Computer Science and Artificial Intelligence Laboratory
Technical Report

MIT-CSAIL-TR-2006-048 June 28, 2006

Was the Patient Cured? Understanding Semantic Categories and Their Relationships in Patient Records

Tawanda Carleton Sibanda

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

CSAIL

Was the Patient Cured? Understanding Semantic Categories and Their Relationships in Patient Records

by

Tawanda Carleton Sibanda

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author . . . .
Department of Electrical Engineering and Computer Science
May 26, 2006

Certified by . . . .
Ozlem Uzuner
Assistant Professor, SUNY
Thesis Supervisor

Certified by . . . .
Peter Szolovits
Professor
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Was the Patient Cured? Understanding Semantic Categories and Their Relationships in Patient Records

by

Tawanda Carleton Sibanda

Submitted to the Department of Electrical Engineering and Computer Science
on May 26, 2006, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this thesis, we detail an approach to extracting key information in medical discharge summaries. Starting with a narrative patient report, we first identify and remove information that compromises privacy (*de-identification*); next we recognize words and phrases in the text belonging to semantic categories of interest to doctors (*semantic category recognition*). For disease and symptoms, we determine whether the problem is present, absent, uncertain, or associated with somebody else (*assertion classification*). Finally, we classify the semantic relationships existing between our categories (*semantic relationship classification*). Our approach utilizes a series of statistical models that rely heavily on local lexical and syntactic context, and achieve competitive results compared to more complex NLP solutions. We conclude the thesis by presenting the design for the Category and Relationship Extractor (CaRE). CaRE combines our solutions to de-identification, semantic category recognition, assertion classification, and semantic relationship classification into a single application that facilitates the easy extraction of semantic information from medical text.

Thesis Supervisor: Ozlem Uzuner
Title: Assistant Professor, SUNY

Thesis Supervisor: Peter Szolovits
Title: Professor

Acknowledgments

I would like to thank my advisor, Professor Ozlem Uzuner, for her guidance throughout my research. She has that rare ability of giving a student enough rope to be creative, while providing sufficient guidance to prevent entanglement. She challenged me to go beyond my own initial expectations and rescued me when my expectations became too expansive. I would also like to thank Professor Szolovits for his continuous feedback, new insights, and helpful pointers to related work.

This work was supported in part by the National Institutes of Health through research grants 1 RO1 EB001659 from the National Institute of Biomedical Imaging and Bioengineering and through the NIH Roadmap for Medical Research, Grant U54LM008748. Information on the National Centers for Biomedical Computing can be obtained from http://nihroadmap.nih.gov/bioinformatics.

Was it not for my family, I would never have found the motivation to actually write and complete this thesis. The first few pages were written between the sun-drenched gardens of our Harare home, and the beautiful beaches of Cape Town, with Misheck, Doreen, Tambu, Rutendo, and Raldo cheering from the sidelines.

My friend, James Mutamba, helped me keep my sanity with incessant joking and conversation. Our talk ranged from whether hypotension is a disease as I was annotating data, to the virtues of the Poisson distribution.

Finally, I would like thank Tsitsi for her endless support and love. She allowed me to bore her for almost two years with talk of F-measures without complaints. Most importantly, she provided a shoulder to cry on during the frustrating times and a hand to high five during my triumphs.

Everything I do, I do *ad maiorem Dei gloriam.*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Since its inception in the 1800s, the narrative patient record has been a goldmine of information for members of the medical community. For the physician, medical records chronicle previous diagnoses and allergic reactions of a patient, providing continuity of information as the patient is transferred from one doctor to another. Without this data, doctors can make mistakes in treatment and diagnosis, as Moore et al. recently revealed [33]. They reported that out of 86 monitored patients, 42 experienced at least one medical error in moving from in-patient to out-patient care due to a lack of communication between physicians. The study concluded that "the poor dissemination of discharge information to outpatient PCPs create[s] an environment in which discontinuity errors [...] become relatively common".

For researchers interested in studying various clinical trends, medical records provide a large archive of data. Separate studies have shown the importance of medical discharge summaries in determining trends in adverse events [23] and disease incidences [34]. Furthermore, clinical records can be used to generate statistical models that, for example, predict patient outcome given clinical findings, provide diagnoses given symptoms, and identify epidemic outbreaks on the basis of regional data.

Unfortunately, much of the information of narrative text is incomprehensible to the most powerful analysis tool of our day—the computer. Machines excel at manipulating information stored in explicit structures such as databases or semantic frames. Automatically extracting information from natural language text is more challenging. For example, consider the sentence "John presented with hypothyroidism secondary to amiodarone". Concluding that `John` has the disease `hypothyroidism` caused by the medication `amiodarone` requires recognizing the words `hypothyroidism` and `amiodarone` as a disease and a treatment respectively, and identifying the causation relationship implied by the construction `secondary to`.

The difficulty of natural language processing in medical text has left a vast repository of potentially informative clinical data largely ignored. It is likely that this repository will continue to expand since doctors often prefer the descriptive freedom of narrative language to the structure of electronic forms for clinical transcription [25]. Against this backdrop, there is a growing demand for computer systems that can "understand" free clinical text.

In this thesis, we tackle the challenge of "understanding" patient records. We describe an approach to extract salient information from natural language medical text and create appropriate representations of this information. We focus on medical discharge summaries which are documents that a physician dictates after a patient's hospital stay, highlighting the patient's reasons for hospitalization, test results and findings, diagnoses, and prescribed medications. We address three sub-problems which we believe are pivotal in medical language understanding:

- *De-identification*: Before discharge summaries can be used in research, explicit personal health information (PHI) must be removed to ensure patient privacy. We perform automatic de-identification.

- *Semantic Category Recognition and Assertion Classification*: Given a discharge summary, we automatically determine the semantic category of each word, e.g., *diseases* or *tests*, and distinguish between positive, negative, and uncertain assertions related to these categories.

- *Semantic Relationship Extraction*: Given a sentence that includes two phrases corresponding to two semantic categories, we extract the relationship between them.

The first problem is dictated by privacy laws. The second and third problems reflect our bottom up approach to language understanding. We first identify the concepts of interest in our text and then determine relationships between the concepts. Unlike many previous approaches, our solutions to these problems incorporate syntactic information (from the Link Grammar Parser [40]). Subsequent sections describe the problems and our solutions in more detail.

1.2 De-identification

1.2.1 The Problem

Before medical reports can be used by researchers, they must be free of personal health information (PHI), such as patient names and telephone numbers, in order to protect patient privacy. In the United States, the Health Information Portability and Accountability Act (HIPAA) provides guidelines for protecting the confidentiality of health care information. HIPAA lists seventeen pieces of textual PHI to be removed of which the following appear in medical discharge summaries: first and last names of patients, their health proxies, and family members; doctors' first and last names; identification numbers; telephone, fax, and pager numbers; hospital names; geographic locations; and dates. The goal of de-identification is to recognize instances of PHI in discharge summaries and replace them with anonymous tags. For example, in the sentence "The patient is to see Dr. Davids", the doctor's name, `Davids`, should be replaced with an anonymous tag, e.g., `REMOVED`, yielding the sentence "The patient is to see Dr. REMOVED".

Difficulties arise because some non-PHI instances overlap with PHI, e.g., the word `Huntington` in `Huntington's disease`. A greedy approach to de-identification might incorrectly remove this word from the disease phrase even though it is not PHI in context. Conversely, in the absence of contextual information, some PHI could be mistaken for non-PHI, e.g., the word `test` in "Please see Dr. Test".

Another obstacle to successful de-identification is spelling errors in PHI, e.g., `John` spelt at `Jhn`. An approach that simply looks up words in a dictionary of proper nouns might incorrectly ignore misspelt PHI.

1.2.2 Our Solution: Statistical De-identifier

We present a statistical de-identifier, a scheme that recognizes patients' first and last names; doctors' first and last names; identification numbers; telephone, fax, and pager numbers; hospital names; geographic locations; and dates, even in the face of ambiguity between PHI and non-PHI, and even in the presence of spelling errors in PHI. We frame the problem as a binary classification task. Given a sentence to be de-identified, the statistical de-identifier considers each word in isolation and uses a Support Vector Machine (SVM) with linear kernel to classify the word as PHI or non-PHI. The SVM is trained on human-annotated data. During our studies, we observed that discharge summary text is often ungrammatical and fragmented; and perhaps the best indicator of PHI versus non-PHI is the local context of the word. Hence, we developed representations of local context that can be used with an SVM classifier. We represented each word with:

- The target itself, i.e., the word to be classified.
- The words within a +/- 2 context window of the target; we refer to these as the left and right *lexical bigrams* of the target.
- Left and right *syntactic bigrams* of the target, where a *syntactic bigram* captures syntactic dependencies between words as identified by the Link Grammar Parser.
- The part of speech of the target.
- Capitalization of the target.
- The length of the target.
- The MeSH ID of the noun phrase containing the target.
- Presence of the target, of the word before the target, and of the word after the target in name, location, hospital, and month dictionaries.
- The heading of the section in which the target appears.
- Whether or not the target contains "-" or "/" characters.

Using these features, the statistical de-identifier recognizes at least 94.3% of PHI and misclassifies at most 0.51% of non-PHI in four different discharge summary corpora (a de-identified corpus that was re-identified with PHI randomly selected from dictionaries; a corpus obtained directly from a hospital containing genuine PHI; a corpus engineered to contain PHI instances that frequently overlap with non-PHI words; and a corpus engineered to contain PHI not occurring in dictionaries). The statistical de-identifier significantly outperforms a dictionary-based scheme [18], as well as the IdentiFinder [6] and SNoW [38] named-entity recognizers.

1.3 Semantic Category Recognition and Assertion Classification

1.3.1 The Problem

There are two parts to this task. The first part, referred to as *semantic category recognition*, is to identify the semantic category of each word in a narrative medical discharge summary. Based on consultation with two doctors in our research group, we defined eight semantic categories of interest to researchers and practitioners: *diseases*, *treatments*, *abusive substances* (also referred to as *substances*), *dosage information* (also referred to as *dosages*), *medical practitioners* (also referred to as *practitioners*), *diagnostic tests* (referred to as *tests*), *results*, and *signs and symptoms* (also referred to as *symptoms*). Each word in the corpus must be classified as belonging to one of these eight categories or to the category *none*, reserved for irrelevant words. For example, in the sentence "The patient was admitted with coronary artery disease", every word in the phrase `coronary artery disease` should be labelled as a *disease* and the words `the`, `patient`, `was`, `admitted`, and `with` should be labelled as *none*. We refer to a phrase composed of words that belong to the same semantic category as a concept. Thus, `coronary artery disease` is a concept.

Perhaps the most important function of a discharge summary is to serve as a documentation of the medical problems associated with a patient, where medical problems refer to *diseases* and *symptoms*. To fully appreciate the implication of medical problems listed in discharge summaries, we must be able to distinguish between positive, negative, and uncertain assertions of these problems, as the following examples illustrate:

- The patient has coronary artery disease.
- The patient's brother has coronary artery disease.
- The patient may have coronary artery disease.
- The patient does not have coronary artery disease.

In the first sentence, the disease is asserted as being present in the patient, in the second sentence the disease is associated with someone other than the patient, in the third case the disease possibly occurs in the patient, and in the final case the disease is asserted as not being present. The second part of our task is, for each problem, to distinguish between the four possible scenarios highlighted: the patient has the problem; the problem is associated with someone other than the patient; the patient may have the problem; and the patient does not have the problem[1]. We refer to this as *assertion classification*.

1.3.2 Our Solutions: Statistical Semantic Category Recognizer and Rule-based Assertion Classifier

The statistical semantic category (SC) recognizer is our solution for semantic category recognition. We frame the problem as a multi-class classification task. Given a sentence, the statistical SC recognizer considers each word in isolation and uses SVMs with linear kernel to classify the word as belonging to one of the eight semantic categories or the *none* category. The statistical SC recognizer incorporates the following features which capture the contextual, ontological, and surface cues that human annotators use in determining semantic categories of words and phrases:

- The target itself.
- Left and right lexical bigrams.
- The heading of the section that the target appears in.
- Syntactic bigrams of the target.
- The head of the noun phrase that the target is part of and the syntactic bigrams of the head.
- The part of speech of the target and the words within a +/- 2 context window.
- The UMLS semantic types of the noun phrase containing the target.
- Whether or not the target is capitalized.
- Whether or not the target contains numerals.
- Whether or not the target contains punctuation.

[1]We assume that these four classes are exhaustive, i.e., if a *disease* or a *symptom* is mentioned in a discharge summary, then it belongs to one of our four categories.

Using these features, the statistical SC recognizer obtains F-measures above 90% for most categories. These results are significantly better than the performance of a baseline which simply maps phrases in the text to UMLS semantic types.

To distinguish between positive, negative, uncertain, and non-patient-related assertions of each medical problem identified by semantic category recognition, we employ a regular-expression-based algorithm referred to as the rule-based assertion classifier. Initially, we used a development corpus to create the following dictionaries:

- Common phrases that precede or succeed a problem and imply that the problem is absent (later referred to as *negation phrases*).
- Common phrases that precede or succeed a problem and imply that the problem is uncertain (later referred to as *uncertainty phrases*).
- Common phrases that precede or succeed a problem and imply that the problem is associated with someone other than the patient (later referred to as *alter-association phrases*).

To classify a problem, the rule-based assertion classifier determines which phrases occur within a four word window of the problem. The classifier is greedy, first checking for alter-association phrases, followed by negation phrases, and finally uncertainty phrases. If at any stage a match is found, the assertion classifier labels the problem according to the assertion implied by the matching phrase. If no match is found we label the problem as present. The rule-based assertion classifier achieves F-measures above 90% .

1.4 Semantic Relationship Classification

1.4.1 The Problem

We define a concept as a phrase composed of words that belong to the same semantic category. Given two concepts in a sentence, the final task is to define the relationship between them. As previously mentioned, an important function of discharge summaries is to highlight the problems a patient has. Hence, we focused on interactions involving *diseases* and *symptoms*; after consulting doctors, we defined six types of binary relationships that encapsulate most of the information pertaining to medical problems: relationships between present *diseases* and *treatments*, between uncertain *diseases* and *treatments*, between *diseases* (both uncertain and present) and *tests*, between *diseases* and *symptoms*, between present *symptoms* and *treatments*, and between uncertain *symptoms* and *treatments*. For example, we define the following possible relationships between a *disease* and a *test*: Test reveals disease, Test conducted to investigate disease, and None (if no relationship exists).

1.4.2 Our solution: Statistical Semantic Relationship Recognizer

Our statistical semantic relationship (SR) recognizer consists of six different multi-class SVM classifiers corresponding to the six binary relationship types. Thus, there is an SVM classifier for relationships between uncertain *symptoms* and *treatments*, another classifier for *diseases–tests* relationships, and so on. Unmarked input text is passed through our statistical semantic category recognizer and the rule-based assertion classifier which mark semantic categories and problem assertions respectively. For each sentence in the text, and for each candidate pair of concepts covered by one of our relationship types (for example, *diseases–tests* relationships), the statistical SR recognizer uses the appropriate SVM classifier to determine which specific relationship exists between the concepts (e.g.,Test reveals disease, Test conducted to investigate disease, or No relationship). The multi-class SVM classifiers for the relationship types all employ the same features. We list the features for the *diseases–tests* classifier for clarity:

- The number of words between the candidate concepts.
- Whether or not the *disease* precedes the *test*.
- Whether or not other concepts occur between the *disease* and the *test*.
- The verbs between the *disease* and the *test*.
- The two verbs before and after the *disease* and the two verbs before and after the *test*.
- The head words of the *disease* and the *test* phrases.
- The right and left lexical bigrams of the *disease* and the *test*.
- The right and left syntactic bigrams of the *disease* and the *test*.

- The words between the *disease* and the *test*.

- The path of syntactic links (as found by the Link Grammar Parser) between the *disease* and the *test*.

- The path of syntactically connected words between the *disease* and the *test*.

Using these features, the statistical SR recognizer achieves a micro F-measure of 84.5%, and a macro F-measure of 66.7%, which are significantly better than the baseline of selecting the most common relationship.

1.5 Contributions

In this thesis we present solutions to de-identification, semantic category recognition, assertion classification, and semantic relationship classification. Our solutions recognize semantic categories and their relationships in medical discharge summaries. Using this information, we can build an extensive list of a patient's medical problems. One of the more prevalent schemes that attempts to semantically interpret medical reports as extensively as we do is the rule-based MedLEE parser [24]. The disadvantage of rule-based approaches is the amount of manpower required to generate the specialized rules. Moreover, the performance of the resulting systems tends to deteriorate in new domains. Our solutions, on the other hand, are easily extensible (by defining new semantic categories and semantic relationships), and we have found that the features used for our various classifiers work equally well for named entity recognition in newswire corpora and for relationship classification in MEDLINE abstracts.

Furthermore, we have developed a novel way of incorporating the syntactic information provided by the Link Grammar Parser into a feature-based classifier. For de-identification and semantic category recognition, we found that the syntactic information extracted is a more informative feature than lexical context traditionally used in information extraction.

Finally, we have shown that, due to the repetitive structure and language of clinical text, the lexical and syntactic context of words in discharge summaries is often a more useful indicator of semantic categories than external ontologies (such as UMLS).

1.6 Thesis Structure

Chapter 2 of this thesis presents related work in de-identification, semantic category recognition, and relationship classification. It also discusses tools and techniques that we employ, such as SVMs and the Link Grammar Parser. Chapter 3 studies the statistical de-identifier, justifying our selection of features and presenting results of our de-identifier compared to other named entity recognition schemes. Chapter 4 presents the statistical semantic category recognizer in more detail and chapter 5 describes the statistical semantic relationship recognizer. Chapter 6 presents the Category and Relationship Extractor (CaRE): a system that integrates the solutions of chapters 3, 4, and 5 into a single interface for information extraction in medical discharge summaries. Finally, chapter 7 reviews our work and discusses the implications of our research.

Chapter 2

Background

2.1 Related Work

This thesis falls within the domain of medical informatics—an emerging multidisciplinary field that endeavors to incorporate computer applications in medical care. A key area within this discipline is Medical Language Processing (MLP). MLP and the more general field of Natural Language Processing (NLP) encompass various efforts to understand and generate natural human languages using computers. MLP focuses on language in medical text, while NLP has traditionally focused on non-medical narrative. Both NLP and MLP are difficult. The problem of understanding non-medical narrative is exacerbated by the intrinsic ambiguity of human languages. Medical text also poses challenges caused by language ambiguity: in addition, difficulties arise due to the ungrammatical sentence fragments and abbreviations characterizing medical text. Consequently, many of the NLP techniques developed over the last 30 years do not directly apply to the medical domain: MLP requires new insights and novel reworking of old techniques.

In this section, we review a selection of these new MLP approaches as well as some general NLP algorithms that inspired our solutions to the tasks of de-identification, semantic category recognition and assertion classification, and semantic relationship classification.

2.1.1 De-identification

De-identification refers to the removal of identifying information from medical records. Traditionally, PHI has been removed by human scrubbers through a slow and labor-intensive process.

Automated solutions to de-identification are gaining popularity. For example, Sweeney's Scrub system [42] employs numerous algorithms, each one specialized for the detection of a specific PHI. Each algorithm uses lexicons and morphological patterns to compute the probability that a given word belongs to the class of PHI that the algorithm specializes in. The word is labelled with the class of PHI corresponding to the algorithm with the highest probability. On a test corpus of patient records and letters, Scrub identified 99–100% of all PHI.

Taira et al. [44] present a de-identification scheme that uses regular expressions to identify PHI with "well-defined syntax", such as telephone numbers and dates. To identify the more complicated patient names, they use the idea of semantic selectional restrictions—the belief that certain word classes impose semantic constraints on their arguments. For example, the verb `vomited` strongly implies that its subject is a patient. Using these constraints, Taira et al. achieve a precision of 99.2% and recall of 93.9% for identification of patient names in a clinical corpus.

In designing our approach to de-identification, we were inspired by the aforementioned systems as well as NLP schemes for named entity recognition (the identification of entities such as people, places, and organizations in narrative text). There are two main classes of solution to this problem: weakly-supervised solutions (which employ very little annotated training data) and supervised solutions (which require large amounts of annotated data).

Weakly-supervised Named Entity Recognition

Weakly-supervised methods employ the "bootstrapping" technique, whereby a small seed set of words belonging to the same semantic category is used to extract contextual cues for that category. These cues are then used to extract more instances of the category which are then used to identify additional contextual cues and so on, until a vast number of instances are identified in context. Collins et al. [11] and Riloff et al. [36] use variations of this approach for named entity recognition. Unfortunately, despite the advantage of not requiring a large amount of annotated data, the accuracies obtained by weakly-supervised methods (Collins et al. report 92% accuracy) are too low for de-identification.

Supervised Named Entity Recognition

Supervised approaches use training data to learn a function mapping inputs to desired outputs. These approaches can be generative or discriminative.

