from a partially pruned decision tree), with somewhat low recall but high precision. The Support Vector Machine (Weka's SMO) reaches F1 = 88.3%, a performance that rivals that of handcrafted systems. The Bayesian net also performs well. The OneR (one rule) classifier is shown as a baseline. Table 27 includes our replication of Schwartz and Hearst (2003) for comparison. Note that all results in this table are based on the same corpus.

Table 27: Performance of various classifiers on the Medstract corpus

Learning Algorithm	Pr	Re	F1
OneR	69.0%	33.1%	44.7%
Bayesian Net	89.6%	81.7%	85.5%
PART rules	95.3%	79.6%	86.7%
SVM (SMO kernel degree = 2)	92.5%	84.4%	88.3%
Schwartz and Hearst (our replication)	88.7%	88.1%	88.4%

We claim that our system has weaker hand-coded constraints than competing approaches. To support this claim, it is worth mentioning that 1,134 acronym-definition pair candidates fulfilled the constraints in Section 6.2.1 and Section 6.2.2, but the supervised learning algorithms only classified 141 candidates (12%) as positive. Therefore, the hand-coded part of our system allowed more candidates than did Schwartz and Hearst's system. In comparison, the latter system considered 220 patterns that involve parentheses, and 148 (67%) are accepted by the rule-based system. In our system, the decrease from 1,134 candidates to 141 is done by the supervised learning component rather than by hand-coded constraints. The advantage of this approach is that the supervised learning component can easily be retrained for a new corpus. The hand-coded constraints are designed to be weak enough to not require modifications for a new corpus.

6.5 Discussion

In this section, we interpret the results of our experiments.

6.5.1 The Parenthesis Feature

In our examination of previous work (Section 6.1), we criticized many authors using overly constrained patterns. One of the problems is the use of parentheses. Many authors only accept acronym-definition pairs when one of the expressions is between parentheses. To avoid this kind of limitation, we did not impose this constraint on our model. However, the only way we were able to perform as well as hand-built systems was to use the feature "whether the acronym or the definition is between parentheses" (feature 16 in Section 6.2.3).

The learner uses this feature since it works well on the Medstract corpus. Our relatively few constraints (Section 6.2.1 and Section 6.2.2) allow 889 acronym-definition pair candidates for which the parenthesis feature is false (neither the acronym candidate nor the definition candidate is between parentheses). In the Medstract corpus, these 889 candidates are negative instances (none are true acronym-definition pairs). Thus, this feature dramatically increases precision with no loss of recall. It is a very informative feature, but we do not wish to hard-code it into our constraints, since we believe it may not adapt well to other corpora. With a new corpus, our system can learn to use the feature if it is helpful, or ignore it if it does not apply. This robustness is an advantage of using few constraints combined with supervised learning.

6.5.2 The Best Features

When evaluating the contribution of the individual features (using the Chi Square Test), we found that three features significantly outperform others. Those features are, in order of predictive power: the distance between the definition and the acronym (feature 6); the number of acronym letters that match the first letters of definition words (feature 1); and the parentheses feature (feature 16).

6.5.3 Effects of the Space-Reduction Heuristics

In Section 6.2, we presented heuristics for reducing the space of possible acronym-definition candidates. A particular case can be misleading for the supervised learning algorithm.

Consider a case in which our heuristics identify <PKA, protein kinase A> but the corpus annotation is <PKA, cAMP-dependent protein kinase A>. It is tempting to say that <PKA, protein kinase A> must count as a negative example for the supervised learner, but this could confuse the learner since the PKA and protein kinase A match is actually very credible and reasonable. Instead of counting <PKA, protein kinase A> as a negative example, we found that it is better to ignore this case during training. It would be incorrect to count this case as a positive example, but it would be misleading to count it as a negative example, so it is best to ignore it. During testing, however, such instances are added to the false negatives, thus

reducing recall, because this is an error and the system must be penalized for it.

6.5.4 External Evaluation

In her master's thesis, Dannélls (2006) evaluated four acronym detection systems, including ours, on the Swedish language. She provided us with Swedish training data and testing data. Our system performs at significantly higher levels than other systems that were tested. Dannélls's results are presented in Table 28.

Table 28: Acronym detection on Swedish texts

Learning Algorithm	Pr	Re	F1
Acrophile (Larkey et al. 2000)	97%	20%	33%
Stanford (Chang et al. 2002)	77%	66%	71%
Simple (Schwartz & Hearst 2003)	100%	6%	11.3%
Our approach (Nadeau & Turney 2005)	96%	91%	93%

6.6 Improving Alias Resolution in NER Systems

The acronym detection module described in this chapter can be integrated with BaLIE's alias resolution algorithm (Section 3.2.3, Figure 3). When a definition is added to a set of aliases "a_i," the corresponding acronym is also added. Moreover, a side-effect of identifying an acronym definition is identifying the exact boundary of the potential entity. For instance, let's look at this sentence containing one "organization" type entity:

"The court convicted the head of the South <ENAMEX TYPE="ORG">**Lebanon Army**</ENAMEX> (SLA) of collaborating with Israel."

In this sentence, the acronym detection module recognizes the acronym SLA:

SLA, [S]outh [L]ebanon [A]rmy

On the one hand, it corrects the organization boundary, and on the other hand, it associates "SLA" to the "South Lebanon Army" alias set. Eventually, the annotations are corrected

accordingly:

> "The court convicted the head of the <ENAMEX TYPE="ORG">**South Lebanon Army**</ENAMEX> (<ENAMEX TYPE="ORG">**SLA**</ENAMEX>) of collaborating with Israel."

In our experiments, we measured no significant improvements on the MUC-7 and the BBN corpora. In fact, only four acronyms are identified in MUC-7, and no acronyms are found in BBN. However, the CONLL corpus is rich in acronyms, and the improvement in organization recall is important, as shown in Table 29. We identified 19 acronyms in the CONLL corpus, and one was a false positive (<New, [N]orm H[e] [w]itt>).

Table 29: BaLIE's performance on the CONLL corpus with acronym detection

	Without acronym			With acronym		
Type	**Precision**	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
Person	49.5	52.10	50.77	49.69	52.16	50.90
Location	65.49	72.71	68.91	65.52	72.71	68.92
Organization	43.26	51.27	46.93	44.60	52.43	48.20
Miscellaneous	61.37	52.35	56.50	61.59	52.35	56.60

6.7 Conclusion

In this chapter, we described a supervised learning approach to the task of identifying acronyms. The approach consists in using few hand-coded constraints to reduce the search space, and then using supervised learning to impose more constraints. The advantage of this approach is that the system can easily be retrained for a new corpus when the previously learned constraints no longer apply. The hand-coded constraints reduce the set of acronym-definition pair candidates that must be classified by the supervised learning system, yet they are weak enough to be transferable to a new corpus with little or no change.

In our experiments, we tested various learning algorithms and found that an SVM is comparable in performance to rigorously designed handcrafted systems, as presented in the literature. We reproduced experiments by Schwartz and Hearst (2003) and showed that our

testing framework was comparable to their work.

We integrated the acronym detection module with BaLIE's alias resolution. We demonstrate that it brings an interesting improvement, particularly at the level of organization recall in an acronym-rich corpus.

Our future work will consist of applying the supervised learning approach to different corpora, especially corpora in which acronyms or definitions are not always indicated by parentheses.

Chapter 7

Discussion and Conclusion

This thesis is about creating a semi-supervised NER system. It has the desirable property of requiring, as input, that an expert linguist lists a dozen examples of each supported entity type. It contrasts with the annotation of thousands of documents with hundreds of entity types, which is required for supervised learning. It also contrasts with manually harvesting NE lists and designing a complex rule system, which are usually required for handmade systems. The NER system we present in this thesis therefore requires very little supervision and we've included this human input in the Appendix.

The system presented in this thesis falls in the new category of semi-supervised and unsupervised systems. Work in this category is relatively rare and recent, and we believe ours to be the first that is devoted exclusively to the autonomous creation of an NER system.

Our overall goal is to create proof-of-concept software. In completing this system, we claim four major contributions that impact the NER field, and also have the potential to be used in other domains. First, we designed the first semi-supervised NER system that performs at a comparable level to that of a simple supervised learning-based NER system (Chapter 3). Second, we present a noise filter for generating NE lists based on computational linguistics and statistical semantic techniques (Chapter 4). This noise filter outperforms previous systems devoted to the same task. Then, we demonstrate a simple technique based on set intersections that can identify unambiguous examples for a given NE type (Chapter 5). Unambiguous NEs are a requirement for creating semi-supervised disambiguation rules. Finally, our fourth contribution is an acronym detection algorithm—part of an alias resolution system—that outperforms previous system and allows improvement in NER for a "less common and very difficult problem" (Chapter 6).

These contributions are crucial components to a successful semi-supervised NER system, and they are explained in the context of the whole system, for which the architecture is detailed in Figure 1. In the course of completing this system, however, we met many

limitations and difficulties, which we discuss in Section 7.1. We conclude this thesis by presenting our future work and some general long-term research ideas.

We believe the resulting system requiring little supervision has two important advantages over past systems put forth in the literature, and this is generally in favour of a shift towards semi-supervised and unsupervised techniques in the machine learning community. Our system is first extensible to new entity types. The design we adopted is free of linguistic knowledge or type-dependant heuristics. Therefore, we can modify the hierarchy or add new types, and let the system generate lists and rules. The system is also easily maintained over time. While supervised learning-based systems get most of their knowledge from large static training corpora, the system we present gets most of its knowledge from the Web. Recrawling the Web and periodically verifying the Web pages from which lists were extracted is a straightforward approach to maintenance.

7.1 Limitations

7.1.1 NE Hierarchy Design

The choice of NE types supported in this thesis is not completely free of constraints. Instead of choosing to recognize "cities," we may want to divide this type into "North American cities," "European cities," etc. However, our technique depends on the "natural order of things" or, more precisely, the way the majority of people decided to list entities on the Web. Noteworthy examples include the "cathedral," "hospital" and "hotel" types, which we decided to recognize in order to fit Sekine's hierarchy and/or BBN corpus. However, these entities rarely occur in exhaustive Web lists. We may have had more success in defining finer types by dividing according to country (there are a lot of regional and national hospital lists, for instance).

7.1.2 Controlling the Uncontrollable

When a bootstrapping algorithm is started, it is well-known that noise can impact and bias the result. If we have relative success in handling noise, we are still exposed to problems of

concept drift. For instance, cities, states, and countries are often mixed up in a single Web page, and the drift potential is very high. The worst case we met was with the "nationality" type. Not only it is highly ambiguous with the "language" type, but it also tends to drift toward "cuisine" type (an unsupported entity type). In our final nationality list, we find entries such as "sandwiches," "buffet," "seafood," etc.).

7.1.3 Extending to Other Languages

One natural extension for our system is applying it to many languages. However, it is limited in at least two ways. First, there is a need for a critical mass of structured information on the Web. In English, the "airport" list relies on 311 Web pages, the "drug" list was created from 1,323 Web pages, and common types such as "city" or "first_name" are usually aggregated from tens of thousands of Web pages. The Web seems to have reached this critical point for many entity types, in English. In some preliminary French-language experiments, we noticed that the task is much more difficult since many entity types may not meet the pages' critical mass. A second limitation is the very large corpus used for creating rule disambiguation. In this thesis, we use a Terabyte-sized corpus mainly composed of English texts. We would need comparable corpora for other languages.

7.1.4 Inaccurate Prior Probabilities

The rule disambiguation creation technique presented in Chapter 5 has an important problem with prior probabilities. For example, suppose we want to classify the word "April" between the "month" and "first_name" classes. A Bayesian classifier will evaluate each class probability by using the entity's prior probability and the conditional probabilities of the words in context. In this case, the prior probability that April is a month is much higher than its prior probability of being a first_name. However, our technique has absolutely no information on real priors. Instead of learning on a corpus representing the distribution of entities in the world, we create a data set of passages using unambiguous examples of entities. Because it is not representative of the reality, we choose to balance our data set so that priors are 50% for both classes. Correcting this prior is discussed in the section on future work.

7.1.5 Out-of-Vocabulary Entities

We presented a technique based primarily on lexicon look-up. When an entity is not part of our lexicon, it cannot be recognized. This is a limitation compared to a supervised learning-based system that usually classifies unknown words NEs if their contexts are sufficiently similar to the context of an entity class. On the one hand, we may use the disambiguation rules to recognize out-of-vocabulary entities. On the other hand, we may combine our system with an existing supervised learning-based system.

7.1.6 Lower-Case Entities

Some entity types in our hierarchy occur naturally in lower case. This is particularly true with species (e.g., bird, mammal) and substances (e.g., food, mineral). In this case, the heuristics aimed at resolving noun-entity ambiguity (Section 3.2.1) basically filter out every entity. In our experiments, we noticed very low recall for these types.

7.2 Future Work

The limitations presented in the previous section are all topics to broach in our future work. We believe none of them are completely beyond resolution. However, we identified three interesting research avenues that may impact the NER field or semi-supervised learning methods at large.

