FROM TEMPORAL EXPRESSIONS TO SYMPTOM ONSET DATE IDENTIFICATION IN EMERGENCY DEPARTMENT NOTES – A TEMPORAL INFORMATION EXTRACTION PROCESS

Deepika Mahalingam

A thesis submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Master of Science in the Department of Computer Science.

Chapel Hill
2011

ABSTRACT

DEEPIKA MAHALINGAM: From Temporal Expressions to Symptom Onset Date Identification in Emergency Department Notes
– A Temporal Information Extraction Process
(Under the direction of Wei Wang)

A patient's visit to the Emergency Department (ED) starts with the triage nurse making a note of the patient's account of the reason for the visit. This triage note (TN) contains symptoms the patient is suffering from, prior treatments if any, related events and sometimes the nurse's evaluation of the situation. Public health officials may use these TNs to identify features of disease outbreaks. Here we present a system that processes triage notes, producing a timeline of events leading to the ED visit and identifying patterns in occurrence of symptoms across patients. This system is designed as an initial step in the process of automatically extracting signals/symptoms defining a disease outbreak based on the details (symptoms and temporal information) associated with ED visits.

ACKNOWLEDGEMENTS

This project would not have been possible without two people: Professor Stephanie Haas, who inspired me to take up this work and my advisor Professor Wei Wang, who constantly guided me and kept me on track during the 6-7 months we have been working together. Special thanks to Professor Hye-Chung Kum for her helpful review and feedback.

My sincere thanks to Richard Medlin for helping me with manual annotation of the data in such short notice, to Ann Irvine for graciously sharing her implementation of TN-TIES with me, to Debbie Travers and Javed Mostafa and the rest of G08 group for being a constant source of motivation. I would also like to thank people at NC DETECT, especially Anna Waller and John Crouch for providing me with the data and related statistics.

This section will not be complete without acknowledging the love, understanding and constant support of my family throughout the two years at UNC-Chapel Hill. Their faith in me and the joy they expressed in hearing from me even during those phone calls I made at weird hours gave me hope and more than enough reason to keep going.

Data for this study were provided by The NC Public Health Data Group and NC DETECT. The NC Public Health Data Group and NC DETECT do not take responsibility for the scientific validity or accuracy of methodology, results, statistical analyses, or conclusions presented.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

State Public Health officials are charged with the task of studying millions of health records collected over a period of time, trying to find patterns of health and illness across population in specific geographic regions. Though many tools have been developed to help them do such analysis, often they end up looking through the records manually to identify details pertaining to the patients' health condition before their Emergency Department (ED) visit. In such situations, triage notes (TNs) act as an important source of documentation.

Epidemiologists are people who study patterns of diseases or health risks in population groups, mostly to look at the impact of these diseases on the population, for public health surveillance purposes and to track these health problems. To understand certain outbreaks, epidemiologists need to first understand how it started. In other words, they need to identify the onset of the earliest symptoms associated with patients who visit the ED with complaints of a particular health problem. Currently, they use a manual approach to identify these onset dates after identifying the ED visits of interest for their study. This research aims to build a system that can reduce the innumerable man-hours spent in analyzing hundreds of thousands of ED records for extracting such information.

A patient's Emergency Department record usually contains date/time details of the visit, the chief complaint (CC) the patient comes in with and the triage note (TN) field written by a nurse (collected as part of the initial documentation step) in addition to the physician's notes, lab tests and results, diagnoses, treatment and other related information. The CC field contains a few words (2-5 words) describing the patient's health condition while the TN field is a free-text field written by the nurse and contains natural language terms as well as medical terms giving a detailed description of the patient's health complaint. TN usually contains associated symptoms and events, along with any medication(s) taken, usually as narrated by the patient or by someone accompanying the patient. TN might also include a quick evaluation of the patient's condition by the triage nurse.

Objective

The major goal of this research is to develop a system that can identify all symptoms and events found in TNs and represent them on a timeline relevant to the visit based on domain knowledge. We then look for similar symptoms across patients and the relative time(s) at which they occurred, thus producing an aggregate timeline of distribution of particular symptoms in a population within a specific period of time before the visit. This can help exploit the wealth of patient information found in TNs without the need for extensive manual labor in making decisions as to where in the timeline each event/symptom should be placed.

In addition to automating the process of extracting, classifying and arranging the temporal information in TNs on a timeline, this research demonstrates how a simple combination of natural language processing (NLP) and supervised learning techniques with some domain knowledge can be used to extract useful information from TNs. Text like TNs pose great challenges to the application of generic natural language processing or machine learning algorithms. This is just a starting step towards building a totally automated system that can process such free text in the medical domain to better serve the healthcare community.

The importance and applications of such research are manifold. Automatic processing of important medical text makes the data more accessible and useful for healthcare professionals. Moreover, such a demonstration will motivate people interested in this area of research to push boundaries and develop new applications combining NLP and biomedical informatics. This work is a simple attempt at understanding a particular genre of text like TN better, and extracting meaningful information from it so as to develop applications that can reduce much of the human effort spent on this. It should be remembered that there is much more scope in the area of temporal analysis of clinical text than what will be discussed in this work and I describe my research in detail here so that it can serve as a basis for such future works.

CHAPTER 2

BACKGROUND

I. Triage Notes

Documentation of a patient's ED visit begins with a triage nurse recording the details of the visit. The major part of this information includes the patient's account of events that lead to the ED visit, i.e., symptoms, medications and related issues either in the patient's own words or by someone accompanying the patient (usually in the case of children or if the patient is not in a position to give the details) along with the nurse's evaluation of the situation or the nurse's observations about the patient condition. Based on this detailed information, the patient's major complaint is noted in the CC field and rest of the account goes into the TN field along with the date/time of the visit and initial vitals such as temperature and blood pressure. in separate fields, all together forming an ED record. Further interactions with physicians, laboratory tests and results, diagnosis, medications prescribed also become a part of this documentation at later stages of the ED visit.

North Carolina Disease Event Tracking and Epidemiologic Collection Tool (NC DETECT) receives these details from various institutions in North Carolina and accumulates them electronically in its database, making it available for later analyses and specific research purposes. Here, we use the electronic version of a sample of Triage Notes and the timestamp associated with ED visits made in the state between 2006 and 2008. TNs carry very rich information about the patient's condition, usually a combination of present illness, relevant past medical history, treatments and medications, which are invaluable to clinicians.

At the same time, since TN is not a structured field, extracting much needed information is not straightforward, hence the need to apply NLP and Information Retrieval techniques. From the following sample triage note,

> states that on wednesday i had a stomach virus , fever and vomiting , diarrhea on tuesday night . felt a little better on thursday , was able to keep some fluids down . states that she took some tylenol last night and started vomiting again . pt . has a drain in right thigh that is draining fluid related to injury from being struck by car , states that her leg is a little more painful , unsure if more swollen . c/o headache

we can see that there is a need to format it in a way so as to enable easy interpretation for medical professionals. Another major issue with TN is that there is no standard way of creating one, i.e., though the kind of information entered in a TN is largely similar, TNs vary from one institution to another. Hence, we need to identify a common method to mine information from TN irrespective of its source. Since ED data are time-sensitive, the best way to structure a TN would be to create a timeline of all the 'events' in it. That way, clinicians can easily assimilate the note without having to manually go through a monotonously long piece of text.