A generative model is one that can be used to randomly generate observed data. One of the most successful generative named entity recognizers is IdentiFinder [6]. IdentiFinder uses a variant of the generative HMM model to learn the characteristics of the names of entities such as people, locations, geographic jurisdictions, organizations, dates, and contact information. For each named entity class, this system learns a bigram language model, where a word is defined as a combination of the actual lexical unit and various orthographic features. To find the names of all entities, the system finds the most likely sequence of entity classes in a sentence given the observed sequence of words and associated features. In chapter 3, we show that our de-identification system outperforms IdentiFinder.

Discriminative supervised models use training data to directly estimate the probability of the output class given observed features. They do not have the ability to generate example instances. Isozaki and Kazawa [27] use Support Vector Machines (SVMs) to recognize named entities in Japanese. They classify the semantic category of each word, employing features of the words within two words of the target (a +/- 2 context window). The features they use include the part of speech of the word, the structure of the word, and the word itself.

Roth and Yih's SNoW system [38] uses discriminative and generative components to recognize people, locations, and organizations. SNoW labels the entities in a sentence and the relationships between the entities. It operates in two stages; in the first stage, it uses weak, discriminative classifiers to determine the probability distribution of labels for entities in the sentence. In the second stage, it uses Bayesian networks to strengthen or weaken its hypothesis about each entity's type based on the constraints imposed by the relationships.

Our de-identification solution is a discriminative supervised model. Like Isozaki et al., we use SVMs to identify the class of individual words (where the class is PHI or non-PHI), and employ orthographic properties of the target word, the part of speech of the word, and lexical context as features.

However, what separates our de-identification solution from many prior approaches, both for named entity recognition and de-identification, is our use of deep syntactic information.

We believe that PHI is characterized by syntactic context, a hypothesis that is supported by experimental results. Few feature-based systems use syntactic information as a feature.

Syntax is used extensively by models that simultaneously extract syntactic and semantic information from text. These models augment traditional syntactic parse trees with semantic information [32].

2.1.2 Semantic Category Recognition and Assertion Classification

Semantic category recognition refers to the identification of semantic categories such as diseases and tests in the text; and assertion classification determines whether problems (diseases and symptoms) are present in the patient, absent, uncertain, or associated with someone other than the patient.

Semantic category Recognition

Semantic category recognition is similar to the task of named entity recognition. Whereas named entity recognizers identify proper nouns such as people, places, and organizations, semantic category extractors search for concepts[1], such as proteins, genes, and diseases. It is not surprising that similar approaches are used for semantic category recognition as for named entity recognition.

As before, the methods can be divided into weakly supervised and supervised techniques. In addition, there are a host of solutions that use external knowledge sources, such as the Unified Medical Language System (UMLS), and exhaustive look-up algorithms to identify medical concepts in clinical text.

Weakly Supervised Solutions: Badger [41] employs a weakly-supervised method that combines semantic category recognition with assertion classification. This method identifies multi-word phrases referring to diagnoses and signs and symptoms of disease. Given a seed set of annotated data, it uses bootstrapping to generate a dictionary of semantic and syntactic rules that characterize semantic categories.

Supervised Solutions: Collier [10] use HMMs to identify genes and gene products in biomedical text. He uses orthographic features of words such as capitalization, whether

[1] In this thesis, a concept refers to a phrase composed of words that belong to the same semantic category.

the word contains certain punctuation symbols, and whether the word is a Greek letter. Zhao [48] builds on Collier's work by incorporating word similarity-based smoothing to overcome data sparseness.

Takeuchi et al. [45] use SVMs to identify technical terms in biomedical text. They employ features of the words within a +/- 3 context window. Their features include the target word and its orthographic properties, previous class assignments, and POS tags.

As for de-identification, there are few supervised solutions to semantic category recognition that use syntax. Finkel et al. [22] describe a maximum entropy Markov model for biomedical entity recognition. This model uses uses local, i.e., surface, and syntactic features. To obtain syntactic features, Finkel et al. fully parse the data using the Stanford parser and for each word in a noun phrase use the head and the governor of the phrase as features.

UMLS-based Solutions: The development of the UMLS Metathesaurus (see Section 2.2.3) has facilitated the proliferation of various schemes that map free text to concept names in the Metathesaurus. These systems tend to use some form of shallow parsing to identify candidate phrases and then exhaustively search for these phrases in the Metathesaurus [4, 16]. Some systems first map phrases to the semantic types defined by UMLS and then map these to a narrower domain of semantic categories (such as diseases and treatments). Long [29] extracts diagnoses and procedures this way.

Our solution to semantic category recognition is a hybrid of Collier's discriminative SVM-based scheme and Long's UMLS search. Our solution differs from previous work in that, firstly, we set out to solve a more complex task. Previous efforts have focused on extracting diseases, symptoms, and treatments that frequently consist of single words or short noun phrases. We additionally identify the results semantic category that includes longer phrases and entire clauses. Furthermore, while most prior approaches have focused on lexical context, we use deep syntactic information in the form of syntactic dependencies from the Link Grammar Parser in order to identify semantic categories.

Assertion Classification

In medical texts, knowing that a disease is referenced in a sentence does not provide enough information about how the disease relates to the patient. The more important questions are “Does the patient have the disease?”, “Has the disease been ruled out?”.

Most approaches for determining positive, negative, or uncertain assertions of an entity are rule-based. Our scheme is based on the NegEx algorithm [9]. NegEx first identifies candidate diseases and findings in clinical text (using UMLS), and then employs a dictionary of phrases indicative of negation to identify which phrases are absent. NegEx uses heuristics to limit the scope of negation phrases and achieves a recall of 94.5% and a precision of 91.73% on negation detection.

Elkin et al. [20] use a similar pattern-matching approach to assign positive, negative, or uncertain assertions to entities identified by SNOMED. They achieve a recall of 97.2% and a precision of 91.1% on negation detection.

There are very few machine learning approaches to negation detection. Chapman et al. [26] use Naive Bayes and Decision trees “to learn when ‘not’ correctly predicts negation in an observation”.

2.1.3 Semantic Relationship Classification

We define semantic relationship classification as inferring the relationship existing between concepts in a sentence. We assume that the concepts are given (by some automated semantic category recognizer). There are numerous schemes addressing this and similar problems. The solutions can be divided into rule-based versus machine-learning approaches.

Rule-based Approaches to Semantic Relationship Classification

Rule-based schemes recognize that relationships between concepts are often categorized by lexical patterns (e.g., the phrase `works for` is a strong indicator of the employer–employee relationship), syntactic patterns (e.g., if a disease is the object of a verb and a treatment is the subject of the same verb, then a relationship is likely to exist between the concepts), and semantic patterns. Rule-based schemes frequently combine such patterns to create integrated templates for identifying relationships [21, 24, 28].

Machine Learning Approaches to Semantic Relationship Classification

Machine learning approaches attempt to automatically “learn” the patterns characterizing different semantic relationships [37, 14]. Zelenko et al. [47] use kernel methods to identify person–affiliation and organization–location relationships. They first parse the data and identify noun phrases. Then for each candidate pair of phrases, they find the lowest common node subsuming both phrases and extract the sub-tree rooted at this node. They define a kernel function that directly computes a similarity score between syntactic trees, and incorporate this into an SVM to classify new instances.

More recent schemes build on Zelenko et al.'s idea of using the syntactic structure linking the concepts as a feature in classifying the inter-concept relationship. For example, Culotta et al. [15] use a dependency tree augmented with surface features of the words in the sentence. Ding et al. [17] decide whether two biochemicals are related by determining whether or not a short dependency path (in the form of links from the Link Grammar Parser) exists between them. Singh [39] describes a method that extracts employer–employee, organization–location, family, and person–location information. Her relationships are binary (either a relationship exists or not). For each candidate pair of concepts, she extracts as features the non-terminal/part of speech sequence between the concepts (provided by fully parsing the sentence), the full text string between the concepts, the ordering of the concepts, and the local context of individual concepts.

Our approach trains various SVM classifiers in order to identify relationships within a specific relationship type (e.g., disease–test relationships, disease–treatment relationships). We use similar features to Singh's approach. But instead of considering the non-terminal path, we use the dependency path between concepts (in the form of links from the Link Parser). Whereas Singh and other researchers concentrate on determining whether or not a relationship exists, we aim to distinguish between 22 different relationships (spread over 6 relationship types), using SVM classifiers specialized for each relationship type.

2.1.4 Integrated Solutions

One of the ultimate goals of our research was to build the Category and Relationship Extractor (CaRE), an integrated system that uses components for de-identification, semantic category recognition, and semantic relationship classification in order to interpret medical text. Friedman's MedLEE [24] parser is an existing system that, like CaRE, extracts information from medical text.

MedLEE

MedLEE [24] maps important concepts in narrative text to semantic frames, where a frame specifies the semantic category of the concept and modifiers of the concept. For example, the sentence "Pneumonia was treated with antibiotics." yields the frames `[problem, pneumonia, [certainty,moderate]]` and `[med, antibiotic, [certainty, high], [status,therapy]]`.

The MedLEE system consists of four components: a preprocessor, a parser, a phrase regularizer, and an encoder. Narrative text is initially passed through the pre-processor. The pre-processor determines the boundaries of words and sentences in the text, and replaces words and phrases with semantic and syntactic categories from a hand-built lexicon.

The output of the first component is then passed to a parser which uses a grammar of syntactic and semantic rules to extract relationships between concepts in the sentence. For example, the rule DEGREE + CHANGE + FINDING applies to phrases containing degree information followed by change and finding information, such as **`mild increase in congestion`**. The rule specifies that the degree concept modifies the change concept and both modify the finding. Thus, the frame produced for the example is `[problem, congestion, [change, increase, [degree, low]]]`.

The frame-based representation is then post-processed by the phrase regularization component that merges words to form multi-word terms. Finally, the encoder maps terms in the sentence to controlled vocabulary concepts (such as UMLS concept identification numbers).

One of MedLEE's strengths is its robust parser. If a parse is not found for the complete sentence, then the sentence is segmented and parses are generated for the separate segments.

The disadvantage of MedLEE is that it requires extensive knowledge engineering to expand to new domains. To extend the parser's functionality to discharge summaries, an additional 300 hand-tailored rules were added to the grammar (initially, there were 450 rules), and 6,000 new entries were created in the lexicon (originally 4,500). This extension represents a significant number of man-hours.

2.2 Resources

In this section, we briefly describe the resources and techniques used in this thesis.

2.2.1 Support Vector Machines

CaRE reduces semantic category recognition and relationship extraction to a series of classification problems. We used SVMs for classification. Given a collection of data points and corresponding classification classes, SVMs endeavor to find a hyperplane that separates the points according to their class. To prevent over-fitting, the hyperplane is chosen so as to maximize the distance between the plane and the closest data point in both classes. The vectors that are closest to this hyperplane are called the support vectors. Given a data point whose class is unknown, an SVM determines which side of the hyperplane the point lies and labels it with the corresponding class (for a more in depth analysis refer to [13]).

Often, the data points are not linearly separable. In this case, a popular approach is to transform the vector of features into a higher dimensional space and search for a linear separator within that space. The hyperplane obtained is non-linear in the original input space, and is determined using the dot-product of the feature vectors. A non-linear function (called a kernel) is used to efficiently compute the dot products of the transformed feature vectors. Several types of separators (and hence kernels) are popular, such as the Polynomial and Gaussian radial basis function (RBF) kernels.

Throughout this paper, we use the simple inhomogeneous linear kernel:

$$(\vec{x} \cdot \vec{x'} + 1)$$

We have found that more complicated kernels achieve better results on the training data, but tend to over-fit to the particular dataset. Furthermore, complex kernels often require cross-validation over a development set to determine optimal values for parameters used in the functions. We have observed that such cross-validation takes a long time and the final classification performance is sensitive to the parameter values used. We prefer to abstract from the details of SVMs and concentrate on the impact that various feature combinations have on classification.

We use SVMs particularly because:

- They robustly handle large feature sets [5].
- They do no make any independence assumptions regarding features (unlike Naive Bayes).
- They have achieved high performance in comparable text categorization tasks [19].

However, SVMs do have limitations. Firstly, they are binary classifiers. For semantic category recognition and semantic relationship classification, we require multi-class labelling. We overcome this problem by using the LIBSVM software library [8], which implements the SVM algorithm and also builds a multi-class classifier from several binary classifiers using *one-against-one voting* [8].

A second limitation of SVMs is that they do not explicitly handle categorical features. They are designed for numerical features, which have real number values. Categorical features, such as bigrams and words, do not have explicit numerical values.

To get around this problem, we use a *one-out-of-m* encoding. Suppose a categorical feature has m possible values $(x_1, x_2, \ldots, x_m)$. To represent that feature, we create an m-dimensional vector. If the feature has value x_j, then we set the value of the j^{th} component of the vector to 1, and set the remainder of the components to 0.

2.2.2 Link Grammar Parser

The importance of syntax as a first step in semantic interpretation has long been appreciated [30]. We hypothesize that syntactic information plays a significant role in MLP. Consider, as an example, the task of semantic category recognition. Words that are objects of the verb `consult` tend to be practitioners, and words that are objects of the verb `reveal` tend to be diseases and results.

To extract syntactic dependencies between words, we use the Link Grammar Parser. This parser's computational efficiency, robustness, and explicit representation of syntactic dependencies make it more appealing than other parsers for MLP tasks.

The Link Grammar Parser [40] models words as blocks with left and right connectors. The connectors impose local restrictions by specifying the type of connections (referred to as links) a word can have with surrounding words. A successful parse of a sentence satisfies the link requirements of each word in the sentence. In addition, two global restrictions must be met:

1. Planarity: The links must not cross. Thus, the following "parse" of the sentence "John loves nice food." is invalid.

```
      +------------+
+---- | -----+     |
|     |      |     |
John  loves  nice  food.
```

2. Connectivity: All the words in the sentence must be indirectly connected to each other.

The parser uses a $O(n^3)$ dynamic programming algorithm to determine a parse of the sentence that satisfies the global and local restrictions and has minimum total link length.

The parser has several features that increase robustness in the face of ungrammatical or complex sentences. Firstly, the lexicon contains generic definitions "for each of the major parts of speech: noun, verb, adjective, and adverb." When it encounters a word in a sentence that does not appear in the lexicon, the parser replaces the word with each of the generic definitions and attempts to find a valid parse in each case. Secondly, the parser can be set to enter a less scrupulous "panic mode" if a valid parse is not found within a given time limit. In panic mode, the parser suspends the connectivity restriction and considers high cost, longer links. Often the result of "panic-mode" parsing is that the phrases and clauses in the sentence are fully parsed, but they are not connected to each other. As we are concerned with local context in our MLP application, such partial parses are often sufficient to extract useful semantic information (see Chapters 3 and 4).

Using Link Grammar Parser Output as an SVM Input

The Link Grammar Parser produces the following structure for the sentence:

"John lives with his brother."

```
    +------------------Xp-----------------+
    |                    +----Js----+     |
    +---Wd--+--Ss-+--MVp-+   +--Ds--+     |
    |       |     |      |   |      |     |
LEFT-WALL John lives.v with his brother.n .
```

This structure shows that the verb `lives` has an Ss connection to its singular subject `John` on the left and an MVp connection to its modifying preposition `with` on the right.

A priori, we wanted to extract syntactic dependency information from the Link Grammar Parser. Since most of our machine learning algorithms classify individual words, we wanted a representation that captured the syntactic context, i.e., the immediate left and right dependencies, of each word, while being simple enough for use with a feature-based classifier such as an SVM. We ultimately developed the novel idea of syntactic n-grams which capture all the words and links within n connections of the target word. For example, for the word `lives` in the parsed sentence, we extracted all of its immediate right connections (where a connection is a pair consisting of the link name and the word linked to)—in this case the set {*(with, MVp)*}. We represented the right syntactic unigrams of the word with this set of connections. For each element of the right unigram set thus extracted, we found all of its immediate right connections—in this case {*(brother, Js)*}. The right syntactic bigram of the word `lives` is then {{*(with, MVp)*}, {*(brother, Js)*}}. The left syntactic bigram of the word `lives`, obtained through a similar process, is {{*(LEFT_WALL, Wd)*},{*(John, Ss)*}}. For words with no left or right links, we created their syntactic bigrams using the two words immediately surrounding them with a link value of *NONE*. Note that when words have no links, this representation implicitly reverts back to lexical n-grams.

To summarize, the syntactic bigram representation consists of: the right-hand links originating from the target; the words linked to the target through single right-hand links (call this set R_1); the right-hand links originating from the words in R_1; the words connected to the target through two right-hand links; the left-hand links originating from the target; the words linked to the target though single left-hand links (call this set L_1); the left-hand links originating from the words in L_1; and the words linked to the target through two left-hand links. This idea can be generalized to produce semantic n-grams for all values of n.

2.2.3 Unified Medical Language System

CaRE benefits from external knowledge sources for de-identification and semantic category recognition. Specifically, CaRE uses the Unified Medical Language System (UMLS) [2] as an external knowledge source. UMLS was developed by the National Library of Medicine to aid automated MLP systems, and comprises three databases: Metathesaurus, the Semantic

Network, and the SPECIALIST lexicon.

Metathesaurus integrates information from different thesauri of biomedical and health-related concepts such as SNOMED and ICD-10. It groups under a single concept ID various words and phrases that refer to the same concept. For each concept, Metathesaurus supplies unique identifiers, source vocabulary information, lexical variants, and semantic types matching the concept (e.g., ANIMAL). In our approach to de-identification, we make extensive use of the MeSH (Medical Subject Headings) Metathesaurus knowledge source. MeSH maps biological terms to descriptors, which are arranged in a hierarchy. There are 15 high-level categories in MeSH: e.g., A for Anatomy, B for Organism, and so on. Each category is recursively sub-divided up to a depth of 11 levels. MeSH descriptors have unique tree numbers which represent their position in this hierarchy.

The semantic types for each concept in the Metathesaurus are defined in the Semantic Network. There are currently 35 semantic types, e.g., ORGANISM and PLANT AND BODY LOCATION OR REGION.

2.3 Natural language processing techniques

Discharge summaries that we obtain from hospitals are haphazardly formatted. Sentences run over multiple lines and section headings are not separated from the text. In this section, we describe various natural language processing techniques that we use to pre-process the summaries for use by MLP components.

2.3.1 Sentence breaking and Tokenization

Initially, we pre-process the discharge summaries to identify sentence boundaries, and format the text so that each sentence occurs on an individual line. This part of the pre-processing stage is essential because our unit of semantic interpretation is the sentence: we endeavor to extract the semantics of individual sentences and not the entire document. The discharge summaries we obtain from hospitals are formated in an ad hoc manner. As a result, automatically sentence-breaking the text is challenging. Our current solution is based on hand-tailored heuristics and does not guarantee correctness. Manual review is usually required before the output can be used by other MLP components. After identifying the sentences, the next step is tokenization (the separation of punctuation from words). We separate all punctuation from words unless the word is an abbreviation, a list marker, or the possessive `'s`. Thus, `car's` becomes `car 's`, `I saw him` becomes `I saw him .`, but `Dr. John` remains `Dr. John`.

2.3.2 Normalization

Many morphological variants of the same word are usually observed in narrative text. For example, `was`, `are`, and `is` are all variants of the verb `be`. In most cases, these variants have the same meaning. Our solutions use statistical machine learning and are thus susceptible to the problem of sparse data—too few examples of certain classification categories and/or features to reliably make predictions. This problem can be alleviated by abstracting over lexical variants of words, effectively increasing the number of instances with similar lexical features.

We use the **norm** tool, part of the UMLS SPECIALIST LEXICON software package [1] to obtain normalized strings for each word we encounter in the discharge summaries. The normalized word is defined as "a version of the original string in lower case, without punctuation, genitive markers, or stop words, diacritics, ligatures, with each word in its uninflected form". For example, the sentence "The patient was covered with an unknown antibiotic." becomes, after normalizing each word, "the patient be cover with an unknown antibiotic.".

2.3.3 Evaluation Metrics

Precision, Recall, and F-measure

To evaluate the effectiveness of our classification tasks, we use *precision*, *recall*, and *F-measure*.

Given predictions for a set of data instances, precision for class x is defined as:

$$\text{Precision} = \frac{\text{number of correctly classified instances of } x}{\text{total number of instances classified as } x}$$

Recall for class x is defined as:

$$\text{Recall} = \frac{\text{number of correctly classified instances of } x}{\text{total number of instances of } x \text{ in the instance set}}$$

F-measure for class x is the harmonic mean of precision and recall; it is defined as:

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

Z-test

We use the Z-test to determine whether differences in our evaluation metrics are significant. We first define $\hat{p}$ as follows:

$$\hat{p} = \frac{n_1 p_1 + n_2 p_2}{n_1 + n_2}$$

where n_1 and n_2 refer to the sample sizes of the two categories being compared, while p_1 and p_2 refer to the classification accuracies or F-measures being compared. Given the $\hat{p}$ values, the z statistic is calculated with the formula below:

$$Z_0 = \frac{p_1 - p_2}{\sqrt{\hat{p}(1-\hat{p})(1/n_1 + 1/n_2)}}$$

$$H_0 : p_1 = p_2$$

$$H_1 : p_1 \neq p_2$$

The z-statistic determines whether the two ratios are essentially the same (with any differences attributable to chance), or whether the difference between the two ratios is in fact significant. If $|Z_0| > 1.96$, then one can conclude that the difference is significant with confidence level $\alpha = 0.05$. For all comparisons in this thesis, we measure significance at $\alpha = 0.05$.