7.2.1 Statistical Semantics Technique as a Noise Filter

Our preliminary experiments using latent relational analysis (LRA; see Section 4.5) was only scratching the surface of the problem. We believe this kind of technique is the key to controlling concept drifts and other forms of difficult and subtle noise. One improvement we will explore is applying LRA on specific entities with high probability of being noise, such as ambiguous entities (entities belonging to two or more lists).

7.2.3 Prior Probabilities Correction

We will also address the correction of naive Bayes classifiers' prior probabilities in creating disambiguation rules. One way to do it is to calculate mutual information between an entity and each possible type to which it can belong. For instance, to disambiguate between the city of "Martin" and the first name "Martin," we may calculate Pointwise mutual information and information retrieval (PMI-IR) scores between this word and some unambiguous words of the class. In this case, preliminary experiments suggest that Martin is more likely to be a first name, and therefore, the prior probability that Martin is a first name should be higher.

7.2.4 Hybrid System

In this thesis, we present a standalone NER system. However, it could easily be used to maintain existing handmade or supervised systems. We are particularly interested in creating a hybrid system for which the core engine would be the Oak system (Sekine and Nabota 2004), and on top of which we would add our generated lists and disambiguation rules.

Another way to create a hybrid system would be to use one or many other NER systems to create an ensemble that would vote on each decision, and fall back in case of out-of-vocabulary entities.

7.3 Long-Term Research Ideas

The long-term objective of this research is to recognize and classify all the possible entity types with high precision. Then, it will be necessary to proceed to disambiguation within objects that have the same name and are within the same class. This problem is known as "personal name disambiguation" (e.g., Is "Jim Clark" the race driver, the film editor, or the Netscape founder?), but the idea extends to other types (e.g., Is "NRC" the National Research Council in Canada or in the United States?). To close the loop, it would be necessary to connect entities worldwide by resolving entity references across languages, a problem already garnering much interest from the machine translation community.

Bibliography

Adar, E. (2002) S-RAD A Simple and Robust Abbreviation Dictionary, *HP Laboratories Technical Report.*

Agbago, Akakpo; Kuhn, R. and Foster, G. (2006) Truecasing for the Portage System. *Proc. of International Conference on Recent Advances in Natural Language Processing.*

Alfonseca, Enrique and Manandhar, S. (2002) An Unsupervised Method for General Named Entity Recognition and Automated Concept Discovery. *Proc. International Conference on General WordNet.*

Asahara, Masayuki and Matsumoto, Y. (2003) Japanese Named Entity Extraction with Redundant Morphological Analysis. *Proc. Human Language Technology conference - North American chapter of the Association for Computational Linguistics.*

Basili, Roberto; Cammisa, M. and Donati, E. (2005) RitroveRAI: A Web Application for Semantic Indexing and Hyperlinking of Multimedia News. *International Semantic Web Conference.*

Bick, Eckhard (2004) A Named Entity Recognizer for Danish. *Proc. Conference on Language Resources and Evaluation.*

Bikel, Daniel M.; Miller, S.; Schwartz, R. and Weischedel, R. (1997) Nymble: a High-Performance Learning Name-finder. *Proc. Conference on Applied Natural Language Processing.*

Black, William J., Rinaldi, F. and Mowatt, D. (1998) Facile: Description of The NE System Used For Muc-7. *Proc. Message Understanding Conference.*

Bodenreider, Olivier and Zweigenbaum, P. (2000) Identifying Proper Names in Parallel Medical Terminologies. *Stud Health Technol Inform.* 77. pp. 443-447.

Boutsis, S., Demiros, I. , Giouli, V. , Liakata, M. , Papageorgiou, H. and Piperidis, S. (2000) A system for recognition of named entities in Greek. *Proc. International Conference on Natural Language Processing.*

Borthwick, Andrew; Sterling, J.; Agichtein, E. and Grishman, R. (1998) NYU: Description of the MENE Named Entity System as used in MUC-7. *Proc. Seventh Message Understanding Conference.*

Brin, Sergey (1998) Extracting Patterns and Relations from the World Wide Web. *Proc. Conference of Extending Database Technology. Workshop on the Web and Databases (workshop).*

Carreras, Xavier; Márques, L. and Padró, L. (2003) Named Entity Recognition for Catalan Using Spanish Resources. *Proc. Conference of the European Chapter of Association for Computational Linguistic.*

Chang, J. T.; Schütze, H. and Altman R.B., (2002), Creating an Online Dictionary of Abbreviations

from MEDLINE. *Journal of American Medical Informatics Association (JAMIA)*, 9(6), p.612-620.

Charniak, Eugene. (2001) Unsupervised Learning of Name Structure from Coreference Data. *Proc. Meeting of the North American Chapter of the Association for Computational Linguistics.*

Chawla, Nitesh V., Bowyer, K. W., Hall, L. O. and Kegelmeyer, W. P. (2002) SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research.* 16. pp. 321-357.

Chen, H. H. and Lee, J. C. (1996) Identification and Classification of Proper Nouns in Chinese Texts. *Proc. International Conference on Computational Linguistics.*

Chinchor, Nancy (1999) Overview of MUC-7/MET-2. *Proc. Message Understanding Conference MUC-7.*

Chinchor, Nancy; Robinson, P. and Brown, E. (1998) Hub-4 Named Entity Task Definition. *Proc. DARPA Broadcast News Workshop.*

Cimiano, Philipp and Völker, J. (2005) Towards Large-Scale, Open-Domain and Ontology-Based Named Entity Classification. *Proc. Conference on Recent Advances in Natural Language Processing.*

Coates-Stephens, Sam (1992) The Analysis and Acquisition of Proper Names for the Understanding of Free Text. *Computers and the Humanities.* 26. pp. 441-456.

Cohen, William W. (1995) Fast Effective Rule Induction. *Proc International Conference on Machine learning.*

Cohen, William W. and Fan, W. (1999) Learning Page-Independent Heuristics for Extracting Data from Web Page. *Proc. of the International World Wide Web Conference.*

Cohen, William and Richman, J. (2001) Learning to Match and Cluster Entity Names. *Proc. International ACM SIGIR Conference on Research and Development in Information Retrieval. Mathematical/Formal Methods in IR (workshop).*

Cohen, William W. (2004) *Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data*, http://minorthird.sourceforge.net.

Cohen, William W. and Sarawagi, S. (2004) Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. *Proc. Conference on Knowledge Discovery in Data.*

Collins, Michael (2002) Ranking Algorithms for Named–Entity Extraction: Boosting and the Voted Perceptron. *Proc. Association for Computational Linguistics.*

Collins Michael and Singer, Y. (1999) Unsupervised Models for Named Entity Classification. *Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and*

Very Large Corpora.

Cucchiarelli, Alessandro and Velardi, P. (2001) Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence. *Computational Linguistics*. 27(1). pp. 123-131.

Cucerzan, Silviu and Yarowsky, D. (1999) Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. *Proc. Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.*

Cunningham, Hamish, Maynard, D., Bontcheva, K., Tablan, V. (2002) GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics.*

Dannélls, Dana (2006) *Acronym Recognition: Recognizing Acronyms in Swedish Texts*. Master Thesis. Göteborg University.

Da Silva, Joaquim Ferreira; Kozareva, Z. and Lopes, G. P. (2004) Cluster Analysis and Classification of Named Entities. *Proc. Conference on Language Resources and Evaluation.*

Dimitrov, Marin; Bontcheva, K.; Cunningham H and Maynard, D. (2002) A Light-weight Approach to Coreference Resolution for Named Entities in Text. *Proc. Discourse Anaphora and Anaphor Resolution Colloquium.*

Doddington, George, Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S and Weischedel, R. (2004) The Automatic Content Extraction (ACE) Program – Tasks, Data, and Evaluation. *Proc. Conference on Language Resources and Evaluation.*

Downey, Doug, Etzioni, O, and Soderland, S. (2005) A Probabilistic Model of Redundancy in Information Extraction. In *Proc. International Joint Conference on Artificial Intelligence.*

Etzioni, Oren, 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, pp. 91-134.

Evans, Richard (2003) A Framework for Named Entity Recognition in the Open Domain. *Proc. Recent Advances in Natural Language Processing.*

Ferro, Lisa; Gerber, L.; Mani, I.; Sundheim, B. and Wilson G. (2005) *TIDES 2005 Standard for the Annotation of Temporal Expressions*. The MITRE Corporation.

Fleischman, Michael (2001) Automated Subcategorization of Named Entities. *Proc. Conference of the European Chapter of Association for Computational Linguistic.*

Fleischman, Michael and Hovy. E. (2002) Fine Grained Classification of Named Entities. *Proc. Conference on Computational Linguistics.*

Florian, Radu; Ittycheriah, A.; Jing H. and Zhang, T. (2003) Named Entity Recognition through Classifier Combination. *Proc. Conference on Computational Natural Language Learning.*

Frunza, Oana; Inkpen, D. and Nadeau, D. (2005) A Text Processing Tool for the Romanian Language. *Proc. of the EuroLAN 2005 Workshop on Cross-Language Knowledge Induction.*

Fung, Pascale (1995) A Pattern Matching Method for Finding Noun and Proper Noun Translations from Noisy Parallel Corpora. *Proc. Association for Computational Linguistics.*

Fürnkranz, Johannes (2002) Round Robin Classification. *Journal of Machine Learning Research.* 2. pp. 721-747.

Gaizauskas, Robert., Wakao, T., Humphreys, K., Cunningham, H. and Wilks, Y. (1995) University of Sheffield: Description of the LaSIE System as Used for MUC-6. *Proc. Message Understanding Conference.*

Grishman, Ralph and Sundheim, B. (1996) Message understanding conference - 6: A brief history. *Proc. International Conference on Computational Linguistics.*

Hearst, Marti (1992) Automatic Acquisition of Hyponyms from Large Text Corpora. *Proc. International Conference on Computational Linguistics.*

Heng, Ji and Grishman, R. (2006) Data Selection in Semi-supervised Learning for Name Tagging. *Proc. joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics. Information Extraction beyond the Document (Workshop)*

Huang, Fei (2005) *Multilingual Named Entity Extraction and Translation from Text and Speech.* Ph.D. Thesis. Carnegie Mellon University.

Jansche, Martin (2002) Named Entity Extraction with Conditional Markov Models and Classifiers. *Proc. Conference on Computational Natural Language Learning.*

Kokkinakis, Dimitri (1998), AVENTINUS, GATE and Swedish Lingware. Proc. of Nordic Computational Linguistics Conference.

Kripke, Saul (1982) *Naming and Necessity.* Harvard University Press.

Landauer, Thomas K. and Dumais, S. T. (1997) A Solution to Plato's Problem: The Latent Semantic Analysis Theory of the Acquisition, Induction, and Representation of Knowledge. *Psychological Review.* 104(2). pp. 211-240.

Larkey, L., Ogilvie, P., Price, A. and Tamilio, B. (2000) Acrophile: An Automated Acronym Extractor and Server, *In Proceedings of the ACM Digital Libraries conference.*

Lee, Seungwoo and Geunbae Lee, G. (2005) Heuristic Methods for Reducing Errors of Geographic Named Entities Learned by Bootstrapping. *Proc. International Joint Conference on Natural Language Processing.*

Li, Xin., Morie, P. and Roth, D. (2004) Identification and Tracing of Ambiguous Names: Discriminative and Generative Approaches. *Proc. National Conference on Artificial*

Intelligence.

Lin, Dekang (1998). Automatic retrieval and clustering of similar words. *Proc. International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics.*

Lin, Dekang and Pantel, P. (2001) Induction of Semantic Classes from Natural Language Text. *Proc. of ACM SIGKDD Conference on Knowledge Discovery and Data Mining.*

Ling, Charles X., and Li, C. (1998). Data Mining for Direct Marketing: Problems and Solutions. *Proc. International Conference on Knowledge Discovery and Data Mining.*

Liu, Bing, Dai, Y., Li, X., Lee W. S. and Yu, P. (2003) Building Text Classifiers Using Positive and Unlabelled Examples. *Proc. of the Third IEEE International Conference on Data Mining.*

Mann, Gideon S. and Yarowsky, D. (2003) Unsupervised Personal Name Disambiguation. Proc. *Conference on Computational Natural Language Learning.*

McDonald, David D. (1993) Internal and External Evidence in the Identification and Semantic Categorization of Proper Names. *Proc. Corpus Processing for Lexical Acquisition.*

May, Jonathan; Brunstein, A.; Natarajan, P. and Weischedel, R. M. (2003) Surprise! What's in a Cebuano or Hindi Name? *ACM Transactions on Asian Language Information Processing.* 2(3). pp. 169-180.

Maynard, Diana; Tablan, V.; Ursu, C.; Cunningham, H. and Wilks, Y. (2001) Named Entity Recognition from Diverse Text Types. *Proc. Recent Advances in Natural Language Processing.*

McCallum, Andrew and Li, W. (2003) Early Results for Named Entity Recognition with Conditional Random Fields, Features Induction and Web-Enhanced Lexicons. *Proc. Conference on Computational Natural Language Learning.*

Mikheev, Andrei (1999) A Knowledge-free Method for Capitalized Word Disambiguation. *Proc. Conference of Association for Computational Linguistics.*

Mikheev, A.; Moens, M. and Grover, C. (1999) Named Entity Recognition without Gazetteers. Proc. Conference of European Chapter of the Association for Computational Linguistics.