II. Symptoms

Diseases are characterized by occurrence of specific symptoms or a chain of symptoms that lead to an ED visit. Physicians and clinicians evaluate such symptoms to decide on the course of action to take to treat the patient. Public health officials are interested in finding patterns in these symptoms to study disease outbreaks in a particular geographic location given a timeframe. Hence, creating a timeline of such symptoms has multiple useful applications in healthcare.

For this research, we look at a few symptoms related to Fever Rash syndrome (Waller et al., 2008) so as to be able to identify patient visits with related complaints and create a distribution of the visits representing how these symptoms vary or are similar within a specific number of ED visits in terms of the time of occurrence. These symptoms were chosen based on a frequency analysis (by NC DETECT) of most common symptoms present in the syndrome positive records, with no (or negligible) occurrence in the syndrome negative documents. We expect this to be useful to state health officials interested in looking at such patterns across a specific population during some timeframe. To give it a generalized form, I use the term 'event' in place of 'symptom' throughout this work.

III. Temporal Information

The term 'temporal expression' (TE) is used to refer to any textual phrase or clause in the TNs that can tell when an event occurred and can refer to specific date, day or time. Such details fall under two main categories: direct or inferred. The former refers to expressions that directly state temporal details, (e.g., *on Tuesday at 6 a.m. or on 01/01/2007*) while the latter refers to the ones that require some interpretation to be made (by the user) from the actual expression (e.g., *yesterday, 2 weeks ago, last month, 3 hrs ago*). In this research, we make an effort to identify and interpret TEs from both these categories.

There are many levels of inference that can be made from TEs. Some are straightforward, like *yesterday* translates to a specific date based on today's date. With some expert consensus, it is also possible to relate *yesterday* to a time range. Both categories of TEs can be taken to another level of interpretation which is done with domain knowledge, for example in *Took Tylenol yesterday. Fever started around 6 a.m.,* a person with domain knowledge usually makes the judgment that since fever started around 6 a.m. yesterday, the patient took Tylenol sometime during the course of the day though the order of mention of the events has been reversed. We don't make such interpretations in this work since that introduces a totally new kind of complexity that is not currently critical to this application. This research deals only with TEs previously identified by Zhou et al. (2006) using discharge summaries and subsequently investigated by Irvine et al. (2008) for TN domain. Here we deal with accuracy only at the day/date level as not all TNs contain explicit information at the time level.

IV. Markup Languages for Event and Temporal Expressions

Annotation schemes were developed so that the computer system could *learn* from manually annotated natural language samples and duplicate the same, thereby imitating the way humans manipulate language. Starting from the 1990s, a lot of effort (Ferro et al., 2005) has been made to create a standard for annotating temporal expressions with a normalized representation of the time information the TEs denote. Initially, the schemes considered TEs as standalone targets for extraction and annotation. Examples are early TIMEX versions, based on which a few automatic taggers and annotation tools were designed: TEMPEX (Mani and Wilson, 2000) and Callisto (Day et al., 2004)

Later the focus shifted to developing standards that enabled interpretation of the expressions that refer to time, which requires knowledge of the temporal context in which they occur, and also normalization of TEs, so as to create a standard way of communication between different programs or systems. The 2005 version of TIMEX2 was created with this goal in mind. This was further extended to create the TimeML standard that aims to capture the richness of time information in documents by separating TEs from events. Time analysis is distributed across four major components: TIMEX3 (captures TEs), SIGNAL (words indicating relationship between temporal objects), EVENT (covers all situations that happen in reference to the TEs) and LINK (used to relate the TEs as well as for ordering events), each one represented as a tag with attributes.

Though it is advisable to use such established standards while developing applications, TimeML would provide a more detailed representation than is required for this work at its current stage. This is mainly because of the limited scope of *events* in this context, i.e., the main events we are looking at are occurrences of symptoms in a time-related manner across patients and not the individual symptoms suffered by a single patient, hence we use a simple, straightforward representation that facilitates this kind of analysis. It should be noted that TimeML would be the best possible scheme to use if one wanted to analyze each patient's symptoms across time, trying to find links between their occurrences.

V. Information Extraction from clinical text

Information extraction (IE) has been a challenging type of information retrieval and an exciting area of research, especially when applied to natural language documents, dating back to the late 1970s. The Message Understanding Conferences (MUC) have played a very important role in the development of various IE tasks, like Named Entity Recognition (NER), co-reference resolution, relationship extraction, terminology extraction and so on but most of this work has been done on standard sources of text like news reports and military reports, which, though written in natural language, could be termed as well-formed and come from a standard language vocabulary. In complete contrast are sources of text like Chat messages, e-mails and blog texts, which have no defined structure for the content part. Though they are written in natural language for most part, sometimes they do not have well-formed sentences and contain a lot of abbreviations, acronyms, misspellings and terms known only to specific users.

Clinical text falls under a separate category of natural language and is usually a combination of standard language terms and medical terminology. A few examples are lab reports, physician and nurses' notes, discharge summaries and electronic patient records (EPR). Health Level Seven (HL7) is a well-known standard that enables exchange, integration, sharing and retrieval of electronic health information, by developing standards for related concepts, documents, applications and messaging (packaging information and communication between parties). Not many existing electronic health information systems have adopted such standards, thus making the task of representation of health information in a standard manner, an interesting and exciting challenge.

Much work has been done on text mining and Information Retrieval (IR) from medical records and multiple methods and applications have been identified in this regard. The applications range from cleaning the text and creating a structured representation so as to help IE tasks (syntactic) to some form of actual IE and interpretation like temporal analysis (semantic). Any IR system defines three basic elements: document and query representation, matching function and a ranking criterion (Ruch et al., 2002) and IE systems for medical text are in no way different. Hence, the major preprocessing steps needed by many unstructured or semi-structured medical records are spelling correction, abbreviation and short form expansion and normalization of non-standard term usages based on some vocabulary. To facilitate this, many dictionaries and term lists have been created in addition to programs like the MetaMap Transfer that automatically link terms to standard biomedical concepts (Osborne et al., 2007). Most IE systems that work on patient data fall into one of the following categories: *time-oriented*, *problem-oriented* and *source-oriented* (Yousefi et al., 2009). In time-oriented structure, patient information is categorized into groups of events based on TEs. In problem-oriented structure, the information is grouped under one or more of recognized problem headings (Subjective, Objective, Assessment and Plan) based on the intended application. In source-oriented structure, contents are arranged according to source of the information, for example, notes of visits, X-ray reports and blood tests. This has more to do with syntax when the EHR is at least semi-structured, but becomes a very complex task with unstructured data, when domain knowledge becomes imperative.

Currie et al. (2001) propose a linguistic approach to IR from medical texts, made possible by a minimal tagging of basic document structure characteristics (like section headings) and then querying records using keywords in defined linguistic contexts. However, this approach relies heavily on dictionaries developed specifically for this purpose, to record synonymous and near equivalent expressions, polysemous expressions and ambiguous expressions. In addition to these lexical contexts, the authors also take into consideration syntactic contexts, namely part of speech of occurrence of the keywords and also semantic and pragmatic issues like belief contexts, negation, presupposition and implications. Such an approach will be quite useful when one wants to extract details about a particular patient's visit and relate them to the patient’s past medical history or previous visits.