Chapter 3

De-identification

Medical discharge summaries include explicit personal health information (PHI) which, if released, would jeopardize patient privacy. The Health Information Portability and Accountability Act (HIPAA) specifies seventeen pieces of textual PHI that must be removed from clinical records to ensure patient confidentiality. Of the HIPAA-listed items, the following appear in discharge summaries: first and last names of patients, their health proxies, and family members; doctor's first and last names; identification numbers; telephone, fax, and pager numbers; hospital names; geographic locations; and dates.

In this chapter, we present a statistical de-identifier which is a simple solution for de-identification in medical discharge summaries. This statistical de-identifier uses SVMs and local lexical and syntactic context to determine whether each word in a corpus is PHI. We compare the statistical de-identifier to sophisticated approaches designed for the similar task of named entity recognition. These named entity recognizers use HMMs and Bayesian networks to capture global context (in the form of inter-entity dependencies). Because medical discharge summaries contain short, fragmented sentences, we hypothesize that classifying each word without using global context is sufficient for de-identification. We present experimental results to support this hypothesis. We also show that the statistical de-identifier outperforms a more traditional, dictionary-based de-identification solution.

We end by determining the source of the statistical de-identifier's gains. We demonstrate that syntactic local context, in the form of syntactic bigrams, is one of the most informative features for de-identification; at times , it is more informative than traditionally used lexical bigrams.

3.1 Definitions

After browsing through several medical discharge summaries and HIPAA regulations, we defined the following recurring classes of PHI:

- Patients: This class includes the first and last names of patients, their health proxies, and family members. Titles, such as `Mr.`, are excluded, e.g., "Mrs. [Mary Smith]$_{patient}$ was admitted on 5/3/2001".

- Doctors: This class refers to medical doctors and other practitioners mentioned in the text. Again titles, such as `Dr.`, are not considered PHI, e.g., "He met with Dr.~[John Bland]$_{doctor}$".

- Hospitals: This class includes hospital names, such as `Massachusetts General Hospital`, and the names of specific medical organizations. We categorize the entire institution name as PHI including common words such as `clinic`, e.g., "She was admitted to [Brigham Women's Hospital]$_{hospital}$".

- IDs: This class refers to any combination of numbers and letters identifying medical records, patients, doctors, or hospitals, e.g., "Provider Number: [12344]$_{ID}$".

- Dates: This class includes any dates appearing in text. HIPAA specifies that years are not considered PHI, but all other elements of a date are. We chose to label a year appearing in a date as PHI if the date was a single lexical unit, e.g., `12/02/99`, and as non-PHI if the year existed as a separate word, e.g., `23 March, 2006`. This decision was motivated by the fact that many solutions to de-identification and named entity recognition (including our own) classify entire words as opposed to segments of a word. Besides, once identified, dates such as `12/02/99` can be easily post-processed to separate the year from the month and day.

- Location: This class includes geographic locations such as cities, states, street names, zip codes, and building names and numbers, e.g., "He lives in [Newton]$_{location}$".

- Phone numbers: This PHI class encompasses telephone, pager, and fax numbers appearing in the text.

3.2 Corpora

A successful de-identification scheme must achieve often competing objectives: it should anonymize all instances of PHI in the text. On the other hand, it should not remove any information that is not PHI (such as disease names and treatments). We have identified two significant challenges to successfully achieving these objectives: the existence of ambiguous PHI and the existence of non-dictionary PHI.

Ambiguous PHI are words that can be instances of PHI or not, depending on their context. For example, `Alzheimer` is a person's name, but it is also a disease name: its meaning in a sentence is determined by surrounding text.

Occasionally, an instance of PHI may not appear in dictionaries if, for example, it is misspelt (e.g., `Jhn` instead of `John`) or if it is an uncommon proper noun, e.g., `Mafikizile`. We refer to such words as non-dictionary PHI.

In order to predict how our system would perform on any discharge summary corpus, we created artificial corpora representing the two pathological situations: the case where most of the PHI is ambiguous and the case where most of the PHI does not occur in dictionaries. Performance metrics were then gathered on these challenging corpora.

To create these corpora, we first obtained a collection of already de-identified discharge summaries:[1] that is, summaries in which most PHI (and unfortunately some non-PHI) had been replaced with the anonymous `[REMOVED]` tag.

Using the definitions of the various PHI classes, we hand-annotated the corpus. We employed a simple annotation scheme wherein each instance of PHI was enclosed within XML tags indicating its type. Since most of the PHI had already been replaced with the `[REMOVED]` tag, we employed local contextual cues and common sense to identify the PHI category of the removed phrases. For example, the sentence:

"Mr. [REMOVED] was admitted to [REMOVED] on [REMOVED]."

becomes after annotation:

"Mr. <patient> [REMOVED] </patient> was admitted to <hospital> [REMOVED] </hospital> on <date> [REMOVED] </date>."

[1] Authentic clinical data are difficult to obtain for privacy reasons; obtaining the de-identified text was considerably easier.

Category	Number of instances			
	Re-identified	**Ambiguous**	**Non-dictionary**	**Authentic**
Non-PHI	17,874	19,275	17875	112,669
Patient	1,048	1,047	1,037	294
Doctor	311	311	302	738
Location	24	24	24	88
Hospital	600	600	404	656
Date	735	736	735	1,953
ID	36	36	36	482
Phone	39	39	39	32

Table 3.1: Break-down of PHI in all four corpora.

We then obtained dictionaries of common names, hospitals, and locations from the U.S. Census Bureau and online sources, and a list of diseases, treatments, and diagnostic tests from the UMLS Metathesaurus. Using these dictionaries, we generated three files: a randomly re-identified corpus, a re-identified corpus containing ambiguous data, and a corpus containing non-dictionary PHI.

3.2.1 Randomly Re-identified Corpus

We began by identifying the common patterns for each type of PHI. A patient or doctor name, `John Smith` for example, can have the following structures: `John Smith`; `Smith, John`; `J.S̃mith`; `Smith`; and `John`.

We then wrote a script to work through the corpus sentence by sentence. For each hospital, location, doctor, and patient, the script replaced the generic `[REMOVED]` tag with randomly chosen values from appropriate dictionaries arranged in randomly selected structures. For phone numbers and dates, the script selected appropriate structures and populated the structures with numbers (or months). IDs were generated by randomly choosing an ID length and individual numbers in the ID. The first column of Table 3.1 shows the break-down of PHI in the re-identified corpus.

3.2.2 Re-identified Corpus with Ambiguous PHI

To generate the corpus of ambiguous PHI, we first marked diseases, tests, and treatments in the de-identified corpus. Next, we re-identified this corpus with subsets of PHI such that each instance of PHI appeared in a disease, treatment, or test dictionary. The result was that a large number of PHI in this corpus overlapped with non-PHI.

Category	Number of Instances	Number of Ambiguous Instances
Non-PHI	19,275	3,787
Patient	1,047	514
Doctor	311	247
Location	24	24
Hospital	600	86
Date	736	201
ID	36	0
Phone	39	0

Table 3.2: Distribution of words that are ambiguous between PHI and non-PHI.

Table 3.2 shows the distribution of PHI in the ambiguous corpus and shows the number of instances of each PHI category that also appear as non-PHI in the text.

3.2.3 Re-identified Corpus with Non-dictionary PHI

The corpus containing non-dictionary PHI was created by the same process used to randomly re-identify the text. However, instead of selecting the replacement for PHI from dictionaries, words were generated by randomly selecting word lengths and letters from the alphabet, e.g., "O. Ymfgi was admitted ...". All patient, doctor, location, and hospital names were consequently not in common dictionaries. The third column of Table 3.1 indicates the distribution of PHI in the corpus containing non-dictionary PHI.

3.2.4 Authentic Discharge Summary Corpus

Later in our project, we obtained a discharge summary corpus that had not been previously de-identified[2]: it contained genuine PHI. We hand-annotated this corpus and used it in our experiments along with the three re-identified corpora. The fourth column of Table 3.1 shows the break-down of PHI in the authentic discharge summary corpus.

3.3 Baseline Schemes

We compared our approach to a scheme that relies heavily on dictionaries and hand-built heuristics [18], Roth and Yih's SNoW [38], and BBN's IdentiFinder [6]. SNoW and IdentiFinder take into account dependencies between entities in the text (we refer to the informations captured by the dependencies as global context), while our statistical de-identifier

[2]Institutional Review Board approval was granted for this study.

focuses on each word in the text in isolation, using only local context provided by a few surrounding words. We chose these baseline schemes to answer one of the central questions of this chapter: whether global context is any more useful than local context for de-identification in fragmented, clinical narrative text.

3.3.1 Heuristic+dictionary Scheme

Many traditional de-identification approaches use dictionaries and hand-tailored heuristics to identify PHI. We obtained one such system [18] that identifies PHI by checking to see if the target words occur in hospital, location, and name dictionaries, but not in a list of common words. Simple contextual clues, such as titles, e.g., `Mr.`, and manually determined bigrams, e.g., `lives in`, are also used to identify PHI not occurring in dictionaries.

3.3.2 SNoW

Roth and Yih's SNoW system [38] recognizes people, locations, and organizations. SNoW operates in two stages. In the first stage, weak classifiers are used to determine the probability distribution of entity labels for phrases in the sentence. These classifiers take advantage of words in a phrase, surrounding bigrams and trigrams of words, the number of words in the phrase, and information about the presence of the phrase or constituent words in people and location dictionaries. Similar weak classifiers are used to determine the probability distribution of relationships between the entities in the sentence. After this initial step, the system uses the probability distributions and constraints imposed by relationships on the entity types to compute the most likely assignment to relationships and entities in the sentence. One can think of the system as using its beliefs about relationships between entities (the global context of the sentence) to strengthen or weaken its hypothesis about each entity's type.

3.3.3 IdentiFinder

IdentiFinder uses HMMs to learn the characteristics of entity labels, including people, locations, geographic jurisdictions, organizations, dates, and contact information [6]. For each named entity class, this system learns a bigram language model which indicates the likelihood that a sequence of words belongs to that class. A word is modelled as a combination of the actual lexical unit and various orthographic features. To find the names of all entities, the system finds the most likely sequence of entity types in a sentence given a sequence of words; thus, it uses the global context of the entities in a sentence.

3.4 Statistical De-identifier: SVM with local context

We observed that discharge summaries contain fragmented sentences, such as "No fever", and hypothesized that the global context of the entire sentence (such as sequences of PHI entities or relationships between PHI entities) would play a limited role in de-identification. We also noticed that PHI is often characterized by local context. For example, the word `Dr.` before a name invariably suggests that the name belongs to the doctor PHI category.

Consequently, we devised the statistical de-identifier which uses a multi-class SVM to classify each word in the sentence as belonging to one of eight categories: doctor, location, phone, address, patient, ID, hospital, or non-PHI. The SVM uses features of the word to be classified (the target), as well as surrounding words in order to capture the contextual clues we found useful as human annotators. The full set of features we use includes:

- The target itself. This feature allows the system to learn common words that rarely occur as PHI, e.g., `and` and `they`.
- The uninterrupted string of two words occurring before and after the target (we refer to these as *lexical bigrams*). We noticed that PHI is frequently characterized by immediately surrounding words. For example, the right bigram `was admitted` following a target usually indicates that the target is a patient.
- The left and right syntactic bigrams of the target (see Chapter 2 for a description of syntactic bigrams). Syntactic bigrams capture the local syntactic dependencies of the target, and we hypothesize that particular types of PHI in discharge summaries occur within similar syntactic structures. Patients are often the subject of the passive construction `was admitted`, e.g., "John was admitted yesterday". In this case, the lexical bigram feature captures the same information as syntactic context. However, with the introduction of modifying clauses in the text, lexical bigrams may no longer provide sufficient context to distinguish entity types. In the sentence "John, who had a hernia, was admitted yesterday", lexical bigrams no longer recognize the context `was`

`admitted` for `John` while syntactic bigrams continue to identify `John` as the subject of `was admitted`.

- The MeSH ID (see Chapter 2) of the noun phrase containing the target word. MeSH maps biological terms to a hierarchical ID space. We obtain this feature by first shallow parsing the text to identify noun phrases, and then exhaustively searching each phrase for a MeSH ID from the UMLS Metathesaurus. We conjecture that this feature will be useful in distinguishing medical non-PHI from PHI: medical terms such as diseases, treatments, and tests have MeSH ID's, while PHI usually does not.

- The part of speech of the target and of words within a +/- 2 context window of the target. Non-PHI instances are more likely to be nouns than adjectives or verbs.

- The presence of the target and of words within a +/- 2 context window of the target in location, hospital, and name dictionaries. Dictionaries are useful in detecting common PHI.

- The heading of the section in which the target appears, e.g., `HISTORY OF PRESENT ILLNESS`. Discharge summaries have a repeating structure. We have noticed, for example, names almost always follow the `DISCHARGE SUMMARY NAME` heading, and dates follow the `DISCHARGE DATE` heading. For PHI not occurring in narrative portions of the text, we hypothesize that the section headings will be useful in determining PHI type.

- Whether the word begins with a capital letter. PHI, such as names and locations, usually begin with a capital letter.

- Whether the word contains the "-" or "/" punctuation symbols. Dates, phone numbers, and IDs tend to contain punctuation.

- Whether the word contains numerals. Again dates, phone numbers, and IDs consist of numbers.

- The length of the word. Certain entities are characterized by their length, e.g., telephone numbers.

3.5 Evaluation

We evaluated the statistical de-identifier by running it on our four corpora using 10-fold cross-validation. We computed precision, recall, and the F-measure for each corpus. Because the purpose of de-identification is to remove PHI and not to distinguish between types of PHI, we treated the task as a binary classification problem and grouped the 7 PHI classes into a single PHI category.

We also ran SNoW, IdentiFinder, and the heuristic+dictionary scheme on the four corpora. For SNoW, we used 10-fold cross-validation. SNoW only recognizes people, locations, and organizations, and not our full set of PHI; so, we evaluated it only on the PHI it is built to recognize. Unfortunately, we were unable to train IdentiFinder on our corpus, and used an implementation of the algorithm that was pre-trained on a news corpus.

The metric that is of interest to most researchers in de-identification is recall for PHI. This metric measures the percentage of PHI that are correctly identified. Ideally, recall should be very high. We are also interested in maintaining the integrity of the data, i.e., avoiding the classification of non-PHI as PHI. This is captured by precision. In the remainder of the paper, we compare the systems based on their F-measure which combines precision and recall.

3.6 Discussion

3.6.1 De-identifying Re-identified and Authentic Discharge Summaries

We first de-identified the randomly re-identified and authentic discharge summaries. These corpora represent normal, non-pathological input. Tables 3.3, 3.4, 3.5, and 3.6 show that the statistical de-identifier outperformed all other systems on these corpora.

On the randomly re-identified corpus, the statistical de-identifier recognized PHI with an F-measure of 97.63%, while IdentiFinder gave an F-measure of 68.35%, and the heuristic+dictionary scheme gave an F-measure of 77.82%.

We evaluated SNoW only on the three kinds of entities it is designed to recognize. We found that it recognized PHI with an F-measure of 96.39% on the re-identified corpus. In comparison, the statistical de-identifier achieved an F-measure of 97.46%.[3] Similarly, on the

[3]The difference in PHI F-measures between SNoW and the statistical de-identifier is not significant for the re-identified corpus: all other PHI and non-PHI F-measure differences between the statistical de-identifier

authentic discharge summaries, the statistical de-identifier outperformed all other systems in recognizing PHI. This observations is true for all corpora, and suggests, firstly, that using dictionaries alone is insufficient for effectively recognizing PHI: context provides additional useful information (IdentiFinder, SNoW, and the statistical de-identifier all utilize contextual cues). Secondly, the results suggest that using just the local context captured by the statistical de-identifier performs as well as (and sometimes better than) using local context combined with global context (as in SNoW and IdentiFinder).

Method	Class	Precision	Recall	F-measure
Stat De-ID	PHI	98.34%	96.92%	**97.63%**
IFinder	PHI	62.21%	75.83%	68.35%
H+D	PHI	93.67%	66.56%	77.82%
Stat De-id	Non-PHI	99.53%	99.75%	**99.64%**
IFinder	Non-PHI	96.15%	92.92%	94.51%
H+D	Non-PHI	95.07%	99.31%	97.14%

Table 3.3: Precision, Recall, and F-measure on re-identified discharge summaries. IFinder refers to IdentiFinder, H+D refers to heuristic+dictionary approach, Stat De-ID refers to the statistical de-identifier.

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	98.31%	96.62%	**97.46%**
SNoW	PHI	95.18%	97.63%	96.39%
Stat De-id	Non-PHI	99.64%	99.82%	**99.73%**
SNoW	Non-PHI	99.75%	99.48%	99.61%

Table 3.4: Evaluation of SNoW and statistical de-identifier on recognizing people, locations, and organizations found in re-identified discharge summaries.

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	98.46%	95.24%	**96.82%**
IFinder	PHI	26.17%	61.98%	36.80%
H+D	PHI	82.67%	87.30%	84.92%
Stat De-id	Non-PHI	99.84%	99.95%	**99.90%**
IFinder	Non-PHI	98.68%	94.19%	96.38%
H+D	Non-PHI	99.58%	99.39%	99.48%

Table 3.5: Evaluation on authentic discharge summaries.

and other systems in the sections 3.6.1, 3.6.2 and 3.6.3 are significant.

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	98.40%	93.75%	**96.02%**
SNoW	PHI	96.36%	91.03%	93.62%
Stat De-id	Non-PHI	99.90%	99.98%	**99.94%**
SNoW	Non-PHI	99.86%	99.95%	99.90%

Table 3.6: Evaluation of SNoW and statistical de-identifier on authentic discharge summaries.

3.6.2 De-identifying Data with Ambiguous PHI

On the data containing ambiguous PHI, the statistical de-identifier accurately recognized 94.27% of all PHI: its performance measured in terms of F-measure was significantly better than that of IdentiFinder, SNoW, and the heuristic+dictionary scheme on both the complete corpus (see Table 3.7 and 3.8), and only the ambiguous entries in the corpus (see Table 3.9).

An example of PHI that was missed by all baseline schemes but recognized by the statistical de-identifier is the patient name `Camera` in the following example:

"Camera underwent relaxation to remove mucous plugs."

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	96.37%	94.27%	**95.31%**
IFinder	PHI	45.52%	69.04%	54.87%
H+D	PHI	79.69%	44.25%	56.90%
Stat De-id	Non-PHI	99.18%	99.49%	**99.34%**
IFinder	Non-PHI	95.23%	88.22%	91.59%
H+D	Non-PHI	92.52%	98.39%	95.36%

Table 3.7: Evaluation on the corpus containing ambiguous data.

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	95.75%	93.24%	**94.48%**
SNoW	PHI	92.93%	91.57%	92.24%
Stat De-id	Non-PHI	99.33%	99.59%	**99.46%**
SNoW	Non-PHI	99.17%	99.31%	99.24%

Table 3.8: Evaluation of SNoW and statistical de-identifier on ambiguous data.

3.6.3 De-identifying PHI Not Found in Dictionaries

We hypothesized that on the data set containing non-dictionary PHI, context would gain importance. As expected, the heuristic+dictionary method recognized PHI with the lowest

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	94.02%	92.08%	**93.04%**
IFinder	PHI	50.26%	67.16%	57.49%
H+D	PHI	58.35%	30.08%	39.70%
SNoW	PHI	91.80%	87.83%	89.77%
Stat De-id	Non-PHI	98.28%	98.72%	**98.50%**
IFinder	Non-PHI	92.26%	85.48%	88.74%
H+D	Non-PHI	86.19%	95.31%	90.52%
SNoW	Non-PHI	97.34%	98.27%	97.80%

Table 3.9: Evaluation only on ambiguous people, locations, and organizations found in ambiguous data.

F-measures on this data set. Again, the statistical de-identifier outperformed all other approaches obtaining an F-measure of 97.44% for recognizing PHI not found in dictionaries, while IdentiFinder and the heuristic+dictionary scheme had F-measures of 53.51% and 38.71% respectively (see Table 3.10).

Of only the PHI not found in dictionaries, 96.49% were accurately identified by the statistical de-identifier. In comparison, the heuristic+dictionary approach accurately identified those PHI that could not be found in dictionaries 11.15% of the time, IdentiFinder recognized these PHI 57.33% of the time, and SNoW gave an accuracy of 95.08% (see Table 3.12).

An example of an instance of PHI missed by all of the baseline schemes (IdentiFinder, SNoW, and heuristic+dictionary), but recognized by the statistical de-identifier, is the fictitious doctor name `Znw` in the sentence:

> "Labs showed hyperkalemia (increased potassium), EKG showed atrial fibrillation, discussed with primary physicians (Znw) and cardiologist (P. Nwnrgo)."