Minkov, Einat; Wang, R. and Cohen, W. (2005) Extracting Personal Names from Email: Applying Named Entity Recognition to Informal Text. *Proc. Human Language Technology and Conference Conference on Empirical Methods in Natural Language Processing.*

Nadeau, David (2005a) *Balie – Baseline Information Extraction : Multilingual Information Extraction from Text with Machine Learning and Natural Language Techniques.* Technical Report. University of Ottawa. http://balie.sourceforge.net/dnadeau05balie.pdf

Nadeau, David (2005b) Création de surcouche de documents hypertextes et traitement du langage

naturel. *Proc. Computational Linguistics in the North-East.*

Nadeau, David and Turney, P. (2005) A Supervised Learning Approach to Acronym Identification. *Proc. Canadian Conference on Artificial Intelligence.*

Nadeau, David; Turney, P. and Matwin, S. (2006) Unsupervised Named Entity Recognition: Generating Gazetteers and Resolving Ambiguity. *Proc. Canadian Conference on Artificial Intelligence.*

Nadeau, David and Sekine, S. (2007) A Survey of Named Entity Recognition and Classification. In: Sekine, S. and Ranchhod, E. *Named Entities: Recognition, classification and use.* Special issue of Lingvisticæ Investigationes. 30(1) pp. 3-26.

Narayanaswamy, Meenakshi; Ravikumar K. E. and Vijay-Shanker K. (2003) A Biological Named Entity Recognizer. *Proc. Pacific Symposium on Biocomputing.*

Ohta, Tomoko; Tateisi, Y.; Kim, J.; Mima, H. and Tsujii, J. (2002) The GENIA Corpus: An Annotated Research Abstract Corpus in Molecular Biology Domain. *Proc. Human Language Technology Conference.*

Palmer, David D. and Day, D. S. (1997) A Statistical Profile of the Named Entity Task. *Proc. ACL Conference for Applied Natural Language Processing.*

Park, Y. and Byrd, R. J. (2001) Hybrid Text Mining for Finding Abbreviations and Their Definitions, *Proc. of the 2001 Conference on Empirical Methods in Natural Language Processing.*

Palmer, David D. and Day, D. S. (1997) A Statistical Profile of the Named Entity Task. *Proc. ACL Conference for Applied Natural Language Processing.*

Pasca, Marius (2004) Acquisition of Categorized Named Entities for Web Search. In *Proc. Conference on Information and Knowledge Management.*

Pasca, Marius; Lin, D.; Bigham, J.; Lifchits, A. and Jain, A. (2006) Organizing and Searching the World Wide Web of Facts—Step One: The One-Million Fact Extraction Challenge. *Proc. National Conference on Artificial Intelligence.*

Patrick, Jon; Whitelaw, C. and Munro, R. (2002) SLINERC: The Sydney Language-Independent Named Entity Recogniser and Classifier. *Proc. Conference on Natural Language Learning.*

Pedersen, Ted (2002) A Baseline Methodology for Word Sense Disambiguation. Proc. Third International Conference on Intelligent Text Processing and Computational Linguistics.

Petasis, Georgios, Vichot, F., Wolinski, F., Paliouras, G., Karkaletsis, V. and Spyropoulos, C. D. (2001) Using Machine Learning to Maintain Rule-based Named-Entity Recognition and Classification Systems. *Proc. Conference of Association for Computational Linguistics.*

Piskorski, Jakub (2004) Extraction of Polish Named-Entities. *Proc. Conference on Language Resources an Evaluation.*

Poibeau, Thierry (2003) The Multilingual Named Entity Recognition Framework. *Proc. Conference on European chapter of the Association for Computational Linguistics.*

Poibeau, Thierry (2006) Dealing with Metonymic Readings of Named Entities. *Proc. Annual Conference of the Cognitive Science Society.*

Poibeau, Thierry and Kosseim, L. (2001) Proper Name Extraction from Non-Journalistic Texts. *Proc. Computational Linguistics in the Netherlands.*

Popov, Borislav; Kirilov, A.; Maynard, D. and Manov, D. (2004) Creation of reusable components and language resources for Named Entity Recognition in Russian. *Proc. Conference on Language Resources and Evaluation.*

Pustejovsky, J.; Castao, J.; Cochran, B.; Kotecki, M.; Morrell, M. and Rumshisky, A. (2001) Extraction and Disambiguation of Acronym-Meaning Pairs in Medline, *unpublished manuscript.*

Radev, Dragomir (1998) Learning Correlations between Linguistic Indicators and Semantic Constraints: Reuse of Context-Dependent Descriptions of Entities. *Proc. joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics.*

Raghavan, Hema and Allan, J. (2004) Using Soundex Codes for Indexing Names in ASR documents. *Proc. Human Language Technology conference - North American chapter of the Association for Computational Linguistics. Interdisciplinary Approaches to Speech Indexing and Retrieval (workshop).*

Rau, Lisa F. (1991) Extracting Company Names from Text. Proc. *Conference on Artificial Intelligence Applications of IEEE.*

Ravin, Yael and Wacholder, N. (1996) *Extracting Names from Natural-Language Text.* IBM Research Report RC 2033.

Riloff, Ellen and Jones, R (1999) Learning Dictionaries for Information Extraction using Multi-level Bootstrapping. *Proc. National Conference on Artificial Intelligence.*

Rindfleisch, Thomas C.; Tanabe, L. and Weinstein, J. N. (2000) EDGAR: Extraction of Drugs, Genes and Relations from the Biomedical Literature. *Proc. Pacific Symposium on Biocomputing.*

Sánchez, David and Moreno, A. (2005) Web Mining Techniques for Automatic Discovery of Medical Knowledge. *Proc. Conference on Artificial Intelligence in Medicine.*

Santos, Diana; Seco, N.; Cardoso, N. and Vilela, R. (2006) HAREM: An Advanced NER Evaluation Contest for Portuguese. *Proc. International Conference on Language Resources and Evaluation.*

Schölkopf, Bernhard, Platt, J., Shawe-Taylor, J., Smola, A. J. and Williamson, R. C. (2001)

Estimating the support of a High-Dimensional Distribution. *Neural Computation*, 13, pp. 1443-1471.

Schwab, Ingo and Pohl, W. (1999) Learning User Profiles from Positive Examples. *In Proc. of the International Conference on Machine Learning & Applications.*

Schwartz, A. and Hearst, M. (2003), A simple algorithm for identifying abbreviation definitions in biomedical texts, *In Proceedings of the Pacific Symposium on Biocomputing.*

Sekine, Satoshi (1998) Nyu: Description of The Japanese NE System Used For Met-2. *Proc. Message Understanding Conference.*

Sekine, Satoshi and Isahara, H. (2000) IREX: IR and IE Evaluation project in Japanese. *Proc. Conference on Language Resources and Evaluation.*

Sekine, Satoshi and Nobata, C. (2004) Definition, dictionaries and tagger for Extended Named Entity Hierarchy. *Proc. Conference on Language Resources and Evaluation.*

Settles, Burr (2004) Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. *Proc. Conference on Computational Linguistics. Joint Workshop on Natural Language Processing in Biomedicine and its Applications (workshop).*

Shen D., Zhang, J., Zhou, G., Su, J. and Tan, C. L. (2003) Effective Adaptation of a Hidden Markov Model-based Named Entity Recognizer for Biomedical Domain. *Proc. Conference of Association for Computational Linguistics. Natural Language Processing in Biomedicine (workshop).*

Shinyama, Yusuke and Sekine, S. (2004) Named Entity Discovery Using Comparable News Articles. *Proc. International Conference on Computational Linguistics.*

Smith, David A. (2002) Detecting and Browsing Events in Unstructured Text. *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval.*

Srihari, Rohini and Li, W. (1999) Information Extraction Supported Question Answering. *Proc. Text Retrieval Conference.*

Steinberger, Ralf and Pouliquen, B. (2007) Cross-lingual Named Entity Recognition. In: Sekine, S. and Ranchhod, E. *Named Entities: Recognition, classification and use.* Special issue of Lingvisticæ Investigationes. 30(1) pp.135-162.

Swan, Russell and Allan, J. (1999) Extracting Significant Time Varying Features from Text. *Proc. International Conference on Information Knowledge Management.*

Szarvas, György; Farkas, R and Kocsor, A. (2006) A Multilingual Named Entity Recognition System Using Boosting and C4.5 Decision Tree Learning Algorithms. *Discovery Science 2006.*

Taghva, K. and Gilbreth, J. (1999), Recognizing acronyms and their definitions, *International journal on Document Analysis and Recognition*, pp. 191-198.

Terra, Egidio and Clarke, C. (2003) Frequency Estimates for Statistical Word Similarity Measures. *Proc. Human Language Technology and North American Chapter of Association of Computational Linguistics Conference.*

Thielen, Christine (1995) An Approach to Proper Name Tagging for German. *Proc. Conference of European Chapter of the Association for Computational Linguistics. SIGDAT (workshop).*

Tjong Kim Sang, Erik. F. (2002) Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. *Proc. Conference on Natural Language Learning.*

Tjong Kim Sang, Erik. F. and De Meulder, F. (2003) Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *Proc. Conference on Natural Language Learning.*

Tsuruoka, Yoshimasa and Tsujii, J. (2003) Boosting Precision and Recall of Dictionary-Based Protein Name Recognition. *Proc. Conference of Association for Computational Linguistics. Natural Language Processing in Biomedicine (workshop).*

Tufis, D. and Mason, O. (1998). Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger, *Proceedings of the First International Conference on Language Resources and Evaluation.*

Turney, Peter (2001) Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. *Proc. European Conference on Machine Learning.*

Turney, Peter D. (2005) Measuring Semantic Similarity by Latent Relational Analysis. *In Proc. International Joint Conference on Artificial Intelligence.*

Tzong-Han Tsai, Richard; Wu S.-H.; Chou, W.-C.; Lin, Y.-C.; He, D.; Hsiang, J.; Sung, T.-Y. and Hsu, W.-L. (2006) Various Criteria in the Evaluation of Biomedical Named Entity Recognition. *BMC Bioinformatics.* 7(92).

Vilar, David; Xu, J.; D'Haro, L. F. and Ney, H. (2006) Error Analysis of Statistical Machine Translation Output. *Proc. Language Resources and Evaluation conference*

Wang, Liang-Jyh; Li, W.-C. and Chang, C.-H. (1992) Recognizing Unregistered Names for Mandarin Word Identification. *Proc. International Conference on Computational Linguistics.*

Wang, Lee, Wang, C., Xie, X., Forman, J., Lu, Y., Ma, W.-Y. and Li, Y. (2005) Detecting Dominant Locations from Search Queries. *Proc. International ACM SIGIR Conference.*

Whitelaw, Casey and Patrick, J. (2003) Evaluating Corpora for Named Entity Recognition Using Character-Level Features. *Proc. Australian Conference on Artificial Intelligence.*

Witten, Ian. H.; Bray, Z.; Mahoui, M. and Teahan W. J. (1999) Using Language Models for Generic Entity Extraction. *Proc. International Conference on Machine Learning. Text Mining (workshop).*

Witten, Ian H. and Frank, E. (2000) *Data Mining: Practical machine learning tools with Java implementations,* Morgan Kaufmann, San Francisco.

Wolinski, Francis; Vichot, F. and Dillet, B. (1995) Automatic Processing Proper Names in Texts. *Proc. Conference on European Chapter of the Association for Computational Linguistics.*

Yarowsky, David and Florian, R. (2002) Evaluating Sense Disambiguation across Diverse Parameter Spaces. *Journal of Natural Language Engineering.* 8(2). pp. 293-310.

Yangarber, Roman; Lin, W. and Grishman, R. (2002) Unsupervised Learning of Generalized Names. *Proc. of International Conference on Computational Linguistics.*

Yeates, S. (1999), Automatic Extraction of Acronyms from Text. *In Third New Zealand Computer Science Research Students' Conference.*

Yu, H.; Hripcsak G. and Friedman C. (2002) Mapping abbreviations to full forms in biomedical articles, *Journal of the American Medical Informatics Association (*9) pp. 262-272.

Yu, Shihong; Bai S. and Wu, P. (1998) Description of the Kent Ridge Digital Labs System Used for MUC-7. *Proc. Message Understanding Conference.*

Zahariev, M. (2004). *A (Acronyms)*, Ph.D. thesis, School of Computing Science, Simon Fraser University.