Yousefi et al. (2009) discuss scenario-oriented IE from Electronic Health Records (EHRs) by modeling the relationship between diseases, symptoms, signs and other clinical information as a graph and extracting all possible diagnostic hypothesis related to a specific scenario. Then, they identify information related to the extracted hypotheses and search for matching evidence in the EHR. The basis for this approach lies in the Disease-Graph modeling the authors create based on relationships between diseases, symptoms/signs and EHR categories, which is weighted based on established expert opinion on how much each EHR item contributes in the diagnostic process and resources such as UpToDate (http://www.uptodate.com). They also collect supporting and weakening evidence for each hypothesis based on internal knowledge of disease-attribute relations. One possible application of this approach in the context of my work would be to enable automatic diagnosis based on a timeline representation of the TN by building dynamic graphs that represent the symptoms and signs related to disease outbreaks in the area (if any). For such an approach to be effective, the underlying model should reflect the relationships between symptoms, TN information and diseases in a manner as exact as possible to avoid a large number of false positives and negatives, both of which will be crucial to treating the patient.

VI. Timeline Representation

Event detection and ordering has been the end goal of most of the research done on temporal information extraction, irrespective of the data source. The results of such analyses are best represented in the form of statistics or on a timeline, showing the temporal order of occurrence of events, sometimes including the relation between two or more events (like cause and effect). A timeline representation of events can assign each event to either specific time point or time duration. Though both cases are applicable to this research, currently we relate an event to a single point on the timeline, preferably the starting point, even in cases where duration is mentioned. Depending on the events and the intended application, modifying the system to include duration should be straightforward.

CHAPTER 3

RELATED WORK

Combi and Shahar (1997) identified *Temporal Reasoning* and *Temporal Data Maintenance* as the two broad categories of research directions with respect to time information in medical text. Temporal Reasoning supports tasks like decision support and forecasting while data maintenance deals with storage and retrieval of time-associated data. Since then, a lot of researchers have focused on both these aspects to develop meaningful applications to cater to the needs of clinicians as well as informaticians. Though news text has been the preferred source for doing temporal analysis (Johansson et al. and Ahn et al., 2005), mainly because of the narrative nature and the wealth of information in it, researchers soon realized the importance and application of extending the work to medical text.

Discharge summaries have always been a favorite source of medical text for temporal IE tasks, since these usually describe the events that occurred beginning from the time the patient entered the hospital till the discharge time, thus encompassing the past medical history, present illness, laboratory tests, other procedures, diagnosis, hospital course and medications. Most of these events are accompanied by timestamp(s) since the summary will eventually become a part of the patient's EHR, thus serving as future source of reference regarding the patient's health.

Zhou et al. (2005) describe the architecture of a system they developed to process the temporal information in clinical narrative reports. The major components of the system are:

- Temporal Tagger that aims to represent the temporal information in the narratives in a structured manner using a Temporal Constraint Structure (TCS) to model the TEs.
- NLP System to extract, structure and encode the clinical information in the narratives, for which they use MedLEE, which produces an XML output.
- Post-processor reads the XML data to perform temporal reasoning, linking events to TEs or to one another, based on linguistic, biomedical and domain knowledge.
- Simple Temporal Problem (STP) Model to represent events as nodes and the associated temporal information as constraints of a directed graph.

The authors describe the TCS in detail in their subsequent work in 2006, essentially coming up with a list of prevalent classes of TEs and events found through manual inspection of 100 discharge summaries. Based on this list, they decided on the fields of TCS to best describe the events in the narratives along with associated temporal details. The authors argue that though there are other standards like TimeML (*Pustejovsky et al., 2003)* developed to serve the same purpose, they weren't applicable to this data because they were developed based on news text. Though one must agree that using TimeML might not be simple, TCS is in many ways similar to TimeML and hence the task wouldn't have been impossible. For example, *anchor* tag is basically the *timex3* and *signal* tags put together and the creation of final graph would have been straightforward if they used just the *tlink,slink* and *alink* values. This work also laid the basis for development of **TimeText**, a temporal reasoning system, evaluated in Zhou et al. (2008). The measurements used for evaluation were correctness of generated temporal relations, recall of clinically important relations, and accuracy in answering temporal questions asked through queries.

Bramsen et al. (2006) propose a machine-learning approach for temporal analysis of discharge summaries, using a manually annotated dataset for supervised training. They organize the narrative into *temporal segments,* which is a part of the text that corresponds to a particular time point or frame and then order these segments based on the temporal information. They use a coarse annotation scheme which does not capture event overlap and uses only three ordering relations: *before, after* or *incomparable* between each pair of segments considered. They use a classifier with unigrams, bigrams and trigrams to automatically extract TEs specific to discharge summaries. In order to find a consistent global ordering based on the pair wise ordering, they use the strength of pair wise ordering to resolve cycles while combining pairs of segments to create an ordered summary.

The major differences between the works of Zhou and Bramsen are that while Zhou focuses on events and tries to associate temporal details to events to produce a model of how events are related based on temporal constraints, Bramsen takes a totally different approach by looking at events from a temporal point of view, aiming to create a temporally ordered summary irrespective of any relation between the events in the narratives.

While both these authors take the entire discharge summary into consideration for their analyses, Gaizauskas et al. (2006) emphasize the importance of restricting the extraction of temporal information from medical texts like discharge summaries to serve a particular task (clinical investigations like x-rays and ultrasounds in this case) in their work. They also limit the smallest temporal unit handled by the system to be a day, the links between events to *before*, *after* and *is_included* and do not include any explicit identification of temporal relations between events. The baseline algorithm proposed by them looks for standard time expressions within the same sentence as the event of interest and asserts a link between them. If no link can be found, then the algorithm tries to infer the relation based on tense and aspect information in the sentence.

On the same lines is the work by Harkema et al. (2005) called the Clinical e-Science Framework (**CLEF**) where the authors try to combine multiple sources of patient information to construct a single consistent record of the patient's condition and treatment over time, called *patient chronicle*, by exploiting the temporal information found in these sources. Due to the complexity of the task, the authors rely on some structured source for mention of the event under investigation (standard procedures like X-Rays are considered), and the narratives for the TEs, based on the observation that

"The structured data component of a patient's clinical record will cover all or most of the noteworthy medical events occurring during a patient's clinical history, such as major diagnoses, the initiation and discontinuation of drug treatments, and investigations".

The authors use TimeML for annotating the narratives and use the tense and aspect of verbs in combination with the date of the document to infer temporal information where it is not explicitly available. This is different from the works described previously in its application (the ability to look at multiple occurrences of same event to determine the actual date/time of occurrence). A similar application for TN text would have to use a different time-scale, given that the events in TNs are usually concentrated around the time of visit.

Related to this is the work done by Irvine et al. (2008) in developing **TN-TIES**, a system for extracting temporal information from ED TNs. Their annotation scheme is based on that of Zhou et al. and they use a simple framework to create segments of TNs based on the TEs identified to belong to each of the defined categories, and each segment is then classified into its corresponding category. The contribution of this work to research in the area lies in the interpretation part, where each segment is associated with a start and end time point based on its class and the date/time details of the ED visit. This essentially produces a timeline marking the start (and end) of events related to the patient's visit.

As discussed by Suominen et al. (2007) and Zhou and Hripcsak (2007), the task at hand is very challenging that one has to think well before applying traditional or established NLP techniques to process and extract information from a special genre of medical text like the TN, that does not always conform to the norms of ‘natural language’. To create temporal order among the events, we use an approach similar to Gaizauskas' and limit the refinement of temporal detail to the *day* level, while at the same time, trying to maintain relative ordering based on smaller levels of temporal information (like hours or minutes) within the day level without explicitly disclosing them. When the temporal detail is not to be found, we use the date of visit as the time of occurrence of the corresponding event, even in cases where domain or semantics based inference of occurrence time could possibly be made.