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	98.12%	96.77%	**97.44%**
IFinder	PHI	52.44%	54.62%	53.51%
H+D	PHI	88.24%	24.79%	38.71%
Stat De-id	Non-PHI	99.54%	99.74%	**99.64%**
IFinder	Non-PHI	93.52%	92.97%	93.25%
H+D	Non-PHI	90.32%	99.53%	94.70%

Table 3.10: Evaluation on the corpus containing PHI not in dictionaries.

Method	Class	Precision	Recall	F-measure
Stat De-id	PHI	98.04%	96.49%	**97.26%**
SNoW	PHI	96.50%	95.08%	95.78%
Stat De-id	Non-PHI	99.67%	99.82%	**99.74%**
SNoW	Non-PHI	99.53%	99.67%	99.60%

Table 3.11: Evaluation of SNoW and statistical de-identifier on the people, locations, and organizations found in the corpus containing PHI not found in dictionaries.

Method	Stat De-id	IFinder	SNoW	H+D
Recall	**96.49%**	57.33%	95.08%	11.15%

Table 3.12: Recall on only the PHI not found in dictionaries.

3.6.4 Feature Importance

To understand the gains of our statistical de-identifier, we determined the relative importance of each feature by running the statistical de-identifier with the following restricted feature sets on the randomly re-identified and authentic corpora:

1. The target words alone.
2. The syntactic bigrams alone.
3. The lexical bigrams alone.
4. The POS information alone.
5. The dictionary-based features alone.
6. The MeSH features alone.
7. The orthographic features alone (e.g., whether or not the words contain punctuation).

The results shown in Tables 3.13 and 3.14 indicate that contextual features (lexical and syntactic bigrams) are the most important features for de-identification in the randomly re-identified corpus. In the authentic corpus, the target word is more informative than other features because this corpus contains repeating doctor and hospital names. Nevertheless, both corpora highlight the relative importance of contextual features. In fact, in both corpora, context is more informative than information from dictionaries, reflecting the repetitive structure and language of discharge summaries.

Class	Feature	P	R	F
Non-PHI	target words	91.61%	98.95%	95.14%
PHI		86.26%	42.03%	56.52%
Non-PHI	lexical bigrams	95.61%	98.10%	96.84%
PHI		85.43%	71.14%	77.63%
Non-PHI	syntactic bigrams	96.96%	98.72%	**97.83%**
PHI		90.76%	80.20%	**85.15%**
Non-PHI	POS information	94.85%	98.38%	96.58%
PHI		86.38%	65.84%	74.73%
Non-PHI	Dictionary	88.99%	99.26%	93.85%
PHI		81.92%	21.41%	33.95%
Non-PHI	Mesh	86.49%	100%	92.75%
PHI		0%	0%	0%
Non-PHI	Orthographic	86.49%	100%	92.75%
PHI		0%	0%	0%

Table 3.13: Comparison of features for randomly re-identified corpus.

Another interesting observation is that for both corpora, syntactic bigrams outperform lexical bigrams (all F-measure changes for the two feature sets are significant, except the changes for non-PHI in the authentic corpus). Most prior approaches to de-identification/named entity recognition have used only lexical bigrams, ignoring syntactic dependencies. Our experiments suggest that syntactic context can be more informative than lexical context.

We conjecture that the lexical context of PHI is more variable than their syntactic context because many English sentences are filled with clauses, adverbs, etc., that separate the subject from its main verb. The Link Grammar Parser can recognize these interjections so that the words break up lexical context but not syntactic context. For example, the word `supposedly` gets misclassified by the lexical bigrams as PHI when encountered in the sentence "Trantham, Faye supposedly lives at home with home health aide and uses a motorized wheelchair". This is because the verb `lives` which appears on the right-hand side of `supposedly` is a strong lexical indicator for PHI. If we parse this sentence with the Link Grammar Parser, we find that the right-hand link for the word `supposedly` is *(lives,E)* where E is the link for "verb-modifying adverbs which precede the verb" [40]. This link is not an indicator of patient names and helps mark `supposedly` as non-PHI.

Syntactic context provides more significant gains in the randomly re-identified corpus than the authentic discharge summary corpus. This is because the two corpora are written

Class	Feature	P	R	F
Non-PHI	target words	98.79%	99.94%	**99.36%**
PHI		97.64%	67.38%	**79.74%**
Non-PHI	lexical bigrams	98.46%	99.83%	99.14%
PHI		92.75%	58.47%	71.73%
Non-PHI	syntactic bigrams	98.55%	99.87%	99.21%
PHI		94.66%	60.97%	74.17%
Non-PHI	POS information	97.95%	99.63%	98.78%
PHI		81.99%	44.64%	57.81%
Non-PHI	Dictionary	97.11%	99.89%	98.48%
PHI		88.11%	21.14%	34.10%
Non-PHI	Mesh	96.37%	100%	98.15%
PHI		0%	0%	0%
Non-PHI	Orthographic	96.39%	99.92%	98.12%
PHI		22.03%	0.61%	1.19%

Table 3.14: Comparison of features for authentic corpus.

in different styles. The randomly re-identified text contains longer, more grammatical sentences. 61.2% of the sentences in this corpus parse at least partially. However, only 51.4% of the sentences in the authentic discharge summary corpus parse at least partially. Hence the authentic discharge summary corpus contains less useful syntactic information, leading to a reduction in the predictive power of the syntactic bigram feature.

3.7 Summary

The experimental results presented in this chapter suggest that local context contributes more to de-identification than global context when working with the disjointed and fragmented sentences of medical discharge summaries. Furthermore, local context is more useful than dictionary information, especially when the text contains uncommon PHI instances that are not present in easily obtainable dictionaries.

Finally, we have shown that syntactic context is at least as important as lexical context for the task of de-identification. The more grammatical the input text, the more successful the parser, and the more significant the contribution of syntactic context to de-identification performance.

Chapter 4

Semantic Category Recognition and Assertion Classification

4.1 Motivation

The overall goal of this thesis is to describe a system that maps information in narrative discharge summaries to standardized semantic representations that a computer can then use for inference and focused querying. We argue that a crucial first step for any such transformation is the recognition of *concepts* in the text, where a concept is defined as a word or phrase of words belonging to a semantic category, e.g., *diseases*. We refer to the task of determining the semantic category of each word as semantic category recognition.

Merely detecting the presence of a concept does not provide sufficient information in medical texts, especially for medical problems, i.e., *diseases* and *symptoms*. The more important questions are “Does the patient have the disease?”, “Has the disease been ruled out?”. We refer to the task of determining whether a problem is present, absent, uncertain, or associated with someone other than the patient as assertion classification.

In this chapter, we present our solutions to semantic category recognition and assertion classification in detail, highlighting the data annotation process, the algorithms employed, and the performance of our approaches compared to alternative solutions.

4.2 Semantic Category Recognition

4.2.1 Data

Before devising a solution for semantic category recognition, we first ascertained the semantic categories of interest in our domain. We obtained a collection of 48 medical discharge summaries spanning 5,166 sentences (after sentence-breaking). We then consulted two doctors in our lab to determine the type of information clinical practitioners would like to extract automatically from discharge summaries. According to their advice and our own observations of the clinical corpus, we defined eight recurring categories which serve as the building blocks for sentence-level semantic interpretation in medical discharge summaries. These are *diseases*, *treatments*, *substances*, *dosages*, *practitioners*, *tests*, *results*, and *symptoms*.

In order to ensure that the eight categories are well-defined and agree with previous work, we mapped semantic types in UMLS to our eight categories. For *diseases* our mapping closely agrees with prior work [29].

The definitions of our categories, in terms of UMLS, are listed below.

- The *diseases* category includes the UMLS semantic types PATHOLOGIC FUNCTION, DISEASE OR SYNDROME, MENTAL OR BEHAVIORAL DYSFUNCTION, CELL OR MOLECULAR DYSFUNCTION, CONGENITAL ABNORMALITY, ACQUIRED ABNORMALITY, INJURY OR POISONING, ANATOMIC ABNORMALITY, NEOPLASTIC PROCESS, and VIRUS/BACTERIUM.

- The *treatments* category includes the UMLS semantic types THERAPEUTIC OR PREVENTIVE PROCEDURE, MEDICAL DEVICE, STEROID, PHARMACOLOGIC SUBSTANCE, BIOMEDICAL OR DENTAL MATERIAL, ANTIBIOTIC, CLINICAL DRUG, and DRUG DELIVERY DEVICE.

- The *substances* category encompasses abusive drugs and drug-related practices. Examples include `narcotics`, `alcohol`, `drug abuse`, and `smoking`. The closest UMLS semantic type is HAZARDOUS OR POISONOUS SUBSTANCE.

- The *dosages* category includes information about the quantitative amount of a medication to be taken and the instructions for taking the medication (e.g., `200 mg b.i.d`, `intravenous`, and `p.o.`). This category does not have a UMLS equivalent.

- The *symptoms* category corresponds to the UMLS semantic type SIGNS OR SYMPTOMS. Signs refer to patient characteristics that are visible or easily obtainable, whereas symptoms refer to patient characteristics that are subjective (such as `pain`). We constrained our *symptoms* category to correspond to negative characteristics or problems. Hence, `low blood pressure` is labelled as belonging to the *symptoms* category, but `alert` and `awake` are not.

- The *tests* category includes the UMLS semantic types LABORATORY PROCEDURE, DIAGNOSTIC PROCEDURE, CLINICAL ATTRIBUTE, and ORGANISM ATTRIBUTE.

- The *results* category refers to the results of tests. It also includes findings of a patient that are not easily obtainable, such as `consolidation` in the sentence, "X-rays showed consolidations". *Results*, unlike the *symptoms* category, includes non-deleterious patient properties, such as `afebrile`. Our *results* category encompasses the UMLS semantic types LABORATORY OR TEST RESULT and FINDING.

- The *practitioners* category refers to medical practitioners involved in the treatment and diagnosis of the patients. The closest UMLS semantic types are BIOMEDICAL OCCUPATION OR DISCIPLINE and PROFESSIONAL OR OCCUPATIONAL GROUP.

4.2.2 Annotation

Figure 4-1 displays a screen shot of a GUI we developed for in-house semantic category annotation. The GUI takes as input the file to be annotated. For each file, it maintains the annotator's position within the text in a metadata file. Hence, an annotator can save his/her work, exit the program, and resume at a later date from the same point.

The GUI allows the annotator to navigate through the corpus sentence by sentence using the next (">>") and back ("<<") buttons. At any time, the annotator can highlight a word or phrase and select a semantic category for the phrase from the menu. The GUI immediately updates the text pane by highlighting the phrase with the color corresponding to the selected semantic label (the menu also serves as a color palette). Labels can be removed by selecting the "None" category. Clicking the "save" button updates the input file to reflect the new annotations.

We use an XML-style format to store the annotated text. Labelled concepts are enclosed within tags that denote their semantic category. Table 4.1 shows the correspondence

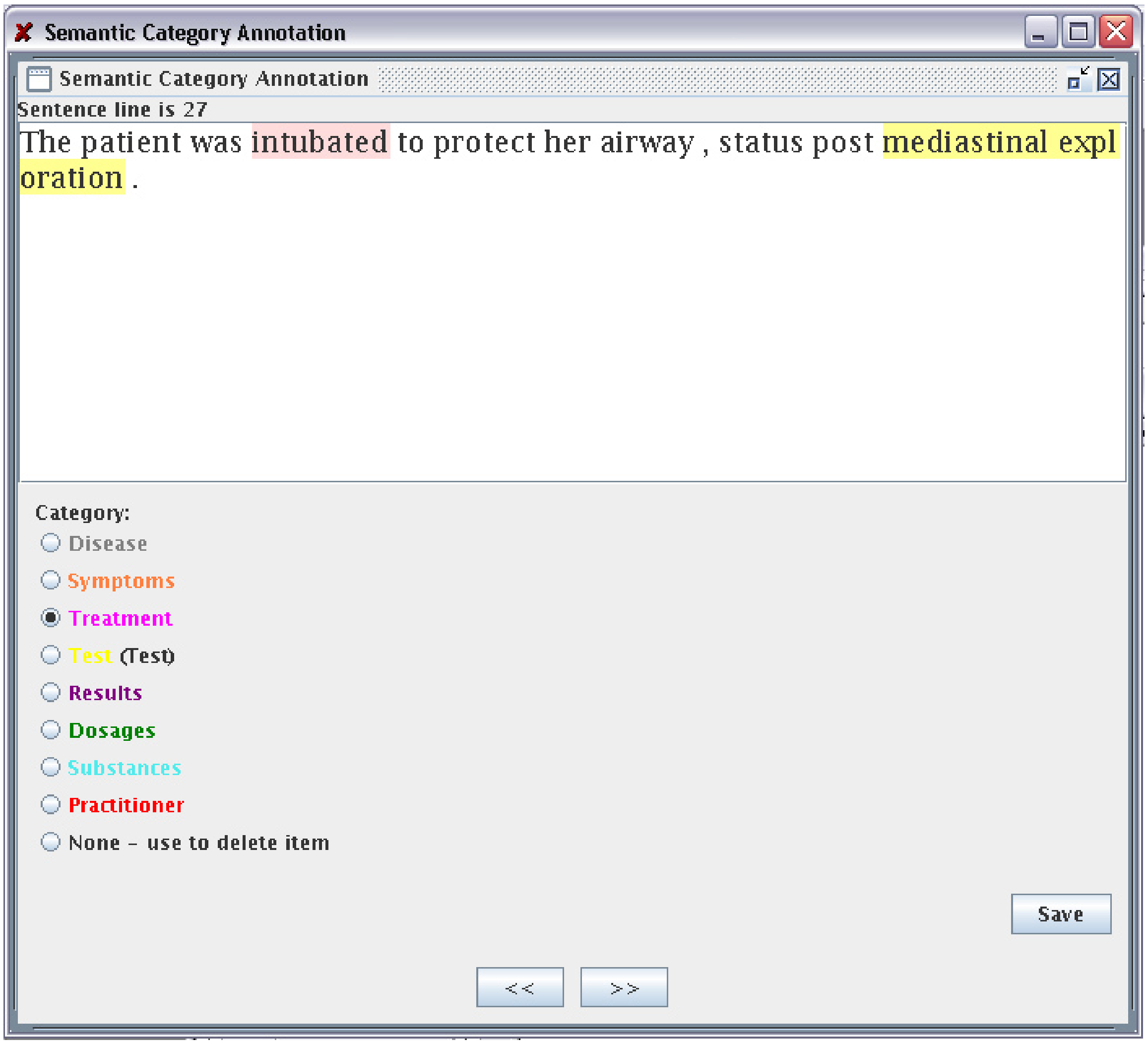

Figure 4-1: Semantic category annotation GUI.

Semantic Category	**XML abbrev.**
Disease	dis
Symptom	symps
Treatment	med
Result	results
Substance	subs
Practitioner	cons
Dosage	dos
Test	test

Table 4.1: XML abbreviations for semantic categories.

between semantic categories and the abbreviations used for the XML tags. For example, the sentence

“He was given antibiotics for his malaria.”

after annotation becomes:

“He was given <med> antibiotics </med> for his <dis> malaria </dis> .”

We produced an annotation guide that initially defined the semantic categories in terms of UMLS semantic types and gave examples of each in context. The guide was then given to the author and another computer science student, who independently annotated the data using the GUI. After initial annotation, we computed the Kappa agreement between the two annotators.

Kappa is “a measure of agreement between two observers, taking into account agreement that could occur by chance” [3]. The Kappa statistic (K) is defined as:

$$K = \frac{P(A) - P(E)}{1 - P(E)} \tag{4.1}$$

where $P(A)$ is the proportion of times the annotators agree, and $P(E)$ is the proportion of times that we would expect them to agree by chance. According to Congalton [12], $K > 0.8$ indicates strong agreement, while $0.4 < K < 0.8$ represents moderate agreement, and a value below 0.4 represents poor agreement.

To compute K, we first found $P(A)$, the proportion of agreement, defined as:

$$P(A) = \frac{\text{number of words annotators labelled identically}}{\text{total number of words in the corpus}} \tag{4.2}$$

We included *None* (referring to terms that belong to none of the semantic categories) as a label in this computation. Then, for each label, i, we defined $P_{1,i}$ as the probability that annotator 1 selects i as a label for a word, and $P_{2,i}$ the probability that annotator 2 selects i as a label for a word. In general $P_{j,i}$ was defined as:

$$P_{j,i} = \frac{\text{number of words annotator } j \text{ labelled as } i}{\text{total number of words in the corpus}} \tag{4.3}$$

For each class, i, we computed, $Q(i)$, the probability that both annotators label a word as belonging to class i:

$$Q(i) = P(1,i) \times P(2,i) \tag{4.4}$$

The expected agreement, $P(E)$ is defined as the probability that both annotators label a word identically, and is the sum of $Q(i)$ values, for all labels, i:

$$P(E) = \sum_{\text{all classes } i} Q(i) \tag{4.5}$$

K is computed from $P(A)$ and $P(E)$ as in Equation 4.1.

The initial Kappa inter-annotator agreement was 79.6%. Sources of disagreement arose primarily in the determining concept boundaries. We had not explicitly decided how to handle determiners and adjectives before concepts, such as `a` and `acute`. Also, determining the boundary of *results* proved to be non-trivial.

We modified our annotation guide to include a set of rules for determining the boundaries of concepts and disambiguating confusing terms. Examples of the rules we developed are shown in Table 4.2.

Both annotators were then retrained with the new guide and proceeded to annotate the same corpus again. The second time around the Kappa agreement was 93.0%, which signifies strong inter-annotator agreement. Disagreement arose in determining concept boundaries. Deciding whether or not an adjective is a dispensable modifier or an integral part of a *disease* turned out to be especially subjective.

Finally, the two annotators reviewed and decided upon correct labels in the face of disagreement and produced a single annotated corpus. In the final corpus the break down of semantic categories is shown in Table 4.3.[1]

[1]The numbers indicate the total number of words tagged as belonging the corresponding semantic cate-

Nested labels and multiple labels for a word or phrase are not allowed.

For the *diseases* category, do not include determiners or adjectives within the category, unless they are parts of commonly used collocations. For example, consider the *disease* `chronic obstructive pulmonary disease`. In this case, `chronic` is an integral part of the *disease*, so is included within the concept name. However, for the phrase `mild pulmonary edema`, `mild` is considered a non-essential modifier and is consequently excluded from the concept name.

Include body parts occurring before the disease name, such as `pulmonary` in the *disease*, `pulmonary artery disease`. Exclude body parts occurring after the disease name in prepositional phrases. Thus, only label `lesion` as a *disease* in the phrase `lesion in the right artery`.

Measurements after a *disease* should be labelled as *results* and should be excluded from the *disease* concept. Thus, `30s` is labelled as a *result* and `ventricular tachycardia` is labelled as a *disease* in the phrase `ventricular tachycardia to the 30s`.

For *symptoms*, include body parts that precede the *symptom*, but exclude body parts occurring after the *symptom* in prepositional phrases.

Numbers after *symptoms* should be labeled as *results*, such as `100` in the phrase `fever to 100`.

Results can include entire clauses. For example, "lungs were clear to auscultation bilaterally", is an instance of *results*. In general include as much of the phrase as is required to understand the concept.

Results can be mixed with *diseases* as in "X-ray showed lesions, consolidation, and opacity". Here, `lesions` is labelled as a *disease*, and `consolidation` and `opacity` are labelled as *results*.

The *results* category takes precedence over the *treatments* category. That is, even though `face mask` is a medical device, and thus a *treatment*, it is labelled as part of the *results* phrase `20% on face mask` in the sentence "Saturations of 20% on face mask".

Table 4.2: Annotation rules.

Category	**Instances**	**Category**	**Instances**
None	27,228	Diseases	2,757
Treatments	2,685	Symptoms	1,410
Results	5,059	Tests	3,185
Substances	116	Practitioners	715
Dosages	3,086		

Table 4.3: Breakdown of words into semantic categories in our corpus.

4.2.3 Baseline: UMLS-based system

The UMLS-based system maps free text to UMLS semantic types. We first use a tagger [7] to identify the parts of speech of the words used in the text. Then, we employ a shallow parser [35] that uses the part of speech tags to identify noun phrases. Subsequently, we exhaustively search each noun phrase for UMLS semantic types. We attempt to find the longest string involving the head noun, that maps to one of our 8 recognized categories. We assume that the head noun is usually the right most element of a noun phrase. We begin by searching sub-phrases containing the head noun for UMLS semantic types before considering other sub-phrases. Consider the phrase, `coronary artery disease`. Our algorithm searches for a semantic type for this phrase by considering its sub-phrases in the following order: `coronary artery disease`, `artery disease`, `disease`, `coronary artery`, `artery`, and `coronary`.

We normalize each phrase and look up the resultant string in the UMLS Metathesarus. The search halts when one of the sub-phrases (considered in order) returns one or more UMLS semantic types that map to one of our 8 semantic categories. Every single word in the noun phrase is then labeled with the representative semantic category. Returning to our example, the phrase `coronary artery disease` corresponds to the UMLS semantic type DISEASE OR SYNDROME, which maps to our *diseases* category. Hence, every word in the phrase is labeled as an instance of *diseases*.