Zhu, Jianhan; Uren, V. and Motta, E. (2005) ESpotter: Adaptive Named Entity Recognition for Web Browsing. *Proc. Conference Professional Knowledge Management. Intelligent IT Tools for Knowledge Management Systems (workshop).*

Zhu, Xingquan, Wu, X. and Chen Q. (2003) Eliminating Class Noise in Large Data-Sets, *Proc. of the International Conference on Machine Learning.*

Appendix: Seed words (system input)

```
<?xml version="1.0" ?>
<entityHierarchy InspiredBy="Satoshi Sekine, Ada Brunstein" Modification="December 12th 2006">
<entity type="name"><entity type="person">
<entity type="first_name" classwords="first name;given name" estimatedpopulation="10E3">
        <seed>Mary</seed>                    <seed>Elizabeth</seed>
        <seed>Rose</seed>                    <seed>Britney</seed>
        <seed>David</seed>                   <seed>Veronica</seed>
        <seed>Susan</seed>                   <seed>Thomas</seed>
        <seed>Robert</seed>                  <seed>Louise</seed>
        <seed>Aaron</seed>                   <seed>Napoleon</seed>
        <seed>James</seed>                   <seed>Catherine</seed>
        <seed>Michael</seed>
</entity>
<entity type="person_title" classwords="title" estimatedpopulation="10E0">
        <seed>Mr</seed>                      <seed>Msr</seed>
        <seed>Sir</seed>                     <seed>Madam</seed>
        <seed>Ms</seed>                      <seed>Mister</seed>
        <seed>Dr</seed>                      <seed>Jr</seed>
        <seed>Prof</seed>                    <seed>Doctor</seed>
        <seed>Lt</seed>                      <seed>Col</seed>
        <seed>Sgt</seed>                     <seed>Phd</seed>
        <seed>Miss</seed>
</entity>
<entity type="last_name" classwords="last name;familly name" estimatedpopulation="10E3">
        <seed>Smith</seed>                   <seed>Johnson</seed>
        <seed>O'Connor</seed>                <seed>Tremblay</seed>
        <seed>Clinton</seed>                 <seed>Williams</seed>
        <seed>Miller</seed>                  <seed>Fletcher</seed>
        <seed>Woods</seed>                   <seed>Anderson</seed>
        <seed>Clark</seed>                   <seed>Robinson</seed>
        <seed>Peterson</seed>                <seed>Foster</seed>
        <seed>Perkins</seed>
</entity>
<entity type="celebrity" classwords="celebrity;name" estimatedpopulation="10E3">
        <seed>Robert De Niro</seed>          <seed>Edgar Allan Poe</seed>
        <seed>Albert Einstein</seed>         <seed>Marie Curie</seed>
        <seed>Isaac Newton</seed>            <seed>George Washington</seed>
        <seed>Galileo Galilei</seed>         <seed>Charles Darwin</seed>
        <seed>William Shakespeare</seed>     <seed>Abraham Lincoln</seed>
        <seed>Mark Twain</seed>              <seed>Sigmund Freud</seed>
        <seed>Pablo Picasso</seed>           <seed>Ernest Hemingway</seed>
        <seed>John F. Kennedy</seed>
</entity>
<entity type="vocation" classwords="vocation;profession" estimatedpopulation="10E2">
        <seed>firefighter</seed>             <seed>journalist</seed>
        <seed>bodyguard</seed>               <seed>nurse</seed>
        <seed>teacher</seed>                 <seed>Banker</seed>
        <seed>Athlete</seed>                 <seed>Entertainer</seed>
        <seed>Electrician</seed>             <seed>Carpenter</seed>
        <seed>Union leader</seed>            <seed>salesperson</seed>
        <seed>Civil Engineer</seed>          <seed>Police Officer</seed>
        <seed>Dental Hygienist</seed>
```

```
</entity>
<entity type="title" classwords="title" estimatedpopulation="10E2">
        <seed>Executive Director</seed>                    <seed>Chief administrative officer</seed>
        <seed>Chief Executive Officer</seed>               <seed>manager</seed>
        <seed>Foreman</seed>                               <seed>secretary general</seed>
        <seed>vice president</seed>                        <seed>Treasurer</seed>
        <seed>Vice President</seed>                        <seed>chairman</seed>
        <seed>Vice Chairman</seed>                         <seed>representative</seed>
        <seed>Secretary</seed>                             <seed>Press Secretary</seed>
        <seed>President</seed>
</entity>
<entity type="character" classwords="character;fictional" estimatedpopulation="10E2">
        <seed>Mickey Mouse</seed>                          <seed>Pink Panther</seed>
        <seed>Peter Rabbit</seed>                          <seed>Pinocchio</seed>
        <seed>Yoda</seed>                                  <seed>Tarzan</seed>
        <seed>Spider-Man</seed>                            <seed>Batman</seed>
        <seed>Captain America</seed>                       <seed>Superman</seed>
        <seed>Flash Gordon</seed>                          <seed>Peanuts</seed>
        <seed>Donald Duck</seed>                           <seed>Bambi</seed>
        <seed>Woody Woodpecker</seed>
</entity></entity>
<entity type="organization">
<entity type="company" classwords="company" estimatedpopulation="10E3">
        <seed>Citigroup Inc.</seed>                        <seed>Coca-Cola Company</seed>
        <seed>Groupe Danone</seed>                         <seed>Toyota Motor Corp.</seed>
        <seed>Verizon</seed>                               <seed>Continental Airlines</seed>
        <seed>General Electric</seed>                      <seed>Kellogg Company</seed>
        <seed>New York Times</seed>                        <seed>Panasonic</seed>
        <seed>US Airways</seed>                            <seed>Radio Shack</seed>
        <seed>Nokia</seed>                                 <seed>Walgreens</seed>
        <seed>Office Depot</seed>
</entity>
<entity type="company_designator" classwords="company" estimatedpopulation="30">
        <seed>Co</seed>                                    <seed>corp</seed>
        <seed>inc</seed>                                   <seed>Ltd</seed>
        <seed>PLC</seed>                                   <seed>AENP</seed>
        <seed>CPORA</seed>                                 <seed>LLC</seed>
        <seed>L.L.C</seed>                                 <seed>LP</seed>
        <seed>L.P</seed>
</entity>
<entity type="military" classwords="military" estimatedpopulation="10E2">
        <seed>Canadian Forces</seed>                       <seed>Israeli Defense Forces</seed>
        <seed>Marine Troops</seed>                         <seed>Italian Navy</seed>
        <seed>United States Marine Corps</seed>            <seed>West India Regiment</seed>
        <seed>Serenissima Regiment</seed>                  <seed>Marina Militare</seed>
        <seed>Royal Marines</seed>                         <seed>Continental Marines</seed>
        <seed>Korps Mariniers</seed>                       <seed>Spanish Marine Infantry</seed>
        <seed>Italian Army</seed>                          <seed>Russian Naval Infantry</seed>
        <seed>Portuguese Navy</seed>
</entity>
<entity type="association" classwords="association" estimatedpopulation="10E2">
        <seed>Amnesty International</seed>                 <seed>Council of Europe</seed>
        <seed>World Bank</seed>                            <seed>United Nations</seed>
        <seed>Friends of the Earth</seed>                  <seed>International Monetary Fund</seed>
        <seed>Human Rights Watch</seed>                    <seed>World Health Organization</seed>
        <seed>League of Nations</seed>                     <seed>UNICEF</seed>
```

```
        <seed>African Union</seed>                          <seed>Foreign Policy Association</seed>
        <seed>Commonwealth</seed>                           <seed>Asian Development Bank</seed>
        <seed>UNESCO</seed>
</entity>
<entity type="government" classwords="government" estimatedpopulation="10E2">
        <seed>Department of Justice</seed>                  <seed>DARPA</seed>
        <seed>NASA</seed>                                   <seed>Department of Transportation</seed>
        <seed>Food and Drug Administration</seed>           <seed>Department of Defense</seed>
        <seed>Department of Agriculture</seed>              <seed>Department of Energy</seed>
        <seed>Securities and Exchange Commission</seed>     <seed>National Science Foundation</seed>
        <seed>Social Security Administration</seed>         <seed>Central Intelligence Agency</seed>
        <seed>Federal Communications Commission</seed>      <seed>Department of Commerce</seed>
        <seed>Department of Labor</seed>
</entity>
<entity type="political_party" classwords="political party;political" estimatedpopulation="10E1">
        <seed>Democratic Party</seed>                       <seed>Republican Party</seed>
        <seed>Liberal Party of Canada</seed>                <seed>Conservative Party of Canada</seed>
        <seed>New Democratic Party</seed>                   <seed>Libertarian Party</seed>
        <seed>Constitution Party</seed>                     <seed>Green Party</seed>
        <seed>Socialist Party</seed>                        <seed>Bloc Quebecois</seed>
        <seed>Labour Party</seed>                           <seed>Conservative Party</seed>
        <seed>Liberal Democrats</seed>                      <seed>Democratic Unionist Party</seed>
        <seed>The Greens</seed>
</entity>
<entity type="political_line" classwords="political movement;political" estimatedpopulation="50E0">
        <seed>democrat</seed>                               <seed>Communist</seed>
        <seed>socialist</seed>                              <seed>Fascist</seed>
        <seed>republican</seed>                             <seed>Republican</seed>
        <seed>Libertarian</seed>                            <seed>Independent</seed>
        <seed>anarchist</seed>                              <seed>Leninism</seed>
        <seed>Marxism</seed>                                <seed>Trotskyism</seed>
        <seed>nationalism</seed>                            <seed>liberalism</seed>
        <seed>totalitarianism</seed>
</entity>
<entity type="nationality" classwords="nationality;nation" estimatedpopulation="20E1">
        <seed>American</seed>                               <seed>japanese</seed>
        <seed>Israeli</seed>                                <seed>korean</seed>
        <seed>chinese</seed>                                <seed>Dutch</seed>
        <seed>Arabic</seed>                                 <seed>Portuguese</seed>
        <seed>Turkish</seed>                                <seed>Czech</seed>
        <seed>Algerian</seed>                               <seed>Canadian</seed>
        <seed>Taiwanese</seed>                              <seed>Haitian</seed>
        <seed>Mexican</seed>
</entity>
<entity type="market" classwords="market" estimatedpopulation="10E1">
        <seed>Amsterdam Stock Exchange</seed>               <seed>NASDAQ</seed>
        <seed>Toronto Stock Exchange</seed>                 <seed>Tokyo Stock Exchange</seed>
        <seed>NYSE</seed>                                   <seed>Korea Stock Exchange</seed>
        <seed>Philadelphia Stock Exchange</seed>            <seed>London Stock Exchange</seed>
        <seed>Australian Stock Exchange</seed>              <seed>London Metal Exchange</seed>
        <seed>Helsinki Stock Exchange</seed>                <seed>Taiwan Stock Exchange</seed>
        <seed>Winnipeg Commodity Exchange</seed>            <seed>Singapore Exchange</seed>
        <seed>NYMEX</seed>
</entity>
<entity type="religious_group" classwords="religious group;religious" estimatedpopulation="10E1">
        <seed>Jewish</seed>                                 <seed>catholic</seed>
```

```
        <seed>Hindu</seed>                          <seed>Muslim</seed>
        <seed>protestant</seed>                     <seed>Buddhist</seed>
        <seed>Protestant</seed>                     <seed>Taoist</seed>
        <seed>Anglican</seed>                       <seed>Christian</seed>
        <seed>Sindhi</seed>                         <seed>Hindi</seed>
        <seed>Gujarati</seed>                       <seed>Telugu</seed>
        <seed>Tamil</seed>
</entity>
<entity type="sports_team" classwords="team;sports" estimatedpopulation="10E2">
        <seed>Los Angeles Raiders</seed>            <seed>New York Mets</seed>
        <seed>Calgary Flames</seed>                 <seed>Montreal Canadiens</seed>
        <seed>Dallas Stars</seed>                   <seed>Edmonton Oilers</seed>
        <seed>Los Angeles Kings</seed>              <seed>Houston Astros</seed>
        <seed>San Diego Padres</seed>               <seed>Pittsburgh Pirates</seed>
        <seed>Cincinnati Reds</seed>                <seed>New York Knicks</seed>
        <seed>Detroit Pistons</seed>                <seed>Indiana Pacers</seed>
        <seed>Boston Celtics</seed>
</entity></entity>
<entity type="location"><entity type="geo_political">
<entity type="city" classwords="city;town" estimatedpopulation="10E3">
        <seed>Ottawa</seed>                         <seed>Toronto</seed>
        <seed>Paris</seed>                          <seed>Dallas</seed>
        <seed>Sydney</seed>                         <seed>Boston</seed>
        <seed>New York</seed>                       <seed>Amsterdam</seed>
        <seed>Nashville</seed>                      <seed>Rome</seed>
        <seed>Barcelona</seed>                      <seed>Montreal</seed>
        <seed>Dublin</seed>                         <seed>Washington DC</seed>
        <seed>Prague</seed>
</entity>
<entity type="state_province" classwords="state;province" estimatedpopulation="10E1">