CHAPTER 4

SYSTEM ARCHITECTURE AND OVERVIEW

The architecture of the first part of the proposed system is shown in Figure 2. Figure 1 shows the TN used to illustrate the output of each of the three major components of the system. Figures 3, 4 and 5 show the output at each stage.

Chunker

This module parses the triage notes in order to help differentiate between parts of the notes that carry different time-related information. Bramsen (2006) describes a method for *temporal segmentation* in his thesis where he talks about learning from *lexical features, topical continuity, positional feature* and *syntactic features* of the data to determine temporal boundaries. According to him, temporal expressions that mark temporal discontinuity can be identified using lexical, positional or syntactic features of the document and usually, change of topic may imply change in temporal flow. He uses these as markers to identify the temporal segments from the data. Irvine (2007) describes a shallow parser written as regular expressions based on manual analysis of her data to serve the same purpose. She uses certain punctuation markers, conjunctions and a few communication verbs indicative of events found in TNs to identify segment boundaries.

The major difference between these two approaches is the type of data under consideration. While Bramsen worked with mostly well-formed text like discharge summaries, Irvine used triage note data, which sometimes lacked even proper sentence formation. Though this work uses TNs, using the markers Irvine used for her work can identify specific events and the related TE but it cannot help in the identification of all events and TEs in a TN. We needed to do this so as to not omit any information present in the TNs.

Hence we designed a parser that identifies shifts in temporal details. Based on manual observation of the triage note characteristics and the intended application of this research, we used regular expressions (REs) that identified the different TEs that fell into one or more of the previously defined temporal categories from the TNs. The chunker, written in Python, divides a TN into segments with one (or sometimes more) associated TEs belonging to a predefined class. Parts of a TN with no identified temporal information remains undivided but gets included as a part of the chunker output. Thus, the chunker doesn't leave out any information in the TNs.

```
2006-03-02 13:30:00; chief complaint subjective: fever x 4 days, rash started
on her face yesterday drinking but vomiting x 3 since last night. urinating
well. green runny nose and cough.
```

Figure 1: Example TN

Appendix A lists the REs used to capture TEs (and their classification) relevant to this research.

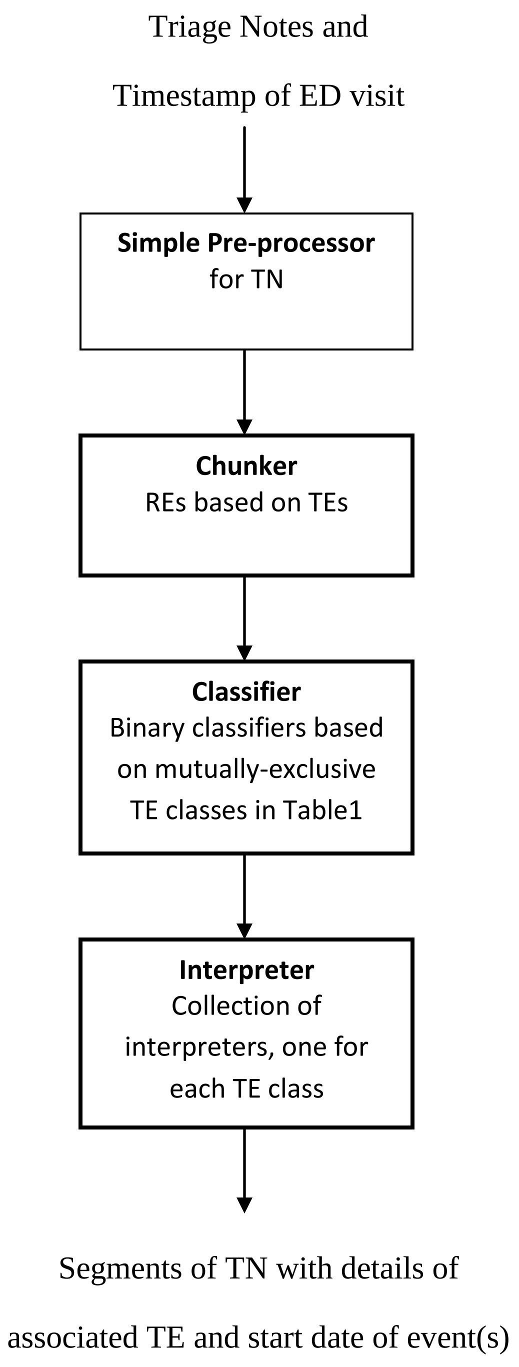
Triage Notes and
Timestamp of ED visit
Simple Pre-processor
for TN
Chunker
REs based on TEs
Classifier
Binary classifiers based on mutually-exclusive TE classes in Table1
Interpreter
Collection of interpreters, one for each TE class
Segments of TN with details of associated TE and start date of event(s)

Figure 2: System Architecture

Visit Date	Visit Time	TN chunk
03/02/06	01:30:00 PM	chief complaint subjective: fever
03/02/06	01:30:00 PM	x 4 days
03/02/06	01:30:00 PM	, rash started on her face
03/02/06	01:30:00 PM	yesterday
03/02/06	01:30:00 PM	drinking but vomiting x 3
03/02/06	01:30:00 PM	since last night
03/02/06	01:30:00 PM	urinating well. green runny nose and cough.

Figure 3: Chunker Output

<u>Classifier</u>

The classifier is a module that could have been easily combined with either the chunker or the interpreter since all the modules work with the temporal categories previously identified as the basis. For example, while determining the segmentation of the TNs using REs, one can do it separately for each category of TE and group those chunks belonging to a category together, thus merging the chunking and classification step. One could also take the output produced by the chunker and based on the identified TE, do direct interpretation of the chunk based on the category it belongs to without explicitly performing the classification.

VisitDate	Visit Time	TE Class	TE	Associated Event Chunk
03/02/06	01:30:00 PM	reldur	x 4 days	chief complaint subjective: fever
03/02/06	01:30:00 PM	reldt	yesterday	rash started on her face drinking but vomiting x 3
03/02/06	01:30:00 PM	timdt	since last night	urinating well. green runny nose and cough.