We refer to this scheme as the UMLS-based system. The UMLS-based system incorporates the following performance-enhancing heuristics:

- It automatically tags noun phrases containing `MED` and `Dr.` as *practitioners*.
- It automatically labels 29 abbreviations, such as `cc`, as belonging to the *dosages* category.
- If numbers occur in *treatment* phrases, the UMLS-based system automatically labels them as *dosages*.
- If a noun phrase refers to a body part (according to UMLS), surrounding adjectives (within a +/- 3 context window) and the noun phrase itself are labeled as *results*. This allows recognition of complex *results* clauses, such as `abdomen was non-tender`.

---

gory. For example, the disease `coronary artery disease` yields three disease instances.

Class	Precision	Recall	F-measure
None	0.828	0.883	0.855
Disease	0.656	0.707	0.680
Treatment	0.548	0.726	0.625
Test	0.764	0.560	0.646
Results	0.404	0.358	0.380
Dosages	0.901	0.597	0.718
Symptoms	0.653	0.334	0.442
Medical Practitioner	0.486	0.733	0.584
Abusive substances	0.685	0.128	0.215

Table 4.4: Results of the UMLS-based system.

- The UMLS-based system has a list of 28 stop words that are ignored during UMLS lookup. Determiners and possessive pronouns are also ignored.
- The UMLS-based system considers gerunds as candidate phrases (e.g., `smoking`).
- If the UMLS-based system encounters a number in a test phrase, it assumes the number is a *result*.
- Numbers preceded by the keywords, `from`, `than`, `to`, `high`, and `low` are assumed to be *results*.

Because a phrase can have multiple matches, we allow the UMLS-based system to return multiple semantic types for a given phrase.

We evaluate the UMLS-based system by computing the precision, recall, and F-measure for the classification of each word in the corpus compared to our manually annotated gold standard. Table 4.4 shows the results obtained.

The F-measures for *disease*, *treatments*, and *tests* are comparatively high (over 60%). This is consistent with prior research showing that concepts such as diagnoses and procedures can be effectively identified by mapping text to UMLS. These tend to be short phrase concepts (rarely more than 4 words). On the other hand, the F-measures for the recognition of *symptoms* and *results* is discouragingly low. Analysis of the errors shows that the *results* and *symptoms* categories are often realized as multi-word phrases or clauses that do not occur in the UMLS ontology. For example, consider the sentence: "A portable chest x-ray dated 9/5/2001 showed the [nasogastric tube to be in good position with a left sided PICC line at the superior vena cava]". In this sentence, the *results* phrase is shown in square brackets. Long phrases like this are very difficult for a simple UMLS look-up scheme to recognize. The phrases `nasogastric tube` and `left-sided PICC line` are separately identified as *treatments*, but are not combined to generate a *results* phrase.

For the *practitioner* class and *substances* class, the majority of errors arise because our definitions of these categories do not match any of the UMLS semantic types.

In general, the errors encountered can be categorized into the following groups:

- Boundary errors: Boundary errors are errors made in determining the extent of a concept.

- Differences in definitions: Occasionally, the way we define a category does not match UMLS's definition. We consider `neck hematoma` a *disease*, for example, but UMLS considers it an instance of FINDING.

- Errors that arise due to ignoring context: Contextual information would be useful in correcting mistakes made by the UMLS-based system. For example, consider the sentence "The daughter's name is Nancy Smith". Our UMLS-based system classifies `Nancy Smith` as a *disease*, because the word `Smith` maps to the UMLS semantic type INJURY OR POISONING. Clearly, an appreciation of the context in which this word occurs would eliminate this type of error.

Theoretically, one could add more rules to the UMLS-based system to increase the classification accuracy of various categories. However, we hypothesize that we can improve performance by simply encoding the lexical and syntactic features of our data, and using statistical machine learning to automatically acquire the necessary rules that characterize the different semantic categories.

4.2.4 Statistical Semantic Category Recognizer

The statistical semantic category (SC) recognizer uses an SVM with linear kernel to classify each word (excluding punctuation) as belonging to one of eight semantic categories or the *none* category reserved for irrelevant words. The SVM employs features that capture as much of the context and characteristics of the target words as possible. We hypothesize that one of the strongest cues of the semantic category of a word is the target itself, since many words, such as `malaria`, are invariably associated with certain categories (*diseases* in this example).

The remainder of the features that we consider can be divided into four groups: orthographic, syntactic, ontological (semantic), and lexical.

Orthographic Features

Orthographic features refer to surface properties of the target word. This category consists of four binary properties:

- Is the word capitalized? Doctor's names and *treatments* tend to be capitalized.
- Is the entire word capitalized? This feature is useful in detecting *tests* that appear as capitalized section headings, e.g., `ELECTROCARDIOGRAM:`.
- Does the word contain numerals? Words containing numbers are frequently instances of *dosages* or *results*.
- Does the word contain punctuation? This feature may be useful in disambiguating between dates and numbers that occur as *results* of *tests*. The former frequently contains the "`/`" punctuation symbol.

Lexical Features

Our lexical features include:

- The left and right lexical bigrams of the target. We hypothesize that lexical bigrams will capture enough local context to characterize semantic categories.
- The section of the discharge summary in which the target appears. Certain sections are more likely to contain specific semantic categories.

Syntactic Features

To obtain syntactic features, we use the Link Grammar Parser as described in Chapter 2. In particular, we extract bigrams of syntactic dependencies referred to as syntactic bigrams.

We use three types of syntactic features:

- Target syntactic bigrams. These are syntactic bigrams of the target word, constructed as described in Chapter 2. As in de-identification, we hypothesize that certain semantic categories are characterized by their syntactic context.

- The head of the noun phrase containing the target, and the syntactic bigrams of the head (later referred to as head syntactic bigrams).[2] We hypothesize that for long noun phrases, the individual syntactic links of modifiers are not indicative of the semantic category of the phrase. Instead, the links of the head are often sufficient to characterize all the words in the noun phrase.

- The part of speech of the target and the parts of speech of words within a +/- 2 context window of the target.

Ontological Features

We used the predictions of our UMLS-based system as a feature for each word. We hypothesize that these predictions will provide useful information for the categories that are well-represented in UMLS, namely *diseases*, *treatments*, and *tests*.

4.2.5 Results

We evaluated the statistical SC recognizer by using 10-fold cross-validation on our corpus of 5,166 sentences. The statistical SC recognizer classified the semantic category of each word in the corpus. We computed F-measures for the following sets of features:

- The complete feature set (all features).
- Lexical features only.
- Syntactic features only.
- Predictions from the UMLS-based system alone.
- The target alone.

Note that the syntactic feature set consists of target and head syntactic bigrams only; the head itself is excluded. Similarly, lexical features include only lexical bigrams and section headings (the target is excluded).

[2]For words not occurring in noun phrases, this feature is identical to the target syntactic bigrams.

Class	Features	Precision	Recall	F-measure
None	All Features	0.938	0.962	0.950
Disease		0.911	0.899	0.905
Treatment		0.924	0.901	0.912
Test		0.931	0.913	0.922
Results		0.857	0.809	0.832
Dosage		0.966	0.941	0.954
Symptoms		0.901	0.815	0.856
Practitioner		0.978	0.934	0.956
Substance		0.934	0.853	0.892

Table 4.5: Results for the statistical SC recognizer when using all features.

4.2.6 Discussion

Comparison of All Features to UMLS-based system

The results in Table 4.5 show that the statistical SC recognizer with all features significantly outperforms the UMLS-based system in all categories. At times, the increase in F-measure is over 60%. Here are some examples that the UMLS-based system misses and the statistical SC recognizer corrects:

- Boundary errors: The UMLS-based system includes the modifier `resultant` as a *disease* in the sentence, "However , the patient had a resultant [mediastinal hematoma]" where the actual *disease* is shown in square brackets. The statistical SC recognizer correctly identifies the *disease* in this case.

- Definition errors: The UMLS-based system classifies `neck hematoma` as a *result* in the sentence "The patient had neck hematoma", while the statistical SC recognizer learns our mapping of `neck hematoma` to *disease*.

- Contextual errors: The UMLS-based system misclassifies `rehabilitation` as a *treatment* when it should be a hospital building or facility in the following sentence: "The patient is to follow up with Dr. Joe, who will see her at rehabilitation". The statistical SC recognizer correctly interprets this sentence.

Despite these features, the statistical SC recognizer is not perfect; it makes contextual errors, for example, `infectious disease` is misclassified as a *disease* instead of a *practitioner* in the sentence, "The patient was continued on antibiotics per infectious disease

Class	Features	Precision	Recall	F-measure
None	Target alone	0.856	0.957	0.904
Disease		0.785	0.745	0.764
Treatment		0.884	0.744	0.808
Test		0.815	0.847	0.831
Results		0.732	0.467	0.570
Dosage		0.827	0.758	0.791
Symptoms		0.876	0.590	0.705
Practitioner		0.874	0.622	0.727
Substances		0.829	0.793	0.811
None	UMLS alone	0.825	0.891	0.857
Disease		0.666	0.718	0.691
Treatment		0.554	0.810	0.658
Test		0.806	0.649	0.719
Results		0.477	0.343	0.400
Dosage		0.903	0.597	0.719
Symptoms		0.648	0.344	0.449
Practitioner		0.545	0.701	0.613
Substances		0.719	0.198	0.311

Table 4.6: Results of the statistical recognizer using target alone and UMLS-based predictions alone.

recommendations". Also, *results* and *symptoms* concepts still pose problems with their length and extent beyond noun phrases.

Comparison of Main Feature Groups

In order to understand where the gains of the statistical SC recognizer originate, we investigated the contribution of our main feature groups: syntactic, lexical, the target words, and UMLS-based predictions. The results show that the UMLS-based features provide the least information among the four feature groups (see Tables 4.6 and 4.7). In fact, the performance of the statistical SC recognizer with the UMLS-based features is very close to the performance of the UMLS-based system. We would not expect the statistical SC recognizer, using the UMLS-based system's predictions as features, to perform much better than the UMLS-based system itself without additional information. The small observed improvement is because the statistical SC recognizer learns a mapping from multiple values to a single prediction in cases where the UMLS-based system returned more than one value.

The target word feature set outperforms the lexical and syntactic feature sets for certain categories (such as *diseases*, *treatments*, *tests*, and *substances*). The target word is useful in these instances, because the same *diseases*, *treatments*, *tests*, and *substances* occur

Class	Features	Precision	Recall	F-measure
None	Lexical	0.844	0.931	0.886
Disease		0.739	0.669	0.702
Treatment		0.771	0.601	0.676
Test		0.812	0.684	0.743
Results		0.751	0.648	0.696
Dosage		0.929	0.884	0.906
Symptoms		0.750	0.522	0.616
Practitioner		0.889	0.727	0.800
Substances		0.825	0.569	0.673
None	Syntactic	0.901	0.939	0.919
Disease		0.765	0.746	0.755
Treatment		0.822	0.769	0.795
Test		0.831	0.803	0.817
Results		0.796	0.733	0.763
Dosage		0.931	0.898	0.914
Symptoms		0.747	0.621	0.679
Practitioner		0.911	0.890	0.900
Substances		0.849	0.629	0.723

Table 4.7: Results of the statistical SC recognizer using lexical features alone and syntactic features alone.

frequently throughout the text. For example, the word `alcohol` occurs 33 times in the corpus—28 times as a *substance*.

However, for complex categories, such as *results* and *symptoms* or categories involving numbers that do not repeat throughout the text, such as *dosages*, contextual features provide more useful information. For example, the *practitioners* concept in square brackets in the sentence "[Infectious disease] was consulted" is misclassified by the target-only model as a *disease* based on the words it contains but is correctly classified by the use of syntactic context. The syntactic features of this system expose the fact that `disease` is the subject of `was consulted` which is a strong indicator of the *practitioners* category.

Finally, we notice that syntactic features outperform lexical features (all the F-measure increases are significant except for the increases in *dosages* and *symptoms* categories).

Comparison of Syntactic Features

To understand why the syntactic feature set outperformed the lexical feature set, we conducted another experiment. We ran the statistical SC recognizer with just lexical bigrams (which are the most important of the lexical features), just target syntactic bigrams, and just head syntactic bigrams, to get a sense of which component of the syntactic features

Category	Feature
None	target syntactic bigrams
Disease	head syntactic bigrams
Treatment	target syntactic bigrams
Test	target syntactic bigrams
Results	target syntactic bigrams
Dosage	lexical bigrams
Symptoms	head syntactic bigrams
Practitioner	head syntactic bigrams
Substances	target syntactic bigrams

Table 4.8: Best performing feature for each category.

gives the feature-set a boost over lexical features. The results show that all three sub-feature sets provide similar amounts of information. Each one outperforms the others in some categories, but there is no clear winner. The feature with the best performance in each category is shown in Table 4.8.

Notice that in all but one case, one of the syntactic features (head syntactic bigrams, or target syntactic bigrams) outperforms lexical bigrams. We noticed that the head syntactic bigrams and target syntactic bigrams complement each other: when combined, the weaknesses of the individual feature sets are overcome. For example, head syntactic bigrams on their own sometimes give false positives due to phrase boundaries. Every single word in a noun phrase has the same head syntactic bigrams, leading to the erroneous labelling of adjectives with head word semantics. This error is corrected by adding target syntactic bigrams. On the other hand, in the sentence "At that time the [stroke fellow] was contacted" head syntactic bigrams correctly mark `stroke fellow` as a practitioner, while target syntactic bigrams alone misclassify the phrase. This is because the head syntactic bigrams replace the context of `stroke` with the context of `fellow`, leading to the correct classification of the phrase. In short, the head and target syntactic bigrams combine synergistically to outperform lexical features.

4.2.7 Summary

In this section, we have described a statistical approach to semantic category recognition in discharge summaries using SVMs. We have shown that in the domain of medical discharge summaries, contextual clues provide stronger indications of the semantic categories of words than the words' UMLS semantic types, and that the syntactic context is the strongest determinant of certain semantic categories.

4.3 Assertion Classification

Assertion classification refers to the task of determining whether a problem, i.e., a *disease* or *symptom*, is present, absent, uncertain, or associated with someone other than the patient. In this section, we define the assertion labels in more detail, describe a statistical assertion classifier, and compare its performance to a rule-based assertion classifier.

4.3.1 Data and Annotation

For this task, we used the same 5,166 sentence discharge summary corpus from our work on semantic category recognition. We additionally obtained a smaller corpus of 3,453 lines that we used as a development set for manually generating rules and selecting features for assertion classification. We defined four types of assertions for medical problems:

- Present: The problem is or was definitely present in the patient. Examples include:[3]
  - "A CT scan showed that she had [airway stenosis]."
  - "Solu-Medrol was given for [tracheal edema]."
- Absent: The problem is absent in the patient. Examples include:
  - "The initial read showed no [demyelinating]."
  - "A subsequent urine culture was negative for any [bacterial growth]."
- Uncertain: The patient possibly has the problem. Examples include:
  - "This was thought possibly to be a [neoplasm]."
  - "The patient felt that her shortness of breath might have just been [anxiety]."
- Alter-association: The problem is not associated with the patient. Examples include:
  - "Sick contact positive for family member at home with a [cough]."
  - "Her brother had a [myocardial infarction] in his 40s."

[3]The problem is shown in square brackets.

We also developed a set of rules to clarify ambiguous situations:

1. The alter-association class takes precedence over all other assertion classes. Hence, in the example, "His brother does not have [myocardial infarction]", the *disease* in square brackets is labelled as being associated with someone other than the patient, instead of being absent.

2. When labelling the assertion class of a problem in a sentence, we ignore information regarding the same problem occurring in other parts of the document. Thus, if one sentence says: "He may have [pneumonia]" and a later sentence says "[Pneumonia] was ruled out", `pneumonia` is labelled as being uncertain in the first case and absent in the second case.

3. If a sentence mentions a medical problem and states that the problem was cured, we still label the problem as being present. This may seem counter-intuitive, but our present assertion class encompasses problems the patient currently has and problems the patient had in the past. Moreover, in Chapter 5 we describe a system that recognizes whether present problems are cured by a *treatment*. Hence, in the sentence "Antibiotics cured his [pneumonia]", `pneumonia` is classified as present.

Using the above rules, the author annotated the corpus. He first identified the concepts in the corpus using the GUI from semantic category recognition. He then manually reviewed the annotated text, looking for *diseases* and *symptoms*. For each *disease* or *symptom*, he determined its assertion class and modified the annotation to reflect this. For example, the input sentence:

"<med> Solu-medrol </med> was given for <dis> tracheal edema </dis>."

becomes, after labelling assertions:

"<med> Solu-medrol </med> was given for <dis-pres> tracheal edema </dis-pres>."

The breakdown of assertion classes in the 5,166 evaluation corpus is shown in Table 4.9.

4.3.2 Baseline Statistical Assertion Classifier

Buoyed by our success with machine learning for the tasks of semantic category recognition and de-identification, we designed a baseline statistical assertion classifier that uses a single

Class	Instances	Class	Instances
Disease Present	1,026	Symptoms Present	511
Disease Uncertain	133	Symptoms Uncertain	36
Disease Absent	154	Symptoms Absent	244
Disease Alter-association	18	Symptoms Alter-association	3

Table 4.9: Assertion class breakdown in validation corpus.

multiclass SVM to label problems as being present, absent, uncertain, or associated with someone other than the patient.

Given a document in which semantic categories have been labelled, the statistical assertion classifier extracts features for each problem, i.e., *disease* or *symptom*, in the text and then labels the problem's assertion class by feeding its features to the SVM.

The features used by the SVM can be divided into two sets: lexical context features and syntactic context features.

Lexical Context Features

Lexical context features include:

- The left and right lexical trigrams of the target problem. In order to obtain this feature, we first replaced each concept in the text with a one word place holder, signifying its semantic category. For example, the sentence:

  "A <test> CT scan </test> showed that she had <dis> airway stenosis </dis> at this time."

  becomes:

  "A TEST showed that she had DIS at this time."

  Replacing a concept, such as `CT scan`, with its semantic category relieves the problem of sparse data: we abstract over lexical variations in semantic categories. We hypothesize that the lexical trigrams will capture enough information for the statistical model to learn the context characterizing the different assertion classes.

- The section of the discharge summary in which the target problem appears. A problem appearing in the `Family History` section of discharge summaries is likely to belong to the alter-association assertion class.

Syntactic Context

Syntactic contextual features include:

- The verb preceding the target problem. Verbs are often indicative of the assertion type of a problem. For example, the verb `showed` preceding a problem suggests that the problem is present.

- The verb succeeding the target problem.

- The syntactic bigrams of the target problem, obtained by extracting the syntactic bigrams of the head noun in the target problem phrase. As for lexical bigrams, we abstract over lexical variations by replacing concepts with their semantic categories. For example, the following parse for the sentence "A chest x-ray revealed airway stenosis":

```
    +---------------------------Xp--------------------------+
    +--------Wd-------+                                     |
    |     +-----Ds----+        +---------Os---------+       |
    |     |    +--AN--+----S---+         +----AN----+       |
    |     |    |      |        |         |          |       |
LEFT-WALL a chest.n x-ray revealed.v airway.n stenosis[?].n .
```

  becomes, after replacing concepts with their semantic categories and removing syntactic links running between words in the same concept:

```
    +----------------Xp----------------+
    +----Wd----+                       |
    |     +-Ds-+---Ss---+----Os---+    |
    |     |    |        |         |    |
LEFT-WALL a  TEST.n revealed.v    DIS.n .
```

- The syntactic bigrams of the verb preceding the target problem and of the verb succeeding the target problem. Knowing that the target problem is the object of a verb whose subject is `brother` suggests that the assertion category is alter-association.

Class	Precision	Recall	F-measure
Present	0.929	0.967	0.947
Absent	0.947	0.900	0.923
Uncertain	0.723	0.556	0.629
Alter-association	1.00	0.810	0.895

Table 4.10: Assertion classification results for statistical assertion classifier.

Evaluation

We evaluated the statistical assertion classifier using 10-fold cross-validation. We classified the assertion category of each *disease* and *symptom* in the 5,166 sentence corpus. Table 4.10 shows the results.

The statistical assertion classifier performs poorly on the assertion classes with the least number of examples, that is the alter-association and uncertain labels. The poor performance is because there is too little data for the SVM to learn a mapping from input features to assertion class.

The statistical assertion classifier makes several recurring mistakes:

- Uncertain medical problems, whose context is rarely seen in the corpus, are routinely misclassified. For example, in the sentence "The patient was given heparin to treat for presumed unstable angina", `unstable angina` is classified as being present instead of uncertain, because the statistical assertion classifier does not recognize the infrequently-used word `presumed` as indicative of uncertainty.

