RECOGNIZING NAMED ENTITIES IN BIOMEDICAL TEXTS

by

Baohua Gu
B.S., Xi'an Jiaotong University, China, 1990
M.S., National University of Singapore, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
Summer 2008

APPROVAL

Name:	Baohua Gu
Degree:	Doctor of Philosophy
Title of thesis:	Recognizing Named Entities in Biomedical Texts
Examining Committee:	Dr. Anoop Sarkar Chair
	Dr. Veronica Dahl, Senior Supervisor Professor, School of Computing Science, SFU
	Dr. Fred Popowich, Co-Senior Supervisor Professor, School of Computing Science, SFU
	Dr. Paul McFetridge, SFU Examiner Associate Professor, Department of Linguistics, SFU
	Dr. Sabine Bergler, External Examiner Associate Professor, Department of Computer Science and Software Engineering, Concordia University
Date Approved:	June 12, 2008

Abstract

Named Entities (NEs) in biomedical text refer to objects that are of interest to biomedical researchers, such as proteins and genes. Accurately identifying them is important for Biomedical Natural Language Processing (BioNLP). Focusing on biomedical named entity recognition (BioNER), this thesis presents a number of novel results on the following topics of this area.

First, we study whether corpus based statistical learning methods, currently dominant in BioNER, would achieve close-to-human performance by using larger corpora for training. We find that a significantly larger corpus is required to achieve a performance significantly higher than the state-of-the-art obtained on the GENIA corpus. This finding suggests the hypothesis is not warranted.

Second, we address the issue of nested NEs and propose a level-by-level method that learns a separate NER model for each level of the nesting. We show that this method works well for both nested NEs and non-nested NEs.

Third, we propose a method that builds NEs on top of base NP chunks, and examine the associated benefits as well as problems. Our experiments show that this method, though inferior to statistical word based approaches, has the potential to outperform them, provided that domain-specific rules can be designed to determine NE boundaries based on NP chunks.

Fourth, we present a method to do BioNER in the absence of annotated corpora. It uses an NE dictionary to label sentences, and then uses these partially labeled sentences to iteratively train an SVM model in the manner of semi-supervised learning. Our experiments validate the effectiveness of the method.

Finally, we explore BioNER in Chinese text, an area that has not been studied by previous work. We train a character-based CRF model on a small set of manually annotated Chinese biomedical abstracts. We also examine the features usable for the model. Our

evaluation suggests that corpus-based statistical learning approaches hold promise for this particular task.

All the proposed methods are novel and have applicability beyond the NE types and the languages considered here, and beyond the BioNER task itself.

Keywords: Biomedical Named Entity Recognition; Support Vector Machine; Conditional Random Field; Annotated Corpus; Biomedical Text; Chinese Text; Nested Named Entity; Noun Phrase Chunking.

Subject Terms: Natural Language Processing (Computer Science); Text Processing (Computer Science); Supervised Learning (Machine Learning); Information Retrieval; Computational Linguistics; Text Mining; Information Extraction.

To all who love and inspire me!

"学无止境 *(The pursuit of knowledge has no end)"*

— A Chinese Proverb

Acknowledgments

I would like to thank my wife Sarah, for her love and support, and for bearing with me in downturns of my study. I would like to thank my parents, Zunjia Gu and Shuhui Zhang, for bringing me up and making me think that pursuing knowledge is always a good thing. I also want to thank my sister Chengfang and her husband Wei, for taking care of our mother over the years while I am abroad.

I express my deepest gratitude to the following people (I apologize for any inadvertent omissions):

Dr. Veronica Dahl, my senior supervisor, for her exceptional mentorship, patient guidance, and moral support, and for teaching me how to think in terms of logic programming.

Dr. Fred Popowich, my co-senior supervisor, for his continuous guidance and support, and for helping me enter and explore the wonderland of natural language processing.

My examiners: Dr. Sabine Bergler and Dr. Paul McFetridge for their constructive suggestions and excellent guidance.

Dr. Anoop Sarkar for many useful discussions and comments over the years.

Everybody in the "Logic and Functional Programming Lab" and the "Natural Language Lab" at SFU for their friendship and support. Particularly, Zhongmin Shi, Dong Song, Yudong Liu, Wendy Yang Wang, Gabor Melli, Chris Demwell, Kimberly Voll, Gholamreza Haffari, Maxim Roy, for many useful and challenging discussions.

All the administrative and technical staff of School of Computing Science.

I would like to thank the open source community, for good quality software tools and hope for the future of software.

I would like to thank School of Computing Science of Simon Fraser University: a great school of a great university!

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Named Entity Recognition

Named entity recognition (NER) is a subtask of information extraction (IE) and text mining (TM). Given a piece of text, the goal is to identify those phrases that correspond to the names of persons, places, organizations, or other entities of interest. In the newswire domain, an NE often refers to the name of a person, an organization, a location, or the expression of a time, a quantity, etc. In the biomedical domain, the NE often refers to the names of proteins, genes and other biomedical objects.

NER is important to IE and TM in the sense that it serves as a building-block for more advanced IE and TM tasks, e.g., relation extraction (RE), which identifies relations between NEs. This is particularly true in the biomedical domain, where millions of biomedical research papers have become electronically available, and thus opened opportunities to facilitate biomedical research by extracting or mining useful knowledge from the biomedical texts. The recognition of biomedical NEs is therefore the very fundamental task underlying further biomedical text mining.

While NER in the newswire domain (NewsNER) has been well studied since the 1990s, NER in the biomedical domain (BioNER) is still in its early stage of development. Techniques proposed for NewsNER have achieved close-to-human performance, whereas BioNER has not matured enough to support advanced IE and TM tasks. For instance, the state-of-the-art performance of BioNER typically falls in 70% to 80% F-score[1] range on benchmark

[1]Please refer to Section 1.2.1 for the definition of F-score.

biomedical data sets, far below that of NewsNER which has an F-score well above 90%. Besides the performance issue, there are quite a number of important yet unsolved issues in BioNER. In this thesis, we will study some of these issues.

1.2 Background and Motivation

1.2.1 NER in Newswire Text

The NER task was first introduced for newswire text in the Sixth Message Understanding Conference (MUC-6) in 1995. The purpose was to recognize various information units, including person, organization and location names, and numeric expressions including time, date, money and percent expressions. Three categories of NEs were defined in MUC-6: TIMEX, NUMEX, and ENAMEX. TIMEX phrases are temporal expressions, which were further divided into date expressions (e.g. July 1) and time expressions (e.g., noon PST). NUMEX phrases were numeric expressions, and further divided into percent expressions (e.g., 3.6%) and money expressions (e.g., $50 million). ENAMEX phrases were proper names, representing references in a text to persons (e.g., Bill Clinton), locations (e.g., Vancouver), and organizations (Simon Fraser University).

In MUC, the NER systems were required to output the recognized NEs by tagging them with the predefined categories. For example, the sentence "Handz bought 100 shares of Google Corp. in 2000" would be tagged as follows:

```
<ENAMEX TYPE="PERSON">Handz</ENAMEX> bought <NUMEX TYPE="QUANTITY">100
</NUMEX> shares of <ENAMEX TYPE="ORGANIZATION">Google Corp.</ENAMEX>
in <TIMEX TYPE="DATE">2000</TIMEX>.
```

Conceptually, NER can be considered as a two-step task: first, identify the phrases of interest, and second, categorize them into predefined categories. Typical approaches adopted by previous NER systems include rule-based as well as statistical learning approaches. Using hand-crafted rules, rule-based systems typically can give satisfactory results for a particular domain, but suffer from the difficulty of adapting to new domains. Statistical learning systems require much training data, but can be ported to other languages or domains more easily. While early approaches tended to be rule-based, the more recent and currently dominating technology is supervised learning. The learning algorithms that have been reported in the literature include Decision Tree (DT) [103], Hidden Markov Model (HMM)

[7], Maximum Entropy model (ME) [10], Support Vector Machine (SVM) [6], Boosting and voted perceptron [25] and Conditional Random Field (CRF) [73].

Benchmarking and evaluation have been conducted in MUC-6, MUC-7 and other conferences, such as the Computational Natural Language Learning (CoNLL) shared tasks 2002 and 2003, and the Multilingual Entity Task Conference (MET). Though the focus was mostly on English text (e.g., as in MUC-6 and MUC-7), recent evaluations have extended to other languages, e.g., Spanish in MET-1, Japanese and Chinese in MET-1 and MET-2. Language-independent NER has also been evaluated, e.g., in CoNLL-2002 and CoNLL-2003. Evaluation of system performance was typically done based on two measures - recall and precision. Recall is the percent of the NEs mentioned in the text and correctly recognized by the system; precision is the percent of the NEs that are recognized by the system and are actually correct. The recall score R and precision score P are then used to calculate a balanced F-score, where $F = 2PR/(P + R)$.

Generally speaking, NER is a relatively simple task in NLP, but it has proven difficult to achieve high accuracy, probably because there is considerable ambiguity involved, such as variations in NE spellings, ambiguity in determining NE types, and ambiguity with non-NE words. Significant progress has been made over the past 10 years, although the task is still not a "solved problem". Human performance on the NER task has been determined to be quite high, with F-scores typically better than 96%. Some NER systems in recent evaluations had performance approaching this human performance [90]. For example, the highest scoring system for NER at MUC-6 showed an F-score of 96.5%, with 96% recall and 97% precision [110]. Existing systems perform very well on mixed-case English newswire corpora, probably due to the years of research and organized evaluations on this specific task in this language. It is still not clear how the existing high-scoring systems would perform on less well-behaved texts, such as single-case texts, non-newswire texts, or texts obtained via optical character recognition (OCR) [90].

1.2.2 NER in Biomedical Text

Historically, NER has been focusing on identifying names appearing in newspapers (we thereby call it newswire NER hereafter in the thesis). Since late 1990s, the NER community has paid more attention to NEs in biomedical domain, probably in coping with the increasingly strong need for mining biomedical text. In order to distinguish from the newswire NER, we will hereafter use BioNER to stand for the NER in biomedical literature.

Type of NE	Examples
Gene	PuB1, peri-kappa B, IL-2 gene
Protein	NF-kappa B, CD28, IL-2
Cell	T-cell, monocytes, Saos-2
Drug	Interferon, Zidovudine, Methotrexate
Chemical	P5E-fe Penta-N-methylpyrrolecarboxamide-edta-Fe(II),1,4-diazabicyclo[2.2.2]octane
Disease	HIV, AA amyloidosis, Haemoglobin C disease

Table 1.1: Examples of biomedical named entities

In the domain of biology and medicine, the NEs include genes, proteins, cells, drugs, chemicals, diseases, etc, which are frequently used in biomedical text and interesting to biomedical researchers. Table 1.1 shows some examples of biomedical NEs. Accordingly, BioNER is the task that aims at automatically recognizing all such NEs in biomedical text. It can be simply viewed as NER applied to the biomedical domain.

The necessity of BioNER mainly lies in the fact that BioNER is often the first step for biomedical text mining, which is urgently needed to cope with the explosive volume of published biomedical research. For example, MEDLINE (Medical Literature Analysis and Retrieval System Online), compiled by the U.S. National Library of Medicine (NLM), is a literature database of life sciences and biomedical information, including medicine, nursing, pharmacy, dentistry, veterinary medicine, and health care. It is freely available on the Internet and searchable via PubMed[2] and NLM's National Center for Biotechnology Information's Entrez system[3]. As of March 24, 2008, PubMed contains 17,857,762 citation records from approximately 5,000 of the world's leading biomedical journals published since 1949. The number is growing quickly, e.g., 10,857 new papers were indexed from March 17 to March 21, 2008[4].

With such tremendous volume and explosive growth, it is extremely challenging for biomedical researchers to keep up with new research results, even within one's own research field. Only with the help of effective and efficient text mining techniques, is it possible for them to absorb knowledge and information from the literature so as to benefit their own research. For biomedical researchers, the NEs in biomedical text are the basic units

[2]http://www.ncbi.nlm.nih.gov/sites/entrez/

[3]http://www.ncbi.nlm.nih.gov/Entrez/

[4]http://www.nlm.nih.gov/bsd/revup/revup_pub.html

IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase.

Los M, Schenk H, Hexel K, Baeuerle PA, Dröge W, Schulze-Osthoff K.

Institute of Biochemistry, Albert-Ludwigs-University, Freiburg, Germany.

Activation of the CD28 surface receptor provides a major costimulatory signal for T cell activation resulting in enhanced production of interleukin-2 (IL-2) and cell proliferation. In primary T lymphocytes we show that CD28 ligation leads to the rapid intracellular formation of reactive oxygen intermediates (ROIs) which are required for CD28-mediated activation of the NF-kappa B/CD28-responsive complex and IL-2 expression. Delineation of the CD28 signaling cascade was found to involve protein tyrosine kinase activity, followed by the activation of phospholipase A2 and 5-lipoxygenase. Our data suggest that lipoxygenase metabolites activate ROI formation which then induce IL-2 expression via NF-kappa B activation. These findings should be useful for therapeutic strategies and the development of immunosuppressants targeting the CD28 costimulatory pathway.

PMID: 7641692 [PubMed - indexed for MEDLINE] PMCID: PMC394448

Figure 1.1: An example abstract retrieved from PubMed

conveying various knowledge and information. It would be impossible to fully understand an article unless all the NEs within it could be precisely identified. Moreover, recognition of the biomedical NEs allows further extraction of relationships and other information by identifying the key concepts of interest and allowing those concepts to be represented in consistent and normalized formats. An example MEDLINE abstract and the annotated NEs are shown in Figure 1.1 and Figure 1.2 respectively.

The BioNER task is non-trivial for several reasons. First, there is no complete dictionary for most types of biomedical NEs. Therefore simply using string-matching algorithms cannot solve the problem. Although there are some manually curated terminological resources (such as gene and protein databases), they are often limited to very specialized sub-domains, and cannot even fully cover NEs in these sub-domains, not to say that new names are continuously created and cannot be collected into the dictionaries immediately.

Second, ambiguity, synonymy and name variations are very common in biomedical literature, largely due to the fact that there are no solid naming conventions in the community, although guidelines for naming do exist. The same word or phrase can refer to a different thing depending on context (e.g., ferritin can be a biologic substance or a laboratory test). Many biological entities have several names (e.g., PTEN and MMAC1 refer to the same

{{IL-2 gene}$_1$ expression}$_7$ and {{NF-kappa B}$_2$ activation}$_7$ through {CD28}$_2$ requires reactive oxygen production by {5-lipoxygenase}$_2$.

Activation of the {{CD28}$_2$ surface receptor}$_3$ provides a major costimulatory signal for {T cell activation}$_7$ resulting in enhanced production of {interleukin-2}$_2$ ({IL-2}$_2$) and {cell proliferation}$_7$.

In {primary T lymphocytes}$_5$ we show that {CD28}$_2$ ligation leads to the rapid intracellular formation of {reactive oxygen intermediates}$_6$ ({ROIs}$_6$) which are required for {CD28}$_2$ -mediated activation of the {NF-kappa B}$_2$ / {{CD28}$_2$ -responsive complex}$_4$ and {{IL-2}$_2$ expression}$_7$.

Delineation of the {{CD28}$_2$ signaling cascade}$_7$ was found to involve {{protein tyrosine kinase}$_3$ activity}$_7$, followed by the activation of {phospholipase A2}$_2$ and {5-lipoxygenase}$_2$.

Our data suggest that {{lipoxygenase}$_2$ metabolites}$_3$ activate {{ROI}$_6$ formation}$_7$ which then induce {IL-2}$_2$ expression via {{NF-kappa B}$_2$ activation}$_7$.

These findings should be useful for {therapeutic strategies}$_7$ and the development of {immunosuppressants}$_7$ targeting the {{CD28}$_2$ costimulatory pathway}$_7$.

Figure 1.2: Annotation of named entities in the abstract Medline No.95369245 as in the GENIA corpus. Tokens in each sentence are separated by space. NEs are enclosed by curly braces, with the entity types labeled by the subscripts defined as following: 1: DNA domain/region, 2: protein molecule, 3: protein family/group, 4: protein complex, 5: cell type, 6: inorganic, 7: other names (i.e., those not yet fully categorized in GENIA ontology).

gene). Often the same name can have minor variations in spelling form (e.g., NF kappa B, NF-kappa B and NF kappa-B). Even if biologists start to use exclusively well-formed and approved names, there are still a huge number of documents containing legacy and ad-hoc names.

Third, biomedical NEs often have multiple words (e.g., *CD28 surface receptor*), and shorter NEs can compound together to form a longer NE, so the problem is additionally complicated by the need to determine name boundaries and resolve overlap of candidate names.

Fourth, the total number of biomedical NEs is expected to be very large, and new names are introduced on a daily basis. Therefore, any NER system must be scalable in the sense that it should be able to identify most, if not all, NEs in millions of biomedical articles in reasonable time. Also, it should possess the ability of resolving previously unseen names.

Lastly, the techniques developed for newswire NER may not be directly applicable to BioNER, because the NEs in the biomedical domain exhibit very different characteristics

from those in newswire domain. For example, while most newswire NEs are proper nouns (appearing as capitalized words), many biomedical NEs are not proper nouns (e.g., more than 60% words in biomedical NEs are in lowercase [140]).

Due to the above mentioned reasons, the BioNER task is much more complex than the newswire NER task. We could still apply the methodologies proven successful for newswire texts (e.g., the corpus-based machine learning methods), however, we have to explore the language phenomena specific to the biomedical domain, so as to develop methods to deal with them appropriately.

1.2.3 Language Phenomena in BioNER

In general, most (if not all) of the challenges brought to BioNER spring from the variety of ambiguities involved in biomedical articles. From an NLP perspective, much ambiguity comes from various linguistic phenomena[5], for example, extensive lexical variations (i.e., one name has several spelling variations, all representing the same concept), synonymy (i.e., one concept is represented by several names), and homonymy[6] or polysemy (i.e., one name has several meanings, representing several concepts). These phenomena are common in natural language, but pose specific challenges to biomedical NER. Besides, problems can also come from tokenization and abbreviation. Below we will discuss some of the most significant language phenomena commonly seen in biomedical text.

(1) NE Variations: Biomedical entity names show considerable variations probably because of the existence of multiple naming conventions. Consider protein naming as an example. Researchers may name a newly discovered protein based on its function, sequence features, gene name, cellular location, molecular weight, or other properties, as well as using abbreviations and acronyms. For example, the *EphB2 receptor*, a protein involved in signalling in the brain, was initially referred to as *Cek5*,*Nuk*, *Erk*, *Qek5*, *Tyro6*, *Sek3*, *Hek5*, and *Drt* before being standardized as *EphB2* [84]. Potential standardization based on publishing guidelines and community consensus is hard to uniformly enforce. Moreover, there are entity names whose status is tentative, and there is also a vast amount of legacy data.

[5]There are other types of ambiguity, e.g., structural ambiguity.

[6]Two words are called homonyms, if they have the same spelling and pronunciation, but are different in meaning or origin. For example, the noun *bear* and the verb *bear* are homonymy.

Type of Variation	Example of Variants
Orthographic	*9-CIS retinoic acid* and *9-cis retinoic acid* *amyloid beta-protein* and *amyloid β-protein*
Morphological	*nuclear receptor* and *nuclear receptors* *Down's syndrome* and *Down syndrome*
Lexical	*hepatic leukaemia factor* and *liver leukemia factor* *human cancer* and *human carcinoma*
Structural	*cancer in humans* and *human cancers* *SMRT and Trip-1 RNAs* and *SMRT RNA and Trip-1 RNA*
Acronyms	*RAR alpha, RAR-alpha, RARA, RARa, RA receptor alpha* *NF-kappaB, NF(kappa)B, NFkappaB, NFKB factor, NF-KB*

Table 1.2: NE variation types and examples

Besides, various orthographic styles (upper cases, lower cases, digits, special characters, and their combinations), Greek/Latin spellings (e.g., *NF-kappa B*, *NF-kappa B*, *NF Kappa B*, *NF kappa B*, *NF-kappaB*, *NF kappaB*, all referring to the same entity), adjectival expressions, gerund expressions (e.g., GTPase-activating protein), nominalizations and prepositional phrases (e.g., activation of NF-kappa B by SRC-1), are all frequently used. Table 1.2, taken from [4], gives a simple categorization of the variation types, along with example names. As will be discussed later, the variety of token formation is a major obstacle for dictionary-based approaches.

(2) Abbreviations: Abbreviations, shortened forms of names or concepts, are prevalent in the biomedical literature. Nearly one half of MEDLINE abstracts contain abbreviations [17]. In 2004, 64262 new abbreviations were introduced, and there is an average of one new abbreviation in every 5 to 10 abstracts [17]. It is believed that more than 800,000 abbreviations exist in biomedical literature, and only a small fraction of them have been compiled into dictionaries of abbreviations [17].

Abbreviations are one of the major sources of ambiguity. It is often the case that an abbreviation can be interpreted as several different definitions (i.e., different full forms of the abbreviation), if it were not explicitly defined in the context. That is, many abbreviations are involved in polysemy. For example, *ACE* can be either *angiotensin converting enzyme* or *affinity capillary electrophoresis*, while *EGFR* might correspond to *epidermal growth factor receptor* or *estimated glomerular filtration rate*, depending on the context.

Sometimes even referring to the same full form, an abbreviation can refer to different

things. For example, *NF2*, simultaneously names a gene, the protein it produces, and the disease resulting from its mutation. Similarly, *CAT* can be a protein, an animal, or a medical device. Although linking an abbreviation to its full form is not the task of BioNER, it is closely related to BioNER, and we will discuss this issue in Chapter 2.

(3) Synonyms: also called aliases, referring to the phenomenon that one entity can be denoted by multiple names in a synonymy relation. One gene, officially designated as *SELL*, or *selectin L*, which controls cell adhesion during immune responses, has 15 aliases. Another gene, *MT1*, is used to describe at least 11 members of a cluster of genes encoding small proteins that bind to metal ions [91]. It is estimated that the HUGO Nomenclature includes more than 26,000 aliases among more than 23,000 human genes. A project conducted by a Columbia research group has collected more than 557,000 gene and protein synonyms from only 32,000 full-text articles. Although recognizing synonyms is not the main task of BioNER, it is certainly related to the task, and is partially addressed by NE mapping as a post-processing of BioNER. We will discuss this issue in Chapter 2.

(4) Tokenization: The text in an article has to be tokenized before the NER step is carried out. The tokenization usually includes splitting the article into paragraphs, the paragraph into sentences, and the sentence into tokens. For languages like English, tokens are typically separated by a space.

Linguistic phenomena that can cause problems for tokenization include abbreviations (e.g., those signalled by a period), apostrophes (e.g., *IL-10's activity*), hyphenation (e.g. *IL-10* or *IL 10*), multiple formats (e.g., *123,456.78* and *123456.78*), various sentence boundaries (e.g., demarcated by period, exclamation mark, question mark, colon, semi-colon, or dash).

In biomedical text, tokenization encounters additional challenges due to domain-specific terminology, and non-standard punctuation and orthographic patterns (e.g., *an alpha-galactosyl-1,4-beta-galactosyl-specific adhesion*, and, *the free cortisol fractions were 4.53 +/- 0.15% and 8.16 +/- 0.23%*).

As another example, consider the sentence "*we here report that interferon inhibits growth and survival of NB4 APL cells in cooperation with RA.*" Here, the acronyms *NB4* and *APL* come as two separate tokens. However, sometimes the two tokens appear as one token *NB4-APL* connected by a hyphen. From a biological point of view, *NB4* is a cell line derived from acute promyelocytic leukemia (APL) cells. From a NLP perspective,

however, it is essential that both formats be tokenized in a consistent and unified way. This example also highlights the importance of acronym detection, because *APL* can stand for *acute promyelocytic leukemia*, and also *antiphospholipid syndrome* (an autoimmune disease), while *RA* can stand for *retinoic acid*, *retrograde amnesia*, *refractory anemia*, or *rheumatoid arthritis*.

(5) Multi-word NEs: the majority of biomedical NEs are multi-token phrases (compounds), containing at least one white space or a hyphen. For example, 90% of the NEs in the GENIA corpus[7] contain typically two or three tokens per NE, while NEs with six or more tokens are rare, although they do exist. The main problem caused by multi-word NEs is the difficulty in detecting their boundaries in a sentence.

(6) Nested NEs: one NE may occur within a longer NE (as a proper string), as well as occur independently. For example, *T cell* is nested within *nuclear factor of activated T cells family protein*. In the GENIA corpus, nested NEs account for about 21.5% of all NE occurrences, with 8.4% of all distinct NEs occurring as nested. Meanwhile, it is reported [88] that about two-thirds of Gene Ontology (GO) terms contain another GO term as a proper substring. Another almost third of all nested NEs appear more than once as nested, while more than a half of nested NEs do not appear on their own elsewhere in the GENIA corpus. These facts suggest that the recognition of the inner structure of NEs cannot rely on spotting the occurrences of the corresponding sub-NEs elsewhere in the corpora.

(7) NEs encoded in Coordinations: Coordination is a multi-word variation phenomenon where lexical parts common for two or more NEs are shared (appears only once), while their distinct lexical parts are enumerated and coordinated with a coordination conjunction (CC). Therefore, coordination encodes at least two NEs. In the GENIA corpus, about 2% of all NE occurrences are involved in coordination, with the majority containing two NEs. About one third of coordinated NEs appear also as ordinary NEs elsewhere in the corpus. For example: *adrenal glands and gonads* can be interpreted as coordination (i.e., *adrenal glands* and *adrenal gonads*) or conjunction (i.e., *adrenal glands* and *gonads*). Note that resolving coordination would involve deep semantic understanding, and likely remains a hard problem in computational linguistics.

[7]The GENIA corpus is a corpus of biomedical term annotation that has been frequently used by previous BioNER study. We also use it in this thesis. More details are given in Chapter 4 and Appendix A.

Lexicon	Size	Unknown Word Rate
Brown	25K	28.2%
WSJ	40K	25.5%
Brown + WSJ	50K	22.4%
GENIA	15K	5.3%
Brown + WSJ + GENIA	60K	4.6%

Table 1.3: Unknown word rates in GENIA using dictionary made from the Brown, WSJ news corpora and GENIA biomedical corpus.

Linguistic Forms	Example Gene and Protein Names
Abbreviation	IL-2
Plural	P38 MAPKs, ERK1/2
Compound	Rpg1p/Tif32p
Coordination	91 and 84 kDa proteins
Cascade	Kappa 3 binding factor
Anaphoric	It, this enzyme
Description	Inhibitor of p53, a protein that binds RNA
Acronym	Phospholipase D (PLD)
Apposition	PD98059, specific MEK1/2 inhibitor

Table 1.4: Example gene and protein names in various linguistic forms

(8) Unknown Words: compared to newswire texts, biomedical texts typically contain more unknown words. Table 1.3, taken from [64], shows the unknown word rate on the GENIA corpus, using lexicons extracted from two well-known newswire corpora (i.e., Wall Street Journal (WSJ) corpus and Brown corpus) and one biomedical corpus (i.e., the GENIA corpus). From the table, we can see that the unknown word rate is undesirably high when using a general purpose lexicon. Even using a domain specific lexicon (such as one made from the GENIA corpus), the unknown word rate would still be quite considerable. For lexicon-based string-matching approaches and some rule-based approaches, the high unknown word rate can be a big obstacle.

In short, biomedical NEs can appear in many linguistic forms in biomedical articles. Table 1.4 summaries various linguistic forms with examples of gene and protein names. Note that the above mentioned linguistic phenomena, though exemplified with NEs appearing in English biomedical text, are typically observed in biomedical text written in other languages (e.g., Chinese). Ideally, a BioNER system should be able to handle most, if not all, of these forms, to become a practical tool for biomedical text mining.

1.3 The Target Problems and Our Methodology

In this thesis we will address the following questions:

1. How can BioNER performance be improved?
2. How can we decrease or remove the dependency of BioNER systems on annotated corpora?
3. How can BioNER be performed on non-English texts?

In addressing these questions we will do the following:

First, we will study whether corpus based supervised learning methods, currently dominant in BioNER, would achieve higher performance by using larger corpora for training. We will evaluate this hypothesis by drawing the learning curves of two popular learning methods, namely SVM and CRF, using a benchmark data set. Our goal is to see whether the hypothesis would hold for BioNER.

Second, we will look into the issue of nested (or embedded) named entities, which has been largely neglected by previous work. The idea is to learn a separate NER model for each nesting level in the training phase and applying to the corresponding level in the testing phase. We will evaluate the idea on a benchmark corpus to see how well it would work for both the non-nested NEs and nested NEs.

Third, we will investigate the applicability of NP chunks to the BioNER task, based on the intuition that an NE is basically an NP. Inspired by the good performance of current NP-chunking systems, we would like to see if NEs can be built on top of NP chunks. We will examine the benefits as well as the problems associated with this idea.

Fourth, we will explore the feasibility of doing BioNER in the absence of annotated corpora, as they are often very expensive to obtain. Noticing that domain-specific dictionaries sometimes are more readily available than annotated corpora, we will see if they can be used to label some of NEs in a sentence, and if these partially labeled sentences can be used to train a classification model by semi-supervised learning.

Finally, we will look at the problem of doing BioNER in Chinese biomedical texts, an area that has not been studied by previous work. Given that there are no publicly available annotated corpus for Chinese BioNER, we will manually annotate a small set of Chinese biomedical abstracts. We will use the corpus to conduct a preliminary evaluation to see if

statistical learning methods are applicable for the particular task, and if so, what kinds of features are usable.

Among the above five problems, the first three address the performance issue, while the fourth deals with the annotated corpora issue, and the fifth studies the issue of non-English texts.

Our methodology, in general, is to consider the BioNER problem as a classification problem and approach it by supervised learning methods. As we will discussed in the next chapter, supervised learning has been currently the dominant approach in BioNER. Given a training corpus with annotated BioNEs, this approach aims to learn a model with the ability to recognize previously unseen entities. The model is characterized by distinctive features associated with the positive and negative examples seen in the training corpus.

1.4 Summary of Contributions

As the result of studying the above problems, this thesis makes the following contributions to BioNER research:

- A comprehensive and up-to-date survey of previous work in BioNER;
- An empirical study showing that using larger corpora to train supervised learning models may not be a good way to improve BioNER performance;
- A supervised learning based method for BioNER that identifies NE boundaries based on noun phrase (NP) chunks;
- A semi-supervised learning method to do BioNER when annotated corpora are not available;
- A character-based supervised learning method as well as an annotated corpus for BioNER in Chinese texts;

Moreover, the thesis provides statistics about language phenomena existing in the studied biomedical texts. These results and the data should be useful for future research in BioNER, BioNLP, general NLP as well as computational linguistics.

1.5 Thesis Organization

The thesis is presented in the article-style format. All chapters can be read independently. The necessary terminology is introduced (and in many cases repeated) wherever used in the stand-alone articles.

The rest of thesis is organized as follows. Chapter 2 presents a survey of previous work in BioNER, from which the subtopics of the thesis are identified. In Chapter 3 we study the relationship between the NER performance and the size of annotated data used for training a supervised model, showing that using a larger annotated corpus may not be a good way to improve the performance for statistical learning approaches. In Chapter 4 we study how to effectively recognize embedded NEs by training a word-based model for each level of nested annotations. In Chapter 5 we study how to do BioNER by NP chunking, assuming that NP chunking would help to detect NE boundaries, based on the intuition that an NE is very likely within an NP. In Chapter 6 we study how to do BioNER when the annotated corpus is not available. In Chapter 7 we study BioNER in Chinese texts. In Chapter 8 we summarize our results and discuss directions for future work.

Chapter 2

Survey of Previous Work

A Named Entity (NE) historically refers to a phrase that denotes a specific object (or a group of objects) of reader's interest, such as person(s), organization(s), and place(s) appearing in news papers. Named Entity Recognition (NER) is a task that seeks to automatically recognize the NEs in a text. In this thesis we study BioNER, aiming at identifying NEs in biomedical text.

This chapter is a survey of the area of Biomedical Named Entity Recognition. It is organized as follows. In Section 2.1, we classify previous BioNER work into four approaches and discuss their strengths and weaknesses. In Section 2.2, we discuss the methods for performance evaluation of a BioNER system. In Section 2.3, we review topics that are closely related to BioNER. We propose future research directions and summarize the chapter in Section 2.4.