Figure 4: Classifier Output

Class	Sample TEs	Sample REs	Occurrence % in evaluation set
Date/Day	on wednesday Friday 2006/03/01	[^a-z]on [a-z]{3,6}day[^a-z] [^a-z][0-9]{1,2}/[0-9]{1,2}/[0-9]{4}[^a-z]	10%
Relative Date/Day/Time	Since last Thursday yesterday 4 days ago	[^a-z][0-9]+[ ]* [(day)|(wk)|(hr)]+[s]*[^a-z]ago	56%
Time	6:30 this morning 09:00:00 PM 0800	[^a-z][0-9]{1,2}:[0-9]{2}[ ]* [a|p]+m	14%
Duration	For 1 week x 2 days for about 24 hrs	[^a-z][(for)|(x)]+[ ]? [about|approx]*[ ]*[0-9]+[ ]*[(day)|(wk)|(hr)]+[s]*	17%
Others (includes fuzzy time and TEs not considered in this research)	month year at that time pta (prior to arrival)		3%

Table 1: TE classes used in this research

We keep the classifier separate in order to introduce uniform modularity in the system. This step also serves two other purposes – one, that of determining if any part of the immediately preceding or succeeding chunks contain the information related to the TE in the current chunk, which is possible when there are punctuation marks like *comma* (,) or *semi-colon* (;) or conjunctions like *and* in the previous chunk. For example, in the TN *fever started yesterday and rash started today*, the chunker divides the note into *fever started yesterday and rash started* and *today* but the classifier relates *rash started* to the TE *today*. The chunker is not trained to do this part since such information can sometimes belong to the same TE and sometimes not. Secondly, the classifier also helps identify chunks belonging to same TE, which is duplicated in various parts of the TN. For example, in the triage note, *fever started 2 days ago. Took Tylenol yesterday. h/a 2 days* *ago*, both fever and headache started two days ago but are separated by another TE (*yesterday*). Identification in this case was easy because of the repeated TE *2 days ago*, but had it or an equivalent expression not been there (for example, *headache too*), it might not have been possible to associate *headache* and *2 days ago*.

```
Input: TN chunks identified by chunker

Output: TN segments assigned to corresponding TE chunk, classified based on table in
appendix A

IF first chunk is not a TE THEN
        STORE chunk to assign to next TE
ELSE
        IF next chunk is not a TE THEN
                IF next chunk has any of the keywords in the list (‘,’, ‘;’, ‘and’ or ‘.’) THEN
                        Split next chunk at last occurrence of keyword
                        Assign first part and previously stored chunk(s) to current TE
                        STORE second part to assign to next TE
                ELSE
                        Assign next chunk and previously stored chunk(s) to current TE
```

Figure 5: Pseudocode for processing of chunks by classifier

The classifier uses the same REs as the chunker, to identify the temporal chunk and class information for each TE. These REs are grouped together based on the TE class they belong to (listed in Appendix A). We use RE matching and the precedence order: time class, relative day/date class, duration class and then day/date class to determine which TE class a TN chunk belongs to. If there are chunks (or part of chunks) preceding the TE and unassigned to other TEs, the classifier relates them to the current TE. To identify the entire TN segment related to a TE, the classifier looks for keywords in the chunk succeeding the TE, to see if there are parts of that chunk that belong to this TE. The keywords currently used (in the order of increasing precedence) are ‘*,*’, ‘*;*’ , ‘*and*’ and ‘.’. When these keywords are found in the next chunk, the classifier assumes that the part of the chunk preceding the last occurrence of such a keyword belongs to the current TE and the part following the keyword would belong to the next TE. If there are no such keywords or if there are no TEs that follow the current TE in the TN, the entire succeeding TN chunk is assigned to the current TE. This approach is illustrated in appendix B.

To produce a consistent classification and interpretation, we used observations from previous works, SQL queries and manual verification of the temporal details found in TNs in order to capture all possible variants of the TEs of interest. For example, most days of the week can be mentioned either using their complete form (like *Tuesday*) or a short form (like *Tue* or *Tues*). The more complex ones are *Wednesday* and *Saturday*, which when shortened, namely as *wed* and *sat,* could also act as verbs. In such cases, we look for other accompanying information that indicates that this is really related to a day (time). For example, we identify *sat* as a TE only if the expression is *on sat*; hence occurrences of *sat* in the beginning of a sentence will not be captured. Another commonly found feature in TNs is time, which is sometimes mentioned using four digits (example *0700*) and sometimes written in *hh*[*:* or .]*mm* format (example *7.00*). Most times, such information is followed by *a.m* or *p.m* when written using the latter notation but one has to look for a modifier like *at* that is ideally expected to precede time to catch the temporal detail. These are just a few specific examples of the kind of learning process involved in designing the modules. Apart from trying to capture such variants, the major challenge is trying not to capture expressions unrelated to temporal detail (like *sat* or *am* as in ‘I am’) in the process.

Based on existing classification of temporal expressions (TEs), chunks associated with TEs are divided into classes, mainly indicating the nature of temporal information related to the chunk. Though classification doesn't play an important role in the final output produced as explained earlier, this step is designed mainly to make the interpretation process simpler. Previous work using these classes produces an overlapping classification of chunks, where some chunks belong to more than one class but for this work, we do a non-overlapping classification since granularity in terms of time is less important compared to capturing all temporal information. Classifying chunks this way enables easy interpretation of the time of occurrence with respect to the timestamp associated with the visit. Similar to the approach adopted in Irvine et al. (2008) we use a series of simple binary classifiers, using Table 1 and a precedence relation based on the associated TEs so as to not classify based on parts of expressions that are actually members of a particular class into another class. For example, while *Tuesday* belongs to the *absolute date/day/time* class, *since Tuesday* should be classified into *relative date/day/time* class that has a different interpretation.

Interpreter

This module takes the classified segments of a TN and arranges them in a relatively progressing order (increasing order of time), associating the segment to a particular date based on the class it belongs to. We define the precedence order in which these classes are to be processed so as to create the best possible global temporal ordering of the TN events. In case of ambiguity, the chunk is associated with the date that is obvious from the other details. For example, if the patient visited the ED at 3am on 01/03/2007, *yesterday night* could refer to the night of 01/02/2007 or to the night of 01/01/2007. In such a case, the interpreter would associate the event with *yesterday*, i.e., 01/02/1007. The datetime Python module has built-in functions to help in converting relative differences in time to actual date/time based on the reference time point.

In its current stage, this research is interested only in associating the information in the triage chunk to its start date based on the TE and the time of patient's ED visit. Though the TE might alone be enough in case of absolute date/time TEs, we need the latter information to make an inference in case of relative TEs like *yesterday* or *3 nights*

Visit Date	**Event Start Date**	**TE**	**TN Event Segment**
03/02/06	02/27/06	x 4 days	chief complaint subjective: fever
03/02/06	03/01/06	since last night	urinating well. green runny nose and cough
03/02/06	03/01/06	yesterday	rash started on her face drinking but vomiting x3

Figure 6: Interpreter Output

ago. This was done to help healthcare professionals determine the beginning of certain events related to a patient's visit, giving lower priority to the knowledge of the duration of the event, i.e., if the event has already ended or is currently in progress. In certain cases, like *I started taking Tylenol 3 days ago and took 3 doses, one each morning and night*, the fact that the patient stopped taking the medicine a day or so ago might play an import part in the course of actions to be taken.

The interpretation part has been made straightforward because of the non-overlapping classes. Had we done an overlapping classification, which is justified, since the identified classes are not necessarily mutually exclusive, some decisions need to have been made at this stage so as to not interpret the same segment in different ways based on the different classifications and also to avoid producing redundant output. On the other hand, choosing such a method might have made it possible to reflect all possible interpretations of the same TE (which is possible even when humans process the information), so as to give the user an option to choose what he/she thinks is the best way to look at the information.

Pre-Processor

This module is not an integral part of the system but was included so as to better prepare the data to suit this kind of analysis. The TNs were initially checked for common misspellings (mostly typographical errors) related to temporal information and the symptoms of interest. This was done since the notes were created by nurses in the ED in a time-critical situation, where spell and grammatical checking are not applicable. Most of these misspellings have been previously identified by researchers in the area (Travers et al. 2007) and most other common ones can be caught using regular expressions. Another class of words that needed preprocessing was abbreviations, found in abundance in TNs. Though there are a few databases and dictionaries that have been built with this in mind (Zhou et al. 2006), they alone did not suffice owing to the subjective nature of notes, where the acronyms and abbreviations vary from one geographic location to another, one medical institution to another and from one nurse to another. So, we developed a comprehensive list of short forms of words found to be most prevalent in the data, to be used along with existing dictionaries. It should be noted that cleaning this kind of text is an area of research in itself; hence we do not use this module to improve the performance of the system but to make sure the relevant information (including all possible variants of it) gets captured.