- Our lexical features encompass words within a +/- 3 context window of the target problem, and our syntactic features extend no further than two syntactic links from the target problem. Hence, we miss contextual assertion clues that fall outside of this window. In the sentence "No history of complaint of chest pain, shortness of breath", the *symptom* `shortness of breath` is separated from the indicative phrase `no history of` by more than three words.[4]

- We found that the word `no` is involved in complex constructions that are ambiguous even for human readers. For example, in the sentence "No JVP, 2+ swelling, no pain", `JVP` and `pain` appear to be absent, while `swelling` is present. This is suggested by

[4]We tried extending our context windows, but noticed a deterioration in performance.

the punctuation and the use of the word `no` multiple times. Our statistical method misclassified `swelling` as absent in the example sentence.

4.3.3 Rule-based Assertion Classifier

The case for a rule-based method

It seems that our statistical assertion classifier does not have enough data to learn the contextual patterns categorizing assertion classes. Often, common sense phrases that suggest uncertainty, such as `probable`, occur too infrequently in the corpus to be statistically correlated with any assertion class.

However, it is relatively easy for a human to build a set of contextual patterns, even from sparse data. For example, seeing the sentence "He has probable pneumonia", we can tell, using common sense, that `probable` is the key word indicative of uncertainty. Furthermore, there is little ambiguity in the patterns generated. `Probable` almost always signifies uncertainty. Indeed, the simplicity of the task of assertion classification makes a rule-based approach attractive for our data. In contrast, the problem of semantic category recognition proved difficult even for a human annotator and we opted for a machine learning approach over a rule-based scheme to avoid the pain-staking hours required to develop adequate rule sets.

Building Dictionaries of rules

Our rule-based assertion classifier resembles NegEx. For each medical problem, it checks whether the problem is surrounded by phrases indicative of negation, uncertainty, or alter-association, and classifies the problem accordingly.

We manually generated dictionaries of common phrases for the different assertion classes by reviewing the development corpus and identified the keywords surrounding annotated medical problems. We developed the following dictionaries:

- Common phrases that precede a problem and imply that the problem is absent (later referred to as *preceding negation phrases*). Examples include: `absence of`, `fails to reveal`, and `free of`.

- Common phrases that succeed a problem and imply that the problem is absent (later referred to as *succeeding negation phrases*). Examples include `was ruled out` and `is`

Phrase type	Instance
Preceding Negation	68
Succeeding Negation	15
Preceding Uncertainty	56
Succeeding Uncertainty	7
Alter-association	14

Table 4.11: Breakdown of the different phrase types.

`negative`.

- Common phrases that precede a problem and imply that the problem is uncertain (later referred to as *preceding uncertainty phrases*). Examples include `believe to be`, `assumed`, and `question of`.
- Common phrases that succeed a problem and imply that the problem is uncertain (later referred to as *succeeding uncertainty phrases*). Examples include `versus`, `is probable`, and `might be active`.
- Common phrases that precede a problem and imply that the problem is associated with someone other than the patient (later referred to as *preceding alter-association phrases*). Examples include: `cousin`, `sister`, and `brother`.
- Common phrases that succeed a problem and imply that the problem is associated with someone other than the patient (later referred to as *succeeding alter-association phrases*). This dictionary is identical to the dictionary of preceding alter-association phrases.

We also added morphological variants of the phrases to the dictionaries. For example, if `ruled out` was found to be a common succeeding negation phrase, then we also added the phrases `ruling out`, `rules out`, and `rule out` to the succeeding negation phrases dictionary. Table 4.11 indicates how many instances of each phrase type we identified.

The Rule-based Assertion Classifier Algorithm

Given a document in which semantic categories have been labelled, the rule-based assertion classifier replaces multi-word problems (*diseases* and *symptoms*) with a single word that uniquely identifies the problem in that sentence. In particular, each problem is replaced with the unique tag <`B-E`>, where B is the position of the first character of the problem within the unannotated sentence and E is the position of the last character. Thus

"The CT scan was consistent with <dis> airway stenosis </dis> at this time."

becomes:

"The CT scan was consistent with <32-46> at this time."

Then, for each problem, the rule-based assertion classifier uses regular expressions to determine which dictionary phrases occur within a four word context window. More specifically, the rule-based assertion classifier first checks for alter-association phrases preceding or succeeding the problem within four words. It then checks for negation phrases and finally uncertainty phrases, all within four words of the problem. The rule-based assertion classifier is greedy: if a match is found at any stage, the problem is labelled with the assertion corresponding to the phrase type.

The search order is deliberate. The alter-association class takes precedence over all other assertion classes, thus we label it first. Negation phrases are often very close to the problem they modify, e.g., `no headache`, hence we search for them before looking for uncertainty phrases.

The rule-based assertion classifier incorporates the following performance enhancing heuristics:

1. If a problem is preceded by a possessive pronoun, e.g., `his malaria`, it is automatically labelled as being present.

2. Like NegEx, the rule-based assertion classifier maintains a list of conjunctions that limit the scope of the dictionary patterns. For example, in the sentence: "He has no headache but has fever", the word `but` limits the scope of the prenegation phrase `no` to `headache`; thus, `headache` is absent, while `fever` is present.

3. The rule-based classifier maintains a separate dictionary of preceding negation and uncertainty phrases whose scope can extend over long lists of problems. For example, the word `denies` negates six problems in the sentence "Patient denies headache, fever, stomach cramps, shortness of breath, pain, and nausea". For these special phrases, the four word window constraint is lifted.

Class	**Precision**	**Recall**	**F-measure**
Present	0.981	0.986	0.984
Absent	0.975	0.967	97.10
Uncertain	0.92	0.894	0.907
Alter	1.00	1.00	1.00

Table 4.12: Assertion classification results for rule-based assertion classifier.

4. The rule-based classifier handles the preceding negation phrase `no` as a special case. Consider the following examples:

    - "No headache, pain, fever."
    - "Headache, no pain, fever."

    In the first example, all three *symptoms* are absent. In the second example, `headache` and `fever` are present, but `pain` is absent. The rule-based assertion classifier incorporates regular expressions to detect these cases and labels the problems involved accordingly.

Evaluation

We ran the rule-based assertion classifier on the 5,166 line corpus. The results are shown in Table 4.12.

Discussion

The results show that the rule-based assertion classifier significantly outperforms the statistical assertion classifier, confirming our hypothesis that, in our corpus, the assertion type of a medical problem is characterized by easily recognizable local context. The rule-based assertion classifier corrects many of the errors that the statistical assertion classifier makes. For example:

- `angina` is correctly labelled as being uncertain in the phrase `presumed angina`. `Presumed` is one of our preceding uncertainty phrases.
- `shortness of breath` is identified as being absent in the sentence "No history of complaint of chest pain, shortness of breath", since `no history of` is one of our preceding negation phrases.

- Our `no`-rules facilitate the correct recognition of `swelling` as being present in "No JVP, 2+ swelling, no pain".

The rule-based assertion classifier misclassifies medical problems surrounded by contextual clues that were not in the development corpus, such as `believe to be` preceding uncertain problems, and `in the event of` also preceding uncertain problems.

From our results, we conclude that, given a small annotated data set, a system based on contextual phrases manually extracted from the corpus outperforms a machine learning method. With larger corpora, it is likely that machine learning becomes a more competitive approach: unfortunately, we did not have sufficient annotated data to test this claim.

Summary

In this section, we have described a rule-based method based on NegEx for detecting the assertion classes of *diseases* and *symptoms* in medical discharge summaries. We have shown that this scheme outperforms a statistical approach.

Chapter 5

Semantic Relationship Classification

5.1 Motivation

Having identified the main concepts in a sentence (where a concept is a word or a phrase of words belonging to the same semantic category), the next step is to recognize the relationships that can exist between these concepts. For example, the following relationships can occur between *treatments* and *diseases*: *treatment* cures *disease*, *treatment* prevents *disease*, and *disease* is a side effect of *treatment*.

In this chapter, we define binary relationships that are of interest for semantic relationship classification in this thesis. We highlight the data annotation process, describe a simple baseline solution, and present the statistical semantic relationship (SR) recognizer.

5.2 Relationship Definitions

In chapter 4, we defined 8 semantic categories: *diseases*, *symptoms*, *substances*, *practitioners*, *dosages*, *tests*, *results*, and *treatments*. Considering relationships between each pair of categories yields 28 relationship types, e.g., *disease–test*. Within each relationship type, there are possibly multiple relationships, such as *test* reveals *disease*, and *test* conducted to investigate *disease*. Due to time constraints and the small size of our corpus, we decided not to address all 28 relationship types. Instead, we focused on relationships involving the patient's medical problems, that is the *diseases* and *symptoms* that the patient has or might have. This decision was motivated by Weed's proposed Problem-Oriented Medical Record [46], which is described as a "a formal summary of the patient's illnesses" [31] and "a tool for organizing the routine documentation of the physician's decision-making process and the plan for and results of care" [31]. Weed argues that organizing medical records according to the patient's medical problems addresses the current lack of consistent structure and content in clinical text.

With Weed's idea in mind, we defined our semantic relationships so that they provide enough information to automatically construct a problem-oriented record from a discharge summary. In this section, we go over the relationship definitions providing examples of each in context, and then show how the relationships can be used to extract a problem list.

5.2.1 Disease and Symptom Relationships

Our relationships involve problems that are present or uncertain. Recall that in Chapter 4, we described an assertion classifier that can determine whether a problem is uncertain, present, absent, or associated with someone other than the patient. Absent problems and problems associated with someone other than the patient are not included in a list of patient problems, so we ignore them in our definitions.

Present Disease and Treatment Relationships

We defined the following relationships (and corresponding abbreviations) between a *disease* that is asserted as being present and a *treatment* in the same sentence:

1. Treatment administered for disease (TADP): The *treatment* is given to the patient to treat the *disease*, but the outcome of the *treatment* is not specified in the sentence, e.g., "[Solu-Medrol]$_{treat}$ was given for [tracheal edema]$_{dis}$".

2. Treatment causes disease (TCDP): The *treatment* given to the patient causes the specified *disease*, e.g., "The patient experienced [femoral neuropathy]$_{dis}$ secondary to [radiation therapy]$_{treat}$".

3. Treatment cures disease (TXDP): The *treatment* cures the *disease*, e.g., "The patient had resolution of her [acute renal failure]$_{dis}$, with [hydration]$_{treat}$ only".

4. Treatment does not cure/worsens disease (TNDP): The *treatment* given to the patient for the *disease* actually exacerbates or does not cure the *disease*, e.g., "The patient had also been treated for a [pneumonia]$_{dis}$ that was believed to have progressed despite treatment with [azithromycin]$_{treat}$".

5. Treatment discontinued in response to disease (TDDP): The *treatment* is discontinued (or not even started) because of the *disease* that the patient has, e.g., "[ACE inhibitor]$_{treat}$ was held because of [renal failure]$_{dis}$".

Uncertain Disease and Treatment Relationships

We defined the following relationships between a *disease* that is asserted as being uncertain and a *treatment* in the same sentence:

1. Treatment administered for uncertain disease (TAD): The *treatment* is given to the patient to treat or prevent the uncertain *disease*, e.g., "The patient had been started on [Levaquin]$_{treat}$ for her apparent [pneumonia]$_{dis}$".

2. Treatment causes uncertain disease (TCD): The *treatment* given to the patient might cause the uncertain *disease*, e.g., "The patient was started on [Dilantin]$_{treat}$ leading to presumed [Stevens-Johnson syndrome]$_{dis}$".

3. Treatment discontinued in response to uncertain disease (TDD): The *treatment* is discontinued (or not even started) because of the *disease* that the patient might have, e.g., "At the time of this admission, the patient's [Dilantin]$_{treat}$ was discontinued, given the patient's risk of [seizures]$_{dis}$".

Present Symptoms and Treatment Relationships

We defined the following relationships between a *symptom* that is asserted as being present and a *treatment* in the same sentence. We only give examples here as the definitions are similar to the *disease* relationships.

1. Treatment administered for symptoms (TASP): e.g.,"She has a history of [headaches]$_{symptom}$ for which she receives [Darvon]$_{treat}$".

2. Treatment cures Symptoms (TXSP): e.g., "Her [hypotension]$_{symptom}$ responded to [fluid boluses]$_{treat}$".

3. Treatment does not cure/worsens symptoms (TNSP): e.g., "The patient was started on [Diltiazem]$_{treat}$, but the [pains]$_{symptom}$ continued to worsen".

4. Treatment causes symptoms (TCSP): e.g.,"This was followed by [nausea]$_{symptom}$ after taking an [iron supplement]$_{treat}$".

5. Treatment discontinued for symptoms (TDSP): e.g.,"The patient was noted to have [bloody stools]$_{symptom}$ and her [heparin]$_{treat}$ was discontinued".

Uncertain Symptoms and Treatment Relationships

We defined the following relationships between an uncertain *symptom* and a *treatment* in the same sentence:

1. Treatment administered for uncertain symptoms (TAS): e.g., "[Tylenol]$_{treat}$ every four hours p.r.n. for [pain]$_{symptom}$".

2. Treatment causes uncertain symptoms (TCS): e.g., "[Tylenol]$_{treat}$ causes [rash]$_{symptoms}$ in the patient".

3. Treatment discontinued in response to uncertain symptoms (TDS): e.g., "Her [antihypertensive medications]$_{treat}$ were held because of the risk for [hypotension]$_{symptom}$" .

Disease and Test

We defined the following relationships between a *disease* that is uncertain or present, and a *test* in the same sentence.

1. Test reveals disease (TRD): The *test* reveals the presence of the *disease*: e.g., "A [CT scan]$_{test}$ showed an [aneurysm]$_{dis}$".

2. Test conducted to investigate disease (TID): The *test* is conducted to gather more information about the *disease*, or check if the *disease* is in fact present: e.g., "A [workup]$_{test}$ was done to evaluate a [thyroid nodule]$_{dis}$".

Disease and Symptoms

We defined the following relationships between an uncertain or present *disease*, and an uncertain or present *symptom* in the same sentence.

1. Disease causes symptoms (DCS): The *symptom* is caused by the *disease*. e.g., "Her [high blood pressure]$_{symptom}$ might be related to her [pulmonary hypertension]$_{dis}$".

2. Symptoms suggest presence of disease (SSD): The *symptoms* suggests the presence of the *disease*, e.g., "Given patient's [respiratory distress]$_{symptom}$, diagnosis includes [pneumonia]$_{dis}$".

5.2.2 Building a problem list

Given a discharge summary in which all the defined semantic relationships have been annotated, creating a problem list is relatively simple. First, we extract all the present and uncertain *symptoms* and *diseases*—these are the patient's problems. For each problem, we use semantic relationships involving that problem to fill in the semantic frame shown in Figure 5-1.

Given a target problem, the **Problem** and **Certainty** fields correspond to the target problem phrase itself and its assertion class respectively. The **Treatments** field lists *treatments* administered for the target problem and whether the outcome of the *treatments* was positive (successful), negative, or unknown. Information for the **Treatments** field is extracted from the *treatments* relationships involving the target problem. The **Causes** field lists causes of the target: that is concepts related to the target by TCS, TCSP, TCDP, TCD, or DCS relationships. The **Associated Problems** section refers to other problems caused by the target problem. This field is only applicable if the target is a *disease* and consists of *symptoms* that the target is related to by the Disease causes symptoms relationships. Finally, the **Proof** field also applies to a target *disease* and lists *tests* and *symptoms* that were used in obtaining a diagnosis of the *disease*. This information is extracted from the Test Reveals Disease and Symptoms Suggest Disease relationships.

We have not written the code to create these semantic frames, but will do so as future work (see Chapter 6).

Problem: pneumonia

Certainty: present

Treatments:

> **Treatment:** Antibiotic
> **Successful:** Yes

Causes: None

Associated Problems: respiratory distress

Proof: Chest x-ray

Figure 5-1: Semantic frames for problems.

5.3 Annotation

We obtained a corpus of 11,619 medical discharge summary sentences, which had been previously annotated to identify semantic categories and problem assertions. Next, we filtered the text, retaining only those sentences containing pairs of concepts encompassed by our relationships: *symptoms* and *treatments*, *symptoms* and *diseases*, *diseases* and *treatments*, and *diseases* and *tests*.[1]

We developed a GUI for semantic relationship classification (see Figure 5-2). The interface allows an annotator to navigate through the text sentence by sentence using the "<<" and ">>"buttons. For each sentence, the GUI extracts all possible candidate pairs of concepts for relationship classification. For each candidate pair extracted, the interface highlights the concepts in the text pane and presents the user with a drop-down menu of possible relationships that can exist between the concepts. The contents of the menu vary depending on the semantic categories of the candidate concepts. For example, if the candidate pair consists of a *disease* and *test*, then the possible relationships are Test reveals disease, Test conducted to investigate disease, and None, where None indicates that no relationship exists between the concepts. When a user finishes labelling all possible relationships in a sentence, the GUI creates and stores a marked-up version of the input sentence, which can later be written to an output file by clicking the save button.

[1]We excluded absent problems and problems associated with someone other than the patient.

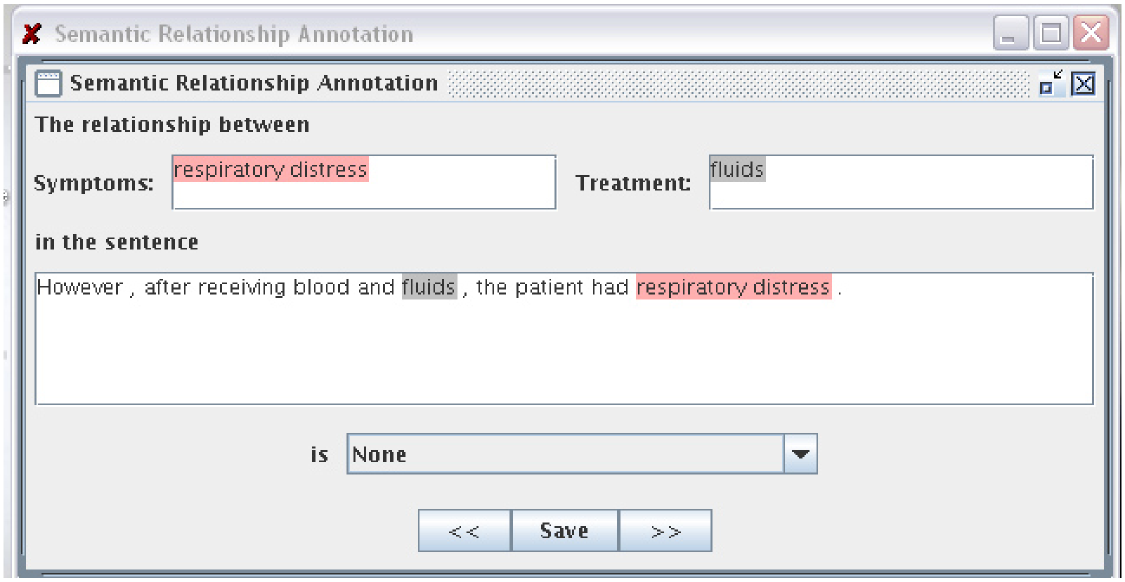

Figure 5-2: Semantic category relationship GUI.

The author annotated the 11,619 line corpus. After annotation, the filtered text consisted of 822 lines. Table 5.1 shows the breakdown of relationships in the final corpus.

5.4 Baseline for Semantic Relationship Classification

We created a simple baseline scheme that, given a candidate pair of concepts, first determines the relationship type implied by the concepts (e.g., *diseases–tests* or *diseases–symptoms*). It then selects, as its prediction, the most common relationship for that relationship type. Hence, for every *diseases–tests* pair, the baseline solution always selects the Test reveals disease relationship.

We evaluated our baseline scheme by computing precision, recall, and F-measures for the recognition of each of the 26 relationships present in our corpus (see Table 5.1). To facilitate comparison with other models, we computed micro-averaged and macro-averaged F-measures for each relationship type, and micro-averaged and macro-averaged F-measures over all relationships.

The macro-averaged F-measure gives equal weight to each relationship, and corresponds to the traditional arithmetic mean. To compute the macro-averaged F-measure, we found the sum of all input F-measures and divided the result by the number of summands. This measure captures how well the classifier performs for all classes, including classes with very

Relationship	Relationship type	Count
none	Present diseases–treatments	342
TADP		259
TXDP		32
TNDP		16
TCDP		32
TDDP		24
none	Uncertain diseases–treatments	49
TAD		30
TCS		10
TDD		8
none	Present symptoms–treatments	132
TASP		62
TXSP		36
TNSP		13
TCSP		14
TDSP		9
none	Uncertain symptoms–treatments	124
TAS		41
TCS		9
TDS		2
none	Diseases–tests	402
TRD		426
TID		58
none	Diseases–symptoms	294
SSD		12
DCS		35

Table 5.1: Breakdown of semantic relationships in corpus.

few instances.

On the other hand, the micro-averaged F-measure takes into account the relative number of instances in each class and computes a weighted mean. To compute the micro-averaged F-measure over a set of classes, we formed an aggregate confusion matrix by adding up the individual confusion matrices from each class and computed the F-measure for this aggregate matrix. The micro-averaged F-measure takes into account the size of classes: overall performance is dominated by the performance of the larger classes.

Table 5.2 shows the performance of the baseline in terms of precision, recall, and F-measures for individual relationships, and micro-averaged and micro-averaged F-measures for each relationship type, and the overall task.