2.1 Approaches to BioNER

2.1.1 Problem Definition

We now formally define the BioNER problem. Given a biomedical article, we assume that it has been properly tokenized and sentence boundaries have also been properly detected. As an NE usually does not span across two sentences, we can reduce the problem of recognizing NEs in the article into one of recognizing NEs in each sentence. Here we do not address the problem of anaphora resolution, which has been long studied and still remains a hard problem. In other words, we only recognize those NEs that are explicitly expressed or

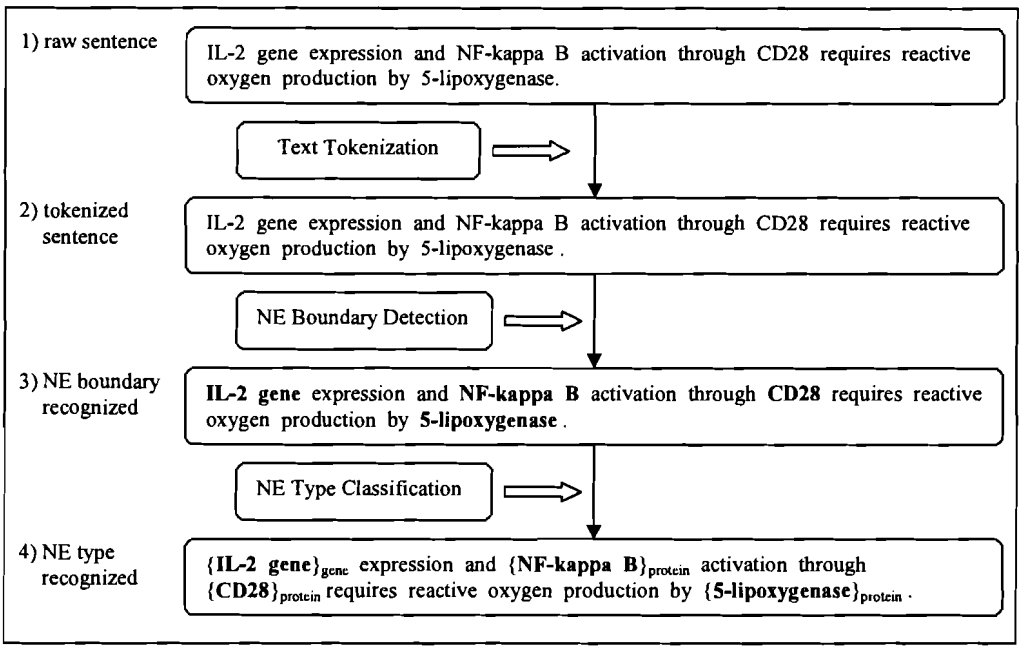

Figure 2.1: An example of typical procedure of BioNER

mentioned in the sentence. Whenever encountering an anaphoric expression, we simply ignore it. Under these assumptions, a sentence S that has n tokens (i.e., words) in sequence can be denoted as $S =< t_1, t_2, t_3, ..., t_n >$. Suppose we want to extract entities that belong to k biomedical categories, $C_1, C_2, ..., C_k$. The problem of BioNER is to find all proper subsequences of S that identify as entities of the k categories. That is, each token is associated with one of the k category names.

Procedurely, the BioNER task can be done in two steps: NE boundary detection and NE type classification. The NE boundary detection step is to decide if a single token or several adjacent tokens represent an NE without considering the type of the NE. In other words, it only distinguishes NEs from non-NEs. The NE type classification step is to assign a specific type to each NE identified in the boundary detection step. In practice, the two steps can also be merged into one step, as in some machine-learning based BioNER systems that we will discuss later. Figure 2.1 shows an example of typical procedure of BioNER, where NE boundary detection and NE type classification are separated.

Previous approaches to the BioNER problem roughly fall into four groups: dictionary-based, rule-based, machine-learning, and combined approaches. The dictionary-based approaches try to find names in the text based on well-formed dictionaries, where the dictionaries can be constructed manually or automatically. The rule-based approaches employ pre-defined rules or patterns for matching NEs in text, where the rules and the patterns can be manually crafted or automatically learned. The machine-learning approaches use machine learning techniques, such as HMMs or SVMs, to induce statistical models for biomedical NEs. Each of the approaches typically has its own strengths as well as weakness. Therefore, two or more approaches can be combined in order to achieve better performance, if the weakness of one approach can be remedied by the strength of another.

2.1.2 Dictionary-based Approaches

The dictionary-based approaches identify NEs in a text by looking up (e.g., string matching) a provided list of known NEs, typically compiled from existing lexicons or databases. The only assumption is the existence of such a list. In the biomedical domain, there are some well-managed lexicons or databases especially for genes (e.g., GenBank), proteins (e.g., UniProt), and chemicals (e.g., ChemIDplus). The string matching can be exact or approximate (e.g., using edit distances to tolerate minor spelling variations).

The strength of this approach is that it is simple and efficient, because the string matching problem has been well studied in computer science and there are various efficient algorithms available. The major problem is that the performance in terms of accuracy usually is not satisfactory. The main reason is that most lexicons have limited coverage, typically specialized to some particular domains and usually not up-to-date. Other reasons include spelling variations (e.g., punctuation and word order) and homonymy (e.g., many abbreviations share lexical forms with common English words).

The spelling variation problem can be alleviated by using approximate string-matching techniques, however, such elastic-matching methods have much greater computational cost than exact-matching techniques. The homonymy problem is more difficult to solve than spelling variation, and typically requires computationally expensive analyses of syntax and semantics. Below we will review some representative works that adopt this approach.

Hirschman et al [48] employed a simple pattern matching strategy, to locate genes using gene names in FlyBase. They reported very low precision rate (2% in full articles and 7% in abstracts) with recall ranging from 31% for abstracts to 84% for full articles. They suggested

that the main reason for such poor precision was homonymy, as many gene names shared their lexical forms with common English words (e.g., gene name/abbreviations such as *an*, *by*, *can*, and *for*).

Tuason et al [123] reported that name variations could account for up to 79% of the missing genes if straightforward string matching was used. They indicated that punctuation variation (e.g., *bmp-4* and *bmp4*), using different numeral styles (e.g., *syt4* and *syt iv*), different transcriptions of Greek letters (e.g., *iga* and *ig alpha*), and word order variations (e.g., *integrin alpha 4* and *alpha 4 integrin*) were the most frequent causes of gene name recognition failures.

Tsuruoka and Tsujii [122] designed a probabilistic generator of spelling variants based on edit distance operations (namely substitution, deletion, and insertion of characters and digits). Only NEs with edit distance less or equal to one were considered as spelling variants. They reported improved lookup performance especially for long NEs. In [121] they further presented an adjusted method for approximate string matching against a dictionary of protein names. By fine tuning the cost function for edit operations and using a naive Bayesian classifier trained on protein names in GENIA corpus, they reported precision 73.5% and recall 67.2% using the same corpus.

Tanabe and Wilbur [113] processed MEDLINE documents to obtain a collection of over two million names, using a gene and protein name tagger (called ABGene) derived from Brill's POS tagger [11] trained on Medline abstracts to full articles. In [114] they further explored methods to purify the collection, in order to obtain a high quality subset of gene/protein names. Their approach was based on the generation of certain classes of names that are characterized by common morphological features. Within each class, inductive logic programming (ILP) is applied to learn the characteristics of gene/protein names. The criteria learned in this manner were then applied to the collection.

Egorov et al [33] proposed a simple and practical dictionary-based approach for recognizing proteins in Medline abstracts. They first constructed a dictionary of mammalian proteins in a semi-automatic way from various public-domain sequence databases, followed by an intensive expert curation step. In order to handle the variations in protein names, they designed a specialized tokenization algorithm, instead of using approximate string matching algorithms, to identify and tag protein name occurrences in biomedical texts. The idea was to convert the input text into a sequence of tokens that are made from the longest sequences of characters belonging to the same class. This conversion actually reduced the variations

by normalizing them before the NER step. They evaluated the system, ProtScan, using 1000 randomly selected and hand-tagged Medline abstracts, reporting 98% precision and 88% recall. The processing speed was about 300 abstracts per second.

In summary, research on the dictionary-based approach has been mainly on these aspects: (1) performance evaluation, (2) how to handle the variations in names, (3) how to (semi-) automatically construct the dictionary. These results have suggested that the dictionary-based approach, if used alone, would not achieve satisfactory performance, and therefore more effort should be made on using the approach in combination with rule-based and/or machine-learning approaches. We will discuss this issue later.

2.1.3 Rule-based Approaches

The rule-based approaches generally attempt to recognize NEs by predefined rules. The rules typically describe common naming structures using either orthographic or lexical clues, or morpho-syntactic features, e.g., word alphanumerical composition, the presence of special symbols, capitalization, and special nouns or special verbs. Crafting such rules typically requires extensive knowledge of linguistics, biology, medicine, and possibly computer programming languages. In some cases, small lists of typical name constituents (e.g., terminological heads, prefixes, suffixes, acronyms, positive names and negative names) are also used.

The strength of rule-based approach is that rules can be carefully designed to deal with specific linguistic phenomena. The main problem is that developing good rules usually requires extensive domain knowledge and can be very time-consuming. Moreover, the defined rules are usually specific to a particular domain, which makes it difficult to be applied to other domains. Some representative works that adopt this approach are reviewed below.

Ananiadou [2] implemented a computational morphological grammar and lexicon, and applied it to medical term recognition. Observing that medical terminology heavily relies on Greek and Latin neoclassical elements for creating terms such as *erythrocyte* and *angioneurotic*, the author proposed a four-level ordered morphology to describe the term formation patterns. However, no evaluation was reported for the proposed methodology.

Fukuda et al [37] presented a system KEX to recognize protein names using surface clues on character strings. The system first identifies core terms (e.g., containing special characters) and feature terms (e.g., protein and receptor). It then concatenates the terms

by handcrafted rules and patterns, and extends the boundaries to adjacent nouns and adjectives. They reported very good performance (precision=94.7%, recall=98.8%) on a small collection of abstracts of specific domains.

Proux et al [94] studied gene names for Drosophila and found that they fall into three categories: (1) names including special characters (32%) (e.g., *Hrp54*); (2) names using only lower case letters and common English words (32%) (e.g., *vamp* and *ogre*); and (3) names using only lower case letters but not common English words (36%) (e.g., *ynd* and *zhr*). Instead of using the approach proposed by [37], they employed a tagger with a nondeterministic finite-state automaton that works in three steps: tokenization, lexical lookup, and disambiguation using HMM. They reported performance of (precision=91.4%, recall=94.4%).

Gaizauskas et al [38] used a manually constructed context-free grammar for protein name recognition. They first split a protein name into component terms, based on its apparent syntactic structure. They then added corresponding grammar rules in the process of recombining the components. For example, for the enzyme name *calmodulin N-methyltransferase*, they recognized the first word *calmodulin* as a potential "enzyme modifier" by looking up the dictionary of enzyme modifiers manually constructed from Swiss-Port and EMTREE. They also identified the last word *N-methyltransferase* as a potential "enzyme head", as suggested by the suffix "*-ase*". They finally derived two context-free grammar rules from the phrase:

```
rule1: enzyme --> enzyme_modifier, enzyme;
rule2: enzyme --> character, '-', enzyme_head;
```

In total, they constructed 160 rules for protein names and reported good performance on recognizing enzyme names (precision=96%, recall=98%).

Thomas et al [116] customized existing general-purpose NE recognizers (e.g., those used in FASTUS) for the protein NE task. Recognition was carried out in several phases using a cascade of finite-state transducers, which recognized complex units (e.g., *3,4-dehydroproline*) and "basic phrases" that are extended to the surrounding words using domain-independent rules for the construction of complex noun groups. Although the performance was not as good as the above domain-dependent systems, they argued that the customization to a new domain can be fast, reliable and cost-effective.

Franzen et al [36] developed a system called Yapex by extending the idea of KEX [37].

In particular, they added data sources (e.g., "core" terms compiled from Swiss-Prot), additional heuristic lexical filters and results of syntactic parsing for detecting NE boundaries. They reported performance of (precision=66.4%, recall=67.8%) for strict match, tested on a human-annotated collection of 100 Medline abstracts.

Narayanaswamy et al [83] used an idea similar to that of KEX [37] to recognize chemical and source NEs (e.g., cells, cell parts and organisms). They used chemical roots and suffixes to identify chemicals, while different classes of "feature" terms are used to perform more sophisticated classification. In addition, context and surrounding words are used for further classification (e.g., in a context such as "expression of *CD40*", the word "expression" indicates that *CD40* is a protein or gene). The performance was evaluated on a manually annotated collection of 55 Medline abstracts, achieving performance of (precision=90.4% and recall=95.6%).

Hou and Chen [50] utilized protein/gene collocates extracted from biological corpora as restrictions to filter false candidates for improving precision of protein/gene name recognition. In addition, they integrated the results of multiple NE recognizers (e.g., Yapex and KeX, and ABGene) to improve the recall rates. They reported significantly improved performance, using both filtering and integration strategies together (e.g., the F-score increases by 7.83% compared to the pure Yapex method).

In summary, most rule-based approaches have focused on designing specialized high quality rules for specialized subdomains, while some studied customizing general-purpose NER systems to the biomedical domain. Probably because of the carefully crafted rules, accuracy obtained with this approach typically was satisfactory. However, there are few works addressing the essential difficulties faced by the rule-based approaches: hard to adapt to new domains, and time-consuming in rule construction. As we will see in the next section, the machine-learning approach should be preferred, if domain adaptability is desired. As for reducing the time in rule construction, automatic rule construction methods might be a future direction for the rule-based approach.

2.1.4 Statistical Machine Learning Approaches

Most machine learning methods used in BioNER have been supervised learning, using training data to learn features useful for NE boundary detection and NE type classification. The main strength of the ML approach is that it has demonstrated better performance than the

rule-based approach. Moreover, it is easier to adapt to new domains, compared to the rule-based approach. The main problem with this approach is that it requires reliable training resources.

Another main challenge is how to select discriminating features. Besides, as the number of features for machine learning systems increases to cover more linguistic information, data sparseness can be a serious problem for the ML methods used, resulting in degraded generalization capability.

The supervised ML-methods that have been exploited for NER include Hidden Markov models (HMM), Maximum Entropy (ME), Support Vector Machines (SVM), and more recently Conditional Random Field (CRF). Note that all these machine learning methods have also been used in NLP tasks other than NER, e.g., POS tagging and syntactic parsing. Depending on the tasks and the types of NEs, these methods yield more or less similar results.

Collier et al [24] used a bi-gram HMM based on lexical and character orthographic features for recognizing NEs of 10 classes. For each sentence, the model took an input that consists of the sequence of words in the sentence and their features. For each given class, the model then calculated the probability of a word belonging to the class. Finally, it produced the sequence of classes with the highest probabilities for the sentence. The evaluation was done on a small corpus of 100 Medline abstracts, reporting an F-score of 73%.

Kazama et al [54] trained multi-class SVMs on the GENIA corpus, using position-dependent features (such as POS, prefix, and suffix features), as well as a word cache (captures similarities of patterns with a common keyword) and HMM state features in order to address the data sparseness problem. In general, an F-score of 50% was achieved.

Morgan et al [81] used HMMs based on local context and simple orthographic and case variations. They first applied simple pattern matching to identify gene names in the associated abstracts and filtered these entities using the list of curated entries for the article. This process created a data set that was then used to train the HMM entity tagger. The results from the HMM tagger were comparable to those reported by other groups (F-score of 75%). The authors suggested that their method has the advantage of being rapidly transferable to new domains that have similar existing resources.

Shen et al [107] trained HMM using prefix/suffix information, part-of-speech (POS) tags, and noun heads as features, in addition to orthographic features. They achieved F-scores of 16.7-80% depending on the class (overall F-score 66.1%; the protein class F-score

was 70.8%), and reported that POS tags (obtained by a tagger trained on the biomedical domain) proved to be among the most useful features.

Takeuchi and Collier [112] trained SVMs with surface word forms, part-of-speech tags, head-nouns and orthographic features. They reported better performance (Fscore of 74.2% for 10 classes) on a small corpus of 100 Medline abstracts.

Yamamoto et al [128] combined boundary features (based on morpheme-based tokenization) with morpho-lexical (POS tags, stems), "biomedical" (whether a given word exists in a compiled database of biomedical resources), and syntactic features (head morpheme information) to train SVMs on the GENIA corpus. They reported an F-score of 0.75 for protein names, and determined that "biomedical" features were crucial for recognizing them. (Lee et al, 2003) suggested strict separation of the recognition and classification steps in the SVM-based NER. For NE boundary detection, they used "standard" features (orthographic, prefix, and suffix information) coupled with a simple dictionary-based refinement of boundaries of the selected candidates (by examining the adjacent words-if they appeared in the dictionary, they were included as part of the term). For NE type classification, they combined a set of class-specific "functional" words and contextual information as features. They reported that the two-phase model showed better performance compared to the "standard" approach, mainly because discriminative features were selected for each subtask separately.

Zhou et al [140] presented another HMM with various features, including word formation patterns, morphological patterns, part-of-speech, semantic triggers, and name alias features. A k-NN algorithm was proposed in order to deal with the data sparseness problem caused by the large number of used features. They also conducted post-processing based on rules automatically drawn from training data to deal with nested NEs phenomenon. They reported the F-score ranging from 83.6% to 86.2% evaluated on the GENIA corpus.

Lin et al [66] adopted ME for biomedical NER, incorporated with dictionary-based and rule-based methods for post-processing. They considered orthographical features, head nouns, and morphological features in training the ME model. Evaluated on the GENIA corpus, they reported overall performance of precision/recall/F-score 51.2%/53.8%/52.5% for 23 categories when simply using ME for NER. In order to overcome inaccurate boundary detection of NEs and misclassication with the ME classifier, they incorporated a post-processing stage, using dictionary-based and rule-based methods to extend the boundary of partially recognized NEs and to adjust classification. After the post-processing the performance was significantly improved to precision 72.9%, recall 71.1%, and F-score 72%.

Finkel et al [35] built an NER system based on the Maximum Entropy Markov Model, an ME enhancement of traditional HMMs. The system made extensive use of local and syntactic features within the text, as well as external resources including the web and gazetteers. The total number of features added up to over a million. It achieved an F-score of 70% on the COLING 2004 NLPBA/BioNLP shared task of identifying five biomedical named entities in the GENIA corpus.

Settles [104] built an NER system using Conditional Random Fields (CRF), which could recognize multiple entity classes at the same time. They used orthographic features including capitalization, affixes, and word shapes. In particular, semantic domain knowledge was incorporated through prepared lexicons. Some lexicons were manually collected (e.g., those of Greek letters, amino acids, chemical elements, known viruses, and abbreviations), whereas others were automatically created (e.g., the lexicon of signal words by means of the Chi-Square test). The system achieved overall performance of an F-score around 0.70, near the current state of the art, with only simple orthographic features.

In summary, research that adopts the statistical and machine learning approach has frequently focused on (1) adopting new ML algorithms that were successfully used in other NLP tasks to BioNER, and (2) incorporating linguistic information as much as possible into features. In general, the performance of ML methods is better than that of dictionary- and rule-based approaches.

2.1.5 Combined Approaches

Since the three approaches discussed above all have their own strength and weakness, there is a clear need for combining them to obtain better performance. In fact, some research works introduced above already used a hybrid of different approaches. For example, Hanisch et al [47] utilized a machine learning technique for computing optimized parameters of scoring measures in a dictionary-based system, while Zhou et al [140] automatically constructed rules to deal with cascaded NEs for their machine learning system. Below we will introduce some other representative works.

Proux et al [94] used a cascade of finite state lexical tools to recognize single-word gene names. Their method was based on a morphological POS tagger, which used a special tag ("guessed") for tokens that cannot be matched with classical transducers. Most gene names were tagged with the "guessed" tag, and eventually confirmed through contextual analysis

(e.g., the presence of a word gene next to a candidate token validates its status as a gene-name). Special post-processing steps are necessary to recover or remove erroneously tagged tokens, including the use of a dictionary of general expressions from biology. They reported a performance of 91% precision and 94% recall on a small corpus of 1200 sentences selected from Flybase , a database of Drosophila genome.

Rindflesch et al [97] built a system called ARBITER for recognizing binding terms by combining several approaches. A binding term is a noun phrase referring to a *binding entity*, which can be a molecule, a genomic structure, a cell or cell component, or some topographic aspect of a molecule, cell or cell component. They selected NPs as potential binding terms if the NPs map to the UMLS or GenBank, or exhibit abnormal morphological characteristics compared to a constrained list of regular English terms or contain heads (e.g., ligand or subunit). Similar to PROPER's extension rules, simple binding terms are joined into complex expressions under specific conditions (e.g., prepositional modification, appositival complementation, etc). Overall, the reported precision was 79% at 72% recall.

Tanabe and Wilbur [113] proposed a combination of statistical and knowledge-based strategies to extract gene and protein names from MEDLINE abstracts. First they applied automatically generated rules from the Brill POS tagger [11] to extract single word gene and protein names. These results were then filtered extensively using manually generated rules formed from morphological clues, low frequency tri-grams, indicator words, suffixes, and part-of-speech information. A key step during this process was the extraction of multi-word gene and protein names that were prevalent in the literature but inaccessible to the Brill tagger. Finally, they applied Bayesian learning to rank the documents by similarity to documents with known gene names and showed the effect of an assumption that documents below a certain threshold do not contain gene/protein names. They tested the method on a collection of 56469 Medline abstracts, and obtained an F-score of 89%.

Mika et al [75] proposed a system NLProt that combines a pre-processing dictionary- and rule-based filtering step with several separately trained SVMs to identify protein names in the MEDLINE abstracts. The NLProt is capable of extracting protein names with a precision of 75% at a recall of 76% after training on a corpus, which was used before by other groups and contains 200 annotated abstracts. For the dictionary to filter irrelevant words, such as common words, medical terms, and species names, they utilized the online version of the Merriam-Webster dictionary (http://www.m-w.com), a dictionary of medical terms (http://cancerweb.ncl.ac.uk/omd), and the species names in UniProt Knowledgebase

(http://us.expasy.org/cgi-bin/speclist). They also used a rule to filter names that are followed by cell(s) or cyte(s) (e.g., *CD4+T lymphocytes*, *Streptococcus mutans cells*).

In summary, the combined approach has demonstrated the potential to obtain better performance than any single approach. Previous works differ in the ways that the combination was implemented.

2.2 Performance Evaluation

The reported performance of most research discussed in the previous section cannot be directly compared with each other, because they were not tested on common test corpora or did not use the same evaluation metrics. To deal with this problem, some evaluation metrics have been proposed, and some evaluation conferences have been held for the BioNER task, as will be discussed below.

2.2.1 Evaluation Metrics

Most NER systems use precision, recall, and F-score to measure performance. Precision (P) is the number of NEs a system correctly detected divided by the total number of NEs identified by the system. Recall (R) is the number of NEs a system correctly detected divided by the total number of true NEs contained in the input text. F-Score (F) combines these two into a single score and is defined in the equation: $F = 2 * P * R/(P + R)$.

As has been shown in the previous section, many BioNER systems also used these metrics to report their performance. However, it has been noticed that the P/R/F performance measure might not be sufficient for the BioNER task, because the boundaries and the categories of biomedical NEs are often ambiguous. Therefore, it is necessary to design various matching criteria to meet the needs of different applications.

Tsai et al [120] proposed some other useful metrics, as summarized in Table 2.1. Their experiments showed that the NER performance varied with different matching criteria. They suggested that relaxed matching criteria can be used to realistically cope with various application requirements.

Exact match	A candidate NE can only be counted as a match if both its boundaries and its class fully coincide with an annotated NE.
Left match	If the left boundary matches exactly, the tagged NE is scored as a match.
Right match	If the right boundary matches exactly, the tagged NE is judged as correct.
Left-or-Right match	If a tagged NE exactly matches either boundary of the human-annotated NE, the hit is counted as a match.
Partial match	A detected NE is counted as correct when any fragment composing the NE is correctly detected.
Approximate match	A tagged NE must be a substring of the human-annotated NE or vice versa. Less restricted than Left-or-right match criteria.
Name-Fragment match	Considering each token in an NE separately, it assesses what percentage of an NE has been correctly recognized.
Core-Term match	machine-annotated NEs must contain a core term to be considered correct. The core term can be defined based on the task.

Table 2.1: Various matching criteria for BioNER

2.2.2 Evaluation Conferences

Gene Name Extraction Task in BioCreAtIvE

Held in 2004, the first BioCreAtIvE challenge (Critical Assessment of Information Extraction in Biology) was to provide a set of common evaluation tasks to assess the state-of-the-art for text mining applied to biological problems. BioCreAtIvE Task 1A was for gene name extraction, in which 15 teams participated. Participants were given 10,000 annotated sentences for training, and another 5000 sentences for testing. All the sentences were selected from MEDLINE, tokenized and marked for mentions of "names" related to genes, including binding sites, motifs, domains, proteins, promoters, and so on. All systems' output were compared to the "golden standard" manually annotated names using exact match.

Among the 15 teams, four teams achieved scores over 80% F-score, with the highest being 83%, all using some kinds of Markov model at the system's top level. However, the teams used different techniques on their Markov models: maximum entropy, hidden Markov models (HMM) and conditional random fields. In addition, one team also had an SVM system at the top level: decisions were made by having two HMMs and an SVM system vote. There were teams using manually generated rules for the task, achieving F-score around the 67% level.

The results are good, but still somewhat lag the best scores achieved in some other domains such as English newswire (e.g., extraction of organization names has been reported higher than 90% F-score). Yeh et al [131] provided an analysis of what causes the gap. They suggested that approximately one-half of the difference in the F-score can be attributed to the fact that gene names are often longer than organization names. The remaining discrepancy may be due to annotation inconsistency in the biological corpora, i.e., the ambiguity about what constitutes a gene or protein name. The common difficulty of this task was determining the boundaries of the names. For a detailed overview of the task, we refer readers to [131].

BioNLP Shared Task in JNLPBA

Also held in 2004, the BioNLP Shared Task was organized by the international Joint Workshop on. Natural Language Processing in Biomedicine and its Applications (JNLPBA), focusing on recognizing biological NEs. The training data came from GENIA version 3.02, containing 2000 annotated MEDLINE abstracts that were selected based on keyword hits for the MeSH terms *human*, *blood cells*, and *transcription factors*. There were 36 classes of biological NEs, however, only 5 were used for the shared task: *protein*, *DNA*, *RNA*, *cell line* and *cell type*. The testing data were created by annotating 404 new abstracts with the 5 classes. About one-half of the abstracts were chosen from the same domain as the training data (i.e., retrieved using the same set of keywords), while the other half were chosen from a more general domain, using only the MeSH keywords *blood cells* and *transcription factors*. The test set was also subdivided according to the year of publication, resulting in somewhat different proportions of the five classes of NEs. Results were measured in F-scores, using exact match, right boundary match, and left boundary match.

Eight teams participated in the task. The best F-score for the exact match was around 0.73, using a combination of HMM and SVM. All teams used the machine learning approach as the backbone of their systems. Roughly four types of classification models were applied: Support Vector Machines (SVMs), Hidden Markov Models (HMMs), Maximum Entropy Markov Models (MEMMs) and Conditional Random Fields (CRFs). The most frequently applied models were SVMs. Maximum Entropy Models were applied by only one system [35], while it was the most successfully applied model in the CoNLL-2003 Shared Task of Language-Independent Named Entity Recognition. [56] gives a detailed description of the shared task.

Evaluation Task	Target NEs	Best F-score	Approach used by the Best System
BioCreAtIvE (2004)	Gene (and related proteins)	83%	Maximum Entropy
BioNLP (2004)	protein, DNA, RNA, cell line, cell type	73%	HMM + SVM

Table 2.2: Performances in BioCreAtIvE and BioNLP BioNER Tasks

In summary, at least two evaluation competitions have been held in the BioNER community. The best performance is summarized in Table 2.2, where one can immediately notice the discrepancy in performance between the two competitions. One reason could be the differences in task difficulty. BioNLP had 5 classes of NEs to identify, while BioCreAtIvE had only 1 class of NEs. Another reason could be due to the annotation quality in the corpora used in the two competitions. Dingare et al [32] found that about 70% of errors could be attributed to inconsistent annotation in the training or evaluation data used in BioNLP.

2.3 Related Research Topics

So far we have seen the general approaches to tackling the BioNER task and their evaluation. However, there are some outstanding issues that are directly related to the BioNER task but are examined in related areas. In this section, we will briefly describe three of them, namely acronyms/abbreviations detection, named entity mapping, and research in general NER.

2.3.1 Detecting Acronyms/Abbreviations

Biomedical NEs often appear in shortened or abbreviated forms. The purpose of acronym detection is to recognize and link an acronym to its expanded long form. In order to locate potential acronym definitions in text, a majority of approaches use pattern matching based on parenthetical forms (i.e., occurrences of acronyms within parentheses). Then, an optimal definition candidate string is selected and the candidate expanded form is analyzed with the aim of discovering the relation between the given acronym and the expanded candidate. Approaches usually are dictionary-based, rule-based and machine-learning-based. Chang and Hinrich [15] provided a detailed account of this area. Zahariev [138] proposed a universal

explanatory theory for acronym acquisition and disambiguation.

2.3.2 Mapping Named Entities

Most NER systems have done their job when the NEs in a raw text have been recognized and output in specified format. However, in practice these recognized NEs often need to be mapped to predefined concepts of a certain ontology, or be mapped to database entries of previously recognized entities. This is especially important for biomedical text mining.

For example, consider two hypothetical sentences: (1) "*SELL* regulates *p53 protein*", and (2) "*p53 protein* is regulated by *Selectin L*". The NER step recognizes *p53 protein* as a protein, and *SELL* and *Selectin L* as a gene respectively. If we want to extract relations based on the NER results, we will probably obtain two regulation relations onto *p53 protein* by two different genes. However, the *SELL* and the *Selectin L* actually refer to the same gene which controls cell adhesion during immune responses. Therefore, the two relations actually mean the same. Even worse, it has been found that *SELL* actually has at least 15 aliases used in biomedical literature [91], but the NER step does not recognize them as synonyms.

From an NLP perspective, NE mapping deals with NE variations and NE ambiguity. NE variation can be orthographic, morphological, lexical, structural variations, and abbreviations/acronyms, as we have mentioned in Section 1.3. These variations are often combined, to make things worse. The simplest approach to handle morphological variation is based on stemming. We can also recognize NE variants by looking up existing dictionaries with approximate string matching and edit distance techniques. Rule-base approaches and various machine learning approaches can be used to deal with the rest types of variations [133].

NE ambiguity arises when an NE has multiple meanings. This is typically the case when the same NE is used to denote several different views or aspects of a concept. For example, an occurrence of the CAT protein can be associated with several different protein entries in a protein database, depending on the species in question. Meanwhile, some NEs (especially acronyms) can have multiple independent meanings. For example, CAT can be a protein, animal, or medical device. Disambiguation methods typically rely on contextual analysis of a given occurrence, mainly using various machine learning strategies (e.g., classification) to decide which sense is correct in the given context.

2.3.3 Research in General NER

To gain a better understanding of the current status of biomedical NER research, it would be worthwhile to examine some approaches to the NER task in the general domain, and compare them with those used in biomedical NER. In general NER, the learning task is often formulated as a classification of single words. Every word is classified into the predefined name categories and the additional is-not-a-name class. This notation assumes that all adjacent items tagged with the same class-label form part of one single name (e.g., *Foreign Ministry* is an organisation and *George Bush* is a person).

The evaluation of NER is typically based on the exact match criteria, i.e., all partially tagged names count as mismatch (e.g., tagging only *Ministry* as ORG would be counted as complete failure). The testing and training data for learning are prepared from all candidate words for the given categories. An instance consists of the word, its features, and the correct category (label). Among the learning methods are HMM [7], Maximum Entropy [10], SVM [112], Boosting and Voted Perceptron [26].

Features for names can represent different aspects: mere surface form of a word, linguistic knowledge about the word (morphological, syntactic, semantic), and statistics of the occurrence of the word in a document collection. In general, the approaches to biomedical NER have evolved similarly to those in general domains, probably because many of the methods for biomedical NER are actually adopted from those in general domains.

The state-of-the-art performance in general NER is achieved by a hybrid method [10] and some have shown accuracy close to human performance [90], but the hybrid approaches to biomedical NER have not yet shown such performances, possibly due to the lack of high quality linguistic resources in biomedicine. By far biomedical NER has assumed the language is English, possibly due to the fact that the majority of biomedical literature is published in English, while general NER has begun to explore many other languages, as well as the problem of NE alignment among multiple languages. Most approaches to biomedical NER have focused on gene and protein names, while approaches to general NER have dealt with language-independent issues as well.

2.4 Chapter Summary and Future Directions

We have surveyed previous research in biomedical NER, its origin from NER in the newswire domain, and research issues particular to it. We have reviewed different approaches to the

task, and the performance evaluation of BioNER systems. We have also briefly discussed some research topics that are closely related to BioNER. We note that the reported performance of some BioNER systems have been good enough for practical applications.