Symptom Onset Date Identifier

The second half of this research is about using these time-interpreted TN segments to analyze occurrences of similar symptoms across patients. The purpose of this research is to see when particular signs/symptoms related to particular disease syndromes start in a population with respect to the patient's ED visit and identify similarity/dissimilarity patterns (if found) in the progress of symptoms among the patients. For example, there might be a few patients who got Symptom A two days before their visit, followed by Symptom B the next day (the day before visit) while another set of patients might have it the other way around, i.e., Symptom B followed by symptom A.

Since this analysis is done with the day of visit as reference point, the analysis will reflect variations among patients who decided to go to the ED after they had been experiencing the symptoms for some time (say, a week after the initial symptoms) and patients who visited the ED immediately after the start of the symptoms. The general assumption here is that patients commonly visit the ED as early as possible, especially in case of uncommon or alarming symptoms like rash, bleeding, etc. and that the most recent symptoms (less than 2 weeks) are usually the most relevant ones (based on NC DETECT syndrome definition). This analysis produces bubble plots indicating distribution of patients from a population corresponding to the onset date of two sets of symptoms belonging to Fever Rash syndrome. These are, namely, *fever or constitutional* symptoms like *fever, body ache, chills,* etc. and *syndrome-specific* symptoms like *rash, blisters*, etc. We then look at co-occurrence patterns and identifying subsets of population that suffered from the symptoms at around the same day(s) which could be an indication of how the symptoms progress in general.

Fever/Body Ache/Chills Onset Day	**Rash Onset Day**
-3 (3 days before ED visit)	-1 (1 day before ED visit)

Figure 7: Symptom Onset Date Identifier Output

CHAPTER 5

EVALUATION PLAN

This chapter describes the test data set, modules to be evaluated, evaluation parameters, and expected results, based on human judgment of the test set.

Data

North Carolina Disease Event Tracking and Epidemiologic Collection Tool (NC DETECT) is North Carolina's statewide syndromic surveillance system and was created to address the need for early event detection and timely public health surveillance in North Carolina using a variety of data sources. One such source is the Emergency Department (ED) data collected from institutions state-wide, which is used to provide syndrome-based monitoring. The ED syndromes are used for both trend analyses and outbreak detection (NC DETECT 2008 Annual Report). Existing tools determine if an ED record represents a case belonging to one or more syndromes based on different fields like *Temperature*, *ChiefComplaint* and *TriageNote*.

Each syndrome is usually associated with two or more classes of symptoms (*fever/constitutional* and *syndrome-specific*) that are expected in patients suffering from that syndrome. For the purpose of this research, we create a dataset 200 ED TNs classified as belonging to a particular syndrome (in this case *Fever Rash*) so as to be able to capture notes that carry similar symptoms and signs. Based on statistics provided by NC DETECT, *fever/chills/body ache* and *rash* were identified as the most common symptoms related to *Fever Rash* syndrome. 174 of the records had an onset date associated with *fever/chills/body ache* and 144 records had an onset date for *rash* with 118 TNs having both sets of symptoms, resulting in a total of 200 TNs.

Chunker

Ideally, the chunker should identify only those TEs it was designed to identify and their variants and use them as TN-splitting keywords. The desired output is a list of TN segments, including the TEs that divided the TN into these segments. The unique random ID assigned to the TN is used to differentiate between the TNs. Evaluation involves two parts – one, to check if the TE chunks include only intended expressions and no unwanted terms or groups of terms have been identified as TEs and secondly, check for any missed TEs in the other TN segments.

Classifier

Once the chunks are in place, the next step is to see if the classifier associates each TE with the correct TN segment(s) and to make sure that the TEs are identified correctly to the class they belong to based on Table1. The aim of evaluation at this stage is to compare the system's segmentation performance with the human-created gold standard, including associating each segment with the correct TE, irrespective of the chunker's performance.

Interpreter

As discussed earlier, the two major goals of this step are to identify the start time related to the segment and to arrange the segments in a TN in increasing order of occurrence times. Verification here is used to check how the system performs in comparison with manual interpretation of the classified segments identified in the previous stage.

Gold Standard

Our approach for temporal analysis of TN requires annotated data for training as well as testing. Two annotators manually annotated the data to reflect the chunking, classification and interpretation desired of the system based on the same rules used to develop the system. The resulting set acts as the gold standard data set used to evaluate different parts of the system developed to automate the process. Recall and precision values are used as quantitative measures to evaluate system performance. An ideal balance between the recall and precision is desired since high precision alone does not necessarily mean the system captures all desired information and high recall alone does not say enough about where the system fails.

For evaluating the second part of the research, namely identifying co-occurrence patterns of different sets of symptoms in the TNs, we create a MySQL database of the interpreted TN segments and the unique identifiers. We then use SQL queries to get counts of the TNs with and without the occurrences of the symptoms of interest. This was done to reduce the unnecessary human effort that would be needed if a gold standard were to be developed for obtaining the counts. Also, since we are, at present, looking at a few simple terms as evidence of the symptoms, SQL queries produced the desired results. It is important not to include negated symptoms, i.e., not counting a symptom when it is associated with a term that indicates its absence. For example, *fever* is not experienced by the patient when the TN says *no fever*, *absence of fever* or (-) *fever*. Some domain knowledge and the training data were used to design the queries to reflect such criteria.

CHAPTER 6

RESULTS

The system was tested to evaluate the performance of each module individually in comparison with the gold standard as well as that of the entire system in the final stage, i.e., finding onset dates of the symptoms of interest. The results of this on a sample of 200 triage notes belonging to the *Fever Rash* syndrome are reported and discussed in detail in this chapter.

Chunker

The input TNs were divided into 1001 chunks by the chunker based on the identified TEs, out of which 402 were TEs and the rest were the events listed in TNs. There were 404 TEs identified in the Gold Standard set and the system's output were checked against these to quantify the performance of the chunker (Table 2).

	Gold Standard		
System Output		+	-
	+	386	16
	-	18	16

Table 2: Chunker results vs Gold Standard TE identification

It was interesting to note that almost all False Positives (TEs identified by chunker but not by gold standard) involved digits which were picked up by the chunker trained to find time and dates. For example, parts of TN that matched the regular expressions designed for time, like *3 this am* from *102.3 this am* were included in the chunker's output. On the other hand, most False Negatives (TEs identified by gold standard but missed by chunker) did not contain digits; examples include a few unexpected TEs like *all day, during the week*, *30 mns* and *jan6*, abbreviations like *yest* and numbers in words like *in a couple of days* (indirect) or *about two days ago* (direct). With some clever and careful design of REs, one can clearly overcome these errors in the future.