        <seed>Quebec</seed>                         <seed>Ontario</seed>
        <seed>Texas</seed>                          <seed>Alaska</seed>
        <seed>California</seed>                     <seed>Rhode Island</seed>
        <seed>British Columbia</seed>               <seed>Virginia</seed>
        <seed>Louisiana</seed>                      <seed>North Carolina</seed>
        <seed>Manitoba</seed>                       <seed>Michigan</seed>
        <seed>Hawaii</seed>                         <seed>Delaware</seed>
        <seed>Florida</seed>
</entity>
<entity type="county" classwords="state;province" estimatedpopulation="10E1">
        <seed>Autauga County</seed>                 <seed>Albany County</seed>
        <seed>Washington County</seed>              <seed>York County</seed>
        <seed>Pacific County</seed>                 <seed>Tuscaloosa County</seed>
        <seed>Montgomery County</seed>              <seed>Taylor County</seed>
        <seed>Williamson County</seed>              <seed>Tallapoosa County</seed>
        <seed>Sevier County</seed>                  <seed>Summit County</seed>
        <seed>Rockingham County</seed>              <seed>Wayne County</seed>
        <seed>Winneshiek County</seed>
</entity>
<entity type="country" classwords="country" estimatedpopulation="20E1">
        <seed>Canada</seed>                         <seed>United States</seed>
        <seed>France</seed>                         <seed>Egypt</seed>
        <seed>Morocco</seed>                        <seed>New Zealand</seed>
        <seed>South Africa</seed>                   <seed>Indonesia</seed>
        <seed>Iraq</seed>                           <seed>Togo</seed>
        <seed>Germany</seed>                        <seed>Brasil</seed>
```

```
        <seed>Netherlands</seed>                    <seed>Austria</seed>
        <seed>Mexico</seed>
</entity></entity>
<entity type="region" classwords="region;subregion" estimatedpopulation="10E1">
        <seed>Latin America</seed>                  <seed>Caribbean</seed>
        <seed>Middle East</seed>                    <seed>Scandinavia</seed>
        <seed>Australasia</seed>                    <seed>Asia Pacific</seed>
        <seed>North America</seed>                  <seed>Mediterranean</seed>
        <seed>Eastern Europe</seed>                 <seed>Western Europe</seed>
        <seed>North Africa</seed>                   <seed>South Asia</seed>
        <seed>East Africa</seed>                    <seed>Central America</seed>
        <seed>South America</seed>
</entity>
<entity type="geological">
<entity type="landform" classwords="landform;mountain" estimatedpopulation="10E1">
        <seed>Mount Everest</seed>                  <seed>K2</seed>
        <seed>Kangchenjunga</seed>                  <seed>Lhotse</seed>
        <seed>Makalu</seed>                         <seed>Manaslu</seed>
        <seed>Dhaulagiri</seed>                     <seed>Annapurna</seed>
        <seed>Cho Oyu</seed>                        <seed>Broad Peak</seed>
        <seed>Nanga Parbat</seed>                   <seed>Gasherbrum II</seed>
        <seed>Gasherbrum I</seed>                   <seed>Shishapangma</seed>
        <seed>Shisha Pangma</seed>
</entity>
<entity type="waterform">
<entity type="river" classwords="river;waterway" estimatedpopulation="10E2">
        <seed>Tennessee River</seed>                <seed>Mississippi river</seed>
        <seed>Buller river</seed>                   <seed>Amazon river</seed>
        <seed>Orange River</seed>                   <seed>Susquehanna River</seed>
        <seed>Savannah River</seed>                 <seed>Missouri River</seed>
        <seed>Mackenzie River</seed>                <seed>Murray River</seed>
        <seed>Arkansas River</seed>                 <seed>Niagara River</seed>
        <seed>Hudson River</seed>                   <seed>Colorado River</seed>
        <seed>Columbia River</seed>
</entity>
<entity type="lake" classwords="lake" estimatedpopulation="10E2">
        <seed>Lake Michigan</seed>                  <seed>Lake Superior</seed>
        <seed>Lake Ontario</seed>                   <seed>Lake Huron</seed>
        <seed>Lake Erie</seed>                      <seed>Lake Baikal</seed>
        <seed>Lhagba Pool</seed>                    <seed>Nettilling Lake</seed>
        <seed>Lake Toba</seed>                      <seed>Lake Wanapitei</seed>
        <seed>Lake Enriquillo</seed>                <seed>Lake Victoria</seed>
        <seed>Lake Vostok</seed>                    <seed>Lake Eyre</seed>
        <seed>Lake Titicaca</seed>
</entity>
<entity type="sea" classwords="sea" estimatedpopulation="10E1">
        <seed>Dead sea</seed>                       <seed>Mediterranean Sea</seed>
        <seed>Red sea</seed>                        <seed>Caspian Sea</seed>
        <seed>Irish Sea</seed>                      <seed>Gulf of St. Lawrence</seed>
        <seed>Gulf of Mexico</seed>                 <seed>Philippine Sea</seed>
        <seed>Bohai Sea</seed>                      <seed>Timor Sea</seed>
        <seed>North Sea</seed>                      <seed>Baltic Sea</seed>
        <seed>Beaufort Sea</seed>                   <seed>Norwegian Sea</seed>
        <seed>Gulf of Oman</seed>
</entity>
<entity type="ocean_bay" classwords="ocean;bay" estimatedpopulation="10E0">
```

```
        <seed>Hudson bay</seed>                    <seed>Atlantic ocean</seed>
        <seed>Pacific ocean</seed>                 <seed>Indian ocean</seed>
        <seed>Artic ocean</seed>                   <seed>Antartic ocean</seed>
        <seed>Baffin Bay</seed>                    <seed>Bay of Fundy</seed>
        <seed>Bay of Biscay</seed>                 <seed>James Bay</seed>
        <seed>Khanty Ocean</seed>                  <seed>Paleo-Tethys Ocean</seed>
        <seed>Proto-Tethys Ocean</seed>            <seed>Pan-African Ocean</seed>
        <seed>Southern Ocean</seed>
</entity></entity>
<entity type="continent" classwords="continent" estimatedpopulation="10E0">
        <seed>North America</seed>                 <seed>South America</seed>
        <seed>Europe</seed>                        <seed>Asia</seed>
        <seed>Africa</seed>                        <seed>Antarctica</seed>
        <seed>Antarctic</seed>                     <seed>Australia</seed>
</entity></entity>
<entity type="astral_body">
<entity type="planet" classwords="planet" estimatedpopulation="10E0">
        <seed>Pluto</seed>                         <seed>Mercury</seed>
        <seed>earth</seed>                         <seed>Mars</seed>
        <seed>Venus</seed>                         <seed>Saturn</seed>
        <seed>Ceres</seed>                         <seed>Jupiter</seed>
        <seed>Eris</seed>                          <seed>Uranus</seed>
        <seed>Neptun</seed>                        <seed>Moon</seed>
        <seed>Titan</seed>                         <seed>Miranda</seed>
        <seed>Phobos</seed>
</entity>
<entity type="star" classwords="star" estimatedpopulation="10E1">
        <seed>Solar system</seed>                  <seed>Orion</seed>
        <seed>Great Bear</seed>                    <seed>Zeta Ophiuchi</seed>
        <seed>Rigel</seed>                         <seed>Altair</seed>
        <seed>Procyon A</seed>                     <seed>Sun</seed>
        <seed>Epsilon Indi</seed>                  <seed>Proxima Centauri</seed>
        <seed>BPM 37093</seed>                     <seed>Nemesis</seed>
        <seed>P Cygni</seed>                       <seed>Zeta Bootis</seed>
        <seed>Cygnus X-1</seed>
</entity></entity></entity>
<entity type="facility"><entity type="property">
<entity type="amphitheatre" classwords="amphitheatre" estimatedpopulation="10E1">
        <seed>Molson amphitheatre</seed>           <seed>Universal amphitheatre</seed>
        <seed>Hollywood Bowl</seed>                <seed>Colosseum</seed>
        <seed>Ford amphitheatre</seed>             <seed>Arles Amphitheatre</seed>
        <seed>Mediolanum Santonum</seed>           <seed>Perigueux</seed>
        <seed>Amphitheatrum Castrense</seed>       <seed>Ludus Magnus</seed>
        <seed>Porolissum</seed>                    <seed>Verona Arena</seed>
        <seed>Augusta Raurica</seed>               <seed>Caerleon</seed>
        <seed>Venta Silurum</seed>
</entity>
<entity type="cathedral" classwords="cathedral" estimatedpopulation="10E1">
        <seed>St Patrick's Cathedral</seed>        <seed>Saint Louis Cathedral</seed>
        <seed>Peterborough Cathedral</seed>        <seed>Saint Raphael's Cathedral</seed>
        <seed>Bristol cathedral</seed>             <seed>Canterberry Cathedral</seed>
        <seed>Cathedral of Sao Paulo</seed>        <seed>Wells Cathedral</seed>
        <seed>Wawel Cathedral</seed>               <seed>Cathedral of Parma</seed>
        <seed>Saint Louis Cathedral</seed>         <seed>Lutheran Helsinki Cathedral</seed>
        <seed>Mexico City Metropolitan Cathedral</seed>  <seed>Ulm Munster</seed>
        <seed>Saint Isaac's Cathedral</seed>
```

```
</entity>
<entity type="castle" classwords="castle" estimatedpopulation="50E0">
        <seed>Prague Castle</seed>                    <seed>Uppsala Castle</seed>
        <seed>Bran Castle</seed>                      <seed>St. Olaf's Castle</seed>
        <seed>Trakai Island Castle</seed>             <seed>Medina del Campo</seed>
        <seed>Wawel Castle</seed>                     <seed>Craigievar Castle</seed>
        <seed>Cahir Castle</seed>                     <seed>Moscow Kremlin</seed>
        <seed>Harlech Castle</seed>                   <seed>Aberdeenshire</seed>
        <seed>Hunyad Castle</seed>                    <seed>Turku Castle</seed>
        <seed>Castrum Danorum</seed>
</entity>
<entity type="skyscraper" classwords="skyscraper;building" estimatedpopulation="10E1">
        <seed>Taipei 101</seed>                       <seed>Sears tower</seed>
        <seed>Empire State Building</seed>            <seed>Bank of China Tower</seed>
        <seed>Chrysler Building</seed>                <seed>Central Plaza</seed>
        <seed>John Hancock Center</seed>              <seed>CITIC Plaza</seed>
        <seed>Petronas Towers</seed>                  <seed>Shun Hing Square</seed>
        <seed>AT&amp;amp;T Corporate Center</seed>     <seed>Jin Mao Building</seed>
        <seed>Emirates Office Tower</seed>            <seed>Baiyoke Tower II</seed>
        <seed>First Canadian Place</seed>
</entity>
<entity type="sport_place" classwords="stadium;arena" estimatedpopulation="10E1">
        <seed>Wrigley Field</seed>                    <seed>Busch Stadium</seed>
        <seed>Yankee stadium</seed>                   <seed>Allianz Arena</seed>
        <seed>Cameron Indoor Stadium</seed>           <seed>Tiger Stadium</seed>
        <seed>Minute Maid Park</seed>                 <seed>Bank One Ballpark</seed>
        <seed>Comiskey Park</seed>                    <seed>Crosley Field</seed>
        <seed>Candlestick Park</seed>                 <seed>Rogers Centre</seed>
        <seed>Fenway Park</seed>                      <seed>Qualcomm Stadium</seed>
        <seed>Griffith Stadium</seed>
</entity>
<entity type="school" classwords="school;college;university" estimatedpopulation="10E1">
        <seed>University of California</seed>         <seed>University of Ottawa</seed>
        <seed>Harvard university</seed>               <seed>Acadia University</seed>
        <seed>University of Otago</seed>              <seed>University of Michigan</seed>
        <seed>University of Canterbury</seed>         <seed>Northwestern University</seed>
        <seed>University of Toronto</seed>            <seed>University of Minnesota</seed>
        <seed>La Trobe University</seed>              <seed>University of Utah</seed>
        <seed>University of Cambridge</seed>          <seed>University of Connecticut</seed>
        <seed>Johns Hopkins University</seed>
</entity>
<entity type="museum" classwords="museum" estimatedpopulation="10E1">
        <seed>Metropolitan Museum of Art</seed>       <seed>Museum of Natural History</seed>
        <seed>Louvre</seed>                           <seed>Guggenheim museum</seed>
        <seed>British Museum</seed>                   <seed>National Gallery of Art</seed>
        <seed>Art Institute of Chicago</seed>         <seed>Museum of Modern Art</seed>
        <seed>Whitney Museum of American Art</seed>   <seed>National Gallery</seed>
        <seed>Tate Gallery</seed>                     <seed>Philadelphia Museum of Art</seed>
        <seed>National Palace Museum</seed>           <seed>Museo del Prado</seed>
        <seed>High Museum of Art</seed>
</entity>
<entity type="airport" classwords="airport" estimatedpopulation="10E1">
        <seed>Croydon Airport</seed>                  <seed>Heathrow airport</seed>
        <seed>Linate Airport</seed>                   <seed>Zurich International Airport</seed>
        <seed>Auckland International Airport</seed>   <seed>Frankfurt International Airport</seed>
        <seed>Taliedo Airport</seed>                  <seed>Philadelphia International Airport</seed>
```

```
        <seed>Indira Gandhi International Airport</seed>   <seed>O'Hare International Airport</seed>
        <seed>Los Angeles International Airport</seed>     <seed>Liverpool John Lennon Airport</seed>
        <seed>John F. Kennedy International Airport</seed> <seed>Gardermoen Airport</seed>
        <seed>Stewart International Airport</seed>
</entity>
<entity type="port" classwords="port" estimatedpopulation="10E1">
        <seed>Port of New York</seed>                      <seed>Sydney Harbour</seed>