Based on the above survey, we identify the following issues as future directions to advance the research and application of BioNER. They are **performance**, **annotated corpora**, and **non-English texts**.

First, **the ways to improve performance**. For an application-oriented technique like BioNER, performance is always the most important issue. However, the state-of-the-art performance of gene and protein NER systems achieves F-scores between 75 and 85 percent, as reported by different research groups using a variety of approaches on different data sets. Such a performance level is still substantially inferior to that of newswire NER systems, which typically are above 90% F-scores, and that of human beings (over 97%). Obviously, more research is needed to fill up the remarkable gap.

As discussed above, previous approaches tend to do feature engineering, introduce new machine learning frameworks, and/or integrate more domain knowledge as postprocessing to improve the performance. All of these approaches have proven to only achieve limited success. For example, the best performance evaluated on the BioNLP-2004 Shared Task data set is 72.94% F-score [14], just a little bit better than that of the Shared Task. This make us think of other ways to improve the performance. In this thesis, we will explore the following potential ways:

- to use larger annotated corpora for training a model;
- to recognize nested NEs as well as non-nested NEs;
- to detect NE boundaries more effectively (possibly with the help of NP chunking);

The second issue is about **annotated corpora**. This issue is important in that all statistical machine learning based methods require annotated corpora as the training data. However, it is very costly to create a good corpus such as the GENIA corpus for the biomedical domain. Besides, our own work has shown that even if a larger corpus were available, we would not be able to improve the performance as much as we expected. Given the result, we believe that there would be applications where BioNER is wanted without having an annotated corpus. In this thesis, we will study this issue.

The third issue is about **non-English biomedical texts**. Currently most BioNER research is done for English texts, while there are millions of non-English biomedical texts

which have been collected and are ready for bio-text mining but have been largely ignored. Non-English texts will likely present language processing challenges different from English, and thus can not be addressed by simply porting the techniques developed for English texts. In this thesis, we will study how to do BioNER in Chinese texts.

Besides, we also identify the following issues that are important but still open problems in BioNER. We will not address them in this thesis and will leave them as future work.

- Inferencing across Sentences: Given a text, most approaches so far recognize the NEs sentence by sentence, without considering clues across sentences. This could be one reason for the low reported performance. Intuitively, information across sentences can be very useful for the BioNER task, especially when one NE appears in more than one sentence of the same text. For example, the NE might be easier to identify in one sentence than in another, and thus if it had already been identified in the easier one, it is not necessary to solve it in the harder one.

- From abstract to full text: Most previous works have been focusing on Medline abstracts, probably due to the easy access via the PubMed service. For the ultimate purpose of text mining in biomedical literature, there is also an obvious need to mine the full text. Intuitively, there would be remarkable linguistic differences between an abstract and its full text. A simple migration from the former to the latter would lead to degraded performance. Therefore, it is necessary as well as non-trivial to investigate the task in the full text environment.

Chapter 3

BioNER by Machine Learning

Supervised machine learning methods have been widely used for NER and other NLP tasks. Differing in the way of learning, these methods all require a set of labeled data for training and attempt to learn a model from the data. Some popular methods that have been exploited for NER are Hidden Markov models (HMM), Maximum Entropy (ME), Support Vector Machines (SVM), and more recently Conditional Random Fields (CRF). These methods have also been used in other NLP tasks, e.g., POS tagging and syntactic parsing.

When applying supervised learning to a classification problem, a rule-of-thumb is that the more labeled data used for training, the better the classification performance. Therefore, it seems natural to use a larger annotated corpus to train an NER model as a straightforward way of improving the NER performance (perhaps also a feasible way if not considering the cost of annotation). In this chapter, we evaluate this idea by empirically studying the relationship between the size of labeled data and the performance of the learned NER model, which we call *learning curve* thereafter. In particular, we use SVM and CRF as the supervised learning methods to evaluate the relationship on the BioNLP-2004 Shared Task data set. Our results show that the learning curves will quickly become flat. This suggests that increasing training data size substantially would not lead to significant improvement in performance.

The rest of the chapter is organized as follows. In Section 3.1, we briefly describe how the NER task can be modeled as a classification problem so as to be handled by supervised learning methods. In Section 3.2 and 3.3, we introduce the principles of SVM and CRF respectively. We present how feature selection is done for supervised learning in Section 3.4. We evaluate the learning curves of SVM and CRF in Section 3.5.

3.1 Modeling NER as a Classification Problem

To apply supervised learning methods to our NER task, here we formulate it as a classification problem. Since an NE usually does not span across two sentences, we can reduce the problem of recognizing NEs in the article into one of recognizing NEs in each sentence. Given a sentence in a biomedical article, we assume that it has been properly tokenized. Here we do not address the problem of anaphora resolution [79], which still remains a hard problem in NLP. In other words, we only recognize those NEs that are explicitly expressed or mentioned in the sentence. Whenever encountering an anaphoric expression, we simply ignore it.

Under these assumptions, a sentence S that has n tokens in sequence can be denoted as $S =< t_1, t_2, t_3, \ldots, t_n >$. Suppose we want to extract entities that belong to k biomedical categories (i.e., k types of entities), $C_1, C_2, \ldots, C_k$. Thus the problem of BioNER actually becomes to find all proper sub-sequences of S that identify as entities of the k categories. That is, each token of the sentence is identified either as one of the k category names or as not of any entity (which can be thought of as a null entity). Therefore, the task of finding which category a token is supposed to associate with can be thought as a classification problem which can be solved by any supervised learning methods, e.g., SVM or CRF.

There are many ways to represent the association, though. And among them, the IOB2 [99] notation is probably the one that is most commonly used in the NER community. In this notation, each word (or token) is associated with one of the three tags B, I, O, where a B tag is given for every token which is at the beginning of an NE, an I tag is given for all the following token in that NE, and an O tag is given for all the rest tokens that are not in any NEs. An example sentence labelled using the IOB2 notation is given in Table 3.1. Note that there are other ways of tagging the NEs, readers are referred to [108] for further references, which experimentally compares different tagging methods for text chunking tasks (NER can also be seen as a text chunking task).

3.2 Support Vector Machines

SVM is based on the structural risk minimization principle from statistical learning theory introduced by V. Vapnik [124]. It learns a linear decision hyperplane to separate positive and negative examples by maximizing the distance to the closest data points from both classes.

Token	Label
IL-2	B-DNA
gene	I-DNA
expression	O
and	O
NF-kappa	B-protein
B	I-protein
activation	O
through	O
CD28	B-protein
requires	O
reactive	O
oxygen	O
production	O
by	O
5-lipoxygenase	B-protein
.	O

Table 3.1: An example sentence labelled using the IOB2 notation.

In its basic form, the SVM learns the linear decision hyperplane $h(\vec{x}) = sign\{\vec{w} \cdot \vec{x} + b\}$, described by a weight vector $\vec{w}$ and a threshold b. The input is a set of n training examples $S_n = ((\vec{x_1}, y_1), \ldots, (\vec{x_n}, y_n)) \in R^N, y_i \in \{+1, -1\}$.

If S_n is linearly separable, the SVM finds the hyperplane with maximum Euclidean distance to the closest training examples. This distance is called the margin δ, as depicted in Figure 3.1. If S_n is not linearly separable, the SVM uses either a kernel function to map the original data into another dimensional space where the data points are linearly separable, or tries to find a hyperplane with a soft margin the allows some degree of training error in order to obtain a large margin. Computing the hyperplane is equivalent to solving an optimization problem. The resulting model is actually the decision hyperplane, expressed by $\vec{w}$ and b.

The problem of finding the hyperplane can be stated as the following optimization problem:

Minimize: $\frac{1}{2}\vec{w}^T\vec{w}$,

Subject to: $y_i(\vec{w}^T \cdot x_i + b) \geq 1, i = 1, 2, ..., n$,

where $\vec{w}^T$ is transpose of vector $\vec{w}$.

To deal with the cases where there may be no separating hyperplane due to noisy labels

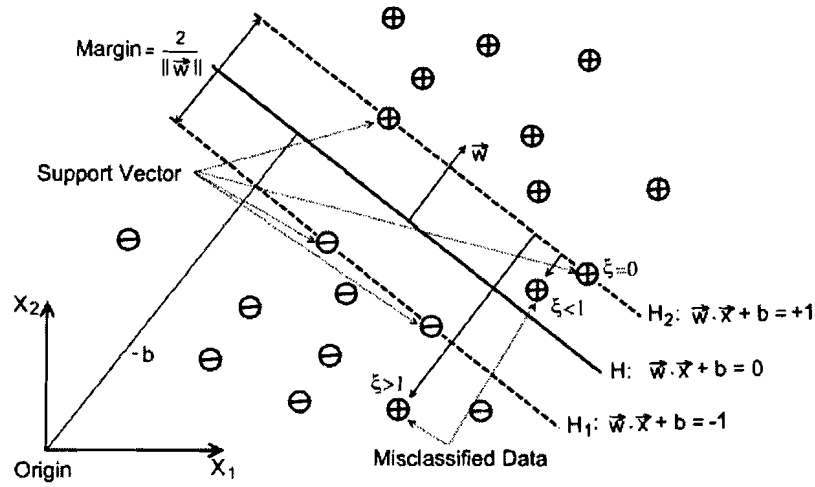

Figure 3.1: A linear SVM for a two dimensional training set

of both positive and negative training examples, the following soft margin SVM can be used [51]:

Minimize: $\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i$,

Subject to: $y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, 2, ..., n$,

where $C \geq 0$ is a parameter that controls the amount of training errors allowed, and ξ_i is a slack variable which measure the degree of misclassification of the datum x_i.

Generally speaking, the SVMs have the following advantages over other machine learning algorithms (e.g., Naive Bayes and Decision Trees). First, they have the ability to handle high dimension feature space (e.g., hundreds of thousands or even millions of features). This is very useful for NER tasks where a large number of features are often necessary to achieve high performance.

Second, they have the ability to handle learning problems where there are very few irrelevant features. This is the case for NER tasks: most features are relevant but not very strong in distinguishing the NEs. To make a good classifier, one has to combine many such features, although the combination might result in some degree of redundancy. SVMs have proven particularly good at learning from such features [51].

Third, theoretical and empirical study has shown that SVMs are good at learning from

a sparse data space [51]. This is also desirable for NER tasks, where annotated data are available in limited amounts.

Fourth, the SVMs in principle are able to maximize their generalizability in the presence of noisy data [51]. This is particularly helpful for NER tasks, where the data sets often contain many mislabelled noisy data.

The above four properties make SVMs desirable for NER, however, it is often not very straightforward to apply SVMs to NER tasks due to two reasons. First, NER tasks often involve multi-classes[1], whereas SVMs are in principle designed for binary classification problems. Extra adaptive processing is required to allow SVMs to handle multi-classes. Second, it is sometimes more convenient to model NER as a sequence labeling problem, in order to better reflect the sequential properties of the language. Although the sequence labeling problem can be further reduced to a classification problem and solved by general supervised learning algorithms such as SVMs, it would be ideal to have an algorithm naturally designed for the sequence labeling problem. Conditional Random Fields (CRFs), to be introduced in the next section, are just one such algorithm.

3.3 Conditional Random Fields

CRFs belong to a type of discriminative probabilistic model that is useful for the labeling sequential data, such as natural language text or biological sequences. In principle, a CRF is an undirected graphical model in which each vertex represents a random variable whose distribution is to be inferred, and each edge represents a dependency between two random variables. The distribution of each random variable Y in the graph is conditioned on an input sequence X. In practice, the Y can be simply interpreted as "labels" for each element in the input sequence X. A model can be trained by learning the conditional distributions between the Ys and feature functions from training data. The obtained model can be used to determine the most likely assignment of a new Y given a new input sequence X and the trained conditional distributions.

CRFs are undirected statistical graphical models, a special case of which is a linear chain that corresponds to a conditionally trained finite-state machine. Such models are well suited

[1] With the IOB2 notation described earlier, if the task involves k types of NEs, there will be $2k + 1$ class labels. In the example sentence shown in Table 3.1, there are two types of NEs: DNA and protein. So five class labels are to be used, namely, *B-DNA*, *I-DNA*, *B-protein*, *I-protein*, and *O*.

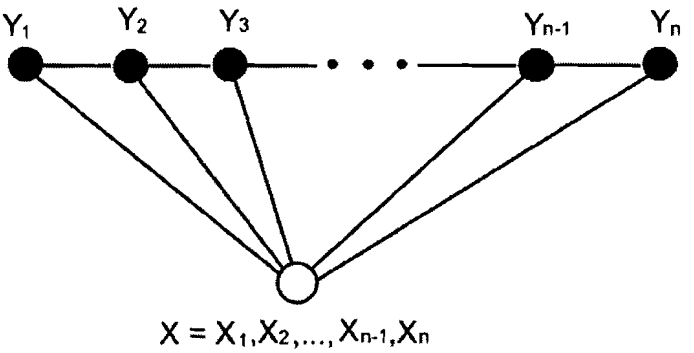

Figure 3.2: A linear chain CRF

to sequence analysis, and CRFs in particular have been shown to be useful in part-of-speech tagging [63], shallow parsing [105], and named entity recognition for newswire data [73].

Biomedical named entity recognition can be thought of as a sequence tagging problem: each word is a token in a sequence to be assigned a label (e.g., *B-protein* for the beginning of a protein entity, *I-DNA* for inside a DNA entity, *O* for outside any entities). Let $o = \langle o_1, o_2, ..., o_n \rangle$ be an sequence of observed words of length n. Let S be a set of states in a finite state machine, each corresponding to a label $l \in L$ (e.g., *B-protein*, *I-DNA*, etc.). Let $s = \langle s_1, s_2, ..., s_n \rangle$ be the sequence of states in S that correspond to the labels assigned to words in the input sequence o. Linear chain CRFs define the conditional probability of a state sequence given an input sequence to be:

$$P(s|o) = \frac{1}{Z_o} exp(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(s_{i-1}, s_i, o, i)) \tag{3.1}$$

where Z_o is a normalization factor of all state sequences, $f_j(s_{i-1}, s_i, o, i)$ is one of m functions that describes a feature, and λ_j is a learned weight for each such feature function.

Usually a first order Markov independence assumption with binary feature functions is used to construct the features. For example, a feature may have a value of 0 in most cases, and value 1 when s_{i-1} is in a state with the label "O", and s_i in a state with the label "B-Protein", and f_j is the feature function "Word=ATPase $\in o$ at position i" in the sequence. Other feature functions that could have the value 1 along this transition are "Capitalized", "MixedCase", and "Suffix is ase".

Intuitively, the learned feature weight λ_j for each feature f_j should be positive for features that are correlated with the target label, negative for features that are anti-correlated with the label, and near zero for relatively uninformative features. These weights are set to maximize the conditional log likelihood of labeled sequences in a training set $D = \langle o, l \rangle(1), ..., \langle o, l \rangle(n)$:

$$LL(D) = \sum_{i=1}^{n} log(P(l_{(i)}|o_{(i)}) - \sum_{j=1}^{m} \frac{\lambda_j^2}{2\sigma^2}. \quad (3.2)$$

When the training state sequences are fully labeled and unambiguous, the objective function is convex, thus the model is guaranteed to find the optimal weight settings in terms of $LL(D)$. Once these settings are found, the labeling for an new, unlabeled sequence can be done using a modified Viterbi algorithm. CRFs are presented in more complete detail by [63] and [111].

Compared to other sequential analysis algorithms such as HMMs, CRFs are conceptually more complicated, but do not suffer from strong Markov assumptions on the input and output sequence distributions of HMMs. Similar to SVMs, CRFs also have the potential to incorporate large number of features. These two properties make CRFs particularly suitable for our NER task.

3.4 Feature Selection for BioNER

To apply supervised learning methods, raw data (e.g., words in sentences) must be featurized, that is, be expressed in such a way that they can be distinguished from each other as much as possible. Previous studies have found the following types of features are generally useful for BioNER.

- **Orthographical features:** they are used to capture information about capitalization, digitalization, special characters and their combination about a word or a token. For example, many protein names contain capital letters and digits.
- **Morphological features:** these refer to the roots, prefixes and suffixes of words. This is based on the observations that many biomedical named entity names contain roots, prefixes and suffixes of biomedical meanings. For example, *haemo* means *blood*, *aero* implies *oxygen*.

	in an NE	not in an NE	subtotal
having the feature	O_{11}	O_{12}	$O_{11} + O_{12}$
not having the feature	O_{21}	O_{22}	$O_{21} + O_{22}$
subtotal	$O_{11} + O_{21}$	$O_{12} + O_{22}$	$N = O_{11} + O_{12} + O_{21} + O_{22}$

Table 3.2: Contingency table of feature occurrences for chi-square test of independence

- **Part-of-Speech features:** they have been shown useful in determining boundaries of biomedical NEs. POS tagging tools can be used to obtain this information, e.g., GENIA tagger [2].

- **Frequent Surface Words:** many words or phrases are more (or less) often used in biomedical names, and therefore might be helpful in determining whether a word is in an NE or not. For example, *factor, receptor, protein, kinase* are often seen in protein names, while *gene, promoter, site, enhancer* are often used in DNA names.

- **Contextual Features:** words sometimes appear together. Thus information about surrounding words might help to make decision on the current word. That is, features of contextual words can also be features of the current words.

Not all features are useful for the classification, and sometimes it is not obvious to see whether a feature is useful or not. While too few features might not be sufficient to learn a good classifier, using too many features may not necessarily classify better. Also using too many features would incur too much computation. Therefore, feature selection is often used to filter out features that are not of good distinguishing power.

Among many feature selection methods, the chi-square test of independence (also called Pearson's chi-square test)[3] has proven very effective. What it does is test the association between two categorical variables. In our case, we want to test whether having a feature is associated with having an NE label. In order to use the chi-square test, we organize related information into a contingency table of two categorical variables as shown in Table 3.2, where O_{ij} refers to occurrences of the corresponding items.

Here the null hypotheses is that there is no association between the two variables (i.e., they are independent from each other), while the alternate hypotheses is that there is an association between the two variables.

[2] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/

[3] http://en.wikipedia.org/wiki/Pearson's_chi-square_test

Note that in order to use the chi-square test, we assume that (1) none of the expected values may be less than 1, and (2) no more than 20% of the expected values may be less than 5. In cases where these two assumptions are violated, we may speculate that the feature in consideration might not be useful for the classification, because they are not even frequently seen in the data.

The general formula for expected cell frequencies is:

$$E_{ij} = \frac{T_i \times T_j}{N} \tag{3.3}$$

where E_{ij} is the expected frequency for the cell in the ith row and the jth column, T_i is the total number of item jth column, and N is the total number of items in the whole table. The formula for chi-square test for independence is:

$$\chi^2 = \sum_{i,j=1}^{2} \frac{(E-O)^2}{E} \tag{3.4}$$

Having the chi-square value, we can refer to a chi-square table to see if it is significant. For the contingency table, the degree of freedom is equal to $(R-1)(C-1)$, where R is the number of rows and C is the number of columns. In our case, $R = 2$ and $C = 2$ (because each variable takes only two possible categorial values), thus $df = 1$. From the chi-square table, we can find the critical values for the desired significance levels. If the chi-square value is greater than the critical value, we accept the null hypothesis, i.e., that the two variables are independent; otherwise, we reject it. For example, the critical value for 0.05 significance level is 3.8415, which means that we have about 95% chances to observe the numbers in the contingency table, provided that the null hypothesis is correct. By the way, the critical values for significance level 0.01 and 0.005, corresponding to 99% and 99.5% chances, are 6.6349 and 7.8794 respectively.

We applied the chi-square independence test on a set of orthographical features and obtained 30 features whose chi-square test results are significant (see Table 3.3).

3.5 Experiments

In this section, we ran experiments on a real BioNER data set, including a training set and an independent testing set, to obtain the learning curves of SVM and CRF. Given the training set, we take a random sample of it to train an SVM and a CRF model. We then

Feature f for a word	# of words with f and in an entity	# of words with f but not in any entity	# of words not with f but in an entity	# of words not with f and not in an entity	χ^2 value
contains '-'	26411	3076	144350	317104	41509.24
contains '('	2091	5007	168670	315173	89.97
contains ')'	2096	5048	168665	315132	94.68
contains '/'	1928	798	168833	319382	1561.25
contains '+'	683	391	170078	319789	393.88
contains ','	847	17178	169914	303002	7465.33
contains '.'	500	19585	170261	300595	9627.30
contains ';'	75	491	170686	319689	115.81
contains ':'	32	720	170729	319460	309.41
contains '*'	33	1	170728	320179	58.13
contains '['	29	149	170732	320031	26.84
contains ']'	23	152	170738	320028	36.14
1st char capital	48954	17620	121807	302560	50983.54
1st char digit	3201	3876	167560	316304	345.58
1st char lowercase	111111	250010	59650	70170	9700.13
last char capital	25125	2220	145636	317960	41620.01
last char digit	18029	4035	152732	316145	22430.02
last char lowercase	121812	265164	48949	55016	8796.39
all chars capital	18529	1763	152232	318417	29818.27
all chars digit	1525	2547	169236	317633	12.89
all chars lowercase	100650	247898	70111	72282	18475.0
capital-in-middle	32236	1428	138525	318752	59238.61
lowercase-in-middle	125501	194031	45260	126149	8148.44
digit-in-middle	9291	1817	161470	318363	11960.75
5≥word_len≥1	75685	224314	95076	95866	31037.81
10≥word_len>5	69336	79732	101425	240448	12985.72
15≥word_len>10	21709	15409	149052	304771	9946.00
20≥word_len>15	3294	639	167467	319541	4191.42
word_len>20	737	86	170024	320094	1090.08

Table 3.3: The results of chi-square independence test on orthographic features

		protein	DNA	RNA	cell type	cell line	All
Training Data	# of NEs	30269	9533	951	6718	3830	51301
	# of tokens	55117	25307	2481	15466	11217	109588
Testing Data	# of NEs	5067	1056	118	1921	500	8662
	# of tokens	9841	2845	305	4912	1489	19392

Table 3.4: Entity and token distributions of the BioNLP-2004 data set

test the models against the standalone testing set to see the performance. By varying the sample sizes, we get a group of performance numbers from which we can draw the learning curve. The learning curve thus show how the performance of each learning method changes with the size of training data. Our purpose here is to illustrate the learning ability as well as the limitations of these corpus-based learning methods for the BioNER task.

We used SVM-light[4] and CRF++[5] as the implementations of SVM and CRF respectively. Both are freely available for research purposes, and have been previously used in NER research (e.g., [39] and [55]).

3.5.1 BioNLP-2004 Data Set

The data set we used was the BioNLP-2004 data set. As mentioned in the introduction chapter, this data set was used for the BioNLP-2004 Shared Task on BioNER. The training data contains 2000 annotated Medline abstracts (18546 sentences or 492551 words in total), while the testing set has 404 annotated Medline abstracts (3856 sentences or 101039 words in total). The number of distinct words is 22056 for the training data and 9623 for the testing data. Five types of entities were annotated, namely, *protein*, *DNA*, *RNA*, *cell line*, and *cell type*. The numbers of occurrences of each entity type and the number of token occurrences involved in each entity type are shown in Table 3.4. All the entities are annotated using the IOB2 notation, where B stands for the beginning of an NE, I for inside the NE, and O for being outside any NE. Both the training data and the testing data have been preprocessed into the format as shown in Table 3.1. Note that in case of nested entities, only the outmost entities (i.e., the longest tag sequence) are considered for the shared task. We will discuss more about the nested entities in next chapter.

[4]available at http://svmlight.joachims.org/

[5]http://crfpp.sourceforge.net/

Relative Position	Token	Feature (capitalized?)	Feature (prefix='gen'?)	Feature (suffix='tion'?)	Feature (...)	Label
-4	IL-2	1	0	0	...	B-DNA
-3	gene	0	1	0	...	I-DNA
-2	expression	0	0	0	...	O
-1	and	0	0	0	...	O
0	NF-kappa	1	0	0	...	B-protein
+1	B	1	0	0	...	I-protein
+2	activation	0	0	1	...	O
+3	through	0	0	0	...	O
+4	CD28	1	0	0	...	B-protein
+5	requires	0	0	0	...	O
+6	reactive	0	0	0	...	O
+7	oxygen	0	0	0	...	O
+8	production	0	0	1	...	O
+9	by	0	0	0	...	O
+10	5-lipoxygenase	0	0	0	...	B-protein
+11	.	0	0	0	...	O

Table 3.5: An example of the features and the context window used for SVM learning. The example context window ranges from position -3 to position +3.

3.5.2 Features for SVM Learning

For SVM learning, we used all the types of features mentioned in the previous section. In particular, the orthographical features were the 30 features selected by using the Chi-Square test. For the morphological features, we used 10000 frequent prefixes and 10000 frequent suffixes automatically generated from the training set, which are 3-5 characters long and appear more than 3 times. The Part-of-Speech tags were provided by the GENIA Tagger, for a total of 45 distinct POS tags. The frequent token features are made by selecting the top 4925 distinct tokens that appear more than 3 times in the training set.

In order to capture context clues of a token, we incorporated the features of its previous 3 tokens and its next 3 tokens (i.e., the context window size is 7). An example of some features for the tokens in the example sentence given in Table 3.1 is shown in Table 3.5. The first column of Table 3.5 is the position of a token relative to the current token, which is "NF-kappa" in this case. The second column is the token, while the last column is its entity label. The third column shows the value of each token for the specified feature ("is the token capitalized?"). If yes, the value is 1; otherwise, it is zero. The values in the remaining columns are set similarly.

We then converted the features into SVM-light's vector data format. Note that here each specific feature corresponds to one dimension in the resulting vector. For example, the

30 orthographical features as shown in Table 3.3 will correspond to 30 dimensions in the resulting vector. Without considering the context, each token corresponds to a row vector with 25000 columns (i.e., dimensions). With the features of tokens in the context, for each token we generate a row vector, whose dimension number is 175000, and whose values are binary (i.e., 1 for having a specific feature while 0 otherwise).

Note that the dimension number is quite high, and the generated vectors contain many zeros, implying the sparsity of the corresponding feature space. As discussed earlier, SVMs have been shown relatively good at handling the high dimensionality and data sparsity, compared to other machine learning algorithms, e.g., Decision Trees and Naive Bayes [51]. In our experiments, they were amenable to SVM-light, too.

Also note that we have five types of entities annotated with the IOB2 notation. In total, we have 11 class labels, namely, *B-protein*, *I-protein*, *B-DNA*, *I-DNA*, *B-RNA*, *I-RNA*, *B-cell_type*, *I-cell_type*, *B-cell_line*, *I-cell_line*, and *O*. We converted the multi-class problem into a combination of 11 binary classification problems. Basically, we built a binary classifier for each class label. When predicting the class label for a token, we ran all the 11 classifiers and got 11 predicted values, from which we chose the one having the largest absolute value to be the predicted class label. Though the combination looks simple, it works well for the task.

3.5.3 Features for CRF Learning

For CRF learning, we used the feature template provided by the CRF++ package and let the package automatically generate the features, rather than explicitly making a dimension for each individual feature as we did for SVM learning. For this purpose, we prepared five lists from the training set. The first and the second list contain frequent prefixes and suffixes, the same as used for SVM learning. The third list contains about 3000 frequent tokens, all appearing **inside** the entities at least 4 times. The fourth list contains about 5000 frequent tokens, all appearing **outside** any entities for at least four times.

We then prepared the data into the format as shown in Table 3.6. In Table 3.6, the first column is the token, the second column is the POS tag of the token, while the last column is the entity tag. Each of the remaining columns corresponds to one of the lists, and is set as this: if the word matches any item of the list, it is set to 1; otherwise, it is zero. For example, the fifth column corresponds to the list of frequent tokens inside entities. If the current word matches any of the tokens, it is set to 1; otherwise, it is set to zero. Note that

the char	POS tag?	has a freq prefixes?	has a freq suffixes?	is it freq in NEs?	is it freq out NEs?	NE tag
IL-2	NN	1	1	1	1	B-DNA
gene	NN	1	1	1	1	I-DNA
expression	NN	1	1	1	1	O
and	CC	1	1	1	1	O
NF-kappa	NN	1	1	1	1	B-protein
B	NN	0	0	1	1	I-protein
activation	NN	1	1	1	1	O
through	IN	1	1	0	1	O
CD28	NN	1	1	1	0	B-protein
requires	VBZ	1	1	0	1	O
reactive	JJ	1	1	1	1	O
oxygen	NN	1	1	0	1	O
production	NN	1	1	0	1	O
by	IN	0	0	0	1	O
5-lipoxygenase	NN	1	1	1	0	B-protein
.	.	0	0	1	1	O

Table 3.6: Example data prepared in CRF++ format.

here the features are used *collectively* rather than *individually* as in SVM learning. Also note that we did not use the orthographical features, as they are not distinguishing when used collectively.

For each item in each column of Table 3.6, we consider the previous two items, the current item, and the next two items. That is, the context window size is 5, from -2 to +2. The feature template for one column is shown in Table 3.7, and it is basically the same for all the columns except the NE tag column. Note that all the features generated by the template are called "Unigram" features in the CRF++ package, which actually correspond to the state features of the CRF framework. The transition features of the CRF framework are automatically generated in the CRF++ package by considering the bigrams of NE tags.

We now can estimate how many features are actually generated by the template. Assume that there are N distinct items in one column of the data, and there are K distinct NE tags used in the problem, then the CRF++ package will generate KN features for each item T_i, KN^2 features for each combination of two items T_i and T_j, and KN^3 features for each combination of three items T_i, T_j, and T_k. That is to say, we will have $(KN+KN^2+KN^3)$ state features for the column. As we only consider the bigram of NE tags for the transition features, we will have $K^2(KN+KN^2+KN^3)$ such features. So the total number of features

template	data item to be used as feature
U00:%x[-2,0]	T_{-2}
U01:%x[-1,0]	T_{-1}
U02:%x[0,0]	T_0
U03:%x[1,0]	T_{+1}
U04:%x[2,0]	T_{+2}
U05:%x[-2,0]/%x[-1,0]	$T_{-2}T_{-1}$
U06:%x[-1,0]/%x[0,0]	$T_{-1}T_0$
U07:%x[0,0]/%x[1,0]	T_0T_{+1}
U08:%x[1,0]/%x[2,0]	$T_{+1}T_{+2}$
U09:%x[-1,0]/%x[0,0]/%x[1,0]	$T_{-1}T_0T_{+1}$

Table 3.7: The feature template used by CRF++ package for one column of the data shown in Table 3.6.

is order $O(CK^3N^3)$, where C is the total number of columns (as those in Table 3.6) to be used for generating features. In our data set, we have $K = 11$ and $N = 22,000$ for the total number of unique words in the training data. Therefore, the total number of features considered by the CRF++ package would be of order 10^{15}. In fact, lots of these features do not appear in the training data and thus can be treated as zero when we actually compute them. In our experiments, the CRF++ package actually generated 11,510,928 features (and kept them in memory) using the above template.

As our purpose was not to look for the features for the best performance, we did not do further feature engineering, though it is possible as well as interesting to do it. We leave it for future work.

3.5.4 Baseline System and Performance Measure

For comparison purposes, we built a baseline system which takes a dictionary-based approach. Given a sentence and a dictionary of known entity names, it does a longest match through the sentence against each entry of the dictionary. For example, assume a small dictionary contains two proteins *NF-kappa B* and *IL-2*, and one DNA *IL-5 gene*. Given the example sentence shown in Table 3.1, after applying the longest matching against the dictionary, we would have the token labels as shown in Table 3.8.

From Table 3.8, we can see that the baseline system can only identify the token *NF-kappa* and *B* as a protein, because it exactly matches the dictionary entry *NF kappa B*. It

Token	True Label	Dictionary Label
IL-2	B-DNA	B-protein
gene	I-DNA	O
expression	O	O
and	O	O
NF-kappa	B-protein	B-protein
B	I-protein	I-protein
activation	O	O
through	O	O
CD28	B-protein	O
requires	O	O
reactive	O	O
oxygen	O	O
production	O	O
by	O	O
5-lipoxygenase	B-protein	O
.	O	O

Table 3.8: An example of dictionary matching.

misses the other two proteins *CD28* and *5-lipoxygenase*, because they are not contained in the dictionary. Besides, the dictionary matching also misses the DNA *IL-2 gene*. Instead it mislabels the token *IL-2* as a protein, because it uses the longest match. This example shows that the performance of dictionary-based approaches will be affected by the coverage of the dictionary and the matching method in use.

Given a matching method, typically, the more entries the dictionary contains, the more likely a match is found, and thus the better the recognition performance. In our experiments, the dictionary is collected from training data and thus independent of the testing data. In total, the dictionary has 18582 distinct NEs, including 8640 protein, 5278 DNAs, 461 RNAs, 2063 cell types, and 2150 cell lines.