Some of the expressions this chunker was not designed to recognize are those in the fuzzy time category like *at this time, at about the same time, several days ago* and those that indicate a time period more than a few weeks like *for a month or so, for app 2 months*. This was done under the assumption that symptoms that occurred more than 2 weeks prior to the ED visit are not critical for surveillance purposes and hence for the application intended in this research. Another important consideration was to not pick up expressions like *3 months old* that usually indicated age since even with the current design of the chunker, phrases like *11 weeks* that indicated age were being recognized as TEs. Though it could be called a TE, such an expression would increase the chances of confusing the classifier while associating events in TNs with TEs in cases where there is no other TE or when there are multiple TEs in the TN. Nevertheless, building on the current system, it would be relatively easy to include such TEs when we identify applications that will require this information for processing.

Classifier

As described earlier, the classifier was built to process the chunker output to identify those parts of a TN that belongs to each TE and then classify the TEs based on Table 1. The classifier identified 397 TEs and their corresponding TN segments based on the 402 TEs identified by the chunker. Since the classifier is designed to work on the same REs as the chunker, the reason for the difference in their outputs was not obvious but manual verification of the classifier output revealed 2 primary reasons for this; one being absence of any segment of TN to relate to the TE and the other being repetition of same TE in a TN, which resulted in the classifier merging consecutive occurrences of a TE into one in its output. Consider the following TN:

> chief complaint subjective: rash on side of face (left temple area), fever thursday and saturday night and today from nap (wakes up soaking wet, does not know temp)

The chunker produced the following output:

- chief complaint subjective: rash on side of face (left temple area), fever
- thursday
- and
- saturday night
- and
- today
- from nap (wakes up soaking wet, does not know temp)

and the classifier produced:

TE	Associated TN chunk
• thursday	chief complaint subjective: rash on side of face (left temple area), fever and
• today	and from nap (wakes up soaking wet, does not know temp)

From this example, we can see that there was no part of the TN that could be associated with *saturday night* and hence the classifier skipped it altogether. The classifier also showed some erroneous behavior while deciding which segment of the TN gets assigned to which TE, with a precision of 87%. Again, there were three major causes of this based on the way in which the classifier divided a TN chunk to assign it to a TE, which is through a set of 'identifiers' like *comma (,), semi-colon (;), period (.)* and *and.* The presence of one or more of these identifiers caused a TN chunk to be split, under the assumption that the part after the identifier belongs to a different TE. If no such TE followed the chunk, then that part is assigned to the previous TE.

The classifier precision was affected when no such identifier was found in the chunk but a part of the chunk did actually belong to a TE different from the one the chunk was assigned to or when the presence of the identifier did not mean the part of the chunk following the identifier belonged to a different TE or when many such identifiers were found, thus making it difficult to identify a clear boundary between the TN segments belonging to different TEs. An example of the third case is:

since yesterday sorethroat, developed fever and rash today and dry cough.

Though a human reading this TN would identify *fever and rash* to belong to *today*, the system was designed to split the chunk *sorethroat, developed fever and rash* at *and*, thus assigning *fever* to *yesterday* and *rash* alone to *today*. Scenarios like this may be overcome by using additional constraints on dividing the chunk, like giving importance to terms like *developed, onset* or *started*. There are also other keywords like *but* that differentiate between time points but I was critical of adding them to the list of identifiers due to low frequency of occurrence in the dataset (less than 1%).

Interpreter

Built with the aim of identifying the 'onset time' of the events in the classified TN segment and ordering all the segments on a time line, the interpreter produced an output of 365 such segments from the output of the classifier. It is important to remember that the interpreter uses a different logic to process the classified chunks based on the class of TE instead of the REs used by the previous two modules. Hence the interpretation of event start time depends only on the TE and is independent of the event-listing TN segment. It is this step that determines the performance of the interpreter since the temporal ordering process is built on the interpretation part.

Some errors identified in the chunker and classifier propagated to this stage, thus adding to the misinterpretations of the interpreter and resulting in lower performance compared to the other two modules. At the same time, the interpreter overcame a few problems of the chunker (False Positives mainly) since it does not use the same REs, hence reducing the number of false positives related to TE and the classifier (especially in case of missed TEs), reducing the chances of false positives, though not being able to eliminate false negatives.

Though using an approach different from the other two modules has improved the performance in certain ways, it has also resulted in a few more false negatives in the interpreter output (5% of the classifier output) as it failed to recognize a few correct TEs.

Before interpreting the classifier output, the system tries to normalize most TEs to keywords that the interpreter uses to recognize the TEs. For example, all occurrences of the RE *yest[^a-z]* are converted to *yesterday*. Similarly, various forms of representing the weekdays are all normalized to their standard long form naming notation, e.g., *wed[^a-z]* and *wednes[^a-z]* are rewritten as *wednesday*. When this normalization fails due to mismatch of the TE and its RE form that the normalize looks for, it reflects in the interpreter output. Nevertheless, the interpreter, as a standalone module, had a very high precision of 96%, missing correct interpretation of only about 3.4% of its input TEs.

Symptom Onset Identifier

This was the final stage of evaluation for the intended application. As explained earlier, this step was done by querying for known expressions related to the symptoms, hence the performance of queries in identifying each symptom can be assumed to be very close to 100%. So we looked at the overall system performance in relation to this particular task and identify which module(s) affected this performance and how.

	Fever/Bodyache/Chills Symptoms		**Rash Symptoms**		**Overall**
N	174		144		317
	Count	**%**	**Count**	**%**	**%**
Correct onset date Identified	125	72%	107	74%	73%
Wrong onset date Identified	27	15%	29	20%	18%
Onset date missed	22	13%	8	6%	9%

Table 3: System Performance in symptom onset identification

As shown in table 3, overall we were able to correctly determine the onset date for 73% of the TNs. For 9% the system did not detect any onset date and for the remaining 18% (N=56) we were able to correctly identify the onset of the symptom but miscalculated the onset date. Of these 56 records where we incorrectly identified the onset date, all except 4 were within one week. We had one TN for each set of symptoms that was off by more than one month. One of these records had TE in terms of *month*, which the system is not yet capable of identifying whereas in the other record, the chunker wrongly identified temperature in Fahrenheit as the TE and the other modules interpreted it as number of days indicating symptom onset.

The classifier module was the source of a large percentage of errors in case of both sets of symptoms (52% of false positives and wrong onset date identification related to fever/body ache/chills and 60% of false positives and wrong onset date identification related to rash are associated with the classification stage of the system) and the impact of the other two modules varied between symptoms. One possible reason for this could be that the classification errors were actually those of the classifier along with those already introduced by the chunker in the previous step. Analysis of these results has helped identify modules that need improvement and the possible methods that can be employed to improve them.

The primary source of errors in this module was identified to be presence of multiple TEs related to same event (either actually so in the TN or due to the classifier) in which case the identifier was designed to consider the earliest date related to the event.

This lead to errors when the prior mention of the event should ideally not be counted as an occurrence of the event, for example, *noticed rash all over body last night. last week brother had scarlet fever. today morning started with fever.* Since *scarlet fever* is associated with *last week*, the subsequent occurrence of fever *today* is ignored by the interpreter.

CHAPTER 7

FUTURE DIRECTIONS AND CONCLUSION

With increasing realization of the importance of Health Information Technology as a critical research area that directly affects the healthcare industry worldwide, more and more related information, especially patient health records, are being made available electronically for research purposes. This thesis explores just one way of processing the vast amount of critical patient health information found in electronic medical records and intends to demonstrate the extent to which the application of simple NLP and information retrieval techniques with some domain knowledge can serve health officials charged with the task of doing the same manually.