5.5 The Statistical Semantic Relationship Recognizer

Our statistical semantic relationship (SR) recognizer uses SVMs to learn the context characterizing binary relationships. We train six different multi-class support vector machines, one for each of the six relationship types in our corpus. Each SVM is specialized in recognizing the relationships existing between specific semantic categories (for example, the *disease–test* SVM distinguishes between Test reveals disease, Test conducted to investigate disease, and None relationships.)

Given a sentence in which the semantic categories and assertions have been marked (as in Chapter 4), we extract all candidate pairs of concepts whose semantic categories are encompassed by our relationship types. A valid candidate pair, for example, would be a *disease* and a *test*. An illegal pair would be a *disease* and a *result*. To classify the relationship between a candidate pair, we use local features for each concept in conjunction with the SVM specializing in the candidate pair's relationship type. In particular, our features include contextual cues, such as lexical and syntactic bigrams for each concept. We also determine the syntactic path between the two concepts in the form of dependencies from the Link Grammar Parser. Our features can be divided into two sets: lexical features (that focus on the lexical units in the text), and syntactically-informed features that use syntactic information (such as parts of speech and parse trees).

Relationship	Type	Precision	Recall	F-measure
None	Present diseases–treatments	48.51%	100%	65.33%
TADP		0%	0%	0%
TXDP		0%	0%	0%
TNDP		0%	0%	0%
TCDP		0%	0%	0%
TDDP		0%	0%	0%
Macro-average				10.89%
Micro-average				48.51%
None	Uncertain diseases–treatments	50.52%	100%	67.12%
TAD		0%	0%	0%
TCS		0%	0%	0%
TDD		0%	0%	0%
Macro-average				16.78%
Micro-average				50.52%
None	Present symptoms–treatments	49.62%	100%	66.33%
TASP		0%	0%	0%
TXSP		0%	0%	0%
TNSP		0%	0%	0%
TCSP		0%	0%	0%
TDSP		0%	0%	0%
Macro-average				11.06%
Micro-average				49.62%
None	Uncertain symptoms–treatments	70.45%	100%	82.67%
TAS		0%	0%	0%
TCS		0%	0%	0%
TDS		0%	0%	0%
Macro-average				20.67%
Micro-average				70.45%
None	Diseases–tests	0%	0%	0%
TRD		48.08%	100%	64.94%
TID		0%	0%	0%
Macro-average				21.64%
Micro-average				48.08%
None	Diseases–symptoms	86.22%	100%	92.60%
SSD		0%	0%	0%
DCS		0%	0%	0%
Macro-average				30.87%
Micro-average				86.22%
Overall Macro-average				**16.38%**
Overall Micro-average				**55.32%**

Table 5.2: Performance of baseline.

5.5.1 Lexical Features

Lexical features are characteristics of the text that are extracted with no syntactic knowledge of the sentence. Our lexical features include:

- The head of the candidate concept phrases. We assume that the head is the last word in each phrase: the head of `pulmonary disease` is `disease`. This feature allows the model to learn from past relationships between concepts. Certain concepts frequently appear in the same relationship throughout the corpus: for example, `tylenol` and `pain` are often related by the Treatment administered for symptom relationship.

- Whether or not the candidate concepts are reversed with respect to the default ordering (i.e.,*diseases* and *symptoms* before *treatments* and *tests*, and *diseases* before *symptoms*). The relative order of the concepts is indicative of the relationship between them: for example, if a *disease* precedes a *test*, it is likely that there is no relationship between the two concepts.

- The left and right lexical trigrams of the candidate concepts. In certain examples, knowing the semantic category of the concepts and their immediate lexical context is sufficient to determine the type of relationship between them. For example, in the sentence "[Tylenol] was given for his [headache]", knowing that the trigram `was given for` follows the *treatment* `Tylenol`, and that the trigram `given for his` precedes the *symptom* `headache`, suggests that the relationship here is Treatment administered for symptoms. As in chapter 4, before we extract the trigrams, we first replace each concept in the text (besides the candidate concepts) with a one word place holder, signifying its semantic category.

- The inter-concept words, i.e., the words between the two concepts (including punctuation). For example, in the sentence "[Tylenol] was given for his [headache]", the value of this feature is the list (`was`, `given`, `for`, `his`). We hypothesize that certain words are indicative of specific relationships occurring between entities. For example, the word `reveal` occurring between a *disease* and a *test* implies that the inter-concept relationship is Test reveals disease. Punctuation often suggests unrelatedness as in: "[Tylenol] was given for pain: his [headache] has not been treated".

- The number of words between the two concepts. In general, the more separated the candidate concepts in the sentence, the less likely they are related.

- Whether or not there are other semantic categories between the two concepts. Other concepts splitting the candidate pair often imply that the candidate concepts are unrelated, e.g., "[Tylenol] was given for his pain, aspirin for his [headache]".

5.5.2 Syntactically-informed Features

Syntactically-informed features (later referred to as syntactic features) are properties of the text that are extracted using some knowledge of the syntactic structure of the sentence. This knowledge can be as elementary as the part of speech of the words in the text, or as complex as the sentence's entire parse tree. Our syntactic features include:

- The left and right syntactic bigrams of the candidate concepts, extracted as described in chapter 2.[2] As for lexical trigrams, we abstract over lexical variations by replacing concepts with their semantic category, and removing syntactic links running between words in the same concept (see Chapter 4). We hypothesize that knowing the immediate syntactic context of the candidate concepts is informative in deciding the relationship existing between them. The advantage of syntactic context over lexical context is that syntactic context captures the connection between words, even if they are separated by relative clauses.

- The verbs occurring between the candidate concepts. Verbs are often the strongest indicators of the type of relationship between concepts. For example, the verb `cures` between a *treatment* and a *disease* suggests the Treatment cures disease or Treatment does not cure (worsens) disease relationships.

- Up to two verbs occurring before and after the candidate concepts. As we previously stated, verbs are strong determinants of relationships. However, sometimes the most informative verbs do not occur between the candidate concepts. In the sentence fragment, "[Levaquin] for [right lower lobe pneumonia], which has resolved", the verb `resolved` suggests that the relationship present in the sentence is Treatment cures disease, but does not occur between the candidate concepts.

[2] We augmented the Link Grammar Parser with Szolovits' medical lexicon [43].

- The syntactic path between the candidate concepts. We extract this feature by parsing the sentence with the Link Grammar Parser, and following the path of links from the head of one candidate concept to the head of the other concept. For example, the sentence:

  "[Chest x-ray] on Monday revealed [pneumonia]."

  yields the following parse:

```
    +-------------------------Xp-------------------------+
    +-------Wd------+---------S--------+                 |
    |        +--AN--+-Mp-+-ON-+        +----Os----+      |
    |        |      |    |    |        |          |      |
LEFT-WALL chest.n x-ray on Monday revealed.v pneumonia.n .
```

  The path of links from the head of `Chest x-ray` to the head of `pneumonia` is *S-Os*. If there is no link path between the concepts, then this feature has value *None*.

  Ding et al. [17] showed that it is possible to determine whether two biochemicals are related by checking for a link path connecting them. We take this idea a step further by hypothesizing that the link path itself is indicative of the type of relationship between concepts. For example, an *S-Os* path between a *disease* and a *symptom*, where the *disease* is the subject, is suggestive of the Disease causes Symptom relationship.

- The link path words: the words encountered on the syntactic path between the candidate concepts (including punctuation). We predict that using just the link path between entities is susceptible to noise. For example, an *S-Os* path from a *treatment* to a *disease* does not really tell us whether the *treatment* causes the *disease*, cures the *disease*, or worsens the *disease*. Thus, we also use the actual words encountered on the path as features. In our example sentence "[Chest x-ray] on Monday revealed [pneumonia]" the word encountered on the link path between the concepts is `revealed`. In our lexical feature set, we consider all the words between the concepts: in this case `on`, `Monday`, and `revealed`. We hypothesize that by only including words on the syntactic paths between concepts, we avoid spurious words, such as prepositions, adjectives, and other modifiers that do not contribute to the semantic relationship in the sentence. If there is no syntactic path between the two concepts, we fall back to the corresponding lexical feature and use all the words between the concepts.

5.5.3 Evaluation

We evaluated the statistical SR recognizer by running it on our corpus of 822 lines, and used 10-fold cross validation to classify the relationship between each pair of candidate concepts encompassed by our relationship types. We computed F-measures for the following feature sets:

- Complete feature set (all the features).
- Lexical features.
- Syntactically-informed features.

5.5.4 Discussion

Table 5.3 shows the F-measures obtained by the statistical SR recognizer using the complete feature set. The statistical SR recognizer significantly outperforms the simple baseline of picking the most common relationships, scoring micro and macro F-measures of 84.54% and 66.70% respectively, compared to 55.32% and 16.38%.

The statistical SR recognizer is particularly strong in recognizing *diseases–tests* and *diseases–symptoms* relationships. For *diseases–tests* relationships, the micro and macro F-measures are 84.79% and 89.73% respectively. We hypothesize that the strong performance for this relation type is attributable to the fact that there are only 3 classes of *diseases–tests* relationships, and two of the three classes have a comparatively large number of instances in the corpus. Furthermore, *diseases–tests* interactions are characterized by repeating lexical context. For example, the verb `show`, occurring between a *disease* and a *test*, is almost always indicative of the Test reveals disease relationship. Similar reasoning explains the strong performance in the *diseases–symptoms* relationship. The phrase secondary to between a *disease* and a *symptom* strongly suggests the relationship Disease causes symptom.

The statistical SR recognizer performs poorly in the *uncertain diseases–treatments* relationships and the *uncertain symptoms–treatments* relationships. These relationship types have the fewest instances in the corpus. There are only 70 *uncertain diseases–treatments* candidate pairs and 176 *uncertain symptoms–treatments* candidate pairs in the text, compared to an average of 407 pairs for all relationship types.

Table 5.4 shows that most of the statistical SR recognizer's gains come from lexical features (the micro and macro F-measures for syntactic features are worse than the lexical values, though not significantly). This result is encouraging as it suggests that one can effectively classify inter-concept binary relationships in medical discharge summaries by using simple lexical properties of the concepts and the surrounding text. However, the result is also discouraging: we have shown the importance of syntax for the tasks of de-identification and semantic category recognition, but it appears that syntax plays a more limited role than lexical features in the harder task of semantic relationship classification.

To explore this discrepancy further, we determined the most informative lexical features and the most informative syntactic features by running the statistical SR recognizer with each feature separately. Table 5.5 shows the macro and micro F-measures obtained. The inter-concept words are the most important lexical features, while the link path words are the most important syntactic features.

The table also shows that the inter-concept words are more informative than the link path words.[3] The latter is a surprising result as we initially hypothesized that using the syntactic path would remove spurious words occurring between the concepts. Furthermore, we expected the performance of the link-path words to be at least as good as the inter-concept words, because the link path words fall back to the inter-concept words in the event of no path existing between the concepts.

The discrepancy arises because the Link Grammar Parser makes mistakes on the corpus. There are 2,471 candidate pairs in the corpus. Of these, only 34.72% (858) have link paths between them. Of these, 858 pairs with link paths, 472 of the paths, 55.01%, contain null links. Recall from chapter 2 that the Link Grammar Parser first attempts to parse the text so that every word has at least one link to another word in the sentence (the completeness criterion). If no such parse can be found, then the parser ignores some words producing a partial parse. Often, these partial parses are erroneous as in the following example. Given the sentence:

"Chest x-ray showed a right lower lobe infiltrate and enlarged cardiac silhouette with congestive heart failure."

[3] The differences in micro F-measure values are significant while the macro F-measure differences are not.

the Link Grammar Parser output is:

```
           +--------------------------------------------MVp--------------------------------------------+
           +----------------------------------------Os---------------------------------------+         |
           |             +---------------------------------Ds--------------------------------+         |
           +----------------Os----------------+                                              |         |
           |             +---------Ds---------+                                              |         +------------------Jp------+
           |             |    +-------A-------+                      +-----------------A-----+         |                          |
           +             |    |       +---Am--+                      |          +---------A--+         |      +-IDEOID+IDEOIC+    |
           |             |    |       |       |                      |          |            |         |      |       |      |    |
[Chest] x-ray.v [showed] a right.a lower.a lobe.n [infiltrate] and enlarged.v cardiac.a silhouette.n with  congestive heart failure .
```

Notice that the words `chest`, `showed`, and `infiltrate` have no left or right links. The parse is partial and clearly incorrect: for example, the word `x-ray` is interpreted as a verb with direct object equal to the word `lobe`. Consequently, the statistical SR recognizer trained on the link path words misses the relationship between `chest x-ray` and `right lower lobe infiltrate` because of the parsing errors. On the other hand, the statistical SR recognizer, when trained on inter-concept words, recognizes the trigger word `shows`, and correctly classifies the relationship between the candidate concepts as an instance of Test reveals disease.

To truly determine the importance of syntax, we compared the performance of inter-concept words with link path words only for those pairs that have a complete link path between them. Table 5.6 shows that in this case the link path words perform similarly to the inter-concept words.[4] For the concepts with a complete link path between them, we realize the benefits of the link path features. For example, the link path words correctly recognize the Treatment discontinued in response to disease relationship between `declining renal function` and `ACE inhibitor` in the sentence "The patient's Lasix and ACE inhibitor were held initially due to her declining renal function". The inter-concept words miss this relationship. In this case the inter-concept words are `were`, `held`, `initially`, `due`, `to`, and `her`. The link path words are `were`, `held`, `due`, and `to`. The link path words do not include the spurious words `initially` and `her`. As we hypothesized a priori, using the syntactic path boosts performance in this case, because it removes words that do not directly determine the semantic relationship in the sentence.

From our experiments, we conclude that the importance of syntactic information is limited by the poor performance of the Link Grammar Parser on the corpus. The Link Grammar Parser is unable to parse many of the ad hoc sentences in the text, because some

[4] The F-measures for the link path words are higher than those for the inter-concept words, but the changes are not significant.

of the words in the corpus are not located in its lexicon and many of the sentences are not grammatical. For de-identification and semantic category recognition, we used local syntactic links. Even if sentences were incorrectly or only partially parsed, local links around most words were correct and hence syntactic information was still informative in these cases. However, for the task of semantic relationship classification, we are using long distance links (in fact entire paths of links). This is a global property of the sentence and is more susceptible to errors in the parser. We hypothesize that in grammatical text, link path words would be more informative than they are in discharge summaries. We plan to test this hypothesis in the future.

Conclusion

In this section, we have described a statistical SR recognizer for classifying semantic relationships involving *diseases* and *symptoms*. The statistical SR recognizer significantly outperformed a simple baseline that chooses the most common relationship in each case. We used lexical and syntactically informed feature sets and showed that the lexical features (in particular, the words occurring between candidate concepts) were the most informative for relationship classification in medical discharge summaries. We hypothesize that in more grammatical text, syntactic features become increasingly more important.

Relationship	Type	Precision	Recall	F-measure
None	Present diseases–treatment	79.83%	83.33%	81.55%
TADP		69.86%	78.76%	74.05%
TXDP		93.75%	46.88%	62.50%
TNDP		80%	50%	61.54%
TCDP		84.21%	50%	62.75%
TDDP		63.64%	29.17%	40%
Macro-average				63.73%
Micro-average				75.89%
None	Uncertain diseases–treatments	62.96%	69.39%	66.02%
TAD		70.00%	70.00%	70.00%
TCS		50%	40%	44.44%
TDD		20%	12.5%	15.38%
Macro-average				48.96%
Micro-average				61.86%
None	Present symptoms–treatments	83.80%	90.15%	86.86%
TASP		69.12%	75.81%	72.31%
TXSP		90.32%	77.78%	83.58%
TNSP		70.00%	53.85%	60.87%
TCSP		66.67%	42.86%	52.17%
TDSP		66.67%	44.44%	53.33%
Macro-average				68.19%
Micro-average				79.32%
None	Uncertain symptoms–treatments	93.75%	96.77%	95.24%
TAS		92.31%	87.80%	90.00%
TCS		57.14%	44.44%	50.00%
TDS		0%	0%	0%
Macro-average				58.81%
Micro-average				90.91%
None	Diseases–tests	90.43%	89.30%	89.86%
TRD		90.62%	92.96%	91.77%
TID		76.92%	68.97%	72.72%
Macro-average				84.79%
Micro-average				89.73%
None	Diseases–symptoms	96.68%	98.98%	97.82%
SSD		87.5%	58.33%	70%
DCS		93.75%	85.71%	89.55%
Macro-average				85.79%
Micro-average				96.19%
Overall Macro-average				**66.70%**
Overall Micro-average				**84.54%**

Table 5.3: Performance of the statistical SR recognizer with all features.

Relationship Type	Feature	Micro F	Macro F
Present diseases–treatments	Lexical	75.74%	64.21%
Uncertain diseases–treatments		65.98%	49.56%
Present symptoms–treatments		80.45%	68.62%
Uncertain symptoms–treatments		90.91%	60.86%
Diseases–tests		88.03%	84.02%
Diseases–symptoms		96.19%	85.79%
OVERALL		**84.46%**	**67.23%**
Present diseases–treatments	Syntactic	73.62%	64.26%
Uncertain diseases–treatments		59.79%	42.98%
Present symptoms–treatments		77.82%	66.50%
Uncertain symptoms–treatments		90.91%	58.59%
Diseases–tests		88.15%	82.62%
Diseases–symptoms		96.19%	85.68%
OVERALL		**83.08%**	**65.22%**

Table 5.4: Comparison of the performance of the statistical SR recognizer with lexical vs. syntactic features.

Features	Feature Set	Overall Micro F	Overall Macro F
Lexical bigrams	Lexical	78.15%	59.20%
Inter-concept words		**83.53%**	**65.35%**
Head words		72.56%	45.95%
Surface features		66.82%	27.23%
Syntactic bigrams	Syntactic	77.42%	57.31%
Link path words		**80.00%**	**60.65%**
Link path		63.09%	25.98%
Verbs		75.11%	53.16%

Table 5.5: Performance of statistical SR recognizer using different feature sets.

Feature	Overall Micro F	Overall Macro F
Inter-concept words	75.68%	45.27%
Link path words	**76.22%**	**46.44%**

Table 5.6: Performance of the statistical SR recognizer using inter-concept words and link path words on sentences with complete linkages.

Chapter 6

Future Work: Category and Relationship Extractor (CaRE)

In this chapter, we present CaRE: a system for the semantic interpretation of medical discharge summaries. This system integrates the various models we have presented in this thesis into a single application that allows a user to quickly extract important information from input text.[1]

6.1 Design Overview

CaRE consists of four major components: the preprocessor, the predictor, the extractor, and the graphical user interface (see Figure 6-1).

6.1.1 Preprocessor

The input file first passes through the preprocessor. This component uses five tools to extract initial information for use by later components.

The *Tokenizer* tokenizes and segments the text so that each new sentence starts on a new line and words are separated from punctuation. The tokenized output is fed to the *Parser* and the *Tagger*. The former parses the text using the Link Grammar Parser and produces a file specifying the left and right connections of each word in the input sentence. The *Tagger* [7] generates a file containing the original text with each word attached to a

[1] At the time of writing, CaRE is only partially completed. Implementation of the system should be completed by the Fall of 2006.

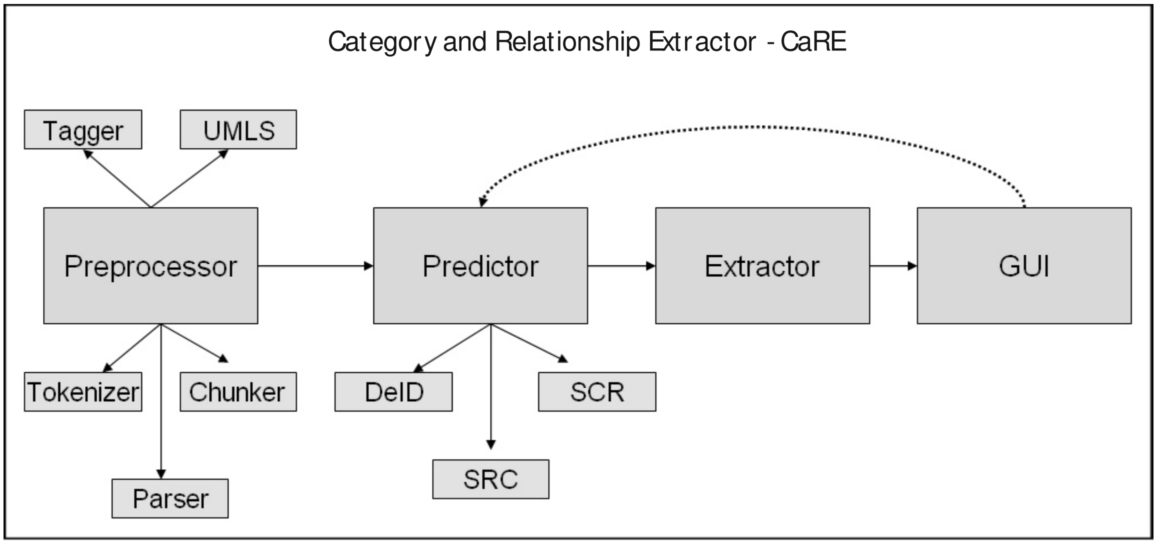

Figure 6-1: Design overview of CaRE.

tag indicating its part of speech. This tagged output is used by the *Chunker* [35] to identify and mark noun phrases in the text. The *Chunker* generates a file in which noun phrases are surrounded by square brackets.