We used precision P, recall R, and F-Score F to measure NER performance. P is the ratio of the number of correctly found NE chunks to that of found NE chunks, while R is the ratio of the number of correctly found NE chunks to that of true NE chunks. F is defined as $(2PR)/(P+R)$. A found NE chunk is considered correct only if it exactly matches the boundary of the true NE chunk. Taking the dictionary matching in Table 3.8 as an example, the baseline system predicts two proteins *IL-2* and *NF kappa B*, but only one (i.e., *NF kappa B*) is correct. So the precision for protein is 50.0%. Originally, there are

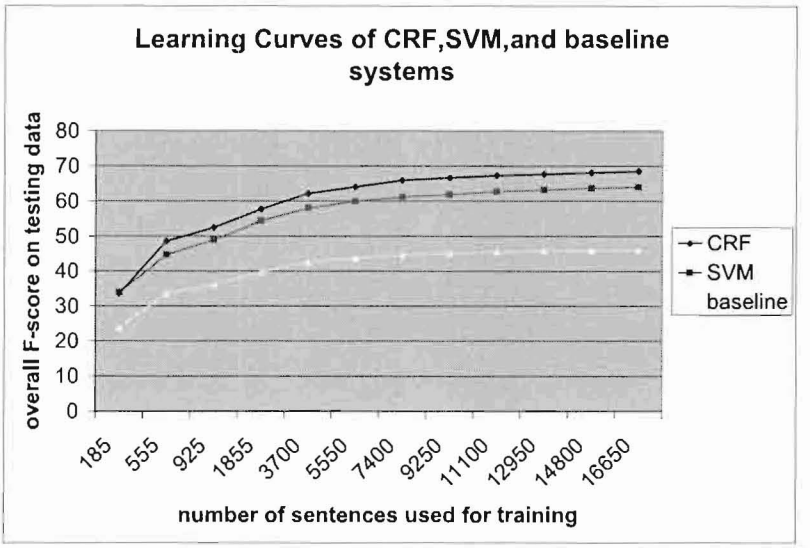

Figure 3.3: Learning Curves of SVM, CRF and the baseline systems. The x-axis is the number of sentences used for training, while the y-axis is the F-score averaged over 5 random runs.

three proteins in the sentence, but only one is correctly identified. So the recall for protein is 33.3%. Similarly, the precision for DNA is 0, while the recall for DNA is 0.

3.5.5 The Resulting Learning Curves

From the BioNLP-2004 training set, we selected a random sample of size x to train the SVM and the CRF. The trained SVM and CRF model was then evaluated on the BioNLP-2004 testing set. We let x be 1%, 3%, 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, and 90% of the total number of sentences (i.e., 18546 sentences). For each size, we ran experiments for 5 times and reported averaged results. The learning curves of SVM, CRF and the baseline system are shown in Figure 3.3. The exact-match performances of the three systems using all of the training data are shown in Table 3.9, which also gives the precision, recall, and F-score for each entity type.

From Figure 3.3, we can see that the SVM and CRF outperform the baseline system on every size of the training data, in terms of the F-Score. This result demonstrates the learning ability of machine learning techniques — they can learn to generalize from the labeled examples provided in the training set. From the figure, we can also see the trend

Entity Type	CRF (P / R / F)	SVM (P / R / F)	Baseline (P / R / F)
Protein	69.03% / 69.80% / 69.41	71.21% / 62.04% / 66.31	58.24% / 47.03% / 52.04
DNA	72.12% / 63.45% / 67.51	53.69% / 67.10% / 59.65	33.33% / 22.81% / 27.09
RNA	63.87% / 64.41% / 64.14	55.93% / 63.46% / 59.46	32.20% / 16.96% / 22.22
cell_type	80.59% / 62.88% / 70.64	57.47% / 80.00% / 66.89	55.44% / 41.86% / 47.70
cell_line	56.60% / 54.00% / 55.27	46.20% / 56.48% / 50.83	32.20% / 29.65% / 30.87
[-ALL-]	70.70% / 66.51% / 68.54	64.37% / 65.19% / 64.78	52.72% / 41.04% / 46.15

Table 3.9: The Performance of CRF, SVM, and the baseline system using the 100% training data

of how the performance changes with the size of the training data: it increases as the size does. That is, the more data used for training, the better the learned model.

However, the learning curves become flat at the end, which means that the increase in performance would be very little even if the size of training data increases a lot. This observation is important in practice. It suggests that increasing training data size is not an ideal way to improve the BioNER performance. Previously people often notice the fact that it would be extremely expensive in terms of human labor to build a large annotated corpus. Our learning curves further suggest that even if the annotation were possible, we would not expect a big improvement in the performance.

3.5.6 Modeling the Learning Curves

In order to obtain a more accurate relation between the training data size and the BioNER performance, we model the learning curves by curve fitting. A previous work [43] on modeling the learning curves of machine learning algorithms suggests that the learning curves could be well fit by the power law model ($y = a \times x^b$, where $a > 0$ and $b > 0$). We thus use it to model the above CRF and SVM learning curves. The curve fitting was done using the Curve Expert[6] software. The resulting model for CRF is $y = 21.7790x^{0.1219}$, while it is $y = 21.2733x^{0.1168}$ for SVM. Here y is the F-score, and x is the number of sentences used for training.

With the above models, we can estimate how many sentences are needed for training in order to achieve a desired F-score, that is, $x = \exp(\frac{1}{b}log(\frac{y}{a}))$. The state-of-the-art performance on the BioNLP data set is about 73% F-score. In order to achieve a 90% F-score, we probably need about 113,500 annotated sentences for training the CRF model, and about

[6]http://www.ebicom.net/ dhyams/cmain.htm

230,700 annotated sentences for training the SVM model. Note that it took the GENIA team about three to four years to annotate the 18546 sentences in the GENIA corpus. Assuming the same annotating speed by the same team, it would take them about 18 years to finish the amount needed for the CRF model, and about 36 years for the SVM model, both of which are not feasible.

Even if the annotation were feasible, it would not be feasible to train an SVM or CRF on such a large corpus. The time complexity of the SVM-light package, currently one of the best SVM implementations, is super-linear, i.e., $O(N^c)$, in terms of the training data size N, where $c > 1$ is a domain-specific or corpus-specific constant (usually between 1.2 and 1.5 as suggested in [129]). The time complexity of CRF is more difficult to estimate, but typically super-linear in terms of the number of training sequences (or sentences in our cases) [23][7]. Given the time complexity, it can be imagined that the computation would quickly become intractable if such a large number of sentences were used for training SVM or CRF.

3.5.7 Discussion and Previous Work

The state-of-the-art performance on the BioNLP-2004 data set is still not satisfactory for BioNER, varying from 70%-80% F-score depending on the targeted data sets. Two reasons might account for it. One is the highly ambiguous nature of biomedical names, and the other is the inconsistent annotation existing in the training and testing data. The former can be seen in Table 3.10, while the latter in Table 3.11. Table 3.10 lists the frequency distribution of NE tags of some common biomedical words in the training data, from which we can see that these words are highly ambiguous and their actual meanings would have to be decided by the context. Table 3.11 compares the frequencies of a list of words that are sometimes labelled as single-word entities but some other times as non-entities. We are not suggesting that all these inconsistencies are annotation errors — we are not biologists anyway thus are not in the right position to judge, but it seems hard to attribute the inconsistent labels purely to the ambiguity of biomedical names, especially when they can stand alone as an entity.

[7]Typically CRF training requires many hundreds or thousands of iterations, each of which involves calculating of the log-likelihood and its derivative. The time complexity of a single iteration is $O(L^2NTF)$ where L is the number of labels, N is the number of sequences, T is the average length of the sequences, and F is the average number of activated features of each labelled clique, which often increases with N. In our

token	total freq	B-P freq	I-P freq	B-D freq	I-D freq	B-R freq	I-R freq	B-CT freq	I-CT freq	B-CL freq	I-CL freq	O freq
cell	2869	42	187	11	38	0	2	4	215	70	835	1465
gene	2307	4	72	4	1381	0	4	0	0	0	0	842
human	2298	197	29	345	26	6	0	639	141	254	106	555
protein	1894	208	1090	9	28	0	2	0	0	0	2	555
binding	1788	20	232	51	258	0	1	0	0	0	0	1226
receptor	1360	28	857	1	124	0	38	0	1	0	7	304
alpha	909	44	613	13	131	1	22	1	2	1	7	74
IL-2	820	562	13	171	30	19	1	1	1	3	3	16
T-cell	513	60	31	17	13	0	0	28	18	30	86	230
erythroid	299	24	3	10	2	3	1	73	30	11	16	126
lymphocyte	213	12	13	3	4	0	0	21	36	1	16	107
c-fos	184	15	3	99	11	30	3	0	0	0	0	23
c-jun	170	19	1	88	4	33	9	0	0	0	0	16
GATA-1	168	123	7	20	6	5	1	0	0	3	1	2
CD4	144	49	4	26	2	0	0	26	14	11	9	3

Table 3.10: The frequency distribution of some common biomedical words' NE labels over the 11 entity tags in the BioNLP-2004 training data set.

In order to achieve better performance, recent efforts have been made mainly in three directions: to use better machine learning algorithms, to use more features, and/or to incorporate more domain knowledge.

In the first direction, people have tried various algorithms, from SVM [65], HMM [140], and Maximum Entropy [66] to CRF [104]. In the second direction, people have tried features from orthographical (e.g., word formation) [140], morphological (e.g., roots/prefixes/suffixes), and surface word features (a.k.a., dictionaries or gazetteers) [140], to shallow syntactic features (e.g., part-of-speech, phrase chunks) [140], and even to deep syntactic features (e.g., parse tree as in [35] to capture longer distance information). While some researchers try to use a minimal set of features [31] by carefully combining different features, others tend to use as many features as possible without paying too much attention to redundancy. As a result, the total number of features can be well above a million, as in [35] and [14]. In the third direction, people have tried to apply domain knowledge as much as possible in feature selection and post-processing [140].

However, with all these efforts, the best BioNER performance on the BioNLP Shared Task data sets, as far as we know, is the 72.94% F-score reported by Chan et al [14]. In order to achieve higher performance, people may have to look in different directions. One

experiments, CRF training takes much longer time than SVM training.

word	frequency as single-word entity	entity types	frequency labelled as non-entity ('O')
IL-2	340	protein	16
NF-kappaB	322	protein	7
monocytes	272	cell type	40
IL-4	242	protein	5
AP-1	240	protein	24
cytokines	207	protein	13
TNF-alpha	175	protein	7
IFN-gamma	170	protein	5
IL-10	167	protein	7
lymphocytes	156	cell type	36
Stat3	134	protein	4
c-Fos	125	protein	3
c-Jun	123	protein	3
NFAT	108	protein	5
macrophages	108	cell type	11
p50	107	protein	9
p65	100	protein	5
IL-6	97	protein	8
TNF	86	protein	13
bcl-2	64	DNA	4
promoter	58	DNA	226
mRNA	55	RNA	146
promoters	52	DNA	25
genes	43	DNA	293
Jurkat	36	cell line	21
enhancer	33	DNA	81
granulocytes	31	cell type	6
leukocytes	29	cell type	3
T-cells	20	cell type	5
HL-60	21	cell line	11

Table 3.11: A frequency comparison of 30 words labelled as single-word entity vs. labelled as non-entity in the BioNLP-2004 training data.

possible direction would be using a larger annotated corpus, which has not been explored by previous research.

Our experimental results suggest that this does not look like a promising direction, even if we do not consider the total cost needed to manually annotate such a corpus, and the potentially expensive computation required by learning from the large corpus. Instead, we would suggest the BioNER community to pay attention to other directions, e.g., improving the quality of the annotation corpus, and making use of clues across sentences.

The issue of annotation quality has been noticed by several researches, e.g., [101], [32] and [140]. It was noticed in [101] that the GENIA corpus was annotated partly by domain experts and partly by linguists, and no interannotator agreement for the annotation was published. In [32], Dingare et al suggested that the low performance on the BioNLP data set stems from high inconsistency in its annotation. They manually checked 50 errors made by their NER system and found that 34 of the errors could be attributed to inconsistent annotation of the training or evaluation data. In [140], Zhou et al manually examined 100 random selected errors from their recognition results, and found that about 50% of errors can be avoided by flexible annotation scheme (e.g., regarding the modifiers in the left boundaries) and consistent annotation. However, due to the highly ambiguous nature of biomedical names and the prohibitive cost of human annotation, making an annotated corpus with the desired quality would probably remain a formidable task.

Given the current level of annotation quality, it might be worth considering NER across sentences. So far most BioNER research takes a text as a set of sentences and tries to identify NEs within each sentence. Even if the context is considered, it is often limited within the sentence, and seldom goes beyond a sentence. However, according to our observation on MEDLINE abstracts, it is rather common that some NEs occur in multiple sentences, and some of the occurrences are easier to identify than others. We believe that making use of such clues across sentences should help to better identify NEs. Due to limited time, we did not explore further along this direction and leave it for future work.

3.6 Chapter Summary

In this chapter, we studied the relationship between the BioNER performance and the size of data used for training a supervised learing model. We began by introducing two popular learning methods, namely SVM and CRF. We described their principles as well as the pros

and cons when used for BioNER. We discussed feature selection for supervised learning. We applied the two learning methods on the BioNLP-2004 Shared Task data sets, and experimentally obtained their learning curves. We also modeled the learning curves by the power law model. While the results show that both supervised learning systems outperform the baseline system which uses exact dictionary matching, we proposed that increasing the size of training data is not an ideal way to further improve the performance of the two learning methods, and perhaps other corpus-based statistical learning methods. We recommended the BioNER research community look for other directions to achieve higher performance.

Chapter 4

Recognizing Nested Named Entities

4.1 Introduction

Nested Named Entities (nested NEs, also called embedded NEs, or cascaded NEs), expressions in which one NE contains another, are commonly seen in biomedical text. For example, the phrase *human immuneodeficiency virus type 2 enhancer* refers to a DNA domain, while its sub-phrase *human immunodeficiency virus type 2* represents a virus. In the well-known GENIA corpus, nested NEs account for 21.75% of all named entities, with up to four nesting levels. Moreover, the nested NEs often represent important relations between entities [85], as in the above example.

In spite of the importance of nested NEs and the fact that significant effort has been made to annotate them (e.g., as done in the GENIA corpus), there has been limited attention paid to recognizing them. For example, most previous studies conducted on the GENIA corpus considered only the outmost containing NEs when encountering nested NEs, as done in the BioNLP/NLPBA 2004 Shared Task [56].

When the nesting structure is ignored, it is convenient to reduce the NER problem to an NE tagging problem (or a sequence labeling problem), which can then be approached by supervised learning methods (such as SVM and CRF, as discussed in Chapter 3), because each token only associates with one NE label (i.e., one NE tag). However, the same techniques cannot be directly applicable to nested NEs, because when the nesting structure is

considered, each token in the nested entities may have multiple NE labels, one for each level of the nesting.

In this chapter, we study how to effectively recognize nested NEs. Instead of ignoring the annotations about the nesting structure and training the NER model all by the outermost level NE tags, we propose to do it level-by-level. That is, we train a set of models, one for each nesting level. When used for prediction, the models can be applied to predict NE tags on each presumed level. The advantage of doing so is that both non-nested NEs and nested NEs can be recognized simultaneously.

We evaluated this idea on the GENIA corpus and found that the model trained on a certain level would perform well only on testing data of the same level (i.e., tested on the data labeled with NE tags on the same level). In other words, a model trained by using the outmost NE tags would only be good at recognizing outmost NEs but not the inner NEs, and vice versa. As for the performance, the combined models achieve 70.13% F-Score on recognizing the outermost NEs, and 41.72% F-Score on the level-two inner NEs.

The rest of the chapter is organized as follows. We describe the NEs as well as the nested NEs in the GENIA corpus in Section 4.2 and 4.3. The proposed level-by-level method is described in Section 4.4, and evaluated in Section 4.5. We briefly discuss previous work in Section 4.6 and summarize the chapter in Section 4.7.

4.2 NEs in the GENIA Corpus

In this section, we will briefly describe the GENIA corpus[1]. The main purpose is to give readers a general idea about the NEs in GENIA. Readers who are familiar with GENIA can skip this section.

The abstracts annotated in GENIA are selected from the search results with keywords (MeSH terms) Human, Blood Cells, and Transcription Factors. In its version 3.0x, it contains 2000 MEDLINE abstracts (1999 distinct ones), 18546 sentences, 490941 tokens (19883 distinct ones), and 97876 named entities (35947 distinct ones) [2]. That is, on average, there are 9.27 sentences per abstract, 26.47 tokens and 5.28 entity occurrences per sentence.

[1]Note that the training data of the BioNLP Shared Task 2004 was actually built on the GENIA corpus, with all the nested structures for NEs discarded. That is, only the outmost NEs are considered there.

[2]Among the 97876 NE occurrences, 5154 are annotated as "null", accounting for 2373 distinct ones of such NEs.

The NEs considered the GENIA Corpus are a subset of the substances and the biological locations involved in reactions of proteins. They are annotated in XML format. For example, one sentence reads as follows:

```
IL-2 gene expression and NF-kappa B activation through CD28
requires reactive oxygen production by 5-lipoxygenase .        (4.1)
```

In the GENIA corpus, it is annotated as follows:

```
<sentence>
  <cons lex="IL-2_gene_expression" sem="G#other_name">
    <cons lex="IL-2_gene" sem="G#DNA_domain_or_region">
      IL-2 gene
    </cons>
    expression
  </cons>
  and
  <cons lex="NF-kappa_B_activation" sem="G#other_name">
    <cons lex="NF-kappa_B" sem="G#protein_ molecule">
      NF-kappa B
    </cons>
    activation
  </cons>
  through
  <cons lex="CD28" sem="G#protein_molecule">
    CD28
  </cons>
  requires reactive oxygen production by
  <cons lex="5-lipoxygenase" sem="G#protein_molecule">
    5-lipoxygenase
  </cons>
  .
</sentence>
```

These NEs are annotated into 36 types, based on the GENIA Ontology (see Appendix A for the ontology tree). However, previous works typically use their super-types, e.g., the

# of words in an NE	total occurrences	distinct occurrences	# of words in an NE	total occurrences	distinct occurrences
1	42232	5915	11	87	86
2	30058	12070	12	46	46
3	14374	8993	13	16	16
4	6033	4474	14	12	12
5	2365	2014	15	7	7
6	1240	1023	16	6	6
7	693	630	17	3	3
8	370	342	18	2	2
9	201	186	19	2	2
10	124	115	$\geq$20	4	4

Table 4.1: Distribution of entity lengths (i.e., number of words) in GENIA corpus

BioNLP-2004 Shared Task considered only five super-types, namely, *protein*, *DNA*, *RNA*, *cell type*, and *cell line*, while taking all other types as non-entities. In this chapter, we will follow this practice by considering six general types: *protein*, *DNA*, *RNA*, *cell type*, *cell line* and *other name*[3]. Note that here *other name* refers to entities that are not of the other five types, but are still entities. For readers' information, statistics about the 36 types of NEs are listed in Appendix A.

The number of words (or tokens) in an entity name, also called entity length, varies from one word to more than twenty words. The length distribution of GENIA NEs is shown in Table 4.1.

4.3 Nested NEs in the GENIA Corpus

As mentioned earlier, about 22% of GENIA NEs are involved in nesting. The maximum number of nesting levels is four, as in the following sentence:

```
The results identify functionally distinct epitopes on the CD4
co-receptor involved in activation of the Ras /protein kinase C
and calcium pathways .                                          (4.2)
```

[3]Note that in the BioNLP Shared Task 2004, only first five entity types were used. The *other name* was not considered.

words in the entity	type of the entity
Ras / protein molecule kinase C pathways	other name
Ras / protein kinase C pathway	other name
Calcium pathways	other name
Ras/protein kinase C	protein molucule
Ras	protein molecule
protein kinase C	protein molecule

Table 4.2: An example of nested NEs in the GENIA with four nesting levels

which contain the nested NEs[4] shown in Table 4.2.

To make the concept of nesting more clearly understood, we distinguish between the following seven terms:

- **nested NEs**: NEs that are involved in nesting; refers to those containing others and those contained in others. For example, in the above mentioned example sentence (4.1), *IL-2 gene expression* refers to an *other name*, while *IL-2 gene* refers to a *DNA*. Both are called nested NEs;

- **non-nested NEs**: NEs that are not involved in nesting; they do not contain others, and are not contained in others. For example, in sentence (4.1), *5-lipoxygenase* refers to a *protein*. It is a non-nested NE.

- **containing NNEs**: NEs involved in nesting and containing other NEs. For example, the above mentioned *IL-2 gene expression* is a containing NNEs.

- **contained NNEs**: NEs involved in nesting but not containing other NEs (i.e., contained by others). For example, the above mentioned *IL-2 gene* is a contained NNEs;

- **outermost NNEs**: the NNEs that contain others but are not contained in others. For example, the *IL-2 gene expression* is an outermost NNEs.

- **middle NNEs**: the NNEs that contain others and are contained in others. For example, if the *IL-2* in *IL-2 gene expression* were annotated as a *protein*, then the nesting would have three levels, with the *IL-2 gene* being the middle NNEs;

[4]Note that the original phrase "the Ras / protein kinase C and calcium pathways" contains a coordination, by which the entity "Ras / protein kinase C pathway" is implicitly expressed. If recovered from the coordination, the phrase would be explicitly read as "the Ras / protein kinase C pathway and the calcium pathway". Later on in Section 4.4, we will use this explicit form as an example to show how the GENIA annotations are converted to IOB2 notations.

NE type	number of occurrences	percentage
nested NEs	36653	21.75% of all NE occurrences (97876)
non-nested NEs	61223	78.25% of all NE occurrences (97876)
containing NNEs	16144	44.05% of all nested NEs (36653)
contained NNEs	21284	58.07% of all nested NEs (36653)
outermost NNEs	15369	41.93% of all nested NEs (36653)
middle NNEs	775	2.11% of all nested NEs (36653)
innermost NNEs	20509	55.95% of all nested NEs (36653)

Table 4.3: The number of occurrences of the NEs involved in nesting

entity type	occurring as outermost NNEs	occurring as middle NNEs	occurring as innermost NNEs
G#protein	2342	210	9088
G#DNA	1849	221	1231
G#RNA	324	106	119
G#cell_type	498	9	791
G#cell_line	544	10	173
G#other_name	7521	109	588

Table 4.4: Numbers of outermost NNEs, middle NNEs and innermost NNEs

- **innermost NNEs**: the NNEs that are contained in others but not containing NEs. For example, the *IL-2* in the above example would be the innermost NNEs.

Note that some containing NNEs are also contained NNEs, as they may stand in the middle levels of a nesting. The number of occurrences of these seven types of NEs is given in Table 4.3.

From Table 4.3, we can see that 21.75% of NE occurrences (36653 out of 97876) of the GENIA corpus are involved in nesting, of which 16144 (or 44.05%) are containing entities, implying that one containing entity may contain more than one other entity (otherwise, there would be only $16144 \times 2 = 32288$ nested NEs, instead of 36653). The nested NEs may locate at the outermost, middle and innermost levels of a nesting. The distributions of different entity types over these different nesting levels are shown in Table 4.4.

From Table 4.4, we can see that a protein, if involved in a nesting, would more likely occur at the innermost level than at the outermost level, which is in turn more likely than at the middle levels. In contrast, an NE of type *other name* would more likely appear as the outermost NNEs than as the innermost NNEs or the middle NNEs.

CC word	occurrences	examples
and	1642	alpha and beta subunits
or	184	wild-type or mutant LTRs
but not	23	OKA but not TNF stimulation
and/or	8	Spi-1 and/or Fli-1 genes
plus	4	CD2 plus CD28 adhesion molecules
as well as	3	PMA- as well as calcium- mediated activation
nor	2	neither LMP1 nor LMP2B mRNA
than	2	neonatal than adult T cells
and not	2	B and not T cells
minus	2	rectal minus oral values

Table 4.5: Distribution of NEs containing coordinating conjunction words in the GENIA corpus

outer entity	entity	count
other_name	protein	4418
protein	protein	2474
DNA	protein	1394
other_name	DNA	775
DNA	DNA	419
other_name	other_name	380
RNA	protein	348
cell_type	cell_type	327
other_name	cell_type	265
cell_line	protein	204

Table 4.6: Top 10 formation patterns of nested NEs in the GENIA corpus

In the GENIA corpus, we also observe that an important way to form nested NEs is coordination. About 2% NE occurrences contain coordinating conjunction (CC) words, i.e., they are composed of multiple shorter entities. The common CC words include: *and*, *or*, *but not*[5], etc, as shown in Table 4.5.

Nested NEs can be formed by various types of NEs. The top ten frequent formation patterns are shown in Table 4.6, while the full ranking of all possible combinations are given in Appendix A.

Assuming the outermost level of a nesting is level one, the number of GENIA entities at each nesting level are shown in Table 4.7, where the entity types are collapsed into 6 major

[5]Here we consider *but not* as one word.

NE type	occurrences at level 1	occurrences at level 2	occurrences at level 3	occurrences at level 4
Protein	24965	9856	448	15
DNA	8552	2386	69	3
RNA	719	381	8	0
Cell Type	6221	1439	30	1
Cell Line	3663	598	18	0
Other Name	19358	2181	55	0

Table 4.7: The numbers of GENIA entities at each embedded level

types: protein, DNA, RNA, cell_type, cell_line, and other_name. From Table 4.7, we can see that most NNEs occur at level one and level two, with less than 700 NNEs at level three and level four. As the number of NE occurrences at level three and four are too few to do machine learning, we only consider level one and level two NNEs in our experiments.

4.4 Methodology

As mentioned earlier, the GENIA corpus provides annotations of both nested NEs and non-nested NEs of 36 entity types. In order to compare with previous work, we consider 6 super-types of the 36 entity types, namely *protein*, *DNA*, *RNA*, *cell type*, *cell line*, and *other name*. We encode the GENIA named entity annotations using the IOB2 notation, where each token is assigned a tag to indicate whether it is the beginning (B), inside (I), or outside (O) of an NE.

As the maximum level of nesting in GENIA is four, we deliberately pad imaginary O tags to those NEs which are non-nested or involved in less-than-four levels of nesting. For example, if a token originally belongs to a non-nested NE, then after the padding, it will have 3 O tags for nesting level one to three. Table 4.8 shows an example of how the padding is done for the multiple nesting levels, where the padded NE tags are shown as italic.

As the result of NE tag padding, each token is associated with four tags, one for each nesting level, with level one corresponding to the outermost level. By now, we are able to reduce the nested NER problem to several one-level non-nesting NER problems, each of which can then be handled by statistical machine learning techniques as we did in the previous chapter. In our experiments to be described below, we will train a set of CRF models for the nested NER problem.

the word	level-one NE tag	level-two NE tag	level-three NE tag	level-four NE tag
the	O	*O*	*O*	*O*
Ras	B-other_name	B-other_name	B-protein_molecule	B-protein_molecule
/	I-other_name	I-other_name	I-protein_molecule	O
protein	I-other_name	I-other_name	I-protein_molecule	B-protein_molecule
kinase	I-other_name	I-other_name	I-protein_molecule	I-protein_molecule
C	I-other_name	I-other_name	I-protein_molecule	I-protein_molecule
pathway	I-other_name	I-other_name	*O*	*O*
and	I-other_name	O	*O*	*O*
the	I-other_name	O	*O*	*O*
calcium	I-other_name	B-other_name	*O*	*O*
patheway	I-other_name	I-other_name	*O*	*O*

Table 4.8: An example of padding NE tags with imaginary O tags (drawn as italic)

As most NNEs in GENIA are located in the first two levels of nesting, we only use these two levels of NE tags in the CRF training. We call the model trained on level one NE tags the *level-one model*, and that trained on level two NE tags the *level-two model*. We will evaluate the models using data of corresponding levels in the section below.

Here we do not consider the NNEs at level three and level four, because the number of NNEs occurrences are too sparse to train a feasible model using a statistical machine learning method. We suggest they might be better recognized using rule-based approaches, and leave them for future work.

4.5 Evaluation

As we did in Chapter 3, we still use the CRF++ package for the CRF implementation. We do 5-fold cross validation on the GENIA data set and report averaged results. In each run, four folds are used as training data, with the remaining one fold as testing data.

As the CRF training time would increase significantly if too many features were considered in the model, here we only used the following types of features: the current token, the POS tag, and the tokens frequently seen inside entities. The context window is set to be (-2, +2). That is, the two words (or tokens) immediately before and after the current token are considered.

As mentioned in the previous section, each token in the whole data set is assigned four

levels of NE tags, corresponding to four nesting levels. We train the level-one model using the level-one NE tags, and the level-two model using the level-two NE tags. We evaluate each of the two models on the two levels of data respectively. The results are given in Table 4.9.

From Table 4.9, we have the following observations:

- the level-one model performs very well on the level-one testing data (overall F-Score 70.13%), but performs very poor on the level-two testing data (overall F-Score 2.30%).
- the level-two model performs fairly well on the level-two testing data (overall F-Score 41.72%), but performs very poor on the level-one testing data (overall F-Score 2.18%).

These observations suggest that the model trained on different nesting levels should only be used to make predictions for the data of the same nesting level. When applied to the data of a different level, the model would not give useful predictions.

We also observe that when applied to the testing data of the same nesting level as the training, the two models have significant differences in the NER performance in terms of the F-Scores. This can be explained by the significant differences of NE occurrences at level-one and level-two of the nesting, as has been shown in Table 4.7: trained on more positive data, the level-one model is supposed to do better.

4.6 Previous Work

Although there have been a considerable number of papers published in BioNER, we know only a few of them addressed the issue of nested NEs. In [140], a rule-based approach was proposed to deal with nested NEs. A set of six patterns (shown in Table 4.10) are hand-crafted, based on language patterns observed in the GENIA corpus, to generate the nesting structure from the innermost NEs, provided that they have been recognized by an HMM model. The evaluation was done by 10-fold cross-validation on the GENIA corpus (Version 3.0). The results show that the rule-based processing of the nested NEs improved the overall F-score of all NEs by 3.9, compared to that without the processing. However, since the approach is basically rule-based, it would suffer the problems associated with general rule-based methods, for example, difficult to adapt to a new domain. Besides, the performance was reported in terms of all NEs, including non-nested NEs as well as nested NEs. It was not clear how well the nested NEs were recognized.

testing level-one model on level-one data			
Entity Type	Precision	Recall	F-Score
Protein	74.85%	72.35%	73.57%
DNA	74.08%	61.22%	66.97%
RNA	83.98%	63.95%	72.382%
cell type	78.25%	72.05%	75.02%
cell line	72.69%	62.57%	67.14%
other name	69.07%	62.75%	65.61%
All types	73.37%	67.21%	70.13%
testing level-two model on level-two data			
Entity Type	Precision	Recall	F-Score
Protein	66.09%	42.11%	51.39%
DNA	53.47%	19.45%	28.48%
RNA	61.93%	30.37%	39.656%
cell type	54.24%	16.04%	24.62%
cell line	40.91%	9.67%	15.44
other name	46.23%	11.40%	18.27%
All types	62.35%	31.37%	41.72%
testing level-one model on level-two data			
Entity Type	Precision	Recall	F-Score
Protein	2.80%	6.86%	3.98%
DNA	0.95%	2.85%	1.42%
RNA	1.48%	2.38%	1.81%
cell type	0.61%	2.48%	0.98%
cell line	0.44%	2.36%	0.74%
other name	0.37%	2.97%	0.65%
All types	1.48%	5.11%	2.30%
testing level-two model on level-one data			
Entity Type	Precision	Recall	F-Score
Protein	10.66%	2.70%	4.30%
DNA	5.81%	0.60%	1.09%
RNA	7.03%	1.62%	2.62%
cell type	3.17%	0.21%	0.39%
cell line	9.93%	0.30%	0.58%
other name	3.06%	0.09%	0.17%
All types	9.16%	1.23%	2.18%

Table 4.9: NER performance of 5-fold cross validation on GENIA data

pattern	example
ENTITY := ENTITY + head noun	PROTEIN binding motif → DNA
ENTITY := ENTITY + ENTITY	LIPID PROTEIN → PROTEIN
ENTITY := modifier + ENTITY	anti Protein → Protein
ENTITY := ENTITY + word + ENTITY	VIRUS infected MULTICELL → MULTICELL
ENTITY := modifier + ENTITY + head noun	(not given)
ENTITY := ENTITY + ENTITY + head noun	(not given)

Table 4.10: The six patterns used in [140] for handling nested NEs.