        <seed>Port of Antwerp</seed>                       <seed>Port of Duluth</seed>
        <seed>Port of Hong Kong</seed>                     <seed>Nhava Sheva</seed>
        <seed>Port of Montreal</seed>                      <seed>Chennai Port</seed>
        <seed>Port of Oakland</seed>                       <seed>Port of Vancouver</seed>
        <seed>Port of Shanghai</seed>                      <seed>Port of Rotterdam</seed>
        <seed>Port Klang</seed>                            <seed>Port of Los Angeles</seed>
        <seed>Port Miou</seed>
</entity>
<entity type="library" classwords="library" estimatedpopulation="10E1">
        <seed>Library of Congress</seed>                   <seed>National Library of Education</seed>
        <seed>Malmo City Library</seed>                    <seed>Geisel Library</seed>
        <seed>British Library</seed>                       <seed>Bodleian Library</seed>
        <seed>Library of Alexandria</seed>                 <seed>Library of Gundishapur</seed>
        <seed>Francis Trigge Chained Library</seed>        <seed>Library and Archives Canada</seed>
        <seed>Library of Alencon</seed>                    <seed>Bibliotheque Nationale de France</seed>
        <seed>Vatican Library</seed>                       <seed>Mitchell Library</seed>
        <seed>Harold B. Lee Library</seed>
</entity>
<entity type="hotel" classwords="hotel" estimatedpopulation="10E1">
        <seed>Waldorf Astoria</seed>                       <seed>hotel Sacher</seed>
        <seed>Grand Hotel Europe</seed>                    <seed>Ritz Hotel</seed>
        <seed>Beverly Hills Hotel</seed>                   <seed>Cecilienhof</seed>
        <seed>Raffles Hotel</seed>                         <seed>Hotel Chelsea</seed>
        <seed>Chateau Marmont</seed>                       <seed>Hotel Hermitage</seed>
        <seed>Palazzo Versace</seed>                       <seed>Hotel George V</seed>
        <seed>Hotel Bel-Air</seed>                         <seed>Grand Hotel Europe</seed>
</entity>
<entity type="hospital" classwords="hospital" estimatedpopulation="10E1">
        <seed>Charite</seed>                               <seed>Guy's Hospital</seed>
        <seed>Allgemeines Krankenhaus</seed>               <seed>hotel-Dieu</seed>
        <seed>Hospicio Cabanas</seed>                      <seed>Pennsylvania General Hospital</seed>
        <seed>Gillette Children's Specialty Healthcare</seed><seed>Turriff Cottage Hospital</seed>
        <seed>Holy Cross Hospital</seed>                   <seed>Shriners Hospital-Canada</seed>
        <seed>Fairview Hospital</seed>                     <seed>Bethlem Hospital</seed>
        <seed>Easton Hospital</seed>                       <seed>Victoria General Hospital</seed>
        <seed>Christ Hospital</seed>
</entity></entity>
<entity type="line">
<entity type="road" classwords="road" estimatedpopulation="10E2">
        <seed>Garden State Parkway</seed>                  <seed>Queen Elizabeth way</seed>
        <seed>European route</seed>                        <seed>Alaska highway</seed>
        <seed>Trans-Canada highway</seed>                  <seed>Jefferson Highway</seed>
        <seed>Lincoln Highway</seed>                       <seed>New Jersey Turnpike</seed>
        <seed>Great River Road</seed>                      <seed>Coquihalla Highway</seed>
        <seed>400-Series Highways</seed>                   <seed>Boston Post Road</seed>
        <seed>Yellowhead Highway</seed>                    <seed>Trans-Canada Highway</seed>
        <seed>Pan-American Highway</seed>
</entity>
<entity type="bridge" classwords="bridge" estimatedpopulation="10E1">
```

```
        <seed>Brooklyn bridge</seed>                    <seed>Golden Gate bridge</seed>
        <seed>Tower Bridge</seed>                       <seed>Confederation bridge</seed>
        <seed>Millau viaduct</seed>                     <seed>Sydney Harbour Bridge</seed>
        <seed>Mackinac Bridge</seed>                    <seed>Tacoma Narrows Bridge</seed>
        <seed>Lake Pontchartrain Causeway</seed>        <seed>Trajan's bridge</seed>
        <seed>Vasco da Gama Bridge</seed>               <seed>Tsing Ma Bridge</seed>
        <seed>Sundial Bridge</seed>                     <seed>Akashi-Kaikyo Bridge</seed>
        <seed>Victoria Falls Bridge</seed>
</entity>
<entity type="station" classwords="station" estimatedpopulation="10E1">
        <seed>London Victoria Station</seed>            <seed>Sydney Bus Depot</seed>
        <seed>Jerusalem Central Bus Station</seed>      <seed>South Station</seed>
        <seed>The Union Station GO Bus Terminal</seed>  <seed>Kyoto Station</seed>
        <seed>Shinjuku Station</seed>                   <seed>Ikebukuro Station</seed>
        <seed>Nagoya Station</seed>                     <seed>Grand Central Terminal</seed>
        <seed>Toronto Bus Terminal</seed>               <seed>Leningradsky Rail Terminal</seed>
        <seed>Lewes railway station</seed>              <seed>Central Station</seed>
        <seed>Union Station</seed>
</entity>
<entity type="railroad" classwords="railroad" estimatedpopulation="10E1">
        <seed>Amtrak</seed>                             <seed>Canadian National</seed>
        <seed>Canadian Pacific Railway</seed>           <seed>Metro-North Railroad</seed>
        <seed>Ferromex</seed>                           <seed>Great Western Railway</seed>
        <seed>London and North Eastern Railway</seed>   <seed>Southern Railway</seed>
        <seed>London, Midland and Scottish Railway</seed><seed>SNCF</seed>
        <seed>Okinawa Monorail</seed>                   <seed>Tsukuba Express</seed>
        <seed>Toden Arakawa Line</seed>                 <seed>Kowloon-Canton Railway</seed>
        <seed>Qinghai-Tibet Railway</seed>
</entity></entity>
<entity type="park" classwords="park" estimatedpopulation="10E1">
        <seed>Elk island National Park</seed>           <seed>Glacier National Park</seed>
        <seed>Jasper national park</seed>               <seed>Banff national park</seed>
        <seed>Forillon national park</seed>             <seed>Wood Buffalo National Park</seed>
        <seed>Yoho National Park</seed>                 <seed>Waterton Lakes National Park</seed>
        <seed>Mount Revelstoke National Park</seed>     <seed>Kootenay National Park</seed>
        <seed>Cape Breton Highlands National Park</seed> <seed>Kouchibouguac National Park</seed>
        <seed>Pukaskwa National Park</seed>             <seed>Fundy National Park</seed>
        <seed>Prince Edward Island National Park</seed>
</entity>
<entity type="amusement_park" classwords="amusement park" estimatedpopulation="10E1">
        <seed>Adventure Island</seed>                   <seed>Busch gardens</seed>
        <seed>Canada's Wonderland</seed>                <seed>Marineland</seed>
        <seed>La Ronde</seed>                           <seed>Luna Park</seed>
        <seed>Disneyland</seed>                         <seed>Epcot Center</seed>
        <seed>The Magic Kingdom</seed>                  <seed>Bakken</seed>
        <seed>Prater</seed>                             <seed>Magic Mountain</seed>
        <seed>Legoland</seed>                           <seed>Alton Towers</seed>
        <seed>Blackpool Pleasure Beach</seed>           <seed>Sea World</seed>
        <seed>Playland</seed>                           <seed>Dreamworld</seed>
        <seed>Phantasialand</seed>                      <seed>Pleasure Island</seed>
</entity>
<entity type="monument" classwords="monument" estimatedpopulation="10E1">
        <seed>Statue of Liberty</seed>                  <seed>Eiffel tower</seed>
        <seed>Great Sphinx of Giza</seed>               <seed>Taj Mahal</seed>
        <seed>Great Wall</seed>                         <seed>The Great Wall</seed>
        <seed>Colosseum</seed>                          <seed>Big Ben</seed>
```

```
        <seed>Dome of the Rock</seed>                    <seed>Mount Rushmore</seed>
        <seed>Tower of Pisa</seed>                       <seed>Wailing Wall</seed>
        <seed>Parthenon</seed>                           <seed>Ming Tombs</seed>
        <seed>Stonehedge</seed>
</entity></entity>
<entity type="product"><entity type="vehicules">
<entity type="car" classwords="car" estimatedpopulation="10E2">
        <seed>Volkswagen Golf</seed>                     <seed>Honda Civic</seed>
        <seed>GMC Yukon</seed>                           <seed>Toyota Prius</seed>
        <seed>Ford Focus</seed>                          <seed>Volkswagen Jetta</seed>
        <seed>Toyota Corolla</seed>                      <seed>Subaru Impreza</seed>
        <seed>Hyundai Elantra</seed>                     <seed>Nissan Sentra</seed>
        <seed>Honda Accord</seed>                        <seed>Hyundai Accent</seed>
        <seed>Fiat Stilo</seed>                          <seed>Chrysler PT Cruiser</seed>
        <seed>Toyota Camry</seed>
</entity>
<entity type="ship" classwords="ship" estimatedpopulation="10E1">
        <seed>Queen Mary 2</seed>                        <seed>Titanic</seed>
        <seed>Wilhelm Gustloff</seed>                    <seed>Great Eastern</seed>
        <seed>Calypso</seed>                             <seed>Queen Mary</seed>
        <seed>Lusitania</seed>                           <seed>Olympic</seed>
        <seed>Exxon Valdez</seed>                        <seed>Bismarck</seed>
        <seed>Lancastria</seed>                          <seed>Queen Elizabeth</seed>
        <seed>Empress of Ireland</seed>                  <seed>Kon-Tiki</seed>
        <seed>Freedom of the Seas</seed>
</entity>
<entity type="train" classwords="train" estimatedpopulation="10E1">
        <seed>Orient Express</seed>                      <seed>Royal Canadian Pacific</seed>
        <seed>Rovos Rail</seed>                          <seed>Palace on Wheels</seed>
        <seed>Trans-Siberian Express</seed>              <seed>Bullet Train</seed>
        <seed>Eurostar</seed>                            <seed>Bulgaria Express</seed>
        <seed>Glacier Express</seed>                     <seed>Golden Arrow</seed>
        <seed>Bluenose</seed>                            <seed>Atlantic Express</seed>
        <seed>20th Century Limited</seed>                <seed>Auto Train</seed>
        <seed>Bluebonnet</seed>
</entity>
<entity type="aircraft" classwords="aircraft" estimatedpopulation="10E1">
        <seed>Constellation</seed>                       <seed>Bell X1</seed>
        <seed>Northrop x15</seed>                        <seed>U2 spy plane</seed>
        <seed>Boeing 747</seed>                          <seed>Lancair</seed>
        <seed>Eurofighter</seed>                         <seed>Voyager</seed>
        <seed>Nemesis</seed>                             <seed>Grippen</seed>
        <seed>Starduster too</seed>                      <seed>Pushy Galore</seed>
        <seed>Piagio Avanti</seed>                       <seed>X15</seed>
        <seed>Boomerang</seed>
</entity>
<entity type="spaceship" classwords="spaceship;spacecraft" estimatedpopulation="10E1">
        <seed>Columbia</seed>                            <seed>Discovery</seed>
        <seed>Endeavour</seed>                           <seed>Apollo 11</seed>
        <seed>International Space Station</seed>         <seed>Atlantis</seed>
        <seed>Challenger</seed>                          <seed>Spacelab</seed>
        <seed>Enterprise</seed>                          <seed>Cassini-Huygens</seed>
        <seed>Skylab</seed>                              <seed>SpaceShipOne</seed>
        <seed>Soyuz</seed>                               <seed>Mir</seed>
        <seed>Shenzhou Spacecraft</seed>
</entity>
```

```
</entity>
<entity type="art">
<entity type="opera_musical" classwords="opera;musical" estimatedpopulation="10E1">
        <seed>The Fantasticks</seed>                    <seed>Les Miserables</seed>
        <seed>Miss Saigon</seed>                        <seed>Chicago</seed>
        <seed>The Lion King</seed>                      <seed>Cabaret</seed>
        <seed>My Fair Lady</seed>                       <seed>Cats</seed>
        <seed>Evita</seed>                              <seed>Annie</seed>
        <seed>Guys and Dolls</seed>                     <seed>Beauty and the Beast</seed>
        <seed>Jesus Christ Superstar</seed>             <seed>Godspell</seed>
        <seed>Rent</seed>
</entity>
<entity type="song" classwords="song;composition" estimatedpopulation="10E2">
        <seed>Aqualung</seed>                           <seed>Bohemian Rhapsody</seed>
        <seed>Hey Jude</seed>                           <seed>Hotel California</seed>
        <seed>Johnny B. Goode</seed>                    <seed>Good Vibrations</seed>
        <seed>Imagine</seed>                            <seed>Heartbreak Hotel</seed>
        <seed>Jailhouse Rock</seed>                     <seed>Every Breath You Take</seed>
        <seed>Brown Eyed Girl</seed>                    <seed>Yesterday</seed>
        <seed>Bridge Over Troubled Water</seed>         <seed>Light My Fire</seed>
        <seed>California Girls</seed>
</entity>
<entity type="painting" classwords="painting" estimatedpopulation="10E1">
        <seed>Mona Lisa</seed>                          <seed>Les Demoiselles</seed>
        <seed>Still Life</seed>                         <seed>Scream</seed>
        <seed>The Kiss</seed>                           <seed>Birds On A Beach</seed>
        <seed>Garden Still Life</seed>                  <seed>Hommage a Brohmann</seed>
        <seed>Garden of Earthly Delights</seed>         <seed>Pear</seed>