Finally, the noun-phrase-chunked file is passed to the UMLS component that exhaustively searches each noun phrase for MeSH IDs and UMLS semantic types. This component generates a UMLS file containing the semantic type and MeSH ID of each word in the input.

6.1.2 Predictor

The predictor consists of the three components described in Chapters 3, 4, and 5: a DeID (de-identification) component, an SCR (semantic category recognition) component, and an SRelC (semantic relationship classification) component.

DeID iterates through each word in the tokenized text, extracting lexical and syntactic features from the pre-processed files. These features are fed to the previously trained statistical de-identifier (see Chapter 3) which predicts the class of each word (the possible classes are doctor, patient, location, date, ID, phone number, hospital, and none.) The predictions are saved in a DeID predictions file in annotated format, where each single or multi-word entity is enclosed in XML tags indicating its PHI class.

SCR, like the DeID component, extracts features for each word in the text and feeds them to the statistical SC recognizer (see Chapter 4) to predict the semantic category of each word (possible classes are *disease*, *treatment*, *test*, *results*, *symptom*, *substance*, *practitioner*,

Problem: pneumonia

Certainty: present

Treatments:

  Treatment: Antibiotic
  Successful: Yes

Causes: None

Associated Problems: respiratory distress

Proof: Chest x-ray

Figure 6-2: Semantic frames for problems.

dosage, and *none*). For predicted *diseases* and *symptoms*, the SCR component uses the rule-based assertion classifier (see Chapter 4) to identify the assertion type of the concept. Finally all semantic category predictions (including assertion types) are saved in an SCR predictions file.

Finally, the SCR output is passed to the SRelC component which extracts candidate pairs of concepts encompassed by our semantic binary relationships. For each concept pair, SRelC finds lexical and syntactic features and applies the statistical SR recognizer (see Chapter 5) to predict the relationship between the concepts. The SRelC component stores its predictions in an annotated file.

6.1.3 Extractor

The extractor takes the output of SRelC and produces a problem list in XML format. To do this it first extracts all the patient problems (present and uncertain *diseases* and *symptoms*) in the file. Then, it uses a coreference engine to group together multiple references to the same problem. For each unique problem, it scans the SRelC file for relationships involving that problem. These relationships are used to populate the semantic frame indicated in Figure 6-2. Finally, the frames are converted into an XML file containing a hierarchy of attributes for each medical problem.

6.1.4 Graphical User Interface

A user interacts with the system via the GUI. When the GUI is launched, it prompts the user for an input file (which is the unprocessed, narrative text). The system preprocesses the user-selected input and passes it through the predictor and extractor components, generating the aforementioned XML-style annotated files. The GUI uses these files to populate three content windows: a de-identification window, a semantic category recognition window, and problem list window (see Figures 6-3 and 6-4).

The de-identification and semantic category windows present the tokenized text in a scroll pane, and highlight the PHI/semantic category of each word with different colors. The menu beneath the scroll pane serves as a color palette and also allows the user to change the class of a word in the text by simply highlighting that word in the text pane, selecting the correct class from the menu, and clicking the update button. For the semantic category window, changing the class of the word and clicking update forces the system to regenerate the problem list. Hence, if the user identifies a disease that was missed by the SRC component, clicking update refreshes the problem list to include this disease.

The problem list window (Figure 6-4) consists of two sections. On the left hand side, the input text is reproduced with annotations indicating the relationships evident in the sentence. Relationships are represented by labelled arrows from one concept to another. The right hand side of the window contains a list of the patient problems in a hierarchical structure reflecting the problems XML file. The user can edit the relationships in the text by selecting candidate concepts and adding/modifying relationship arrows between them. Changes are propagated to the problem list.

After changing the annotations in the text (PHI class, semantic category class, or relationship type), the user can choose to retrain the system by selecting the `Retrain` menu item. Retraining involves incorporating the user's annotations into the system's original training files and retraining the various statistical recognizers with the new data. To cut down on training time, we use incremental learning to update the SVMs used by our recognizers. Incremental learning is the process whereby an SVM updates its hyperplane to accommodate new data instances without retraining the SVM on the entire data set.

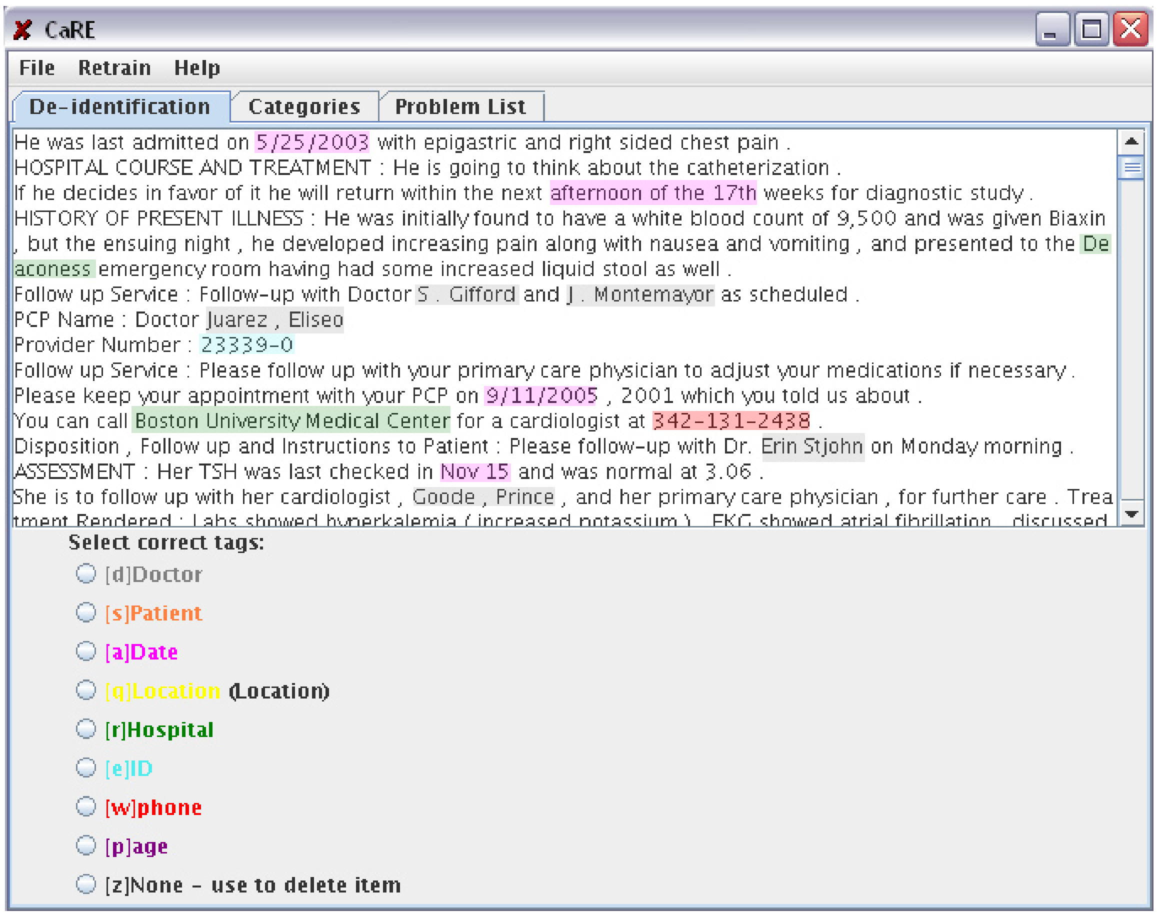

Figure 6-3: De-identification window of CaRE.

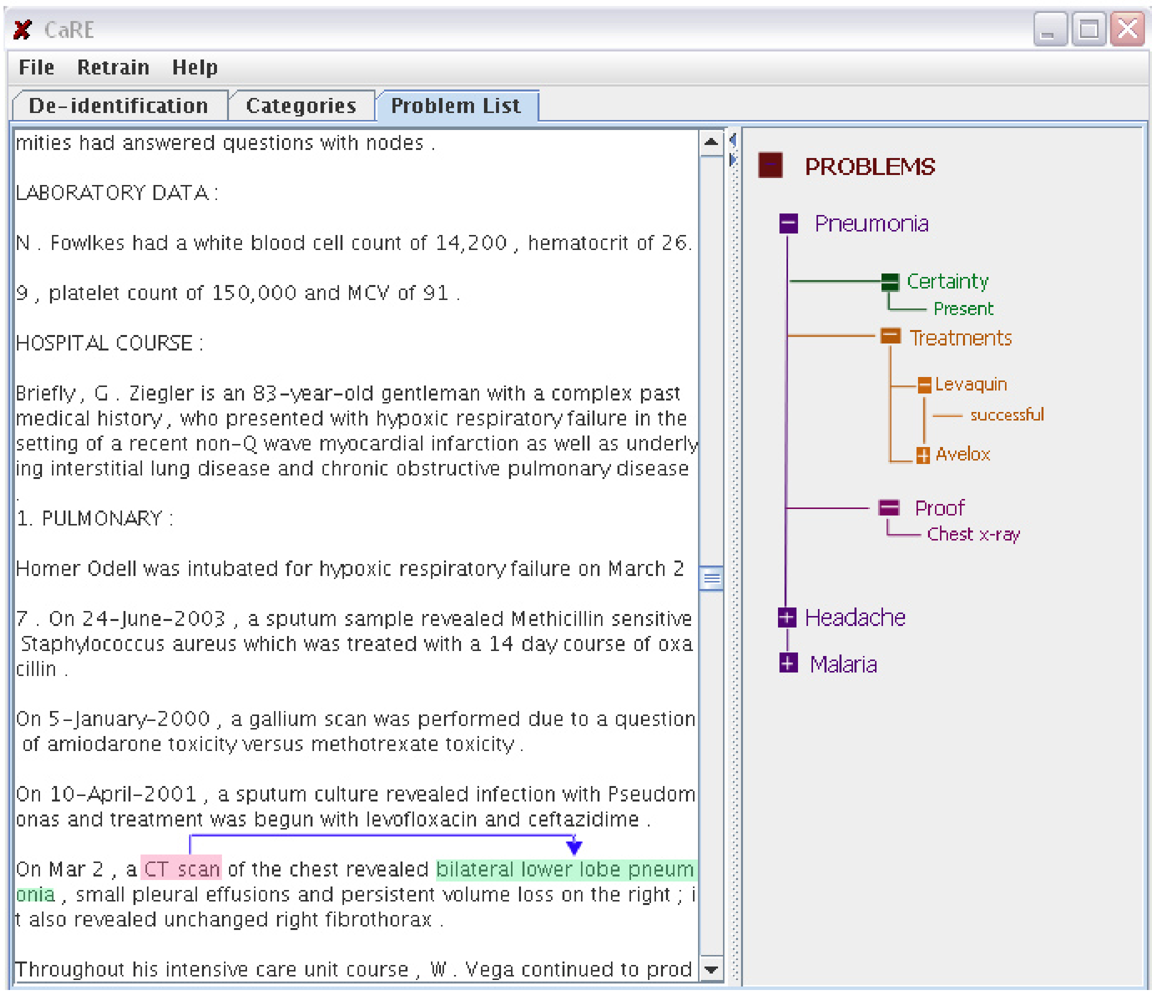

Figure 6-4: Problem list window of CaRE.

6.2 Capabilities

To summarize, CaRE has the following capabilities:

- Given input text, CaRE can generate: a de-identified version of the file with PHI replaced by anonymous tags; a version of the file in which the semantic categories and assertion classes are marked; a version of the file indicating the semantic relationships involving *diseases* and *symptoms*; and a list of the medical problems the patient has and attributes of these problems.

- CaRE can learn from its mistakes. Using the GUI, a user can review and correct the system's predictions and efficiently retrain CaRE to avoid similar mistakes in the future.

- The GUI allows the user to quickly view the problems a patient has without having to read the entire document. This feature is crucial in the emergency department setting where doctors frequently have insufficient time to read a patient's past discharge summaries.

Chapter 7

Conclusion

In this thesis, we have described statistical and rule-based approaches for de-idenfification, semantic category recognition and assertion classification, and semantic relationship classification. Our solutions rely heavily on local context and achieve competitive results compared to more complex NLP solutions. We have also developed novel methods of incorporating syntactic information from the Link Grammar Parser as a feature to SVM classifiers. We have shown that these syntactic features are useful for de-identification and semantic category recognition. However, for semantic relationship classification, the ungrammatical and ad hoc nature of the documents make it difficult to extract any useful, global syntactic information. Finally, we have presented the design of CaRE: a system that integrates our various statistical and rule-based approaches into a single application.

In the emergency room, every wasted second compromises lives: doctors often have insufficient time to review past discharge summaries, even though these records may contain crucial information. One of the overarching goals of our studies is to produce automated systems that doctors can use to rapidly query a patient's discharge summaries for critical information. In this thesis, we have taken some of the first steps in making such a vision a reality.

Bibliography

[1] The SPECIALIST NLP Tools [web page] 2006. URL: http://lexsrv3.nlm.nih.gov/SPECIALIST/index.html. [Accessed 30 April, 2006].

[2] Unified Medical Language System [web page] 2006. URL: http://www.nlm.nih.gov. [Accessed 30 April, 2006].

[3] What is Kappa [web page] 2006. URL: http://www.musc.edu/dc/icrebm/kappa.html [Accessed 30 April, 2006].

[4] A. Aronson. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program. In *AMIA*, 2001.

[5] K. Bennett and C. Campbell. Support Vector Machines: Hype or Hallelujah. *ACM SIGKDD Explorations Newsletter*, 2(2):1–13, December 2000.

[6] D. Bikel, R. Schwartz, and R. Weischedel. An Algorithm that Learns What's in a Name. *Machine Learning Journal Special Issue on Natural Language Learning*, 34(1/3):211–231, February 1999.

[7] E. Brill. A Simple Rule-Based Part of Speech Tagger. In *3rd Conference on Applied Natural Language Processing*, 1992.

[8] C. Chang and C. Lin. LIBSVM: a Library for Support Vector Machines. 2001.

[9] W. Chapman, W. Bridewell, P. Hanbury, G. Cooper, and B. Buchanan. A Simple Algorithm for Identifying Negated Findings and Diseases in Discharge Summaries. *Journal of Biomedical Informatics*, 34:301–310, 2001.

[10] N. Collier, C. Nobata, and J. Tsujii. Extracting the Names of Genes and Gene Products with a Hidden Markov Model. In *18th International Conference on Computational Linguistics*, 2000.

[11] M. Collins and Y. Singer. Unsupervised Models for Named Entity Classification. In *EMNLP*, 1999.

[12] R.G. Congalton. A Review of Assessing the Accuracy of Classifications of Remotely Sensed Data. *Remote Sensing of Environment*, 37:35–46, 1991.

[13] C. Cortes and V. Vapnik. Support-vector Networks. *Machine Learning*, pages 273–297, 1995.

[14] M. Craven. Learning to Extract Relations from MEDLINE. In *AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.

[15] A. Culotta and J. Sorensen. Dependency Tree Kernels for Relation Extraction. In *ACL*, 2004.

[16] T. Delbecque, P. Jacquemart, and P. Zweigenbaum. Indexing UMLS Semantic Types for Medical Question-Answering. *Studies in Health Technology and Informatics*, 116:805–810, 2005.

[17] J. Ding, D. Berleant, J. Zu, and A. Fulmer. Extracting Biochemical Interactions from MEDLINE Using a Link Grammar Parser. In *IEEE International Conference on Tools with Artificial Intelligence*, 2003.

[18] M. Douglass. Computer Assisted De-identification of Free Text Nursing Notes. Master's thesis, Massachusetts Institute of Technology, 2005.

[19] S. Dumais. Using SVMs for Text Categorization. *IEEE Intelligent Systems*, 13(4):21–23, 1998.

[20] P. Elkin, S. Brown, B. Bauer, C. Husser, W. Carruth, L. Bergstrom, and D. Wahner-Roedler. A Controlled Trial of Automated Classification of Negation From Clinical Notes. *BMC Medical Informatics and Decision Making*, 2005.

[21] R. Feldman, Y. Regev, M. Finkelstein-Landau, E. Hurvitz, and B. Kogan. Mining Biomedical Literature Using Information Extraction. *Current Drug Discovery*, pages 19–23, 2002.

[22] J. Finkel, S. Dingare, H. Nguyen, M. Nissim, C. Manning, and G. Sinclair. Exploiting Context for Biomedical Entity Recognition: From Syntax to the Web. In *International Joint Workshop on Natural Language Processing in Biomedicine and its Applications at CoLing*, 2004.

[23] A. Forster, H. Murff, J. Peterson, T. Gandhi, and D. Bates. The incidence and severity of adverse events affecting patients after discharge from the hospital. *Annals of Internal Medicine*, 138(3):161–167, 2003.

[24] C. Friedman, P. Alderson, J. Austin, J. Cimino, and S. Johnson. A general natural language text processor for clinical radiology. *JAMIA*, 1:161–74, March 1994.

[25] A. Van Ginneken, M. De Wilde, E. Van Mulligen, and H. Stam. Can Data Representation and Interface Demands be reconciled? Approach in ORCA. In *AMIA Symposium Supplement*, 1997.

[26] I. Goldin and W. Chapman. Learning to Detect Negation with 'not' in Medical Texts. In *26th ACM SIGIR Conference*, 2003.

[27] H. Isozaki and H. Kazawa. Efficient Support Vector Classifiers for Named Entity Recognition. In *COLING*, 2002.

[28] G. Leroy, C. Hsinchun, J. Martinez, S. Eggers, R. Falsey, K. Kislin, Z. Huang, J. Li, J. Xu, D. McDonald, and G. Nq. Genescene: Biomedical Text and Data Mining. In *ACM/IEEE-CS*, 2003.

[29] W. Long. Extracting Diagnoses from Discharge Summaries. In *AMIA*, 2005.

[30] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*, chapter 3.2.1. MIT Press, 1999.

[31] S. Meystre and P. Haug. Automation of a Problem List Using Natural Language Processing. *BMC Medical Informatics and Decision Making*, 5(30), 2005.

[32] S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. A Novel Use of Statistical Parsing to Extract Information from Text. In *6th Applied Natural Language Processing Conference*, 2000.

[33] C. Moore, J. Wisnivesky, S. Williams, and T. McGinn. Medical Errors related to Discontinuity of Care from an Inpatient to an Outpatient Setting. *Journal of General Internal Medicine*, 18(8):646–651, August 2003.

[34] T. Mudiayi, F. Onyanga-Omara, and M. Gelman. Trends of morbidity in general medicine at United Bulawayo Hostpitals, Bulawayo, Zimbabwe. *Central African Journal of Medicine*, 43:213–219, 1997.

[35] L. Ramshaw and M. Marcus. Text Chunking Using Transformation-Based Learning. In *Third ACL Workshop on Very Large Corpora*, 1995.

[36] E. Riloff and R. Jones. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. In *16th National Conference on Artificial Intelligence*, 1999.

[37] B. Rosario and M. Hearst. Classifying Semantic Relations in Bioscience Texts. In *ACL*, 2004.

[38] D. Roth and W. Yih. Probabilistic reasoning for entity and relation recognition. In *COLING*, 2002.

[39] N. Singh. The Use of Syntactic Structure in Relationship Extraction. Master's thesis, Massachusetts Institute of Technology, 2004.

[40] D. Sleator and D. Temperley. Parsing English with a Link Grammar. Technical Report CMU-CS-91-196, Carnegie Mellon University, 1991.

[41] S. Soderland, D. Aronow, D. Fisher, J. Aseltine, and W. Lehnert. Machine Learning of Text Analysis Rules for Clinical Records. *Technical Report: UMASS at Amherst – Center for Intelligent Information Retrieval*, 1995.

[42] L. Sweeney. Replacing Personally-Identifying Information in Medical Records, the Scrub System. *Journal of the American Medical Informatics Association*, pages 333–337, 1996.

[43] P. Szolovits. Adding a Medical Lexicon to an English Parser. In *AMIA*, 2003.

[44] R. Taira, A. Bui, and H. Kangarloo. Identification of Patient Name References within Medical Documents using Semantic Selectional Restrictions. In *AMIA*, 2002.

[45] K. Takeuchi and N. Collier. Use of Support Vector Machines in Extended Named Entity Recognition. In *6th Conference on Natural Language Learning*, 2002.

[46] L. Weed. Medical Records that Guide and Teach. *The New England Journal of Medicine*, 278(12), 1968.

[47] D. Zelenko, C. Aone, and A. Richardella. Kernel methods for Relation Extraction. In *EMNLP*, 2002.

[48] S. Zhao. Named Entity Recognition in Biomedical Texts using an HMM Model. In *20th International Conference on Computational Linguistics*, 2004.