In a preliminary study [41], we showed that training with outermost labeling yields better performance on recognizing outermost entities, and conversely, using the inner labeling results in better F-scores for recognizing inner entities. The results were obtained by training a binary SVM model on protein and DNA NEs only. In this chapter, we further extend those results by proposing a level-by-level learning method for recognizing both the non-nested NEs and nested NEs. Moreover, we train a CRF model for more entity types.

The most recent work we know on nested NEs is [1]. Three methods were introduced and compared to model and recognize nested NEs. The basic idea was to reduce the nested NER problem to one or more BIO tagging problems so that existing NER tools can be used. One of the methods, called layering, used an idea similar to our level-by-level method. However, it does not explicitly distinguish the NEs at different nesting levels as we do. Rather, it only distinguishes nested NEs by separating them as "containing NEs" or "contained NEs". Another different aspect is that we evaluate the NER performance on different nesting levels, while only the overall NER performance (for all NE types at all levels of nesting) was reported in [1]. Therefore, it was not clear that how well the nested NEs themselves were recognized.

4.7 Chapter Summary

In this chapter, we propose a level-by-level method to train supervised learning models for recognizing nested NEs. The idea is to use the NE tags on different nesting levels separately. We tested the idea using the GENIA corpus, where the maximum level of nesting is four. Particularly, we train a CRF model for each nesting level, and find that the model trained on a particular level would only predict well the NE tags of the same nesting level. Although

evaluated only on the GENIA corpus, we expect the proposed method should be applicable to other corpora that contain nested NEs.

A number of issues are worth consideration for future work. First, we can apply other machine learning algorithms to see which one works best for nested NEs. Second, we can evaluate the proposed method on other corpora that contain nested NEs. Third, we can explore features that are suitable for different levels of nesting. The intuition here is that NEs at different nesting levels might have different characteristics. Fourth, we can consider how to make use of annotation information across the nesting levels. The idea is that knowing the level one annotation of a word should benefit the recognition of its level two annotation and so forth.

Chapter 5

BioNER by NP Chunking

Readers might have noticed that a named entity is basically a noun phrase (NP) or a part of it. They might wonder, is it possible to do NER by an NP-based method? That is, instead of approaching NER using a word-based classification task, can we go with a phrase-based method? For example, could we first identify all NPs in a given sentence and then determine which of the NPs are NEs? This seems intuitively straightforward. Moreover, it seems feasible too, considering the fact that NP chunking has been well studied in NLP, and its performance in the newswire domain has been very close to that of human. In this chapter we will explore this idea for the biomedical NER task.

5.1 Motivation

In previous chapters, we have seen how the NER task can be reduced to a word-based classification problem. In this framework, we manage to train a classifier to assign each word of a given sentence an entity label. For this end, we express each word as a vector in some feature space, where each dimension of the vector corresponds to a feature of the word. We try to use features that can best represent the words as well as to distinguish each from the others. Given a training corpus where each word in a sentence is annotated with an entity label, we run a classification algorithm (e.g., SVM) to learn the mapping from the vector to the label. The resulting classifier can then be used to assign entity labels to words in previously unseen sentences.

Note that the word-based approach has actually combined the two subtasks of NER — entity boundary detection and entity type classification — into one step. This combination

has an obvious drawback in feature selection. The two subtasks may require different features, if combined together, all the features have to be considered together in one model. This would lead to larger feature space (i.e., larger dimensions). Many supervised learning algorithms would suffer from the increased feature dimensions, because their computation complexity is often super-linear in terms of the number of features (e.g., SVM). Therefore, if we do NER in the two steps, we would be able to reduce the number of features required for each step, and thus spend less computational resources in total.

Further, using NP chunking for entity boundary detection would bring us one more advantage. NP chunking has been well studied in NLP. The overall performance has been very good (e.g., in the newswire domain). Thus accurate NE boundary detection could be possible with the accurate NP chunking results, given the special relationship between the two.

Besides the possible saving in detecting entity boundaries, making use of NP chunks may improve the overall performance of NER by helping to identify entity boundaries more accurately. This is possible because previous works on NP chunking have shown close-to-human performance ($F - score \approx 94\%$) on newswire texts [100]. Given that these NP chunkers could be tuned to perform similarly well on biomedical texts, we expect entity boundary detection could become easier with the help of the accurately separated NP chunks. Moreover, the entity type classification step may also benefit from it, because we could use more features specifically good for entity type classification without worrying about how they would affect entity boundary detection.

5.1.1 Dependency Between NPs and NEs

An example of how NP chunks correspond to NEs is given in Table 5.1. In the table we can see that the NEs in the sample sentence are either within base NPs or just the base NPs themselves.

To understand the dependency between being in an NP and being in an NE, we obtained the contingency table from the GENIA corpus (Table 5.2) by counting how many tokens (or words) occur in (or not in) NEs and/or NPs. From the second column of the table, we can see that there are 109588 tokens appearing in NEs, of which 106091 also appear in NPs, while only 3497 do not occur in an NP. This observation suggests that NEs are strongly dependent on NPs. The chi-square independence test[1] on Table 5.2 shows that the

sentence	IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase .
base NPs	IL-2 gene expression, NF-kappa B activation, CD28, reactive oxygen production, 5-lipoxygenase
bio NEs	Proteins: NF-kappa B, CD28, 5-lipoxygenase Genes: IL-2 gene Other types: IL-2 gene expression, NF-kappa B activation

Table 5.1: An example of base NP and NE mapping

Num of Tokens	in an NE	not in an NE	Total
In an NP	106091	189463	295554
Not in an NP	3497	193500	196997
Total	109588	382963	492551

Table 5.2: The contingency table of GENIA tokens being in an NE vs. being in an NP

dependency is statistically significant: the chi-square value = 79553.31 (degree of freedom = 1, p-value = 5.535 for 1% level, p-value = 3.841 for 5% level, p-value = 10.83 for 0.1% level).

5.1.2 Background on NP Chunking

In NLP, the task of dividing a sentence into non-overlapping phrases is called text chunking. NP chunking is part of the task, aiming to recognize the chunks that consist of noun phrases (NPs) [100]. One characteristic of NPs is that they can be recursively defined. Two or more adjacent shorter NPs can form a longer NP, connected by a conjunction, a preposition or other words. For example, the compound noun phrases "IL-2 gene expression *and* NF-kappa B activation", "reactive oxygen production *by* 5-lipoxygenase" as in the above sentence.

In order to avoid the ambiguity introduced by such recursions, here we only consider base NPs. We define base NPs as non-recursive noun phrases ending after their nominal head and excluding any type of postmodification (e.g., prepositional phrases, attributes, appositions).

Base NP chunking has been a standard task in NLP, and a number of systems [100] have shown good performance (F-score 90% $\sim$ 94%) on newswire texts.

[1]Please refer to Chapter 3 for details about the chi-square independence test.

Here we used CRF++ [2], an open source implementation of CRF for base NP chunking[3]. We train the CRF model using the WSJ_15_18 section (containing 8936 sentences) of the Wall Street Journal corpus, and evaluate it on the WSJ_20 section (containing 2012 sentences). Among the 12335 base noun phrases annotated in the testing set, the trained CRF++ chunker found 12311, of which 11503 are correct (accuracy: 97.86%). The precision/recall/F-score are 93.44%/93.25%/93.35% respectively. This is very close to the best performance (F-score 94%) evaluated on the same data set, which was reported in [62] using SVM.

5.2 The Proposed Method

Our proposed method consists of three steps, namely base NP chunking, base NP classification, and NE boundary recovery. The first step is to find out all base NP chunks in a given sentence. In our experiments, this is done by the CRF++ chunker trained on the Wall Street Journal corpus. Given a base NP output by step one, the second step is to determine whether it contains an NE. This is done by training a binary classifier that classifies the NP to 1 (has an NE inside) or 0 (has no NE inside). The type of the NE is also decided in this step. For all the base NPs that are classified as having an NE inside, we decide the NEs' boundaries in the third step. The procedure is illustrated in Figure 5.1 for the example sentence shown in Table 5.1.

As mentioned in the motivation section, the proposed NP-based method is hypothesized to have three advantages over traditional word-based approaches. First, it can take advantage of the high performance of NP chunking techniques. Second, it allows us to use appropriate features for each step, resulting in a smaller feature space to be dealt with. Third, it helps to modulize the NER process, and thus allows easy combination of different techniques and incorporating domain knowledge, for example, using domain-specific rules for the third step, while using machine learning methods for the first and the second step.

Note that here we have made an assumption that an NE can only be within a base NP. Although this has been observed to hold for most NEs, there are some NEs that cross multiple NPs. Besides, base NP chunking sometimes makes mistakes which could result in

[2]The package can be downloaded at http://crfpp.sourceforge.net/

[3]Recall that we have used the same package in Chapter 3 for NE tagging. Both tasks can be modeled by CRF.

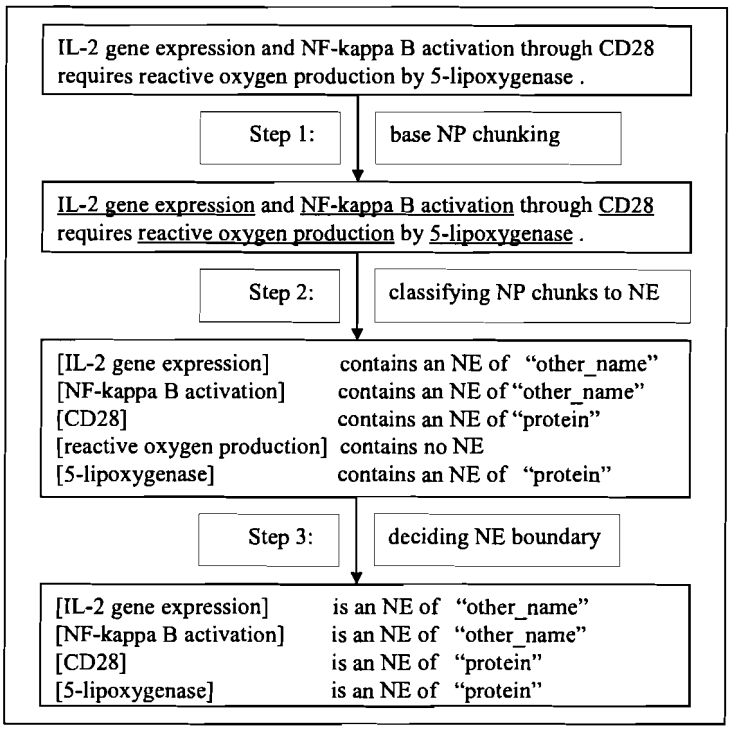

Figure 5.1: Illustration of the procedure of BioNER by NP chunking.

more cases that violate the assumption. We will see how severely the violation would affect the overall performance of NER later.

In order to classify a base NP into a type of NE, an immediate question here is how to represent its features as a vector, in order to apply a supervised learning algorithm. In word-based approaches, we construct the feature vectors by checking each word against the same set of features. This way we can guarantee that all feature vectors have the same dimensions. However, this featurization method does not work in our current setting, because base NPs can have different number of words.

To overcome this problem, we design the following method to make features for each base NP. We first create a feature vector for each word, following the method used in the above mentioned word-based approaches. We then make the feature vector of a base NP from four parts. The first part is made by merging the feature vectors of all tokens in the

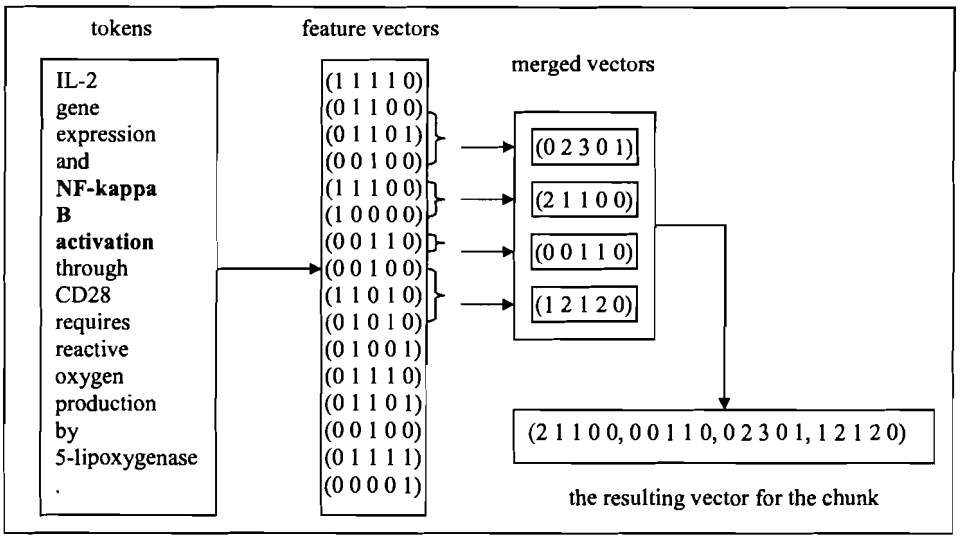

Figure 5.2: Illustration of the featurization method of an NP chunk.

NP except the head token (assuming it is the last word). The merge is done by a simple vector addition. The second part is just the feature vector of the head word. The third part is the concatenated feature vectors of the k tokens in the NP's left context, while the fourth part is the concatenation of features of the k tokens in the NP's right context[4]. Finally, the four parts are concatenated together to form one long vector to represent the base NP. By this method, we not only guarantee that base NPs of different words have feature vectors of the same length, but also take into account the features of head words, the non-head words and the words in context. Figure 5.2 shows an example of the featurization method for the NP chunk "NF-kappa B activation" in the sentence of Table 5.1.

Another problem is how to assign an entity tag to a base NP. For now, we simply set it to that of the head word (usually the last noun) of the base NP. The intuition here is that the head word often suggests the category of the NP[5]. We find that this basically works well

[4]In our experiments, we let $k = 3$ based on our experience gained in word-based approaches as described in Chapter 3.

[5]The linguistic relationship (both syntactic and semantic) between the head and the phrase is similar to that formalized in head-driven phrase structure grammar (HPSG). We refer readers to [93] for more details.

	# of proteins	# of DNA	# of RNA	# of cell_type	# of cell_line
training	30269	9533	951	6718	3830
testing	5067	1056	118	1921	500

Table 5.3: Number of entities in BioNLP-2004 Shared Task data

in our experiments.

5.3 Evaluation

We evaluate the proposed method using the data set of the BioNLP-2004 Shared Task, as we did in Chapter 3. Here we only show the number of NEs of different types in the training and testing set in Table 5.3, and refer the reader to Chapter 3 for other details.

For both the training data and the testing data, we apply our method step by step: to do base NP chunking, to make a feature vector and to assign an entity type for each base NP. We then train an SVM model for classifying a base NP to an entity type. The model is then evaluated on the stand-alone testing data. We use *SVM-light*[6] as the implementation of SVM.

5.3.1 The Assumption about NE Boundaries

We first evaluate how the performance would look like under the assumption that the NE boundaries are exactly the base NP boundaries. Given the base NP chunks output by the base NP chunker, we train an SVM classifier which takes a base NP chunk as input and classifies it into any of the five entity types. The labels for the base NP chunks are set to be the same as the type of the NE contained in it if any; otherwise, the type of the chunk is set to non-entity. The performances for all the five NE types are measured on the testing data set and shown in Table 5.4.

Note that the NP type classification results, although looking good, are not considering the boundaries of the NEs. In order to recognize the actual NEs, we still need to decide the NE boundaries.

If we simply assume the NE boundaries are just the corresponding base NPs, we have the following NER performance for proteins (see Table 5.5), which is far inferior to that of the BioNLP-2004 (around 70% F-score for exact match).

[6]The package is available at http://svmlight.joachims.org/.

Entity type	Precision	Recall	F-score
Protein	82.43%	83.00%	82.7%
DNA	80.86%	65.04%	72.1%
RNA	73.83%	69.30%	71.5%
cell_type	90.90%	66.84%	77.0%
cell_line	82.16%	55.78%	66.4%

Table 5.4: Performance (P/R/F) of classifying base NP chunks into NE types (on testing set)

	# of matches	Precision	Recall	F-score
Exact match	1817	35.86%	48.02%	41.06%
Right boundary match	2942	58.06%	77.75%	66.48%
Left boundary match	2178	42.98%	57.56%	49.21%

Table 5.5: The Performance of recognizing proteins using exact match, right boundary match,and left boundary match, assuming NE boundaries are just the NP boundaries.

By examining the BioNLP-2004 data set, we observe that all the annotated NEs do not contain a determiner (i.e., *a*, *an*, and *the*). However, the results of base NP chunking included the determiner as part of an NP. In particular, we found that in the testing set, 2785 base NPs contain *a* or *the*. So we ignore the first token's tag if the token is *a* or *the*. This adjustment involves the left boundaries of the base NP chunks. The resulting recognition performance is shown in Table 5.6. We can see that we gain about 8% F-score for exact match, and about 10% for left-boundary match.

5.3.2 The Problem of Boundary Mismatch

From the above experiments, we realize that boundary mismatch seems the biggest problem to map a base NP to an NE. In order to see how severe the problem is, we count how many words or tokens are labeled as parts of NEs but not identified as parts of base NPs.

	# of macthes	Precision	Recall	F-score
Exact match	2177	42.96%	57.53%	49.19%
Right boundary match	2942	58.06%	77.75%	66.48%
Left boundary match	2619	51.69%	69.21%	59.18%

Table 5.6: Performance of recognizing proteins after removing all determiners from base NP chunks when mapping them to corresponding NEs.

token	Freq in Protein	token	Freq in DNA	token	Freq in RNA	token	Freq in cell_type	token	Freq in cell_line
(	530	(	379	(	32	(	99	(	109
)	276	)	134	,	26	and	45	,	55
of	174	,	79	and	12	+	40	)	44
,	102	and	78	)	11	)	38	and	30
and	75	;	76	not	1	,	36	clones	21
:	16	to	70	neither	1	-	10	-	14
to	15	of	33	forms	1	subsets	9	+	12
or	12	constructs	33	chain	1	isolated	6	or	9
activated	10	upstream	32	cfms	1	not	5	not	8
+	7	or	14	but	1	to	4	infected	7

Table 5.7: Top 10 words/tokens that appear in different types of NEs but not identified as part of a base NP

entity type	training set (original)	training set (base NP)	testing set (original)	testing set (base NP)
protein	30269	21418	5067	3861
DNA	9533	8688	1056	989
RNA	951	708	118	98
cell_type	6718	6197	1921	1763
cell_line	3830	3696	500	482

Table 5.8: Comparison of the number of true entities and those resulted from base NP chunks

We found that there are a total of 3497 and 866 such occurrences of words/tokens in the training and testing data respectively. We list the top 10 frequent words/tokens in Table 5.7. The mismatches that appeared in the training data will no doubt play a negative role in training the SVM classifier. Besides, the 866 mismatches in the testing data could cause misidentifying up to 17% of the 5067 entities, if we assume one such mismatch accounts for one recognition error. In other words, the boundary mismatch does look like a big, if not the biggest, problem.

Another problem resulting from base NP chunking is that there are fewer ntities (shown in Table 5.8) resulting from base NP chunking, if we assume an entity exactly corresponds to a base NP. We think this problem is also related to the errors made by the base NP chunker we used, which is trained on newswire texts, rather than biomedical texts. We believe that such errors could be greatly reduced if we had used a base NP chunker trained on biomedical texts.

CC word	occurrences	examples
and	1642	alpha and beta subunits
or	184	wild-type or mutant LTRs
but not	23	OKA but not TNF stimulation
and/or	8	Spi-1 and/or Fli-1 genes
nor	2	neither LMP1 nor LMP2B mRNA
than	2	neonatal than adult T cells
and not	2	B and not T cells
as well as	3	PMA- as well as calcium- mediated activation
plus	4	CD2 plus CD28 adhesion molecules
minus	2	rectal minus oral values

Table 5.9: Distribution of NEs containing coordinating conjunction words in the GENIA corpus

CC words	occurrences	examples
and ··· and ···	8	normal and leukemic B and T cell precursors and T lymphocytes
and ··· or ···	3	unstimulated and PMA- or TNF- stimulated cells
or ··· or ···	1	11 alpha-chloromethyl or 11 alpha- or 11 beta-phenyl

Table 5.10: Examples of NEs containing multiple CC in the GENIA corpus

5.3.3 NEs Spanning Multiple Base NPs

There is yet another problem related to our one-base-NP-for-one-NE assumption. We find that there are a considerable number of NEs involved in conjunctions, implying that they are supposed to span multiple base NPs, as shown in Table 5.9. Some NEs are even involved in multiple conjunctions, as shown in Table 5.10. Though it is hard to estimate how much this violation of our assumption accounts for the performance reduction, there is no doubt that the violation should be taken into consideration to improve the performance. As we lack biomedicine domain knowledge, we leave this issue for future work.

5.3.4 Effects of POS Tags on the NP-NE Relation

Since part-of-speech (POS) is probably the most important information used in deciding base NP boundaries, we naturally want to see how it is related to NE boundaries. We therefore count how frequently each type of POS occurs or not in a base NP and an NE. We use the GeniaTagger to do POS tagging on the BioNLP 2004 training data. The tagger

achieved 98.26% F-score tested on the GENIA corpus [7]. The frequencies are given in Table 5.11.

This information can help us make rules to determine NE boundaries. For example, knowing that PRPs (Possessive pronouns) have always appeared in base NPs but not in NEs, we can create a rule to exclude these words when we want to decide the boundary of an NE within a base NP.

5.3.5 Adjusting Base NP Boundaries for NEs

Based on the above error analysis, especially the statistics about POS distributions over base NPs and NEs, we make a rule to adjust the boundaries of base NPs. For those NPs that contain an NE inside (we know this from the NP classification step), the rule prevents words from being included in the NE if their POS tags are any of {DT, CC, RB, PRP, VBG, WDT, FW, JJR, POS, JJS, EX, PDT, RBS, WP, NNP}, based on the frequencies in Table 5.11. All these words, if appearing in an NP, are much more likely outside an NE than inside an NE. The results of overall NER performance on testing data is given in Table 5.12. To compare, we have also given the best performance using a word-based SVM classifier with the same features in Table 5.13.

From Table 5.12 and Table 5.13 we can see that the POS adjustment rule does significantly improve the NER performance, compared to that shown in Table 5.6, although it is still about 8% less than the word-based SVM performance (dominant by protein) which we obtained in Chapter 3. We believe that the performance could be further improved if more domain knowledge could be incorporated to make more effective rules.

As an error analysis, we compare the number of base NPs to the number of true NEs before and after applying the adjustment rule. They are given in Table 5.14 and Table 5.15, respectively. The true NE numbers are given in column two for training data and column four for testing data, while the identified base NP numbers are given in column three and column five. Note that these identified base NPs will be used as unit data to train and test the SVM classifier. We can see from the tables that the adjustment rule does help in identifying NE boundaries more accurately, resulting in more NEs being identified. That is, more positive data can be used in the training phase.

Encouraged by the significant improvement resulting from our simple rule for boundary

[7] See details on http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/

POS Tag	Total Freq	Freq in NP and NE	Freq in NP but not in NE	Freq not in NP but in NE	Freq not in NP and not in NE
NN	151140	72935	77708	142	355
IN	63085	1	194	270	62620
JJ	47013	13009	29870	104	4030
DT	36456	72	36341	7	36
NNS	33019	14990	18005	6	18
.	18440	0	17	9	18414
,	18079	133	760	298	16888
CC	17442	913	6068	299	10162
VBN	12804	4	25	14	12761
RB	11938	145	1420	50	10323
VBD	10164	2	26	96	10040
VBP	9393	2	17	39	9335
VBZ	9322	0	17	117	9188
CD	8898	2335	6547	7	9
TO	7615	35	82	89	7409
)	7444	957	1166	508	4813
(	7395	307	889	1157	5042
PRP	3792	0	3792	0	0
VBG	3364	6	62	23	3273
VB	3102	2	16	24	3060
WDT	2483	0	2482	0	1
MD	1931	0	0	0	1931
FW	1500	23	731	2	744
SYM	1480	118	789	117	456
:	1428	0	0	109	1319
WRB	921	0	2	0	919
JJR	454	17	271	0	166
POS	210	44	166	0	0
JJS	181	0	123	0	58
EX	153	0	153	0	0
PDT	153	0	122	0	31
”	146	33	62	10	41
RBS	103	6	59	0	38
WP	49	0	49	0	0
RBR	22	0	3	0	19
NNP	15	0	15	0	0
“	6	1	4	0	1
PRP$	0	0	0	0	0

Table 5.11: Frequencies of POS occurring in base NPs and NEs

Entity type	Precision	Recall	F-score
protein	53.68%	62.80%	57.88%
DNA	47.54%	57.70%	52.13%
RNA	50.85%	56.07%	53.33%
cell_type	54.66%	73.12%	62.56%
cell_line	41.60%	58.10%	48.48%

Table 5.12: NER Performance of exact boundary match on testing set after applying the boundary adjustment rule based on POS tags

Entity type	Precision	Recall	F-score
protein	71.21%	62.04%	66.31%
DNA	53.69%	67.10%	59.65%
RNA	55.93%	63.46%	59.46%
cell_type	57.47%	80.00%	66.89%
cell_line	46.20%	56.48%	50.83%

Table 5.13: NER performance of word-based SVM classifier using the same features

Entity type	Training set (original)	Training set (base NP)	Testing set (original)	Testing set (base NP)
protein	30269	21418	5067	3861
DNA	9533	8688	1056	989
RNA	951	708	118	98
cell_type	6718	6197	1921	1763
cell_line	3830	3696	500	482

Table 5.14: Numbers of base NPs and true NEs before the adjustment

Entity type	Training set (original)	Training set (base NP)	Testing set (original)	Testing set (base NP)
protein	30269	26628	5067	4832
DNA	9533	10429	1056	1177
RNA	951	862	118	121
cell_type	6718	7134	1921	2096
cell_line	3830	4407	500	578

Table 5.15: Numbers of base NPs and true NEs after the adjustment

Entity type	Precision	Recall	F-score
protein	91.20%	83.73%	87.30%
DNA	74.24%	84.94%	79.23%
RNA	77.97%	90.20%	83.64%
cell_type	79.59%	94.09%	86.24%
cell_line	61.80%	69.59%	65.47%

Table 5.16: NER performance of exact match, assuming base NP chunks exactly match the true NE boundaries

adjustment, we believe that if the base NP chunks reflected the actual boundaries of the NEs very well (e.g., by using a base NP chunker trained specifically for biomedical texts), then the NER performance of the NP-chunking based method could approach that of the word-based performance.

To show this is possible, we did another experiment by assuming that the base NP chunks exactly match the true NEs. This can be done by manipulating the base NP tags in the training and testing data: if np_tag = "I" but ne_tag = "O", then reset np_tag = "O". The resulting NER performance on the five types of NEs is given in Table 5.16.

From Table 5.16 we can see that the NER performance on all entity types would be substantially better than those of the word-based SVM as shown in Table 5.13. This result suggests that if NE boundaries could be accurately recovered from base NP chunks, the proposed NP-chunking based method could classify the chunks very well and thus achieve very good NER performance. From another perspective, this result also implies that how one determines NE boundaries is probably the key or bottleneck issue for the BioNER task.

5.4 Chapter Summary

In this chapter, we have empirically studied how biomedical NER can be done by using base NP chunks. The idea is inspired by the observation that although NER and NP chunking are different tasks, they are closely related — an NE must be an NP or inside an NP. As base NP chunking on newswire texts has achieved close-to-human performance ($F > 0.90$), we conceived that NER could benefit by making use of NP chunks to decide NE boundaries.

We assumed that one base NP corresponds to one NE, and view NER as a classification problem that classifies an NP into a category (type) of entity. Given a sentence, we do NER in two phases. In phase one, we run base NP chunking over the sentence, assigning an NP

tag to each word. In phase two, we decide, for each base NP, whether it corresponds to an NE.

Two treatments were made in order to use the NE annotations of BioNLP-2004 Shared Task data sets. First, a featurization method was designed to express an NP as a vector, where the NP can have a varying number of words, but the vector has fixed dimensions. Second, a set of heuristics were designed to match the boundaries of corresponding NPs and NEs, based on the knowledge of their distributions over POS tags.

An SVM classification model was trained and tested on the BioNLP-2004 data sets. In the training process, a sentence was first chunked into base NPs, then each NP was converted into a vector and assigned a class label based on the NE tag of the last word in the NP, and then an SVM model was learnt from all the base NPs in the training sentences. In the testing process, a sentence was first chunked into base NPs, then each NP was converted into a vector, and assigned a class label based on the model.

We obtained F-score 57% on the BioNLP-2004 evaluation data set, which is about 8% less than that obtained by using word-based SVM learning. We conducted error analysis and found that boundary mismatches introduced most class labeling errors in training and testing process, while some errors were introduced by the assumption of one base NP matching one NE. Provided that the base NP chunks exactly match the NEs, we obtained very good results with F-score close to 87%.

Our study suggests that although NER is closely related to NP chunking, it is not easy to do NER directly based on NP chunking results. The difficulty may be due to the intrinsic difference between the two tasks: while NEs are words grouped together by their semantic meanings, NPs are more dependent on syntactic relations (e.g., which word modifies which) and functions (e.g., used as noun or adjective) of words. Our experience also suggests that determining NE boundaries is probably the most difficult process in BioNER, and rule-based methods that use linguistic patterns in combination with in-depth domain knowledge should have the good potential to help deal with it. In this sense, the proposed NP-chunking based method actually provides a framework to incorporate domain knowledge. Finally, our study also brings an immediate demand to the BioNLP community: an NP chunker for texts of the biomedical domain is highly desired.

Chapter 6

BioNER in the Absence of Human Annotated Corpora

Biomedical Named Entity Recognition (BioNER) is an important task in biomedical text mining. Currently the dominant approach is supervised learning, which requires a sufficiently large human annotated corpus for training. In this chapter, we propose a novel approach aimed at minimizing the annotation requirement. The idea is to use a dictionary which is essentially a list of entity names compiled by domain experts and sometimes more readily available than domain experts themselves. Given an unlabelled training corpus, we label the sentences by a simple dictionary lookup, which provides us with highly reliable but incomplete positive data. We then run a SVM-based self-training process in the spirit of semi-supervised learning to iteratively learn from the positive and unlabelled data to build a reliable classifier. Our evaluation on the BioNLP-2004 Shared Task data sets suggests that the proposed method can be a feasible alternative to traditional approaches when human annotation is not available[1].

6.1 Introduction

As we have reviewed in Chapter 2, approaches to the BioNER task basically fall into three groups: dictionary-based, rule-based, and supervised-learning-based. The dictionary-based

[1]A paper based on the work described in this chapter was accepted for oral presentation in the IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE 2007) and published in its proceedings (http://caai.cn:8086/nlpke07/) [42]

approach assumes the presence of a dictionary of names of target types and identifies names in text by using string-matching techniques, which makes it suffer from low coverage of the dictionary used. The rule-based approach typically uses hand-crafted linguistic rules (or patterns/templates) and seeks named entities by pattern-matching or template-filling. The problem is that good rules require hard work from domain experts, and are not easily adaptable to new domains. In recent years, supervised learning techniques have become dominant, with better performance and adaptability. However, their major drawback is that they need a sufficiently large annotated corpus to build an accurate model.

So far most annotation work is manually done by humans (e.g., domain experts), and the process has proven to be time-consuming and error-prone. For example, the well-known GENIA corpus, debuted in the year 2000 with 500 annotated Medline abstracts [115] took about 4 years to evolve into its third version, which contains 2000 annotated abstracts. Though it is widely used in biomedical NER as a benchmark as in the BioNLP-2004 Shared Task [56], it has been shown to suffer from considerable inconsistencies in its annotation [32]. While annotated texts are difficult to obtain, un-annotated texts (e.g., Medline abstracts) are more readily available. As such, active learning and semi-supervised learning have recently attracted attention, making use of unannotated texts to reduce the requirement of annotated texts. However, both methods, among other things, still require humans' involvement in annotation either at the beginning or during the process of learning.

In this chapter, we consider the BioNER task in a new setting where human annotation is not available. The idea is to use a dictionary (a list of names), instead of a human annotator, to label sentences, and then employ a technique similar to semi-supervised learning to automatically learn a model from the partially labelled sentences. The assumption of having such a dictionary is sometimes easier to satisfy than that of having a human annotator. This is the case in biomedical NER, where many names of proteins and genes can be found in a number of established databases. Although they might have limited coverage, the dictionaries typically are compiled by domain experts and thus are supposedly of good quality.

The difficulty of this new setting lies in that we often cannot completely label all words in a sentence, because some entities in it may not be covered by the dictionary. In fact, after dictionary labeling, we will only have positive examples (i.e., those words matching dictionary entries), but no negative examples (i.e., all other words remain unlabelled because their true labels are not clear). Thus, the problem becomes how to learn from positive and

unlabelled examples.