        <seed>Vase of Flowers</seed>                    <seed>Self-Portrait</seed>
        <seed>Still Life: Vase with Twelve Sunflowers</seed><seed>The Matador</seed>
        <seed>La Joie de Vivre</seed>
</entity>
<entity type="sculpture" classwords="sculpture" estimatedpopulation="10E1">
        <seed>Charging Bull</seed>                      <seed>Fountain of Neptune</seed>
        <seed>The Thinker</seed>                        <seed>David</seed>
        <seed>La Joute</seed>                           <seed>Venus of Melos</seed>
        <seed>Colossus of Rhodes</seed>                 <seed>Venus of Lespugue</seed>
        <seed>Angel of the North</seed>                 <seed>Lady of Auxerre</seed>
        <seed>The Great Bear</seed>                     <seed>Kleobis and Biton</seed>
        <seed>Spiral Jetty</seed>                       <seed>Goddess of Democracy</seed>
        <seed>Falling Autumn Leaves</seed>
</entity></entity>
<entity type="media">
<entity type="broadcast" classwords="broadcast;tv" estimatedpopulation="10E2">
        <seed>60 Minutes</seed>                         <seed>Jeopardy</seed>
        <seed>Seinfeld</seed>                           <seed>Pokemon</seed>
        <seed>Access Hollywood</seed>                   <seed>Good Morning, America</seed>
        <seed>Today</seed>                              <seed>The Outer Limits</seed>
        <seed>The Practice</seed>                       <seed>The Ed Sullivan Show</seed>
        <seed>Oprah</seed>                              <seed>Tonight Show</seed>
        <seed>C.S.I.</seed>                             <seed>Wheel of Fortune</seed>
        <seed>The Early Show</seed>
</entity>
<entity type="movie" classwords="movie" estimatedpopulation="10E2">
        <seed>The Godfather</seed>                      <seed>The Shawshank Redemption</seed>
        <seed>Star Wars</seed>                          <seed>Citizen Kane</seed>
```

```
        <seed>Monty Python and the Holy Grail</seed>    <seed>Casablanca</seed>
        <seed>Schindler's List</seed>                   <seed>Pulp Fiction</seed>
        <seed>2001: A Space Odyssey</seed>              <seed>The Wizard of Oz</seed>
        <seed>Blade Runner</seed>                       <seed>Raiders of the Lost Ark</seed>
        <seed>Goodfellas</seed>                         <seed>Chinatown</seed>
        <seed>One Flew Over the Cuckoo's Nest</seed>
</entity>
<entity type="book" classwords="book;novel" estimatedpopulation="10E2">
        <seed>Harry Potter and the Half-blood Prince</seed><seed>The Da Vinci Code</seed>
        <seed>The Historian</seed>                      <seed>Fast food nation</seed>
        <seed>The world is flat</seed>                  <seed>Master of the Game</seed>
        <seed>The Age of Reason</seed>                  <seed>Animal Farm</seed>
        <seed>Bible</seed>                              <seed>The Blue Lotus</seed>
        <seed>Mein Kampf</seed>                         <seed>1984</seed>
        <seed>Ulysses</seed>                            <seed>Brave New World</seed>
        <seed>The Satanic Verses</seed>
</entity>
<entity type="newspaper" classwords="newspaper" estimatedpopulation="10E1">
        <seed>New York Times</seed>                     <seed>Chicago Tribune</seed>
        <seed>Le Monde</seed>                           <seed>Washington Post</seed>
        <seed>The Globe and Mail</seed>                 <seed>Jerusalem Post</seed>
        <seed>China Daily</seed>                        <seed>The Nation</seed>
        <seed>National Post</seed>                      <seed>USA Today</seed>
        <seed>The Guardian</seed>                       <seed>The Boston Globe</seed>
        <seed>San Jose Mercury News</seed>              <seed>International Herald Tribune</seed>
        <seed>Philadelphia Inquirer</seed>
</entity>
<entity type="magazine" classwords="magazine" estimatedpopulation="10E1">
        <seed>Times Magazine</seed>                     <seed>Forbes</seed>
        <seed>Vogue</seed>                              <seed>Metropolitan</seed>
        <seed>Wired</seed>                              <seed>Current History</seed>
        <seed>Newsweek</seed>                           <seed>Outside</seed>
        <seed>Guitar Player</seed>                      <seed>PC Magazine</seed>
        <seed>Popular Science</seed>                    <seed>Popular Mechanics</seed>
        <seed>American Heritage</seed>                  <seed>Scientific American</seed>
        <seed>Rolling Stone</seed>
</entity></entity>
<entity type="weapon" classwords="weapon" estimatedpopulation="10E2">
        <seed>knife</seed>                              <seed>bayonet</seed>
        <seed>handgun</seed>                            <seed>sniper rifle</seed>
        <seed>Shotgun</seed>                            <seed>Patriot missile</seed>
        <seed>44 Magnums</seed>                         <seed>spear</seed>
        <seed>cannon</seed>                             <seed>Maxim gun</seed>
        <seed>MP5K</seed>                               <seed>Colt .45 Automatic</seed>
        <seed>Thompson submachine gun</seed>            <seed>Uzi</seed>
        <seed>Molotov cocktail</seed>
</entity>
<entity type="food_brand" classwords="food" estimatedpopulation="10E2">
        <seed>Gatorade</seed>                           <seed>dr. Pepper</seed>
        <seed>Cheese Whiz</seed>                        <seed>Oreo</seed>
        <seed>Rice Krispies</seed>                      <seed>Dannon</seed>
        <seed>Equal</seed>                              <seed>Yoplait</seed>
        <seed>Uncle Ben's</seed>                        <seed>Butterball</seed>
        <seed>Cheerios</seed>                           <seed>Crisco</seed>
        <seed>Pringles</seed>                           <seed>McCormick</seed>
        <seed>Carnation Milk</seed>
```

```
</entity>
<entity type="food" classwords="food" estimatedpopulation="10E1">
        <seed>cereal</seed>                          <seed>milk</seed>
        <seed>vegetable</seed>                       <seed>meat</seed>
        <seed>rice</seed>                            <seed>donut</seed>
        <seed>tofu</seed>                            <seed>cheese</seed>
        <seed>mushrooms</seed>                       <seed>couscous</seed>
        <seed>pasta</seed>                           <seed>olive</seed>
        <seed>muffin</seed>                          <seed>soy milk</seed>
        <seed>bread</seed>
</entity>
<entity type="clothes" classwords="clothes" estimatedpopulation="10E2">
        <seed>Gucci</seed>                           <seed>Armani</seed>
        <seed>Ralph Lauren</seed>                    <seed>Tommy Hilfiger</seed>
        <seed>Fruit of the Loom</seed>               <seed>Calvin Klein</seed>
        <seed>Prada</seed>                           <seed>Versace</seed>
        <seed>Dolce &amp;amp; Gabbana</seed>         <seed>Hugo Boss</seed>
        <seed>Diesel</seed>                          <seed>Chanel</seed>
        <seed>Adidas</seed>                          <seed>Burberry</seed>
        <seed>Nike</seed>
</entity>
<entity type="drug" classwords="drug;medication" estimatedpopulation="10E2">
        <seed>Advil</seed>                           <seed>Tylenol</seed>
        <seed>Gaviscon</seed>                        <seed>Gravol</seed>
        <seed>Sudafed</seed>                         <seed>Benadryl</seed>
        <seed>Claritin</seed>                        <seed>Metamucil</seed>
        <seed>Pepcid</seed>                          <seed>Imodium</seed>
        <seed>Nicoderm</seed>                        <seed>aspirin</seed>
        <seed>motrin</seed>                          <seed>Maalox</seed>
        <seed>Triaminic</seed>
</entity></entity>
<entity type="event">
<entity type="game" classwords="game" estimatedpopulation="10E1">
        <seed>Olympic games</seed>                   <seed>Wimbledon</seed>
        <seed>tour de France</seed>                  <seed>US Masters</seed>
        <seed>Superbowl</seed>                       <seed>World Cup</seed>
        <seed>Commonwealth Games</seed>              <seed>PGA Championship</seed>
        <seed>Rugby World Cup</seed>                 <seed>British Open</seed>
        <seed>Roland Garros</seed>                   <seed>Stanlet Cup Finals</seed>
        <seed>Evergreen Tournament</seed>            <seed>FA Challenge Cup</seed>
        <seed>World Poker Tour</seed>
</entity>
<entity type="holiday" classwords="holiday" estimatedpopulation="20E0">
        <seed>christmas</seed>                       <seed>easter</seed>
        <seed>Halloween</seed>                       <seed>thanksgiving</seed>
        <seed>mother's day</seed>                    <seed>Valentine's Day</seed>
        <seed>St. Patrick's Day</seed>               <seed>4th of July</seed>
        <seed>Father's Day</seed>                    <seed>New Year</seed>
        <seed>Labor Day</seed>                       <seed>Memorial Day</seed>
        <seed>Chinese New Year</seed>                <seed>Ramadan</seed>
        <seed>Earth Day</seed>
</entity>
<entity type="war" classwords="war" estimatedpopulation="10E1">
        <seed>Vietnam war</seed>                     <seed>World War II</seed>
        <seed>World War I</seed>                     <seed>Cold war</seed>
        <seed>The Gulf War</seed>                    <seed>Korean War</seed>
```

```
        <seed>Iraq War</seed>                          <seed>War of 1812</seed>
        <seed>American Civil War</seed>                <seed>Russian Civil War</seed>
        <seed>Croatian War of Independence</seed>      <seed>Battle of Carthage</seed>
        <seed>Second Congo War</seed>                  <seed>Romanian Revolution</seed>
        <seed>Iran-Iraq War</seed>
</entity>
<entity type="hurricane" classwords="hurricane" estimatedpopulation="10E1">
        <seed>Hurricane Katrina</seed>                 <seed>Hurricane Andrew</seed>
        <seed>Hurricane Wilma</seed>                   <seed>hurricane Charley</seed>
        <seed>hurricane Ivan</seed>                    <seed>Tropical Storm Arlene</seed>
        <seed>Tropical Storm Bonnie</seed>             <seed>Tropical Storm Alberto</seed>
        <seed>Hurricane Rita</seed>                    <seed>Hurricane Frances</seed>
        <seed>Hurricane Jeanne</seed>                  <seed>Hurricane Hugo</seed>
        <seed>Hurricane Dennis</seed>                  <seed>Hurricane Isabel</seed>
        <seed>Hurricane Lili</seed>
</entity>
<entity type="crime" classwords="crime" estimatedpopulation="10E1">
        <seed>Oklahoma City Bombing</seed>             <seed>September 11, 2001 attacks</seed>
        <seed>Moscow Theatre Siege</seed>              <seed>Oklahoma City bombing</seed>
        <seed>Wall Street bombing</seed>               <seed>Beslan School Siege</seed>
        <seed>King David Hotel bombing</seed>          <seed>Circus arson</seed>
        <seed>Dupont Plaza Hotel arson</seed>          <seed>Bath School Disaster</seed>
        <seed>Erfurt massacre</seed>                   <seed>Postal shooting</seed>
        <seed>Jack the Ripper</seed>                   <seed>Boston Strangler</seed>
        <seed>Scarborough Rapist</seed>
</entity>
<entity type="conference" classwords="conference" estimatedpopulation="10E1">
        <seed>Halifax Summit</seed>                    <seed>APEC</seed>
        <seed>Tokyo Summit</seed>                      <seed>Kyoto conference</seed>
        <seed>Rambouillet Summit</seed>                <seed>Moscow Conference</seed>
        <seed>International Meridian Conference</seed> <seed>Quebec Conference</seed>
        <seed>Geneva Conference</seed>                 <seed>Congress of Berlin</seed>
        <seed>EVA Conferences</seed>                   <seed>World Food Conference</seed>
        <seed>World Summit for Children</seed>         <seed>Pan-American Conference</seed>
        <seed>Conference of Lausanne</seed>
</entity></entity>
<entity type="natural_object"><entity type="living_thing"><entity type="animal"><entity
type="invertebrate">
<entity type="insect" classwords="insect" estimatedpopulation="10E1">
        <seed>Ant</seed>                               <seed>beetle</seed>
        <seed>fly</seed>                               <seed>walking stick</seed>
        <seed>cockroach</seed>                         <seed>bee</seed>
        <seed>mosquito</seed>                          <seed>dragonfly</seed>
        <seed>cricket</seed>                           <seed>butterfly</seed>
        <seed>firefly</seed>                           <seed>locust</seed>
        <seed>spider</seed>                            <seed>centipede</seed>
        <seed>termite</seed>
</entity>
<entity type="sea_animal" classwords="sea animal" estimatedpopulation="10E1">
        <seed>crab</seed>                              <seed>starfish</seed>
        <seed>coral</seed>                             <seed>anemones</seed>
        <seed>sea cucumber</seed>                      <seed>Basket Star</seed>
        <seed>Sponges</seed>                           <seed>Moon Jellyfish</seed>
        <seed>Mantis Shrimp</seed>                     <seed>Squid</seed>
        <seed>Hermit Crab</seed>                       <seed>Jellyfish</seed>
        <seed>Seastar</seed>                           <seed>Sea Urchin</seed>
```

```
        <seed>Octopus</seed>
</entity></entity>
<entity type="vertebrate">
<entity type="fish" classwords="fish" estimatedpopulation="10E1">
        <seed>shark</seed>                          <seed>tuna</seed>