To address this issue, we propose a self-training technique using support vector machines (SVMs) as the base learner. Given the positive and unlabelled examples, we let the SVM iteratively identify reliable (or strong) negatives from unlabelled examples and use its own classification results to teach itself. We evaluated our technique on the BioNLP-2004 Shared Task corpus and obtained encouraging results. The rest of this chapter is organized as follows: the details of our method are presented in Section 6.2. Section 6.3 describes the evaluation. Related work is discussed in Section 6.4. Our conclusions and future work are discussed in Section 6.5.

6.2 The Proposed Technique

We treat the NER task as a word classification problem as we did in Chapter 3. Given a sentence, the task is to assign a class label to each word (or token). If multiple classes are involved, we reduce it to binary classification sub-problems. As mentioned in the introduction, our technique has two steps: (1) labelling sentences using a dictionary; (2) building a stable classifier by learning from the partially labelled sentences. We assume that a dictionary containing hundreds and even thousands of recognized entity names is not difficult to obtain. In the biomedical domain, a number of established databases are available, for example, SwissProt[2] and GenBank[3]. In this section, we will describe the details of the two steps as well as some relevant issues.

6.2.1 Labelling Sentences Using a Dictionary

Given a sentence, we label the words in it by performing longest match through the sentence against all entries in the dictionary. Consider the following sentence from the GENIA corpus:

```
IL-2 gene expression and NF-kappa B activation through CD28
requires reactive oxygen production by 5-lipoxygenase .        (6.1)
```

According to the GENIA annotation, here *IL-2 gene* is a gene, while *NF-kappa B*, *CD28* and *5-lipoxygenase* are all proteins. If the target entity type is protein, and the protein

[2]A curated database of proteins managed by the Swiss Bioinformatics Institute (http://www.ebi.ac.uk/swissprot/

[3]A database of nucleotide sequences maintained by the US National Center for Biology Information (www.ncbi.nlm.nih.gov/Genbank/).

dictionary we have only contains two entries: *NF-kappa B* and *CD28*, then only the 3 words in the sentence will be labelled as positives, while all others will remain unlabelled. Note that the protein *5-lipoxygenase* is not correctly labelled, because it is not in the dictionary.

After labelling sentences using the given dictionary, we have only positive tokens but no negative ones. Among the remaining tokens, some could be positive tokens that are not covered by the dictionary, while all the others are those true negative tokens. We take them as unlabelled data.

The labels for the positive tokens are reliable in most cases, except that when the token appears in embedded entities, we might obtain false positives. This can be demonstrated by the example sentence (6.1). Actually, the token *IL-2* is also annotated as a protein when it appears independently (e.g., not followed by the word gene) in the corpus. However, *IL-2* would be labelled as positive if it were in our protein dictionary, resulting in a contradiction with the GENIA annotation, which labels *IL-2 gene* as a whole as a gene.

We note that GENIA also labels *IL-2 gene expression* in the same sentence as some entity that relates to *IL-2 gene*. Thus, we speculate that it is not unacceptable to label *IL-2* as an embedded entity, as it appears to us that the *IL-2 gene* is a gene that relates to protein *IL-2*. We attribute such issues to ambiguity resulting from inconsistent annotation, and will leave them to biologists to solve. In our later evaluation, we still take the GENIA annotation as the gold standard. For all those words whose dictionary labels are different from the GENIA annotation, we neglect the dictionary labels and treat the words as unlabelled.

6.2.2 Choosing the Base Learner

We use SVM as the base learner. As a binary classification algorithm, SVM has shown outstanding performance in many classification problems (e.g., document classification). Besides, it has two salient properties that are highly desirable for our task: (1) it can handle high dimensions and tolerate sparse instance spaces [51], and (2) for each prediction, it produces the distance from the classification boundary, which can be taken as the confidence of the prediction. The first property allows us to associate hundreds of thousands of features with each word, while the second property helps us to identify reliable negatives from the unlabelled data, which are far from the classification boundary.

Here we do not describe how SVM works, which can be easily found in the SVM literature (e.g., [52]). The basic idea of how SVM builds a classifier is illustrated in Figure 6.1. Note that other classifiers (hyperplanes) may also separate the positive data (marked as "+")

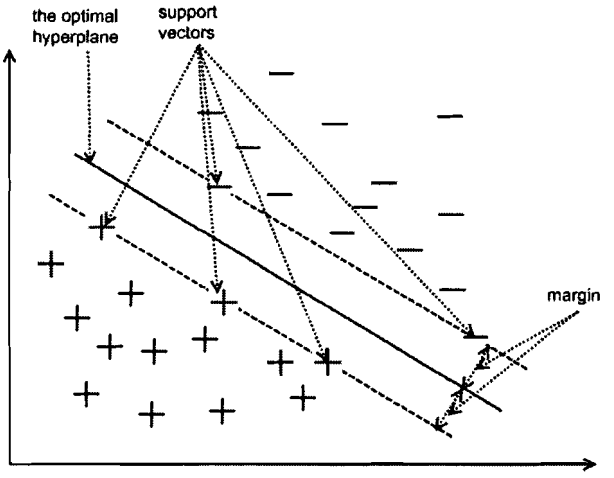

Figure 6.1: An illustration of a linear SVM in a two-dimensional feature space

from the negative data (marked as "-"). However, the SVM hyperplane is the one that maximizes the margins between the two classes. The data points on the boundary are called support vectors.

We speculate that the base learner actually can be any classification algorithm as long as it outputs a value that measures the confidence or goodness of its prediction. In the case of SVM, this value is the distance of a data point from the hyperplane.

As we did in Chapter 3 and Chapter 5, we still use the SVM-light package[4] as the implementation of SVM [52]. For efficiency reasons, we used the default linear kernel in our experiments, and found that the classification accuracy is acceptable. In the future, we plan to explore other types of kernels.

6.2.3 Selecting the Initial Negatives

Now our task becomes building a classifier from the initial positive set P_0 and the unlabelled set U obtained from the dictionary labelling. For this end, we need to select the initial set of negatives N_0. We do this using the algorithm as shown in Figure 6.2.

Here k is an adjustable parameter which can be set by experiments. In our evaluation, we set it to be equal to the size of P_0, which already works well. The initial classifier built in

[4]The package is available at http://svmlight.joachims.org/

1. Assign to each token in P_0 the class label +1;
2. Assign to each token in U the class label -1;
3. Build an SVM classifier using P_0 and U;
4. Classify U with the classifier;
5. Sort the predictions of U by the distance from classification boundary in descending order;
6. Select the bottom k words that are predicted as negatives to form the initial reliable negative set N_0, while the remaining are left in the U by resetting the class labels to 0 (meaning unlabelled);

Figure 6.2: The algorithm for selecting the initial negatives

1. Let $N = N_0$;
2. If (U is empty), go to step 8; otherwise, go to step 3;
3. Build an SVM classifier M from P_0 and N;
4. Classify U using the classifier M;
5. Sort the predictions by the distance;
6. Select the bottom k negatives from the predictions and add them to N; all others are remained in U;
7. If the model is stable, go to step 8; otherwise, go back to step 2;
8. Output the final model M;

Figure 6.3: The algorithm for SVM self-training

this stage is likely not accurate. We will expand the set N by SVM self-training described in the next section.

6.2.4 SVM Self-Training

Self-training allows the learner to teach itself. Given the positive set P_0 and the initial negative set N_0, following the above notations, we describe the self-training algorithm in Figure 6.3.

Note that other stopping criteria can also be used, e.g., the maximum number of iterations. Instead of specifying the parameter k, we can also set a predefined cut-off value of distance to select reliable negatives from unlabelled data. However, in our experiments, we observe that this has actually introduced more misclassified negatives into the expanded negative set. Thus, we did not use it.

The self-training process can be illustrated by Figure 6.4. Though we did not draw the

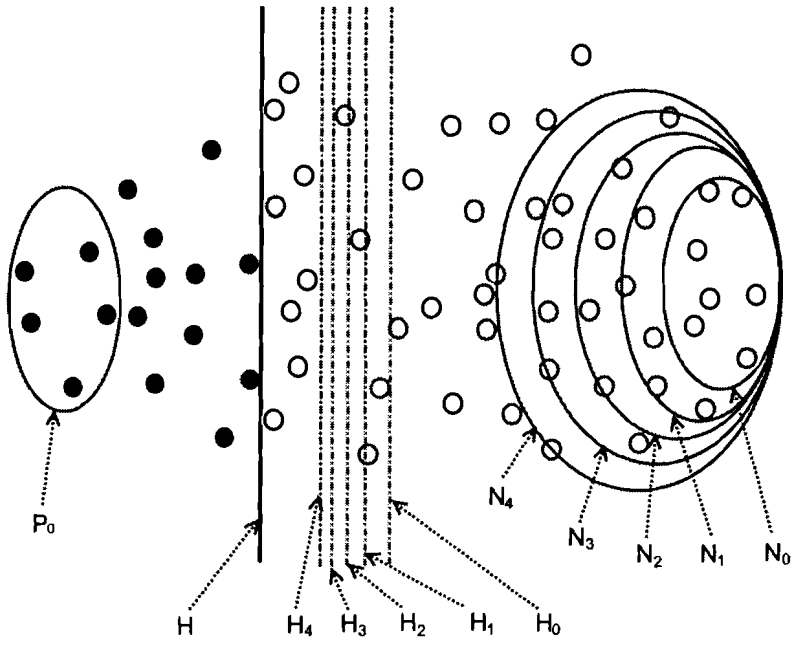

Figure 6.4: Graphical representation of how the negative set is expanded in the self-training process. The solid points are positives, while the empty points are negatives. The ellipse P_0 stands for the set of initial positives, while N_0 is the initial set of negatives obtained by taking all unlabelled data as negatives. Ellipses N_1 to N_4 are the expanded sets of negatives resulted from iteration 1 to 4. H is the true SVM hyperplane, while H_0 to H_4 each corresponds to the hyperplane between P_0 and the corresponding negative set (i.e., N_0, N_1, N_2, N_3, and H_4, respectively).

classification hyperplane generated by the 5th (or a later) iteration, readers can imagine that it is located right in the middle between the positive and the corresponding negative set. And as the self-training continues, it will approach the true SVM hyperplane H.

6.2.5 Learning Without Negative Examples

Why is it possible to learn with no negative examples?

Denis et al [30] gave an explanation from a probability perspective. Using the positive training data, we can estimate the conditional probability of example x given that it is positive: $P(x|+)$. Using the unlabelled data, we can estimate the probability of x: $p(x)$. If the probability of the positive class can be estimated, we can calculate the conditional

probability of x given that it is negative:

$$p(x|-) = [p(x) - p(+)p(x|+)]/p(-) \tag{6.1}$$

where $p(-) = 1 - p(+)$.

With the $p(x|-)$, we can compute the probability of being positive or negative given the probability of x by using Bayes rule as following:

$$p(+|x) = p(+)p(x|+)/p(x), \;\; p(-|x) = p(-)p(x|-)/p(x). \tag{6.2}$$

We can then choose the one with higher probability as the predicted class.

6.2.6 Positive Set Expansion

Why not expand the positive set?

Ideally we hope to expand the positive set as well as the negative set, so that the resulting classifier could approach the one built from fully annotated data. We could do the two expansions at the same time, or at different times (e.g., first expanding the negative set to some predefined size). However, the idea did not work well in our experiments. We tracked the class labels assigned to selected "reliable" positives and negatives during the self-learning. We found that a significant portion of predicted reliable positives are false positives. In our experiments, the misclassification rate sometimes can be 40%, which is far larger than that of the negatives. Therefore, if we also expanded the positive set, many mislabelled data would be introduced, so that would seriously bias the classifier built in later iterations. This could be related to the nature of the problem: positive words are in themselves harder to identify. How to reduce the misclassification rate of the positives is our ongoing work.

6.3 Evaluation

6.3.1 The Data Sets

As we did in Chapter 3, we use the data sets of the BioNLP-2004 Shared Task [56] in the evaluation. Recall that the training data contains 2000 annotated Medline abstracts (in total 18546 sentences, 492551 tokens), while the testing set has 404 annotated Medline abstracts (in total 3856 sentences, 101039 tokens). For the shared task, all entities were annotated into 5 classes: protein, DNA, RNA, cell line, and cell type, using the IOB2 notation [99].

Feature name	Example	Feature name	Example
1st_char_upper	MHC	Has_dot	PU.1
1st_char_lower	Gene	Has_comma	1,25(OH)2D3
1st_char_digit	5-lip	Has_hyphen	peri-kappa
Last_char_upper	ROI	Has_slash	enhancer/promoter
Last_char_lower	c-myc	Has_parenthesis	CCK(B)
Last_char_digit	CD28	Has_plus	K+
Upper_inside	hEpoR	Roman_num	Type-II
Lower_inside	PuB2	Greek_num	Gamma(c)
Digit_inside	E1A	Has_others	FY*B, t(8;21)

Table 6.1: Some orthographic features and examples

6.3.2 The Features

Feature selection is essential to any classification task. Previous studies (e.g., [140]) have found that the orthographic, morphological, gazetteer, syntactic features are usually useful for BioNER. As our goal in this study was not to seek the best features toward the best performance, but rather was to evaluate the proposed method, we followed the research of [140] by only considering the features listed below. These features, though perhaps not the best features and not in the best combination, already gave us comparable performance.

1. Orthographical features: they were used to capture capitalization, digitalization, and other token formation information. Some of these features and examples are given in Table 6.1

2. Morphological features: We used frequent prefixes and suffixes. Given all token occurrences in the training set, we counted the frequencies of each prefix and suffix (3-5 chars long) of a token, and selected those occurring more than 3 times as the frequent prefixes and suffixes respectively.

3. Part-of-Speech features: they have been shown useful in detecting boundaries of biomedical NEs. We used the GENIA Tagger for POS tagging.

4. Frequent token features. This feature set is also called gazetteer features. Among the distinct tokens in the training set, we took those appearing more than 3 times as the frequent tokens.

As such, we had 25000 features for each token. In order to capture context clues of a token, we incorporated the features of its previous 3 tokens and its next 3 tokens (i.e., the context window size is 7). In total, we used 175000 features for each token. Note that the

same set of features have also been used in Chapter 3 for the SVM model.

6.3.3 Handling Multiple Class Labels

As we saw in Chapter 3, there are five types of entities in the data set. Annotated in the IOB2 notation, we have 11 class labels, namely, *B-protein*, *I-protein*, *B-DNA*, *I-DNA*, *B-RNA*, *I-RNA*, *B-cell_type*, *I-cell_type*, *B-cell_line*, *I-cell_line*, and *O*. We converted the multi-class problem into a combination of 11 binary classification problems. Basically, we built a binary classifier for each class label. When predicting the class label for a token, we ran all the 11 classifiers and got 11 predicted values, from which we chose the one having the largest absolute value to be the predicted class label. Though the combination looks simple, it works well for the task.

6.3.4 The Input Dictionary

The dictionary used in the evaluation was built from the training set. By extracting all the distinct entities provided in the training set, we made a 100% coverage dictionary. This dictionary contains 18582 unique entity names, of which 8630 are proteins, 5278 are DNAs, 461 are RNAs, 2063 are cell types and 2150 are cell lines). By randomly selecting a subset of this 100% dictionary, we assessed the performance of our technique under different dictionary coverages.

6.3.5 The Baseline and Skyline System

For comparison purposes, we built both a baseline and what we call a skyline system. The baseline system took purely a dictionary-based approach, doing longest match through the sentences against the input dictionary. The performance varies with the dictionaries used. The overall F-score on the testing set using the 100% dictionary is given in Table 6.2, which is close to the baseline of the BioNLP-2004 Shared Task.

The skyline system adopted a fully supervised learning approach, to simulate the best performance that our proposed technique could possibly achieve. It used all annotations given in the training set to train a SVM classifier (using the features as mentioned in Subsection 6.3.2, and handling multiple class labels as described in Subsection 6.3.3). No further post-processing was done. The performance on the testing set is shown in Table

Entity Type	Skyline System Precision/Recall/F-score	Baseline System Precision/Recall/F-score
Protein	71.21% / 62.04% / 66.31%	58.24% / 47.03% / 52.04%
DNA	53.69% / 67.10% / 59.65%	33.33% / 22.81% / 27.09%
RNA	55.93% / 63.46% / 59.46%	32.20% / 16.96% / 22.22%
cell_type	57.47% / 80.00% / 66.89%	55.44% / 41.86% / 47.70%
cell_line	46.20% / 56.48% / 50.83%	32.20% / 29.65% / 30.87%
[-ALL-]	64.37% / 65.19% / 64.78%	52.72% / 41.04% / 46.15%

Table 6.2: Performance of the skyline and the baseline system using the 100% dictinoary

6.2. The overall F-score would rank the 5th among all the 8 systems participated in the BioNLP-2004 Shared Task.

6.3.6 Our System

We implemented the technique described in Section 6.2. Tokens were featurized as described in Subsection 6.3.2, and class labels were handled as in Subsection 6.3.3. To simulate real world scenarios where a perfect dictionary is hard to obtain, we used a random subset of the above-mentioned 100% dictionary as the input dictionary. By adjusting the sampling size, we tried 10%, 20%, up to 90% of the 100% dictionary. With the dictionary of a given size, we labelled all sentences in the training set from which an SVM model was learned by the self-training process. The model was then evaluated against the testing set. The base learner was the SVM-light package with the default settings. In the self-training process, we set the initial size of the negative set equal to that of the true positives labelled by the dictionary. For each iteration, we select a fixed number of new negatives (i.e., set the parameter k to be 5% of initially total unlabelled tokens). We stopped the training iterations when 80% of unlabelled tokens had been labelled, which is typically sufficient to produce a stable model.

6.3.7 Results and Discussion

Averaged over 3 runs, the overall F-scores of our system and the baseline system on all the entity types are shown in Figure 6.5. From the results, we can see that our system beats the baseline system that uses the same dictionary, and approaches the skyline system performance when using larger dictionaries. For example, using the same 50% random subset of the perfect dictionary, our system achieves a 0.46 F-score, while the baseline system has a 0.30 F-score.

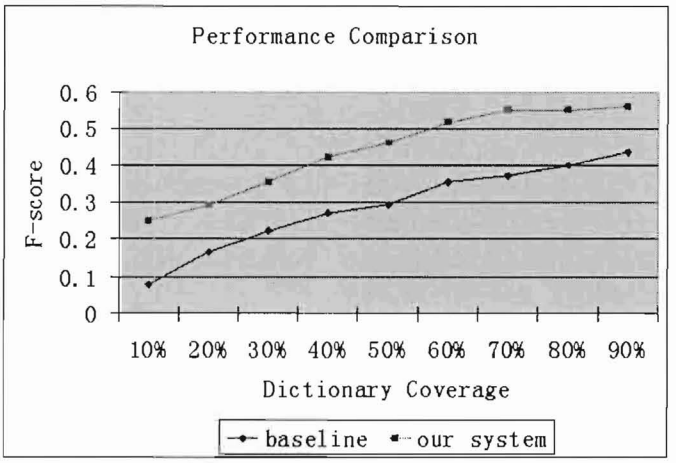

Figure 6.5: The overall F-scores of the prototype and baseline system under different dictionary coverages.

We observed that even using the 100% dictionary to label the corpus, the performance of our system is still about 0.1 F-score less than the skyline performance[5]. This may be due to the fact that we ignored the dictionary labels that contradict the true labels, and did not expand the positive set in the self-training process. As a result, the number of positive data available for our system is smaller than that for the skyline system. Thus the trained model is less accurate. Also note that the improvement of the self-training upon the baseline was nearly constant (about 0.15 F-score) for every size of the used dictionary, which coincides with that of the skyline over the baseline. These observations suggest that, given the learning ability of the base learner, properly expanding the positive set would be the key to further improving the final performance of the self-training method.

[5]Note that the skyline performance is not shown in Figure 6.5, because it assumes all available annotation are used for training and does not vary with the dictionary coverage.

6.4 Related Work

6.4.1 Supervised Learning in NER

A number of supervised learning algorithms have been explored in the newswire domain, for example, HMM [78], Maximum Entropy [9], SVM [72], and CRF [73]. Supervised learning based systems have shown better performance than rule-based systems, e.g., [139]. In the biomedical domain, the representatives are SVM [65], HMM [140], Maximum Entropy [66] and CRF [104]. Two best-known shared tasks held in this area are BioNLP [56] and BioCreAtIvE [131], where most participants used supervised learning techniques.

6.4.2 Using Unlabelled Data in NER

The idea of employing unlabelled data was first introduced to the NLP community by Yarowsky [130] for the task of word sense disambiguation. Collins and Singer [26] probably were the first to use unlabelled data for NE classification. Riloff and Jones [96] proposed a bootstrapping method for NER in web documents by iteratively learning language patterns from unlabelled text. Ando and Zhang [5] presented a structural learning paradigm for semi-supervised learning, which aims to learn the most predictive low-dimensional feature projection from unlabelled data. Shen et al [107] studied active learning for NER by using multi-criteria in example selection.

6.4.3 Using Unlabelled Data in Text Classification

Much previous work using unlabelled data involved text classification (or document categorization), where semi-supervised learning was studied along two lines. One line of work requires a small set of labelled texts and tries to use a large set of unlabelled texts to augment the labelled set. Blum and Mitchell [8] proposed co-training which uses two distinct views of each document. Nigam and Ghani [86] studied the effectiveness and applicability of co-training. Joachims [52] introduced Transductive SVMs to minimize misclassifications of unlabelled documents. We did not use the popular co-training algorithm, because it has two conditions: (1) it requires a natural split of the feature set, and (2) each subset of features is sufficient to train a reliable classifier. In the case of biomedical NER, the second condition seems hard to satisfy, because currently the best performance (F-score 70%) was achieved by using all the available features plus some post-processing heuristics [56].

Another line of work addresses practical tasks where only positive (but no negative) examples and unlabelled examples are provided. In order to build a classifier, one has to first identify reliable negative examples. Algorithms proposed for this purpose include Spy in S-EM [68], 1-DNF in PEBL [136], and NB in [67]. Once reliable negative examples have been selected, an iterative process like that in semi-supervised learning can be applied, for example, Mapping-Convergence of SVM in [136], and NB plus EM in [68].

Our work is more similar to the second line of work. However, ours is different in that we use a dictionary to label the text instead of requiring any human involvement. Besides, we use an SVM to identify reliable negatives, and let the SVM teach itself in later iterations. Also, the task we study is NER, which is different from text classification. To our knowledge, [53] is probably the most similar work to ours, which studied semi-supervised learning for general NER. However, their approach required preliminary chunking of syntactic categories (i.e., noun phrases, verb phrases, and prepositional phrases). In other words, they only considered the potential semantic relations between a noun phrase and its context. If their work can be considered phrase-based, ours is purely token-based in the sense that we do not assume any phrase-chunking as pre-processing.

6.5 Chapter Summary

We address the problem of how to recognize named entities in biological text in the absence of any human annotated corpus. The idea is to use a dictionary lookup to label the training sentences. Given the positive words labelled by the dictionary, we design an SVM-based self-training algorithm to identify negative words from unlabelled text so as to build an accurate classifier.

The purpose is to minimize the requirement of annotated corpora by traditional supervised learning algorithms. Our preliminary experiments suggest that this is possible, largely because the used dictionary is actually made by the domain experts. In this sense, the idea is somewhat equivalent to asking the dictionary to do the annotation job (as a dummy annotator).

We plan to study the following issues based on this work. First, whether the method can be applied to NER tasks in other domains (e.g., clinical text), so as to develop the proposed technique into a general method. Although our intuition suggests that this is the case, we still need to empirically demonstrate it. Second, we shall study how to improve

the performance up to the level achieved by traditional supervised learning. Third, how to effectively identify reliable positive data from the unlabelled data. We believe the solution to the third issue will finally lead to solving the second issue.

Another interesting direction would be to apply our method in active learning for NER. In contrast to selecting the most reliable predictions as we were doing in this work, we could select the most unreliable ones for humans to label.

Chapter 7

BioNER in Chinese Texts

Most research on biomedical named entity recognition has focused on English texts, e.g., MEDLINE abstracts. However, recent years have also seen significant growth of biomedical publications in other languages. For example, the Chinese Biomedical Bibliographic Database[1] has collected over 3 million articles published after 1978 from 1600 Chinese biomedical journals. We present here a Conditional Random Fields (CRF) based system for recognizing biomedical named entities in Chinese texts. Viewing Chinese sentences as sequences of characters, we trained and tested the CRF model using a manually annotated corpus containing 106 research abstracts (481 sentences in total). The features we used for the CRF model include word segmentation tags provided by a segmenter trained on newswire corpora, and lists of frequent characters gathered from training data and external resources. Randomly selecting 400 sentences for training and the rest for testing, our system obtained an 68.60% F-score on average, significantly outperforming the baseline system (F-score 60.54% using a simple dictionary match). This suggests that statistical approaches such as CRFs based on annotated corpora hold promise for the biomedical NER task in Chinese texts.[2]

[1] http://www.imicams.ac.cn/cbm/index.asp

[2] A paper based on the study described in this chapter has been accepted for oral presentation in the 21th Canadian Conference on Artificial Intelligence (AI-2008) [44].

7.1 Introduction

Our study, and most previous work in BioNER, has actually focused on biomedical research papers that are written in English. This is probably due to the availability of the well-known MEDLINE repository[3], about 90% of whose collection is originally written in English. While MEDLINE is currently and will likely remain the largest collection of English biomedical research publications, significant growth has also been seen in other languages. Take Chinese biomedical literature as an example. There are 5 large Chinese biomedical bibliographic databases, and they have collected millions of articles from at least 2500 Chinese biomedical journals, only 6% of which are indexed by MEDLINE [127]. With the increasingly large volume of these research papers becoming available, we believe that there will soon be strong needs for biomedical NER as well as for mining deeper knowledge in Chinese and other non-English languages.

In this chapter, we study biomedical NER in Chinese texts. To the best of our knowledge, this topic has not been studied so far. Previous research has shown that for the counterpart task in English texts, the state-of-the-art performance can be achieved by statistical approaches based on annotated corpora. Thus it is natural for us to assume that these approaches would also work for Chinese texts. To validate this hypothesis, however, we face an immediate difficulty: currently there are no annotated Chinese corpora in the biomedicine domain publicly available. Moreover, it is not clear what kinds of features should be used for the task.

We have created a small annotated corpus by ourselves, on which we trained and tested a CRF model. Given that there are no spaces between Chinese words and that doing word segmentation as a preprocessing step might introduce extra errors, we considered the NER task as a character-based sequence labeling problem. We used lists of frequent characters obtained from the training set and some external resources to construct features for the CRF model. We also tried to use word segmentation labels as features. We obtained encouraging results (Precision(P)/Recall(R)/F-Score(F) = 73.27% / 64.65% / 68.60%), which significantly outperforms our dictionary-based baseline system (P/R/F = 69.89% / 52.88% / 60.14%). The results suggest that statistical approaches based on annotated corpora should still be promising for biomedical NER in Chinese texts.

The rest of the chapter is organized as follows: Section 7.2 summarizes the properties of

[3]http://www.ncbi.nlm.nih.gov/pubmed/

the biomedical NEs in Chinese texts. Section 7.3 describes how the annotation was done. Section 7.4 recaps the CRF framework for sequence labeling and explains how it was used for the Chinese NER. Section 7.5 gives evaluation details, Section 7.6 reviews previous related research, and Section 7.7 concludes this chapter.

7.2 Properties of Biomedical Named Entities in Chinese Texts

Previous studies in the newswire domain have shown that Chinese NER is in general more difficult than English NER. First, while capitalization plays a important role in English names, there is no such information in Chinese text. Second, English words are naturally separated from each other by spaces, however, there is no space between Chinese words. Chinese word segmentation is in itself a difficult task, whose overall performance on newswire texts is still awaiting further improvement.

By reading the Chinese abstracts and consulting domain experts, we observed the following language phenomena that could make Chinese Biomedical NERs more complicated than their English counterparts:

1. mixed use of Chinese words and their English counterparts or synonyms, e.g., 人ACAT1基因P7启动子 (meaning "human ACAT1 gene P7 initiator"), can also be written as: 人酰基辅酶A：胆固醇酰基转移酶1（ACAT1）基因P7启动子；

2. One English term may have different Chinese translations, e.g., *Aptamer* has several equivalent translations in use: 寡核苷酸配基, 核酸适体, Aptamer, RNA适体, 核酸适配子;

3. Chinese has its own printing form for punctuation, special symbols, and even English letters. Many times they are mixed with English counterparts, e.g., periods can appear as "." or "。", dashes can be "-" or "—", commas can be "," or "，", English letters GP can be "GP" or "G P";

4. One term may have spelling variations, often involved in English and Chinese characters, e.g., ACAT1, A-CAT1, A—CAT1; 人白细胞，人类白细胞 (meaning: human leukocytes);

7.3 Annotating a Corpus for Chinese Biomedical NER

Our core annotation was performed by a Chinese Biochemistry PhD student who is familiar with biomedical terms in both English and Chinese. These annotations were then double-checked by the first author, who is also a native Chinese speaker. As most biomedical NER projects are mainly interested in proteins and genes, we decided to only annotate these two types of entities. We issued the query " (酶 or 受体 or 抗体) and 基因 "(meaning: {(enzyme or receptor or antibody) and gene}) to the CQVIP portal[4], which returned us 287 abstracts. We downloaded all of them and selected 103 for the annotation, discarding those containing too few (less than 5) protein/gene terms. One of the abstracts reads as follows:

> 克隆并测定人酰基辅酶A：胆固醇酰基转移酶1（ACAT1）基因P7启动子的序列，进一步探讨ACAT1基因表达的转录调控特点并分析其特异性转录位点。根据GenBank数据库提供的人A-CAT1基因P7启动子核苷酸序列，应用PCR方法从人单核细胞系THP-1扩增分离出ACAT1基因P7启动子全长片段，将PCR产物克隆入T载体，并对所获得的序列进行生物学信息分析。经琼脂糖凝胶电泳及直接测序鉴定，克隆的ACAT1基因P7启动子片段碱基序列与GenBank数据库一致。成功克隆了ACAT1基因P7启动子，为研究在动脉粥样硬化过程中ACAT1基因的转录调控机制奠定基础。

Similar to the GENIA corpus, the annotation guideline was to mark all character sequences that are either proteins or genes according to the context. Embedded named entities are also required to be marked. The original annotation was marked by the first annotator using color markers on paper. Double-checking was done by both annotators going through the annotation together. The annotation was then typed into the computer using the annotation tool Callisto (available from http://callisto.mitre.org/). We then converted the annotation from Callisto's XML format into IOB2 format, where B denotes the beginning character of an NE, I denotes each following character that is inside the NE, and O denotes a character that is not a part of any NE, to agree with the convention of the NER community.

[4] 维普资讯 , http://www.cqvip.com/. According to [127], which compares accessibility and coverage of five large Chinese biomedical bibliographic databases, CQVIP seems to have the best coverage, and allows free abstract search.

7.4 Chinese Biomedical NER using CRF

7.4.1 A Recap of the CRF Model

We will view Chinese sentences as sequences of characters, and use CRFs to model them. Recall that we have described the CRF framework in Chapter 3. For the reader's convenience, we will recap the basic idea of CRFs below, and refer readers to Chapter 3 for more details.

CRFs are undirected graphical models for calculating the conditional probability of output vertices based on input ones. While sharing the same exponential form with maximum entropy models, they have more efficient procedures for complete, non-greedy finite-state inference and training.

A linear chain CRF model defines the conditional probability of a state sequence $s =< s_1, s_2, ..., s_n >$ given an input sequence $x =< x_1, x_2, ..., x_n >$ to be:

$$P(s|o) = \frac{1}{Z_o} exp(\sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j f_j(s_{i-1}, s_i, o, i)) \tag{7.1}$$

where Z_o is a normalization factor of all state sequences, $f_j(s_{i-1}, s_i, o, i)$ is one of m functions that describes a feature, and λ_j is a learned weight for each such feature function.

CRFs allow us to utilize a large number of observation features as well as different state sequence based features and actually any other types of features we want to add. Note that there are other models, e.g., SVMs as described in Chapter 3 that can also deal with a large number of features. However, previous work [73] has shown that CRFs usually work better for sequence analysis. This has been also observed in our experimental results in Chapter 3, suggesting that CRFs tend to have better NER performance than SVMs in English text. Therefore, we decide to use CRFs in our evaluation below.

7.4.2 Chinese NER as a Sequence Labeling Task

We view each sentence as a sequence of characters, each of which is associated with a tag. The tag indicates whether the character is part of an NE, and the location if it is within the NE. Assume there are k types of entities under consideration, we can reduce the NER problem to the problem of assigning one of $2k + 1$ tags to each character. For example, in our small annotated corpus, we label two types of entities: protein and gene. If we are

going to recognize each type, the tag set will contain 5 tags: *B-PROTEIN, I-PROTEIN, B-GENE, I-GENE, O.* Thus we need to assign one of the five tags to each character.

Note that in our experiments to be described later, we did not try to distinguish between protein and gene due to the small scale of our annotated corpus. We did not consider embedded entities either. Rather, we considered both of them as an "ENTITY", an imaginary super type of protein and gene. That is, our tag set is *B-ENTITY, I-ENTITY, O*, and we want to assign one of them to each character.

For English NER tasks, a character-based model introduced in [58] proposed the use of substrings within English words. In Chinese NER, the character-based model is more straightforward, since there are no spaces between Chinese words, and each Chinese character is actually meaningful. Besides, using a character-based model can avoid errors made by a Chinese word segmenter. Moreover, there are currently no word segmentation tools available for the biomedicine domain.

7.5 Evaluation

With the above formulation, we train and test a CRF model on the annotated corpus. In our experiments, we use the CRF++ version 0.42[5] which implements the CRF model and is designed for typical NLP tasks including NER.

7.5.1 The Annotated Corpus

To better understand the text we are dealing with, we obtained some basic statistics from the corpus. It has 106 abstracts in total, containing 481 sentences or 38645 characters. There are 1199 distinct characters, including Chinese and English ones, and all numbers and special symbols. The top 20 frequent uni-gram, bi-gram and tri-gram characters are given in Table 7.1.

In total, there are 1062 mentions of 472 unique named entities, either proteins or genes, annotated in the corpus. The top 30 frequent entities are listed in Table 7.2. The longest entity is spelled as " β-半乳糖苷α1，2-岩藻糖转移酶（α1，2-fucosyltansferase，α1，2-FT） ", which is a protein and occurs only once in the corpus.

[5]CRF++: Yet Another CRF Toolkit, available at http://crfpp.sourceforge.net/.

1-gram characters	Frequency	2-gram characters	Frequency	3-gram characters	Frequency
的	894	基因	507	RNA	88
,	741	表达	289	DNA	84
基	595	细胞	222	PCR	78
A	577	NA	196	的表达	75
因	543	蛋白	123	mRN	60
。	493	PC	113	ERT	58
T	462	RN	96	TER	57
P	459	方法	96	hTE	57
R	442	RT	91	端粒酶	51
1	420	0.	91	cDN	47
C	399	CR	87	基因型	47
性	389	结果	86	. 05	46
0	377	序列	86	基因的	43
2	356	DN	84	多态性	41
表	352	检测	83	位基因	39
a	340	活性	83	表达,	39
N	303	. 0	79	等位基	39
达	302	hT	76	酶活性	38
e	286	的表	75	基因组	37
D	284	患者	72	P<0	36

Table 7.1: Top 20 uni-gram, bi-gram, tri-gram characters.

the Spelling of the Named Entity	the Entity Type	Frequency of Occurrences
端粒酶	PROTEIN	39
hTERT	PROTEIN	36
PEPC	PROTEIN	14
CHS	GENE	13
mRNA	GENE	13
CD147	PROTEIN	11
Survivin	PROTEIN	11
I-PROTEIN	GENE	10
VEGF	PROTEIN	10
survivin	GENE	9
MTHFR	PROTEIN	9
GUS	GENE	9
p53	PROTEIN	8
TS	PROTEIN	8
eNOS	PROTEIN	8
MARs	GENE	8
bax	GENE	7
ST13	GENE	7
MARs序列	GENE	7
SV40	GENE	7
p16基因	GENE	7
TIMP-2	PROTEIN	7
Ha-ras基因	GENE	7
pacs	PROTEIN	7
MOMP	PROTEIN	6
GSTM1	PROTEIN	6
hTNF α	PROTEIN	6
HpcagA	GENE	6
PCF	PROTEIN	6

Table 7.2: Top 30 frequent entities.

7.5.2 Feature Construction

To obtain a good estimation of the conditional probability of the tags assigned to the characters, we should use features that can distinguish between the characters. In our experiments, we used two types of features for the CRF model, i.e., character list features and word segmentation features.

We considered four character lists. The first two character lists were made from our corpus. We randomly selected 400 sentences from our corpus to form the training set, and used the remaining 81 sentences to form the testing set. From the training set, we counted character frequencies and selected those appearing in entities more than twice as the first list (called L1), and those appearing outside entities more than twice as the second list (called L2).

The third list was gathered from external resources. We downloaded from the Web a free dictionary[6] (in PDF format) containing Chinese translations of more than 20,000 human gene names from widely-used gene databases such as NCBI Gene[7] and GO[8]. We converted it into plain text format, and counted the character frequencies. We used the top 1000 frequent characters as the third list (called L3). We assume this list contains the characters that are most frequently seen in protein and gene names.

The fourth list was also from external resources. We retrieved the first 1000 abstracts from the CQVIP web site with the disjunction of four Chinese words: 酶, 抗体, 受体, 基因 (i.e., "enzyme or antibody or receptor or gene"). We used the top 2000 frequent characters as the fourth list (called L4). We assume this list contains the characters that are most frequently seen in Chinese biomedical abstracts.

Corresponding to the above four character lists, we have four columns of binary features for each character in the corpus. If the character is in any list, then we set the value for the corresponding feature to be 1; otherwise, it is set to 0.

We also considered using word segmentation information. We think this information may be useful in deciding entity boundaries. As we could not find a segmentation tool specifically for Chinese biomedical texts, we used the Stanford Chinese Word Segmenter[9]. Although it is trained on newswire texts, we expect it could still provide some basic word boundary

[6] http://wzhangcnster.googlepages.com/hg608.zip

[7] Entrez Gene, http://www.ncbi.nlm.nih.gov/sites/entrez?db=gene

[8] the Gene Ontology, http://www.geneontology.org

[9] http://nlp.stanford.edu/software/segmenter.shtml

the char	is it in L1?	is it in L2?	is it in L3?	is it in L4?	WS tag	NE tag
初	0	1	0	1	B	O
步	0	1	0	1	E	O
判	0	1	0	1	B	O
定	0	1	1	1	E	O
C	1	1	1	1	B	B-ENTITY
H	1	1	1	1	M	I-ENTITY
S	1	1	1	1	E	I-ENTITY
在	0	1	1	1	S	O
蓝	0	1	1	1	B	O
粒	1	1	1	1	E	O
小	0	1	1	1	B	O
麦	1	1	0	1	E	O
中	0	1	1	1	S	O

Table 7.3: Example data prepared in CRF++ format.

information that would be useful for our CRF model. We used the word segmenter's output to assign a segmentation tag for each character. The segmentation tags use the BMES notation, which are commonly used in word segmentation task, where B stands for beginning of a word, M for the middle of the word, E for the end of the word, and S stands for a word consisting of a single character.

Using the corpus and the feature construction described above, we prepared the training and testing data in the format of the CRF++ package, part of which is shown in Table 7.3.

We let the CRF++ package automatically generate feature functions for the CRF model, by defining the feature template provided by CRF++. Table 7.4 is the template we defined to use the characters themselves (i.e., those in column 1 of Table 7.3) and their occurrences in L1 (i.e., those in column 2 of Table 7.3). Detailed usage of the template and associated meanings are given at the CRF++ home page.

As an example of how the actual feature functions are automatically generated, consider the template "U03:%x[1,0]" in Table 7.4. CRF++ automatically generates a set of feature functions (func1 ... funcN) as in Table 7.5.

The total number of feature functions generated by this template amounts to $(L * N)$, where L is the number of output classes and N is the number of unique string expanded

template	data items used in feature
# Unigram	
U00:%x[-2,0]	判
U01:%x[-1,0]	定
U02:%x[0,0]	C
U03:%x[1,0]	H
U04:%x[2,0]	S
U05:%x[-2,0]/%x[-1,0]	判/定
U06:%x[-1,0]/%x[0,0]	定/C
U07:%x[0,0]/%x[1,0]	C/H
U08:%x[1,0]/%x[2,0]	H/S
U09:%x[-1,0]/%x[0,0]/%x[1,0]	定/C/H
U10:%x[-2,1]	0
U11:%x[-1,1]	0
U12:%x[0,1]	1
U13:%x[1,1]	1
U14:%x[2,1]	1
U15:%x[-2,1]/%x[-1,1]	0/0
U16:%x[-1,1]/%x[0,1]	0/1
U17:%x[0,1]/%x[1,1]	1/1
U18:%x[1,1]/%x[2,1]	1/1
U19:%x[-1,1]/%x[0,1]/%x[1,1]	0/1/1
# Bigram	
B	

Table 7.4: The meaning of CRF++ feature template, assuming the current character is "C" in Table 7.3

func1 = 1, if (tag = B-GENE and char ="H")
func2 = 1, if (tag = I-GENE and char ="H")
func3 = 1, if (tag = B-PROTEIN and char = "H")
func4 = 1, if (tag = I-PROTEIN and char = "H")
func5 = 1, if (tag = O and char ="H")
...

Table 7.5: Example feature functions generated from the template

Features	Accuracy	Precision	Recall	F-score
Baseline	90.10	68.77	53.79	60.19
	1.71	4.92	6.53	5.15
C	93.44	74.12	57.98	64.98
	1.35	4.15	6.06	5.15
C+L1	93.55	72.69	60.05	65.70
	1.41	5.06	6.43	5.64
C+L2	93.66	73.45	58.34	64.96
	1.27	4.36	5.97	5.17
C+L3	93.56	73.54	58.17	64.88
	1.31	4.28	6.04	5.16
C+L4	93.48	73.81	58.17	64.98
	1.30	4.17	5.91	5.03
C+WS	93.72	74.36	61.22	67.09
	1.29	3.81	5.16	4.35

Table 7.6: Performance of CRF vs. baseline average over 50 runs

from the given template.

7.5.3 Experimental Results

As mentioned ealier, our annotated corpus contains 481 labeled sentences in total. We randomly selected 400 sentences for training and the remaining 81 for testing. Though we have annotated two types of NEs: protein and gene, we combined them into one type "ENTITY" in our experiments, as the size of the corpus may not be sufficient to support fine classification. Thus the NER task became that of assigning one of three tags to each character: *B-Entity, I-Entity, O.*

For comparison purposes, we built a baseline system, which implements a simple dictionary matching strategy to label the testing set with a dictionary consisting of all distinct NEs found in the training set. To see how useful the different types of features are for the CRF performance, we tried different combinations of the features. The averaged results of exact NE boundary matches over 50 random runs are given in Table 7.6, where "C+L1" means the character and L1 are used as features and so on. In each cell, the number below is the standard deviation.

Hypothesis	p-value
mean(baseline) $\neq$ mean(C)?	1.014e-05
mean(C) $\neq$ mean(C+L1)?	0.5096
mean(C+L1) $\neq$ mean(C+L2)?	0.4982
mean(C+L2) $\neq$ mean(C+L3)?	0.9395
mean(C+L3) $\neq$ mean(C+L4)?	0.9223
mean(C+L4) $\neq$ mean(C+WS)?	0.0275

Table 7.7: The Welch two-sided t-test of two paired sample means for Table 7.6.

Note that the purpose of running the random experiments 50 times is to apply a statistical significance test (e.g., the Welch's t-test[10]) on the sample means, so that we do not assume any knowledge about the distribution of the samples. Probably due to the data sparsity resulted from the small scale of the training data, in some runs of our experiments, the performance of the learned CRF model was a bit worse than that of the baseline. However, it outperforms the baseline in most runs. Table 7.7 shows the results of the Welch two-sided t-test on paired samples of the averaged F-scores for the feature combinations used in Table 7.6. The calculation was done using **R**[11]. The t-test results confirm that the difference between the CRF models and the baseline is significant, suggesting that they can do significantly better than the baseline method.

From Table 7.6 and Table 7.7, we can see that: (1) the CRF model using the character features alone already outperforms the baseline system; (2) adding any of the four character lists as features does not result in any significant increase nor decrease in the F-score of using 'C' alone, while adding word segmentation tags as features can significantly improve the F-score; (3) the two greatest performance improvements over the baseline result from the "C+WS" combination and the "C+L1" combination. This shows that information about word segmentation and frequent entity characters are probably the most important features.

We then used the 'C+WS' combination as the base features and experimented the effects of adding the character list features. The averaged results over 50 independent random runs are given in Table 7.8, showing that the best performance is obtained by using all the available features, with 68.60% F-score vs 60.54% of the baseline. The corresponding Welch's t-test results are given in Table 7.9.

[10]http://en.wikipedia.org/wiki/Welch%27s_t_test

[11]The R Project for Statistical Computing, http://www.r-project.org/

Features	Accuracy	Precision	Recall	F-score
Baseline	90.32	68.44	54.54	60.54
	1.77	6.00	7.28	6.26
C	93.51	74.57	59.45	66.05
	1.42	4.59	6.65	5.58
C+WS	93.61	74.03	61.75	67.24
	1.32	4.49	5.76	4.81
C+WS+L1	93.62	73.90	63.69	68.32
	1.31	4.54	6.31	5.13
C+WS+L1+L2	93.85	73.26	64.40	68.44
	1.27	4.40	6.21	4.91
C+WS+L1+L2+L3	93.89	72.99	64.29	68.27
	1.29	4.55	6.15	5.02
C+WS+L1+L2+L3+L4	93.93	73.27	64.65	68.60
	1.25	4.46	6.07	4.93

Table 7.8: Performance of CRF vs. baseline averaged over 50 runs

Hypothesis	p-value
mean(baseline) $\neq$ mean(C)?	5.352e-06
mean(C) $\neq$ mean(C+WS)?	0.1270
mean(C+WS) $\neq$ mean(C+WS+L1)?	0.1415
mean(C+WS+L1) $\neq$ mean(C+WS+L1+L2)?	0.4512
mean(C+WS+L1+L2) $\neq$ mean(C+WS+L1+L2+L3)?	0.5660
mean(C+WS+L1+L2+L3) $\neq$ mean(C+WS+L1+L2+L3+L4)?	0.3723

Table 7.9: Welsh Two Sample two-sided t-test for Table 7.8

7.5.4 Discussion

We are a bit surprised by the impact of word segmentation information on the NER performance, because the word segmenter we used is trained on newswire corpora and may make many errors in segmenting biomedical texts. For example, in the following abstracts containing only 4 sentences, the segmenter has made at least 4 obvious errors. The errors are given in Table 7.10, together with the ideal segmentation determined by human.

> 目的：评价 自体 外周 血干 细胞 移植 联合 大剂量 化疗 治疗 恶性 血液 病 的 近期 疗效 。 方法：8 例 恶性 血液病（非霍奇 金淋 巴瘤 7 例，多 发性 骨髓瘤 1例）采用 化疗 联合 G - CSF 动员 外周 血干 细胞，Cobe Spectra血 细胞 分离机 采集 循环血 中 的 单 个 核 细胞，8 例 平均 单 个 核 细胞 数 3.0×10E8 • kg-1，平均 CD34 + 细胞 为 3.5 × 10E6 • kg-1，- 80℃ 冰箱 冻存 。 8例 均 在 大剂量 化疗 后 回输 细胞 。 结果：外周 血干 细胞 回输 后 WBC 恢复 至 2 × 10E9 • L - 1 平均 为 11d，血小板 恢复 至 50 × 10E10 • L - 1 平均 为 13d 。 8例 移植 后 短期 内均 达 缓解 。 结论：化疗 联合 G - CSF 是 一 高效 、 低毒 的 动员 方案；自体 外周 血干 细胞 移植 联合 大剂量 化疗 是 治疗 难治性 恶性 血液 病 的 一 种 安全 、 近期 疗效 肯定 的 治疗 方法 。

So why does the poor segmentation information help so much? We think the reason is two-fold. First, though trained on newswire texts, the segmenter provides nearly correct segmentation on the phrases or sentences similar to those in newswire texts. This information can help determine the NE boundaries. Second, many NEs in our corpus contain English abbreviations of the Chinese NEs, and as we have observed, the segmenter rarely made mistakes on these abbreviations. Thus the errors made by the segmenter might not hurt the NER performance.

7.6 Previous Work

The NER task was originally set to identify names of people, locations, and organizations in English newswire texts. Influential approaches make use of either handcrafted rules [76] or machine learning algorithms [7] or their combination [77]. Most recent works have adopted supervised learning algorithms, e.g., HMM [78], Maximum Entropy [9], SVM [72], and CRF [73]. The state-of-the-art performance (about 94%-96% F-scores) was achieved by supervised learning based systems (e.g., [139]).

Raw Phrase	自体外周血干细胞移植
English Meaning	Autologous peripheral blood stem cell transplantation
Segmenter Output	自体 外周 血干 细胞 移植
Human Output	自体 外周血 干细胞 移植
Raw Phrase	非霍奇金淋巴瘤
English Meaning	Non-Hodgkin's lymphoma
Segmenter Output	非霍奇 金淋 巴瘤
Human Output	非霍奇金 淋巴瘤
Raw Phrase	多发性骨髓瘤
English Meaning	Multiple myeloma
Segmenter Output	多 发性 骨髓瘤
Human Output	多发性 骨髓瘤
Raw Phrase	移植后短期内均达缓解
English Meaning	All reached remission shortly after the transplant
Segmenter Output	移植 后 短期 内均 达 缓解
Human Output	移植 后 短期 内 均 达 缓解

Table 7.10: Examples of phrases wrongly segmented by the Standford Chinese Segmenter

Recent years have also seen some NER works in Chinese newswire texts. The dominant approaches are supervised learning algorithms based on annotated corpora, e.g., HMM [137] and CRFs [13] [19] [34]. As the best performance of Chinese word segmentation is still not very satisfactory (about 85%-90% F-Score on newswire texts), and using it as a preprocessing step of NER would introduce errors affecting the final performance, recent works tend to treat the NER task as a character-based sequence tagging problem, similar to the Chinese word segmentation task.

In the biomedicine domain, supervised learning approaches have also been dominant. Representative works include SVM [65], HMM [140], Maximum Entropy [66] and CRF [104]. Two best-known shared tasks held in this area are BioNLP [56] and BioCreAtIvE [131], where most participants used supervised learning techniques. The best performance of the BioNLP 2004 competition was an F-score of about 73%. All these works target at English text.

Our work is different from all the above works in that we study Chinese NER in the biomedicine domain, which is a subarea of NER that has not been explored.

7.7 Chapter Summary

We have studied the problem of recognizing biomedical named entities in Chinese research abstracts. We adopted the statistical approach based on annotated corpora. The model we used is Conditional Random Fields, which has proven very effective in Chinese NER in newswire texts as well as biomedical NER in English abstracts. As there is no publicly available annotated Chinese corpus, we have created a small one by ourselves, which consists of 106 Chinese biomedicine abstracts. We trained and tested the CRF model on this small corpus, and obtained performance that is significantly better than the baseline system by gaining about 8% in F-score. We also experimented on combinations of different features, and found that among all the features, word segmentation information and frequent characters appearing in targeted entities might be the most useful ones. Our results suggest that the CRF model and perhaps other statistical models such as HMM and the Maximum Entropy model, which are based on annotated corpora, hold good promise on the biomedical NER task in Chinese texts. We hope that this work would elicit more attention to BioNER in Chinese language as well as to other non-English languages.

Chapter 8

Conclusion and Future Work

8.1 Summary of Main Results

In this thesis we have presented a number of results aiming to enhance and enrich the research of BioNER.

We have first presented a comprehensive survey of previous work in this area, from which we identified some key issues to address in this thesis.

Performance remains the most important issue for BioNER. As the dominant approaches are statistical machine learning based, it seems straightforward to assume that the models trained on larger annotated corpora would have better performance. However, our experimental results on the GENIA corpus, perhaps by far the best biomedical corpus available to public, suggest that this may not be a cost-effective way, since human annotation costs substantial amount of time and money and tends to be error-prone. This result is novel and is important for future work that attempts to improve the performance.

Nested (or Embedded) named entities are very common in biomedical texts but have been largely ignored in previous work. We have proposed a method to effectively recognize them. Our method assumes nested NEs have been annotated in the training corpus, where each word may have multiple NE tags. For all levels of the nesting, the main idea of the method is to learn a separate classification model for each level in the training phase, while applying the learnt model for the corresponding level in the testing phase. Our experiments show that this level-by-level method can identify both the non-nested NEs and nested NEs very well.

One of the difficulties in BioNER is to determine the NE boundaries. Previous work

based on statistical learning tends to combine the boundary detection and the entity type classification into one model. Motivated by the intuition that an NE is often an NP or part of it, and inspired by the good performance of NP-chunking systems, we proposed an NP-chunking based method to recognize the NEs. We have explored the benefits as well as the problems associated with such an approach. Our experimental results show that this method would be useful for BioNER with the help of rules that can determine the NE boundaries based on the NP chunks. As the state-of-the-art BioNER systems tend to use both machine learning techniques and manually-created rules, we expect the proposed method to provide a novel way to incorporate domain knowledge with machine learning.

Statistical machine learning approaches to BioNER require a sufficiently large annotated corpus to train a good model. In practice, however, annotating the large corpus is often labor-intensive and time-consuming as well as error-prone. We have proposed a method that can do BioNER in the absence of annotated corpus. The idea is to make use of an existing dictionary of NEs to label sentences, and to use these partially labeled sentences to iteratively train a classification model in the manner of semi-supervised learning. We have designed an SVM-based self-training algorithm for this purpose. Our experiments validate the potential usefulness of this approach.

Most previous work has focused on English texts, and to the best of our knowledge, none has addressed Chinese biomedical texts. Conceiving similar needs for other non-English languages, we have studied the BioNER task in Chinese biomedical texts. We manually annotated a small set of Chinese biomedical abstracts. We then applied CRF, a state-of-the-art statistical machine learning algorithm suitable for sequence labeling tasks, to build a classification model for BioNER. Considering the fact that there are no spaces between Chinese words, and Chinese word segmentation itself is an unsolved problem, we decided to build a character-based model. Our experiments show that the CRF learnt from the small corpus significantly outperforms the baseline system that uses a simple dictionary matching for NER. We also experimented on different types of features to find out which would be useful for the task.

Although the above methods are proposed and evaluated for the BioNER task, we believe that they are potentially applicable to other NLP tasks.

8.2 Future Work

In general, we observe that the research of NER has been progressing in the following directions:

- **Higher Performance:** to pursue higher performance aiming at human-like performance (e.g., [139]);
- **More Languages:** to handle more languages, from English to Non-English (e.g., Chinese texts as studied in this thesis), from single language to multiple languages (i.e., cross-lingual and multilingual texts, e.g., as in [29]);
- **More Domains:** to adapt to more domains, from general (e.g., newswire) to specific, for example biomedicine (as in this thesis), chemistry [119], astronomy [82], and law [18].
- **More Entity Types:** to identify more types of entities in the given domains. For example, the Sekine's Extended Named Entity Hierarchy [102] has defined nearly 200 NE types.

This thesis has focused on the specific biomedicine domain, studying five subtopics as recapped in the previous section. In previous chapters, we have discussed the future work for each subtopic. As a summary, we identify the following directions as future work in the biomedical domain:

- to look for ways to further improve the recognition performance. As discussed in Chapter 3, the state-of-the-art performance is still far less than human performance. It has been a bottleneck for downstream applications, waiting for major breakthroughs. Ideally, the community should make more efforts to produce higher quality corpora and introduce better machine learning algorithms. However, when these were not available, some of the methods proposed in this thesis should help for this purpose. For example, to use the method described in Chapter 4 to handle nested entities and the method described in Chapter 5 to help better determine NE boundaries. Another potential way would be to make use of clues beyond one sentence, e.g., the paragraph, the abstract, or even the full text of the article. The intuition behind this is that an NE might be mentioned multiple times in a text, and some might be easier to identify than others.

- to move from abstract to full text. Currently most research focus on abstracts only. However, an abstract is only a summary of a research, and many details about the NEs are covered in the full text. For the ultimate purpose of text mining in biomedical literature, BioNER systems must be able to handle full text. Intuitively, there would be remarkable linguistic differences between an abstract and a full text. Also there would be more noise contained in the full text. A simple migration from the former to the latter would lead to degraded performance. Therefore, it is necessary as well as non-trivial to investigate BioNER in the full text environment.

- to move from English to non-English languages (e.g., Chinese), and possibly mixed-language texts. Exploration along this direction can help advance biomedicine research by making research results published in different languages. For example, disease A is reported to be related to protein B in an English paper, while protein B is reported to be controlled by drug C in a Chinese paper. Then BioNER in different languages would help building the connection between the disease and the drug. The research along this direction may also benefit related areas such as machine translation of biomedical text, e.g., by mapping named entities written in different languages. The research challenge would involve the linguistic properties of different languages as well as different naming conventions in different languages.

- to move down along a concept hierarchy to identify more and finer entity types. Most work in BioNER only considers general entity types. For example, the GENIA corpus has actually annotated 36 entity types as defined in the GENIA Ontology (shown in Appendix Figure A.1), but when it was used as the training data for the BioNLP-2004 Shared Task, all the NEs in it were generalized to only five entity types, namely, *protein*, *DNA*, *RNA*, *cell type*, and *cell line*. According to the GENIA Ontology, the type *protein* actually has six sub-types, namely, *protein family or group*, *protein complex*, *individual protein molecule*, *subunit of protein complex*, *substructure of protein*, and *domain or region of protein*. When all NEs of various sub-types of protein were identified only as protein, the subtle distinction among the sub-types would be overlooked. Recognition of these sub-type NEs allows deeper understanding of their functions as well as the relations among them. Of course, the recognition would become more difficult as more ambiguity would be involved in the finer NE types. An immediate issue would be to prepare more annotations about the finer entity types.

- to improve the recognition speed in order to efficiently process huge amount of texts potentially as large as MEDLINE and CQVIP. This is actually a scalability issue: solving the problem using affordable computational resources in acceptable time. It is especially important for practical applications of BioNER, in order to keep up with the explosive growth of the biomedical literature. Though currently dominant in BioNER, supervised learning algorithms typically spend significant time in model training. Considerable time is also spent in processing raw text, e.g., creating features (the number could be up to several millions) for all the words. We note that this issue, although important, has been largely neglected by the BioNER community. We plan to study it in our future work.

In particular, some immediate extensions of this thesis are:

- How might changes in feature space affect the learning curves and performance of BioNER systems? For example, when the amount of data becomes much larger than that experimented in this thesis, is it possible that the flat pattern of the learning curves changes?

- To recognize nested NEs, would knowledge of NE tags of other nesting levels help? Intuitively, the answer is yes, but it is not clear to what extent.

- To build a good NP chunker for biomedical text. If available, it should benefit BioNER immediately, and very possibly some other biomedical text mining tasks, too.

- For BioNER without annotated corpora, how to effectively expand the positive set? If successful, such a research would bring good insights to the semi-supervised learning research.

- For Chinese BioNER, it would be interesting to map the recognized Chinese NEs to their English counterparts. This would contribute to machine translation of the named entities in different languages.

- Would some combinations of the approaches proposed in this thesis lead to better performance? For example, the combination of the explicit treatment of nested NEs with the NP chunk based approach.

Appendix A

More Statistics about the GENIA Corpus

The GENIA corpus (version 3.02) contains 97876 named entities (35947 distinct) of 36 types, and 490941 tokens (19883 distinct). There are 16672 nested entities, containing others or nested in others (the maximum embedded levels is four). Among all the outmost entities, 2342 are proteins and 1849 are DNAs, while there are 9298 proteins and 1452 DNAs embedded in other entities.

The GENIA entities are annotated based on the GENIA Ontology (Figure A.1). The meanings of the classes (types) used in the ontology are explained in the project's web page[1]. Note that while most GENIA entities are annotated as the leaf types of the ontology tree, some are not. For example, some NEs are labelled as the type *lipid*, which is a super type of the leaf type *steroid* in the ontology tree. Moreover, some entity types are not shown in the ontology tree, but still appear in the annotation. For example, *DNA_N/A*, *RNA_N/A*, and *protein_N/A*, all probably referring to corresponding subtypes which are not defined in the ontology.

The distribution of entity occurrences over the 36 types is shown in Table A.1. Some entities appear more frequent than others, ranging from 21511 for *protein molecule* to 2 for *RNA-substructure*. The distribution perfectly fits the famous Zipf's law[2]: given a corpus of natural language utterances, the frequency of any word is inversely proportional to its rank

[1] http://www-tsujii.is.s.u-tokyo.ac.jp/ genia/topics/Corpus/genia-ontology.html

[2] http://en.wikipedia.org/wiki/Zipf%27s_law

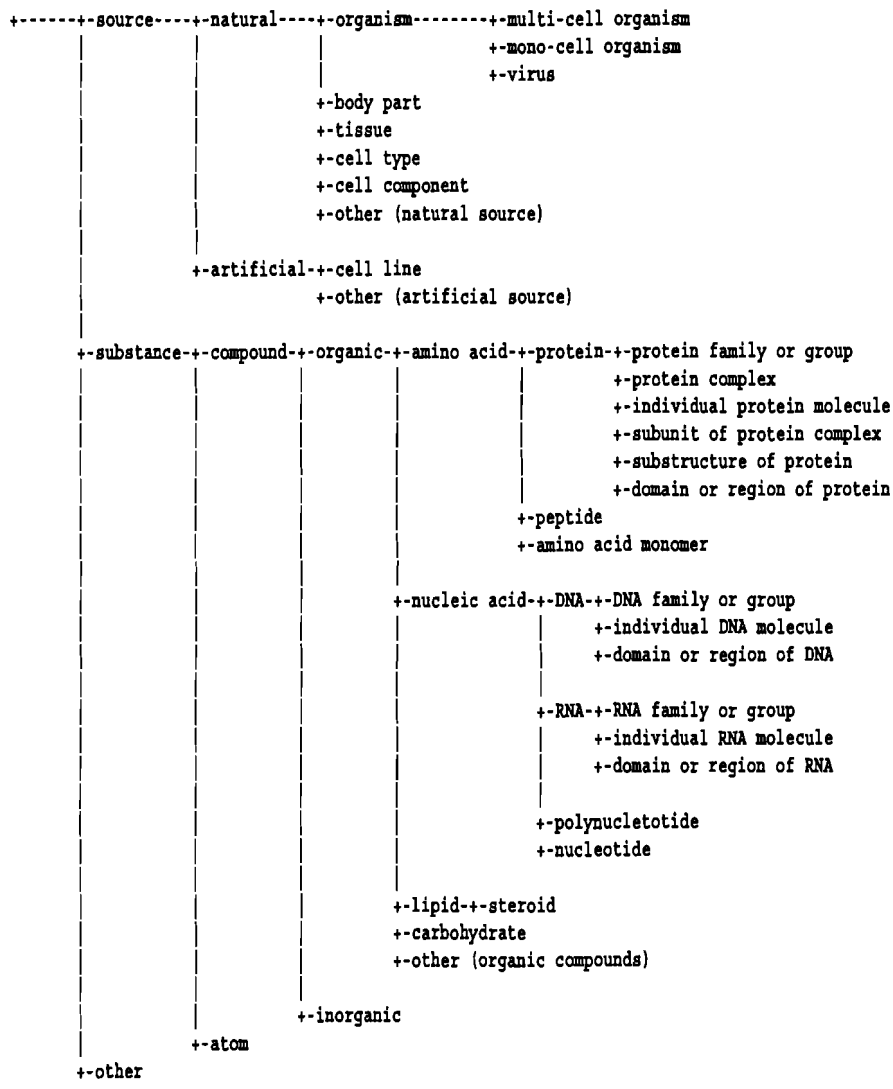

Figure A.1: The GENIA Ontology

Entity type	Occurrences	Entity type	Occurrences
G#protein_molecule	21511	G#DNA_molecule	529
G#other_name	20055	G#peptide	518
G#protein_family_or_group	8247	G#body_part	438
G#DNA_domain_or_region	7810	G#atom	340
G#cell_type	7021	G#RNA_family_or_group	332
G#other_organic_compound	4081	G#polynucleotide	258
G#cell_line	3846	G#inorganic	255
G#protein_complex	2397	G#nucleotide	236
G#lipid	2357	G#mono_cell	222
G#virus	2117	G#other_artificial_source	207
G#multi_cell	1745	G#protein_substructure	127
G#DNA_family_or_group	1511	G#DNA_substructure	106
G#protein_domain_or_region	990	G#carbohydrate	97
G#protein_subunit	895	G#protein_N/A	97
G#amino_acid_monomer	780	G#DNA_N/A	48
G#tissue	678	G#RNA_domain_or_region	39
G#cell_component	662	G#RNA_N/A	14
G#RNA_molecule	557	G#RNA_substructure	2

Table A.1: Distribution of occurrences of 36 types of entities in the GENIA corpus

in the frequency table. In our case, though, we consider the ranks and the frequencies of the 36 NE types, instead of the words. The curve-fitting result is shown in Figure A.2. The resulting function is $y = 24709.7x^{-0.978659}$, where x is the rank of an NE type and y is the frequency of the type.