        <seed>whale</seed>                          <seed>trout</seed>
        <seed>salmon</seed>                         <seed>Swordfish</seed>
        <seed>Mako Shark</seed>                     <seed>Striped Bass</seed>
        <seed>bass</seed>                           <seed>perch</seed>
        <seed>Carp</seed>                           <seed>pike</seed>
        <seed>Perch</seed>                          <seed>Cod</seed>
        <seed>Sole</seed>
</entity>
<entity type="reptile" classwords="reptile" estimatedpopulation="10E1">
        <seed>Tortoise</seed>                       <seed>alligator</seed>
        <seed>Iguana</seed>                         <seed>python</seed>
        <seed>Chameleon</seed>                      <seed>Gecko</seed>
        <seed>Bearded Dragon</seed>                 <seed>Rattlesnake</seed>
        <seed>Anaconda</seed>                       <seed>Turtle</seed>
        <seed>Komodo Dragon</seed>                  <seed>Lizard</seed>
        <seed>Snake</seed>                          <seed>Frog</seed>
        <seed>Copperhead</seed>
</entity>
<entity type="bird" classwords="bird" estimatedpopulation="10E1">
        <seed>snow goose</seed>                     <seed>cormorant</seed>
        <seed>Heron</seed>                          <seed>Eagle</seed>
        <seed>Merlin</seed>                         <seed>Dove</seed>
        <seed>Owl</seed>                            <seed>Swan</seed>
        <seed>Albatross</seed>                      <seed>Blue Jay</seed>
        <seed>Falcon</seed>                         <seed>Peacock</seed>
        <seed>cukoo</seed>                          <seed>American Kestrel</seed>
        <seed>Mocking Bird</seed>
</entity>
<entity type="mammal" classwords="mammal" estimatedpopulation="10E1">
        <seed>bear</seed>                           <seed>cow</seed>
        <seed>horse</seed>                          <seed>human</seed>
        <seed>lynx</seed>                           <seed>cat</seed>
        <seed>whale</seed>                          <seed>lamb</seed>
        <seed>Mouse</seed>                          <seed>Dog</seed>
        <seed>Mule</seed>                           <seed>Deer</seed>
        <seed>Kodiak Bear</seed>                    <seed>Asian Elephant</seed>
        <seed>Orangutan</seed>
</entity></entity></entity>
<entity type="vegetal" classwords="vegetal;plant" estimatedpopulation="10E1">
        <seed>crop</seed>                           <seed>herb</seed>
        <seed>flower</seed>                         <seed>grapevine</seed>
        <seed>tree</seed>                           <seed>tree fern</seed>
        <seed>Green algae</seed>                    <seed>lichens</seed>
        <seed>Bee orchid</seed>                     <seed>oak tree</seed>
        <seed>Giant Sequoia</seed>                  <seed>white pine</seed>
        <seed>Cypress</seed>                        <seed>maple tree</seed>
        <seed>carnivorous plant</seed>
</entity></entity>
<entity type="mineral" classwords="mineral;chemical element" estimatedpopulation="50E0">
        <seed>hydrogen</seed>                       <seed>water</seed>
        <seed>iron</seed>                           <seed>mercury</seed>
```

```
        <seed>copper</seed>                                    <seed>Lead</seed>
        <seed>Silver</seed>                                    <seed>Benzene</seed>
        <seed>uranium</seed>                                   <seed>Calcium</seed>
        <seed>nickel</seed>                                    <seed>Chlorine</seed>
        <seed>Ozone</seed>                                     <seed>Carbon</seed>
        <seed>Oxygen</seed>
</entity></entity>
<entity type="unit">
<entity type="measure" classwords="measure" estimatedpopulation="10E1">
        <seed>liter</seed>                                     <seed>kilogram</seed>
        <seed>ohm</seed>                                       <seed>Decibel</seed>
        <seed>Carat</seed>                                     <seed>gram</seed>
        <seed>rad</seed>                                       <seed>hectare</seed>
        <seed>fathom</seed>                                    <seed>inch</seed>
        <seed>acre</seed>                                      <seed>ton</seed>
        <seed>Pascal</seed>                                    <seed>Volt</seed>
        <seed>Newton</seed>
</entity>
<entity type="currency" classwords="currency" estimatedpopulation="10E1">
        <seed>Yen</seed>                                       <seed>Euro</seed>
        <seed>dollar</seed>                                    <seed>Pound</seed>
        <seed>franc</seed>                                     <seed>Ruble</seed>
        <seed>yuan</seed>                                      <seed>Krone</seed>
        <seed>Peso</seed>                                      <seed>Ringgit</seed>
        <seed>Real</seed>                                      <seed>Rupiah</seed>
        <seed>Rand</seed>                                      <seed>Baht</seed>
        <seed>Krona</seed>
</entity>
<entity type="month" classwords="month" estimatedpopulation="24">
        <seed>January</seed>                                   <seed>February</seed>
        <seed>March</seed>                                     <seed>April</seed>
        <seed>May</seed>                                       <seed>June</seed>
        <seed>July</seed>                                      <seed>August</seed>
        <seed>September</seed>                                 <seed>November</seed>
        <seed>December</seed>                                  <seed>jan</seed>
        <seed>feb</seed>                                       <seed>mar</seed>
        <seed>apr</seed>
</entity>
<entity type="weekday" classwords="weekday" estimatedpopulation="14">
        <seed>Monday</seed>                                    <seed>Tuesday</seed>
        <seed>Wednesday</seed>                                 <seed>Thursday</seed>
        <seed>Friday</seed>                                    <seed>Saturday</seed>
        <seed>Sunday</seed>                                    <seed>mon</seed>
        <seed>tue</seed>                                       <seed>wed</seed>
        <seed>thu</seed>                                       <seed>fri</seed>
        <seed>sat</seed>                                       <seed>sun</seed>
</entity></entity>
<entity type="misc">
<entity type="disease" classwords="disease" estimatedpopulation="10E2">
        <seed>myocardial infarction</seed>                     <seed>stroke</seed>
        <seed>aphasia</seed>                                   <seed>cold</seed>
        <seed>leukemia</seed>                                  <seed>Abscess</seed>
        <seed>Alzheimer Disease</seed>                         <seed>Bipolar Disorder</seed>
        <seed>Bone Neoplasms</seed>                            <seed>Encephalitis</seed>
        <seed>Epilepsy</seed>                                  <seed>Facial Paralysis</seed>
        <seed>Chronic Fatigue Syndrome</seed>                  <seed>Fever</seed>
```

```
        <seed>Laryngitis</seed>
</entity>
<entity type="god" classwords="god;deity" estimatedpopulation="10E0">
        <seed>Allah</seed>                                  <seed>Zeus</seed>
        <seed>Venus</seed>                                  <seed>Jesus</seed>
        <seed>Ra</seed>                                     <seed>Ares</seed>
        <seed>Aphrodite</seed>                              <seed>Odin</seed>
        <seed>Horus</seed>                                  <seed>Nephthys</seed>
        <seed>Minos</seed>                                  <seed>Mars</seed>
        <seed>Osiris</seed>                                 <seed>Valkyries</seed>
        <seed>Diana</seed>
</entity>
<entity type="religion" classwords="religion" estimatedpopulation="10E1">
        <seed>buddhism</seed>                               <seed>Islam</seed>
        <seed>Catholic</seed>                               <seed>Atheism</seed>
        <seed>christianity</seed>                           <seed>Judaism</seed>
        <seed>Hinduism</seed>                               <seed>Sikhism</seed>
        <seed>Jainism</seed>                                <seed>Shinto</seed>
        <seed>Taoism</seed>                                 <seed>Protestant</seed>
        <seed>Zoroastrianism</seed>                         <seed>Baha'i</seed>
        <seed>zen</seed>
</entity>
<entity type="color" classwords="color" estimatedpopulation="10E0">
        <seed>white</seed>                                  <seed>red</seed>
        <seed>blue</seed>                                   <seed>yellow</seed>
        <seed>purple</seed>                                 <seed>Green</seed>
        <seed>Black</seed>                                  <seed>Orange</seed>
        <seed>Pink</seed>                                   <seed>Brown</seed>
        <seed>Gray</seed>                                   <seed>orange</seed>
        <seed>peach</seed>                                  <seed>Lavender</seed>
        <seed>cyan</seed>
</entity>
<entity type="language" classwords="language" estimatedpopulation="10E1">
        <seed>French</seed>                                 <seed>English</seed>
        <seed>Dutch</seed>                                  <seed>Spanish</seed>
        <seed>Japanese</seed>                               <seed>German</seed>
        <seed>Italian</seed>                                <seed>Russian</seed>
        <seed>Chinese</seed>                                <seed>Greek</seed>
        <seed>Korean</seed>                                 <seed>Arabic</seed>
        <seed>Finnish</seed>                                <seed>Czech</seed>
        <seed>Turkish</seed>
</entity>
<entity type="award" classwords="award;prize" estimatedpopulation="10E1">
        <seed>Nobel prize</seed>                            <seed>Academy Award</seed>
        <seed>Pulitzer prize</seed>                         <seed>Genie award</seed>
        <seed>Razzie award</seed>                           <seed>Grammy Award</seed>
        <seed>Crystal Awards</seed>                         <seed>Army Commendation Medal</seed>
        <seed>Mtv Award</seed>                              <seed>Golden Gloves</seed>
        <seed>Library of Congress Living Legend</seed>      <seed>Cross of Valour</seed>
        <seed>Cy Young Award</seed>                         <seed>Grey Cup</seed>
        <seed>Stanley Cup</seed>
</entity>
<entity type="sport" classwords="sport" estimatedpopulation="10E1">
        <seed>football</seed>                               <seed>hockey</seed>
        <seed>baseball</seed>                               <seed>racquetball</seed>
        <seed>tennis</seed>                                 <seed>Giant Slalom</seed>
```

```
        <seed>Soccer</seed>                                    <seed>Basketball</seed>
        <seed>Golf</seed>                                      <seed>Softball</seed>
        <seed>Wrestling</seed>                                 <seed>Boxing</seed>
        <seed>Volleyball</seed>                                <seed>Swimming</seed>
        <seed>Cross Country</seed>                             <seed>Track and Field</seed>
</entity>
<entity type="academic" classwords="academic" estimatedpopulation="10E1">
        <seed>Sociology</seed>                                 <seed>Physics</seed>
        <seed>Philosophy</seed>                                <seed>Medecine</seed>
        <seed>Computer Science</seed>                          <seed>Chemistry</seed>
        <seed>Psychology</seed>                                <seed>History</seed>
        <seed>Biology</seed>                                   <seed>Mathematics</seed>
        <seed>Economics</seed>                                 <seed>English</seed>
        <seed>Music</seed>                                     <seed>Education</seed>
        <seed>Anthropology</seed>
</entity>
<entity type="rule" classwords="rule;law" estimatedpopulation="10E1">
        <seed>U.S. Constitution</seed>                         <seed>Amateur Sports Act</seed>
        <seed>World Trade Accord</seed>                        <seed>Americas Free Trade Agreement</seed>
        <seed>Anti-Monopoly Law</seed>                         <seed>Lindbergh Law</seed>
        <seed>Universal Declaration of Human Rights</seed><seed>Constitutional law</seed>
        <seed>Civil Code of Quebec</seed>                      <seed>Constitution Act</seed>
        <seed>California Penal Code</seed>                     <seed>Uniform Commercial Code</seed>
        <seed>Model Penal Code</seed>                          <seed>Napoleonic Code</seed>
        <seed>Black's Law Dictionary</seed>
</entity>
<entity type="theory" classwords="theory;law" estimatedpopulation="10E1">
        <seed>Zipf's law</seed>                                <seed>Newton's Law</seed>
        <seed>Cook's theorem</seed>                            <seed>Theory of relativity</seed>
        <seed>law of large numbers</seed>                      <seed>Big Bang Theory</seed>
        <seed>Cell theory</seed>                               <seed>Decision theory</seed>
        <seed>Theory of Global Climate Change</seed>           <seed>Chaos theory</seed>
        <seed>De Morgan's law</seed>                           <seed>Euler's theorem</seed>
        <seed>Occam's Razor</seed>                             <seed>Pythagorean theorem</seed>
        <seed>Theory of Relativity</seed>
</entity></entity></entity>
</entityHierarchy>
```