Table A.2 shows the names of top 50 frequent entities together with their types and frequencies, most of which are *protein molecules*.

As we mentioned in Chapter 4, the NEs in GENIA can appear in a nesting structure, as outermost NEs, innermost NEs or in the middle of the nesting. The distribution of these occurrences for the 36 entity types are shown in Table A.3. We can see that NEs of type *other name* most likely appear outermost in a nesting structure (overall 7521 occurrences), while NEs of type *protein molecule* most likely appear innermost in the nesting (overall 6349 occurrences).

Table A.4 shows the frequencies of different nesting patterns in the GENIA corpus. Here the 36 entity types are generalized to six super-types, namely, *protein*, *DNA*, *RNA*, *cell type*, *cell line*, and *other name*. From this table we can see that when being the outer entity of a nesting structure, an NE of type *other name*, *protein*, *DNA*, *RNA* or *cell line*, most likely contains a protein as an inner entity, while a *cell type* entity most likely contain another *cell type* entity inside (the corresponding lines shown in bold fonts).

Entity Name	Entity Type	Occurrences
NF-kappa B	G#protein_molecule	1094
IL-2	G#protein_molecule	742
T cells	G#cell_type	694
NF-kappaB	G#protein_molecule	659
patients	G#multi_cell	587
HIV-1	G#virus	417
AP-1	G#protein_N/A	403
LPS	G#lipid	396
transcription factors	G#protein_family_or_group	389
IL-4	G#protein_molecule	383
transcription factor	G#protein_family_or_group	337
apoptosis	G#other_name	327
monocytes	G#cell_type	315
tyrosine	G#amino_acid_monomer	298
TNF-alpha	G#protein_molecule	285
PMA	G#other_organic_compound	281
IFN-gamma	G#protein_molecule	254
cytokines	G#protein_family_or_group	227
lymphocytes	null	223
EBV	G#virus	221
c-Jun	G#protein_molecule	218
cytokine	G#protein_family_or_group	213
B cells	G#cell_type	211
IL-10	G#protein_molecule	208
c-Fos	G#protein_molecule	205
NFAT	G#protein_family_or_group	199
tax	G#protein_molecule	197
TCR	G#protein_molecule	190
HIV	G#virus	189
IL-6	G#protein_molecule	187
NF-AT	G#protein_molecule	183
RA	G#other_organic_compound	182
T lymphocytes	G#cell_type	177
Stat3	G#protein_molecule	174
CD40	G#protein_molecule	166
gene expression	null	162
GM-CSF	G#protein_molecule	161
CD28	G#protein_molecule	156
PKC	G#protein_molecule	153
tyrosine phosphorylation	G#other_name	151
Sp1	G#protein_molecule	150
GATA-1	G#protein_molecule	148
TNF	G#protein_family_or_group	148
dexamethasone	G#lipid	148
transcription	G#other_name	140
glucocorticoid receptor	G#protein_domain_or_region	140
GR	G#protein_family_or_group	140
IL-12	G#protein_molecule	138
p65	G#protein_subunit	137
p50	G#protein_subunit	135

Table A.2: Top 50 frequent entities in the GENIA corpus

entity type	occurrences as outmost NEs	occurrences as innermost NEs	occurrences as middle NEs
G#amino_acid_monomer	16	396	3
G#atom	16	182	0
G#body_part	11	52	3
G#carbohydrate	7	28	1
G#cell_component	47	32	2
G#cell_line	544	173	10
G#cell_type	498	791	9
G#DNA_domain_or_region	1499	1025	211
G#DNA_family_or_group	193	141	9
G#DNA_molecule	149	59	1
G#DNA_N/A	1	2	0
G#DNA_substructure	7	4	0
G#inorganic	9	45	0
G#lipid	54	660	3
G#mono_cell	1	43	0
G#multi_cell	168	149	2
G#nucleotide	1	107	0
G#other_artificial_source	47	8	2
G#other_name	**7521**	588	109
G#other_organic_compound	182	501	2
G#peptide	42	115	2
G#polynucleotide	18	11	0
G#protein_complex	293	750	9
G#protein_domain_or_region	126	51	5
G#protein_family_or_group	820	1724	82
G#protein_molecule	1006	**6349**	99
G#protein_N/A	11	14	0
G#protein_substructure	9	26	0
G#protein_subunit	77	174	15
G#RNA_domain_or_region	7	1	1
G#RNA_family_or_group	86	56	11
G#RNA_molecule	231	60	94
G#RNA_N/A	0	2	0
G#tissue	35	50	2
G#virus	53	1056	7

Table A.3: Numbers of outermost NNEs, middle NNEs and innermost NNEs

outer entity	inner entity	count
cell_line	cell_line	100
cell_line	cell_type	75
cell_line	DNA	8
cell_line	other_name	15
cell_line	**protein**	**204**
cell_type	cell_line	7
cell_type	**cell_type**	**327**
cell_type	other_name	26
cell_type	protein	87
cell_type	RNA	2
DNA	cell_line	3
DNA	cell_type	8
DNA	DNA	419
DNA	other_name	28
DNA	**protein**	**1394**
DNA	RNA	2
other_name	cell_line	29
other_name	cell_type	265
other_name	DNA	775
other_name	other_name	380
other_name	**protein**	**4418**
other_name	RNA	190
protein	cell_line	14
protein	cell_type	83
protein	DNA	143
protein	other_name	120
protein	**protein**	**2474**
protein	RNA	12
RNA	cell_line	2
RNA	DNA	56
RNA	**protein**	**348**
RNA	RNA	16

Table A.4: Counts of different types of nested entities in the GENIA corpus

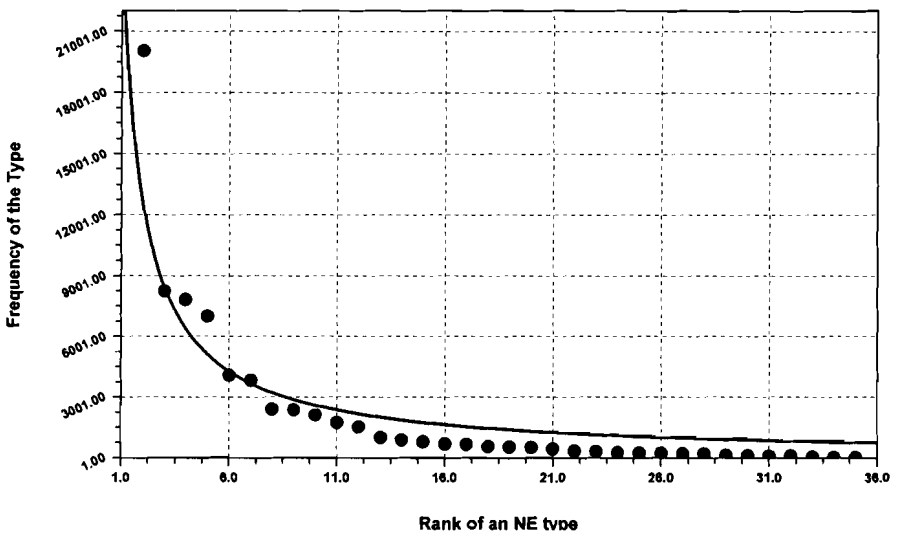

Figure A.2: The curve-fitting result of GENIA NE types' ranks and frequencies in terms of Zipf's law.

Table A.5 shows the top 20 frequent words (tokens) occurring different types of NEs of the GENIA corpus. Here the entities are generalized to five types, namely, *protein*, *DNA*, *RNA*, *cell type* and *cell line*. An interesting observation from this table is that the word *and* appears very frequently in all the five types of NEs, suggesting that many entities are formed by conjunction of smaller entities. This phenomena should be kept in mind when dealing with nested NEs as well as doing NER by NP chunking.

Table A.6 shows the top 20 frequent words appearing inside an NE and outside an NE. Here the entities are also generalized to five types, namely, *protein*, *DNA*, *RNA*, *cell type* and *cell line*. An interesting observation here is that the parentheses can frequently appear both inside and outside an NE, suggesting the complexity of biomedical entity names.

word (freq) in protein	word (freq) in DNA	word (freq) in RNA	word (freq) in cell type	word (freq) in cell line
B (1610)	gene (1385)	mRNA (665)	cells (3005)	cells (1680)
factor (1388)	promoter (1096)	transcripts (75)	T (1469)	cell (905)
protein (1298)	genes (589)	RNA (72)	human (780)	lines (504)
transcription (1124)	site (567)	((51)	lymphocytes (715)	line (431)
NF-kappa (1086)	((430)	) (51)	monocytes (509)	T (399)
receptor (885)	) (429)	**and (47)**	B (476)	human (360)
) (715)	element (395)	mRNAs (44)	blood (315)	Jurkat (298)
((714)	human (371)	c-jun (42)	peripheral (302)	U937 (181)
factors (693)	enhancer (353)	, (38)	cell (219)	B (147)
proteins (659)	**and (329)**	receptor (38)	mononuclear (214)	((134)
alpha (657)	sites (314)	c-fos (33)	**and (193)**	) (134)
NF-kappaB (605)	binding (309)	transcript (32)	macrophages (183)	**and (123)**
IL-2 (575)	region (288)	B (27)	leukocytes (142)	THP-1 (121)
nuclear (574)	elements (260)	1 (25)	activated (130)	T-cell (116)
kinase (528)	sequence (254)	beta (25)	) (120)	clones (92)
kappa (454)	B (251)	alpha (23)	((119)	monocytic (86)
receptors (413)	IL-2 (201)	GR (22)	primary (119)	HeLa (85)
and (382)	DNA (193)	IL-2 (20)	normal (115)	leukemia (85)
I (382)	LTR (187)	c-myc (17)	endothelial (104)	HL-60 (81)
AP-1 (380)	reporter (184)	IL-6 (17)	erythroid (103)	K562 (76)

Table A.5: Top 20 frequent words inside GENIA entities

word in entities	freq	word outside entities	freq
cells	4771	of	21117
B	2511	the	19867
T	2118	.	18412
human	1743	,	17648
factor	1483	in	12857
gene	1465	and	12082
)	1449	to	7215
(	1448	a	6535
cell	1404	)	5748
protein	1339	(	5702
transcription	1207	that	5182
NF-kappa	1202	by	4746
promoter	1103	with	4321
and	1074	is	3966
receptor	1056	expression	3028
alpha	835	was	2933
IL-2	804	for	2828
lymphocytes	785	The	2542
factors	695	activation	2203
mRNA	669	as	1869

Table A.6: Top 20 frequent words inside and outside protein/DNA/RNA/cell-type/cell-line entities

Bibliography

[1] Beatrice Alex, Barry Haddow, and Claire Grover. Recognising nested named entities in biomedical text. In *Proceedings of BioNLP Workshop*, 2007.

[2] Sophia Ananiadou. A methodology for automatic term recognition. In *Proceedings of COLING*, 1994.

[3] Sophia Ananiadou, Sylvie Albert, and Dietrich Schuhmann. Evaluation of automatic term recognition of nuclear receptors from medline. In *Genome Informatics Series*. 2000.

[4] Sophia Ananiadou and Goran Nenadic. *Text Mining for Biology and Biomedicine*, chapter Automatic Terminology Management in Biomedicine. 2005.

[5] Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of ACL*, 2005.

[6] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of HLT-NAACL*, 2003.

[7] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: A high-performance learning name finder. In *Proceedings Of The 5th Conference On Applied Natural Language Processing*, 1997.

[8] Avrim Blum and Tom Mitchell. Combining labelled and unlabelled data with co-training. In *COLT-1998*, 1998.

[9] Andrew Borthwick. *A Maximum Entropy Approach To Named Entity Recognition*. PhD thesis, New York University, 1999.

[10] Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proceedings of Workshop on Very Large Corpora 1998*, 1998.

[11] Eric Brill. A simple rule-based part-of-speech tagger. In *Proceedings of 3rd Conference on Applied Natural Language Processing*, 1992.

[12] Eric Brill. Some advances in rule-based part-of-speech tagging. In *Proceedings of AAAI*, 1994.

[13] Bob Carpenter. Character language models for chinese word segmentation and named entity recognition. In *Proceedings of SIGHAN Bakeoff*, 2006.

[14] Shing-Kit Chan, Wai Lam, and Xiaofeng Yu. A cascaded approach to biomedical named entity recognition using a unified model. In *Proceedings of Seventh IEEE International Conference on Data Mining (ICDM)*, 2007.

[15] Jeffrey T. Chang and Schutze Hinrich. *Text Mining for Biology and Medicine*, chapter Abbreviations in biomedical text. 2005.

[16] Jeffrey T. Chang, Schutze Hinrich, and Altman B. Russ. Creating an online dictionary of abbreviations from medline. *Journal of the American Medical Informatics Association*, 2002.

[17] Jeffrey T. Chang, Schutze Hinrich, and Altman B. Russ. Gapscore: Finding gene and protein names one word at a time. *Bioinformatics*, 2004.

[18] Mark Chaudhary, Christopher Dozier, Gregory Atkinson, Gary Berosik, Xi Guo, and Steve Samler. Mining legal text to create a litigation history database. In *Proceedings of Law and Technology (LawTech)*, 2006.

[19] Aitao Chen, Fuchun Peng, Roy Shan, and Gordon Sun. Chinese named entity recognition with conditional probabilistic models. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, 2006.

[20] Lifeng Chen, Hongfang Liu, and Carol Friedman. Gene name ambiguity of eukaryotic nomenclatures. *Bioinformatics*, 2005.

[21] Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. Chinese named entity recognition with conditional random fields. In *Proceedings of SIGHAN Bakeoff*, 2006.

[22] K. Bretonnel Cohen, Andrew Dolbey, George Acquaah-Mensah, and Lawrence Hunter. Contrast and variability in gene names. In *Proceedings of ACL Workshop on NLP in the Biomedical Domain*, 2002.

[23] Trevor Cohn, Andrew Smith, and Miles Osborne. Scaling conditional random fields using error-correcting. In *Proceedings of ACL*, 2005.

[24] Nigel Collier, Chikashi Nobata, and Junichi Tsujii. Extracting the names of genes and gene products with a hidden markov model. In *Proceedings of COLING*, 2000.

[25] Michael Collins. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proceedings of Annual Meeting of ACL*, 2001.

[26] Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of EMNLP*, 1999.

[27] Veronica Dahl and Baohua Gu. Semantic property grammars for knowledge extraction from biomedical text. In *Proceedings of 22nd International Conference on Logic Programming (ICLP)*, 2006.

[28] Veronica Dahl and Baohua Gu. A chrg analysis of ambiguity in biological texts (extended abstract). In *Proceedings of 4th International Workshop on Constraints and Language Processing (CSLP)*, 2007.

[29] Cesar de Pablo-Sanchez, Jose Luis, and Paloma Martinez. Named entity processing for cross-lingual and multilingual ir applications. In *Processing of SIGIR*, 2006.

[30] Francois Denis, Anne Laurent, Remi Gilleron, and Marc Tommasi. Text classification from positive and unlabelled examples. In *Proceedings of IPMU*, 2002.

[31] Nazife Dimililer and Ekrem Varoglu. Recognizing biomedical named entities using svms: Improving recognition performance with a minimal set of features. In *Proceedings of Workshop on Knowledge Discovery in Life Science Literature (KDLL)*, 2006.

[32] Shipra Dingare, Jenny Finkel, Malvina Nissim, Christopher Manning, and Claire Grover. A system for identifying named entities in biomedical text: How results from two evaluations reflect on both the system and the evaluation. *Comparative and Functional Genomics*, 2005.

[33] Sergei Egorov, Anton Yuryev, and Nikolai Daraselia. A simple and practical dictionary-based approach for identification of proteins in medline abstracts. *Journal of American Medical Information Association*, 2004.

[34] Yuanyong Feng, Le Sun, and Yuanhua Lv. Chinese word segmentation and named entity recognition based on conditional random fields models. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, 2006.

[35] Jenny Finkel, Shipra Dingare, Huy Nguyen, Malvina Nissim, Christopher Manning, and Gail Sinclair. Exploiting context for biomedical entity recognition: from syntax to the web. In *Proceedings of JNLPBA*, 2004.

[36] Kristofer FRANZEN, Gunnar ERIKSSON, Fredrik OLSSON, Lars ASKER, Per LIDEN, and Joakim CASTER. Protein names and how to find them. *International Journal of Medical Informatics*, 2002.

[37] K. Fukuda, A. Tamura, T. Tsunoda, and T. Takagi. Toward information extraction: Identifying protein names from biological papers. In *Proc. of Pacific Symposium on Bio-computing*, 1998.

[38] Robert Gaizauskas, George Demetriou, and Kevin Humphreys. Term recognition and classification in biological science journal articles. In *Proc. Workshop on Comp. Terminology for Medical and Biological Applications*, 2000.

[39] Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics*, 31(4), 2005.

[40] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *Proc.16th Int. Conf on Computational Linguistics*, 1996.

[41] Baohua Gu. Recognizing nested named entities in genia corpus. In *Proceedings of BioNLP*, 2006.

[42] Baohua Gu, Veronica Dahl, and Fred Popowich. Recognizing biomedical named entities in the absence of human annotated corpora. In *Proceedings of IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE)*, 2007.

[43] Baohua Gu, Feifang Hu, and Huan Liu. Modelling classification performance for large data sets – an empirical study. In *Proceedings of the Second International Conference of Web-Age Information Management (WAIM)*, 2001.

[44] Baohua Gu, Fred Popowich, and Veronica Dahl. Recognizing biomedical named entities in Chinese research abstracts. In *Proceedings of the 21th Canadian Conference on Artificial Intelligence (AI)*, 2008. (to appear).

[45] Mona Soliman Habib. Addressing scalability issues of named entity recognition using multi-class support vector machines. In *PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY*, volume 27, 2008.

[46] Udo Hahn and Joachim Wermter. *Text Mining for Biology and Biomedicine*, chapter Levels of Natural Language Processing for Text Mining. 2005.

[47] Daniel Hanisch, Juliane Fluck, Heinz-Theodor Mevissen, and Ralf Zimmer. Playing biology's name game: Identifying protein names in scientific text. In *Proc of Pacific Symposium on Bio-computing*, 2003.

[48] Lynette Hirschman, Alexander A. Morgan, and Alexander S. Yeh. Rutabaga by any other name: Extracting biological names. *Journal of Biomedical Informatics*, 2002.

[49] Jerry R. Hobbs, Douglas E. Appelt, John Bear, David J. Israel, Megumi Kameyama, Mark E. Stickel, and Mabry Tyson. Fastus: A cascaded finite-state transducer for extracting information from natural-language text. In *Finite-State Language Processing.* 1997.

[50] Wen-Juan Hou and Hsin-Hsi Chen. Enhancing performance of protein name recognizers using collocation 2003. In *Proceedings of ACL Workshop on NLP in Bio-medicine*, 2003.

[51] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, 1998.

[52] Thorsten Joachims. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*. 1999.

[53] Rosie Jones. *Learning to Extract Entities from Labelled and Unlabelled Text.* PhD thesis, Carnegie Mellon Univeristy, 2005.

[54] Junichi Kazama, Takaki Makino, Yoshihiro Ohta, and Junichi Tsujii. Tuning support vector machines for biomedical named entity recognition. In *Proceedings of ACL Workshop on NLP in the Biomedical Domain*, 2002.

[55] Junichi Kazama and Kentaro Torisawa. Exploiting wikipedia as external knowledge for named entity recognition. In *Proceedings of EMNLP-CoNLL*, 2007.

[56] Jim-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of International Joint Workshop on NLP in Biomedicine and Its Applications*, 2004.

[57] Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Junichi Tsujii. Genia corpus: A semantically annotated corpus for bio-textmining. *Bioinformatics*, 2003.

[58] Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D. Manning. Named entity recognition with character-level models. In *Proceedings of Conference on Computational Natural Language Learning*, 2003.

[59] Martin Krallinger. Biocreative - critical assessment of information extraction systems in biology. Web, 2004.

[60] Michael Krauthammer and Goran Nenadic. Term identification in the biomedical literature. *Journal of Biomedical Informatics*, 2004.

[61] Michael Krauthammer, Andrey Rzhetsky, Pavel Morozov, and Carol Friedman. Using BLAST for identifying gene and protein names in journal articles. *Gene*, 2000.

[62] Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *Proceedings of NAACL*, 2001.

[63] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[64] Matthew Lease and Eugene Charniak. Parsing biomedical literature. In *Proceedings of IJCNLP*, 2005.

[65] Ki-Joong Lee, Young-Sook Hwang, and Hae-Chang Rim. Two-phase biomedical ne recognition based on SVMs. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[66] Yi-Feng Lin, Tzong-Han Tsai, Wen-Chi Chou, Kuen-Pin Wu, Ting-Yi Sung, and Wen-Lian Hsu. A maximum entropy approach to biomedical named entity recognition. In *Proceedings of the 4th SIGKDD Workshop on Data Mining in Bioinformatics*, 2004.

[67] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. Building text classifiers using positive and unlabelled examples. In *Proceedings of International Conference on Data Management*, 2003.

[68] Hongfang Liu, Alan R. Aronson, and Carol Friedman. A study of abbreviations in medline abstracts. In *Proceedings of AMIA Symposium*, 2002.

[69] Hongfang Liu and Carol Friedman. Mining terminological knowledge in large biomedical corpora. In *Pacific Symposium on Bio-computing*, 2003.

[70] Hongfang Liu, Stephen B. Johnson, and Carol Friedman. Automatic resolution of ambiguous terms based on machine learning and conceptual relations in the UMLS. *Journal of American Medical Information Association*, 2002.

[71] W. H. Majoros, G. M. Subramanian, and M. D. Yandell. Identification of key concepts in biomedical literature using a modified markov heuristic. *Bioinformatics*, 2003.

[72] James Mayfield, Paul McNamee, and Christine Piatko. Named entity recognition using hundreds of thousands of features. In *Proceedings of CoNLL*, 2003.

[73] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature selection and web-enhanced lexicons. In *Proceedings of CoNLL*, 2003.

[74] Gabor Melli, Yang Wang, Yudong Liu, Mehdi M. Kashani, Zhongmin Shi, Baohua Gu, Anoop Sarkar, and Fred Popowich. Description of squash, the sfu question answering summary handler for the duc-2005 summarization task. In *Proceeding of Document Understanding Conference 2005 (DUC)*, 2005.

[75] Sven Mika and Burkhard Rost. Protein names precisely peeled off free text. *Bioinformatics*, 2004.

[76] Andrei Mikheev, Claire Grover, and Marc Moens. Description of the LTG system used for MUC-7. In *Proceedings of 7th Message Understanding Conference (MUC-7)*, 1998.

[77] Andrei Mikheev, Marc Moens, and Claire Grover. Named entity recognition without gazeteers. In *Proceedings of Conference of European Chapter of ACL*, 1999.

[78] Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. BBN: Description of the SIFT system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference*, 1998.

[79] Ruslan Mitkov. Anaphora resolution: The state of the art, 1999.

[80] Behrang Mohit and Rebecca Hwa. Syntax-based semi-supervised named entity tagging. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, 2005.

[81] Alex Morgan, Lynette Hirschman, Alexander Yeh, and Marc Colosimo. Gene name extraction using flybase resources. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[82] Tara Murphy, Tara McIntosh, and James R. Curran. Named entity recognition for astronomy literature. In *Proceedings of the 2006 Australasian Language Technology Workshop (ALTW)*, 2006.

[83] Meenakshi Narayanaswamy, K. E. Ravikumar, and K. Vijay-Shanker. A biological named entity recognizer. In *Proceedings of Pacific Symposium on Biocomputing*, 2003.

[84] Nature. Wanted: A new order in protein nomenclature (editorial opinion). *Nature*, 1999.

[85] Goran Nenadic, Irena Spasia, and Sophia Ananiadou. Mining biomedical abstracts: What is in a term? In *Proceedings of International Joint Conference on NLP*, 2004.

[86] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of CIKM*, 2000.

[87] Chikashi Nobata, Nigel Collier, and Junichi Tsujii. Automatic term identification and classification in biological texts. In *Proceedings of Natural Language Pacific Rim Symposium*, 1999.

[88] P.V. Ogren, K.B. Cohen, G.K. Acquaah-Mensah, J. Eberlein, and L. Hunter. The compositional structure of gene ontology terms. In *Proceedings of Pacific Symposium on Biocomputings*, 2004.

[89] Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. GENIA corpus: An annotated research abstract corpus in molecular biology domain. In *Proceedings of Human Language Technology Conference*, 2002.

[90] David D. Palmer. and David S. Day. A statistical profile of the named entity task. In *Proc. 5th Conf. on Applied Natural Language Processing*, 1997.

[91] Helen Pearson. Biology's name game. *Nature*, 2001.

[92] Georgios Petasis, Frantz Vichot, Francis Wolinski, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D. Spyropoulos. Using machine learning to maintain rule-based named-entity recognition and classification systems. In *Proceedings of ACL*, 2001.

[93] Carl Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. The University of Chicago Press, 1994.

[94] Denys Proux, Fransois Rechenmann, Laurent Julliard, Violaine Pillet, and Bernard Jacq. Detecting gene symbols and names in biological texts: a first step toward pertinent information extraction. In *Proceedings of Ninth Workshop on Genome Informatics*, 1998.

[95] Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *In Proceedings of the Third ACL Workshop on Very Large Corpora*, 1995.

[96] Ellen Riloff and Rosie Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of AAAI*, 1999.

[97] Thomas C. Rindflesch, Lawrence Hunter, , and Alan R. Aronson. Mining molecular binding terminology from biomedical text. In *Proceedings of the AMIA Annual Symposium*, 1999.

[98] Thomas C. Rindflesch, Lorraine Tanabe, and John N . Weinstein. EDGAR: Extraction of drugs, genes and relations from the biomedical literature. In *Proceedings of Pacific Symposium on Biocomputing*, 2000.

[99] Eric Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*, 2002.

[100] Erik Tjong Kim Sang. Np chunking. web, 2005.

[101] Jasmin Saric. Genia and other annotated corpora for bionlp. web, 2005.

[102] Satoshi Sekine. Sekine's extended named entity hierarchy. web, 2007.

[103] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A decision tree method for finding and classifying names in japanese texts. In *Proceedings of Workshop on Very Large Corpora*, 1998.

[104] Burr Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of JNLPBA*, 2004.

[105] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of NAACL*, 2003.

[106] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, , and Chew-Lim Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of ACL*, 2004.

[107] Dan Shen, Jie Zhang, Guodong Zhou, and Jian Su. Effective adaptation of hidden markov model-based named entity recognizer for biomedical domain. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[108] Hong Shen and Anoop Sarkar. Voting between multiple data representations for text chunking. In *In Proceedings of the Eighteenth Meeting of the Canadian Society for Computational Intelligence, Canadian AI*, 2005.

[109] Zhongmin Shi, Baohua Gu, Fred Popowich, and Anoop Sarkar. Synonym-based query expansion and boosting-based re-ranking: A two-phase approach for genomic information rretrieval. In *Proceedings of the Fourteenth Text REtrieval Conference (TREC)*, 2005.

[110] Beth M. Sundheim. Overview of results of the MUC-6 evaluation. In *Proceedings of the 6th Conference on Message Understanding*, 1995.

[111] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[112] Koichi Takeuchi and Nigel Collier. Bio-medical entity extraction using support vector machines. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[113] Lorraine Tanabe and W. John Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics*, 2002.

[114] Lorraine Tanabe and W. John Wilbur. Generation of a large gene/protein lexicon by morphological pattern analysis. *Journal of Bioinformatics and Computational Biology*, 2004.

[115] Yuka Tateisi, Tomoko Ohta, Nigel Collier, Chikashi Nobata, and Junichi Tsujii. Building an annotated corpus from biology research papers. In *In the Proceedings of COLING 2000 Workshop on Semantic Annotation and Intelligent Content*, 2000.

[116] James Thomas, David Milward, Christos Ouzounis, Stephen Pulman, and Mark Carroll. Automatic extraction of protein interactions from scientific abstracts. In *Proc of Pacific Symposium on Biocomputing*, 2000.

[117] Manabu Torii, Sachin Kamboj, and K. Vijay-Shanker. An investigation of various information sources for classifying biological names. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[118] Manabu Torii and K. Vijay-Shanker. Using unlabeled medline abstracts for biological named entity classification. In *Proceedings of Genome Informatics Workshop*, 2002.

[119] Joe Townsend, Ann Copestake, Peter Murray-Rust, Simone Teufel, and Chris Waudby. Language technology for processing chemistry publications. In *Proceedings of the UK e-Science All Hands Meeting*, 2005.

[120] Richard Tzong-Han Tsai, Shih-Hung Wu, Wen-Chi Chou, Yu-Chun Lin, Ding He, Jieh Hsiang, Ting-Yi Sung, and Wen-Lian Hsu. Various criteria in the evaluation of biomedical named entity recognition. *BMC Bioinformatics*, 2006.

[121] Yoshimasa Tsuruoka and Junichi Tsujii. Boosting precision and recall of dictionary-based protein name recognition. In *Proceedings of ACL Workshop on NLP in Biomedicine*, 2003.

[122] Yoshimasa Tsuruoka and Junichi Tsujii. Probabilistic term variant generator for biomedical terms. In *Proceedings of 26th Annual ACM SIGIR Conference*, 2003.

[123] O. Tuason, L. Chen, H. Liu, J.A. Blake, and C. Friedman. Biological nomenclature: A source of lexical knowledge and ambiguity. In *Proceedings of Pacific Symposium on Biocomputing*, 2004.

[124] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[125] Chia-Wei Wu, Shyh-Yi Jan, Tzong-Han Tsai, and Wen-Lian Hsu. On using ensemble methods for chinese named entity recognition. In *Proceedings of SIGHAN Workshop*, 2006.

[126] Youzheng Wu, Jun Zhao, Bo Xu, and Hao Yu. Chinese named entity recognition based on multiple features. In *Proceedings of HLT/EMNLP*, 2005.

[127] Jun Xia, Judith Wright, and Clive E. Adams. Five large chinese biomedical bibliographic databases: accessibility and coverage. *Health Information and Libraries Journal*, (25):55–61, 2008.

[128] Kaoru Yamamoto, Taku Kudo, Akihiko Konagaya, and Yuji Matsumoto. Protein name tagging for biomedical annotation in text. In *Proceedings of Workshop on NLP in Biomedicine*, 2003.

[129] Yiming Yang, Jian Zhang, and Bryan Kisiel. A scalability analysis of classifiers in text categorization. In *Proceedings of SIGIR*, 2003.

[130] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of ACL*, 1995.

[131] Alexander Yeh, Alexander Morgan, Marc Colosimo, and Lynette Hirschman. BioCreAtIvE task 1A: Gene mention finding evaluation. *BMC Bioinformatics*, 2005.

[132] Mikio Yoshida, Ken ichiro Fukuda, and Toshihisa Takagi. PNAD-CSS: A workbench for constructing a protein name abbreviation dictionary. *Bioinformatics*, 2000.

[133] Hong Yu and Eugene Agichtein. Extracting synonymous gene and protein terms from biological literature. *Bioinformatics*, 2003.

[134] Hong Yu, Vasileios Hatzivassiloglou, Andrey Rzhetsky, and W. John Wilbur. Automatically identifying gene/protein terms in medline abstracts. *Journal of Biomedical Informatics*, 2003.

[135] Hong Yu, George Hripcsak, and Carol Friedman. Mapping abbreviations to full forms in biomedical articles. *Journal of American Medical Information Association*, 2002.

[136] Hwanjo Yu, Jiawei Han, and Kevin Chen-Chuan Chang. PEBL: Positive example based learning for web page classification using SVM. In *Proceedings of KDD*, 2002.

[137] Shihong Yu, Shuanhu Bai, and Paul Wu. Description of the kent ridge digital labs system used for MUC-7. In *Proceedings of 7th Message Understanding Conference*, 1998.

[138] Manuel Zahariev. *A (Acronyms).* PhD thesis, School of Computing Science, Simon Fraser University, 2004. Governor General's Medal Award.

[139] GuoDong Zhou and Jian Su. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of 40th Annual Meeting of ACL*, 2002.

[140] GuoDong Zhou, Jie Zhang, Jian Su, Dan Shen, and ChewLim Tan. Recognizing names in biomedical texts: A machine learning approach. *Bioinformatics*, 2004.