Effective processing of such information has to take into consideration the ways in which these data are used at various levels in the healthcare domain. In the case of TNs, after the nurse creates the note, a doctor is the first person the note serves as input to but in most cases, the doctor processes the information in the note in the presence of the patient (or the person accompanying the patient) who reported to the nurse. At this stage, information extraction from the note is not that important since the process might actually be repeated by the doctor, hence it might serve the process better if one could just structure the note so as to present it in a reader-friendly manner to the doctor. The doctor then records his/her own observations of the patient's condition as a part of the patient's medical record, which is then used during the rest of the ED process, for the patient's treatment and billing. Epidemiologists, who study health and disease patterns, are interested in understanding the reasons for occurrence of these patterns and hence need to study the very first report of a patient's illness which has been recorded in the form of TNs. Since they are interested in studying patterns of diseases across time, the best way to represent this information is on a timeline, starting with the initial onset of symptoms experienced by the patient ending with the ED visit.

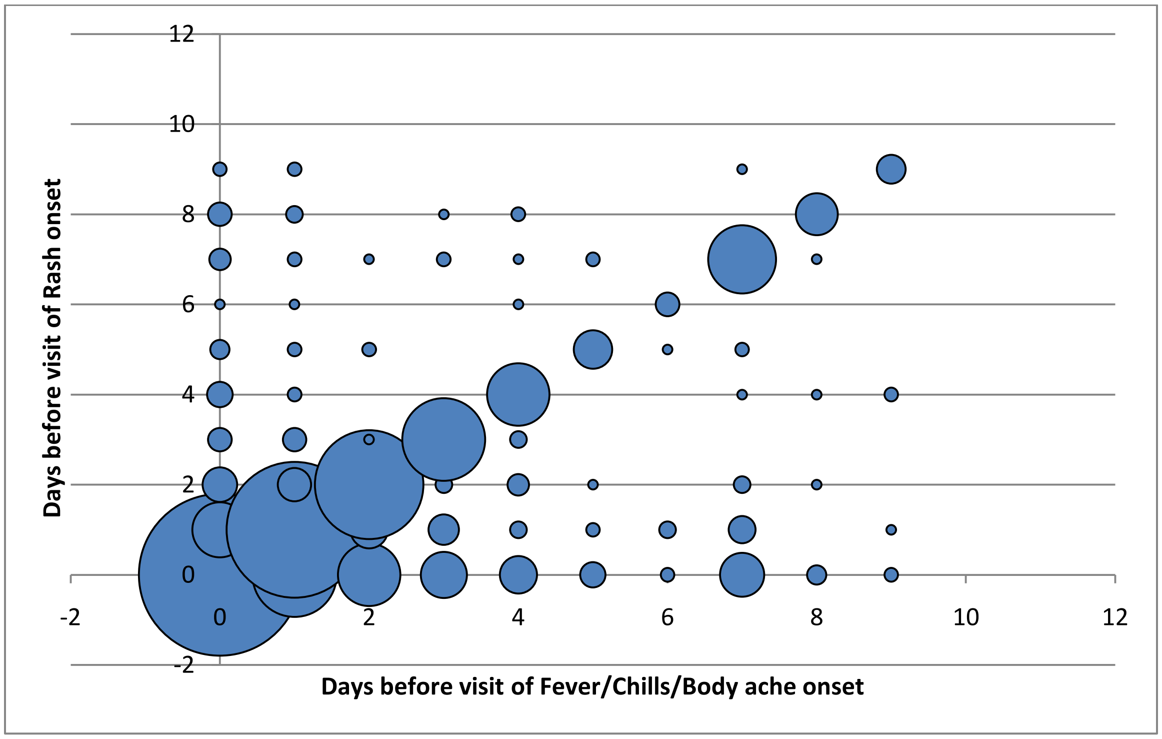

Figure 8: Fever/Chills/Body ache vs Rash onset days in 2000 Fever Rash TNs

Figure 8 is a sample output of the system showing co-occurrence of the two groups of symptoms in a set of 2000 TNs belonging to the Fever Rash syndrome. The horizontal axis indicates the day of onset of *fever/body ache/chills* before the date of ED visit while the vertical axis represents the same for *rash*. Day 0 indicates the day of the visit, day 1 stands for 1 day before the visit and so on until day 8, which actually represents 8-15 days before the visit while day 9 indicates days beyond 15 days prior to the visit. TNs without any identified TEs have been excluded from the bubble plot. From the figure, it is clearly evident that in a great majority (68.5%) of the TNs, the two sets of symptoms occur together. Such an illustration of onset dates can help epidemiologists clearly identify individual as well as co-occurrence patterns of symptoms related to ED visits along with the visits of interest without having to manually go through the records.

There are many ways to improve the performance of each of the three main modules, ultimately improving the overall system performance. The building blocks of chunker are the REs it uses, so creating more robust and highly-specific expressions will have a major impact on the chunking process. For example, by studying a larger dataset, one will be able to identify specific patterns of occurrences of TEs like either being preceded by or followed by certain phrases like *had*, *developed*, *experienced*, *on and off* and so on. Also, developing an alternative to using *[^a-z]* to mark word boundaries might help in picking up only the intended TE. Use of more TN-specific knowledge and parts-of-speech tagging while identifying the TN segment belonging to each TE in the classifier could help in avoiding most errors in that step. In addition, one can create subclasses within the established categorization of TEs to better handle cases of overlapping classification. For example, in *for the past 10 hours*, though *for* clearly indicates that this expression belongs to duration class, if the RE belonging to relative day/date/time class had chunked *10 hours*, then this TE would have been classified into reldt class. Though this does not cause any errors in the interpretation step for this particular application, it would have created problems when one started looking at the corresponding event duration. This is a crucial area in need of betterment. As discussed earlier, building a more accurate normalizer will take care of errors caused during interpretation.

The previous chapter identified and discussed the strengths and weaknesses of the system in detail. To summarize, one has to first concentrate on the REs that form the basis of the chunker and classifier modules in order to improve the overall performance of the system. Clear delineation of class boundaries and identification of class members go a great way towards building a system that can handle conflict resolution and fuzziness well. Once these steps are in place, building an interpreter to suit the intended application becomes a simple and straightforward process.

I would like to re-emphasize that the application discussed here is just one useful manner of representing the required information. For example, if a person was interested only in studying patients suffering from *vomiting* and *diarrhea*, and looking at other symptoms that occur along with these, a time line representation may not really be necessary. Instead, one might divide the TNs based on these symptoms and search for keywords that indicate occurrences of other symptoms, say *started with fever, rash onset* etc, essentially redesigning the chunker and classifier to identify such expressions and interpret them based on the order of mention or time of occurrence or as required by the application. Thus, this work is intended to serve as a general framework that can be tweaked based on domain knowledge and the end goal.

A possible application that can be built on top of this system is to study the co-occurrence of symptoms around a particular time in a certain population as the ED data is made available so as to be able to identify disease outbreaks in very early stages. Such a system will essentially keep a count of the number of cases reported in a locality of patients suffering from same or related symptoms occurring within a particular timeframe of the visit, irrespective of the time of occurrence of the symptoms relative to each other and alert health officials when this count reaches a particular level when these co-occurring symptoms can be called to constitute an outbreak. However, to accomplish this, we need to extend this system to be able to identify all or at the least most possible TEs and make clear decisions while encountering fuzzy times.

A larger scale application of this research would be to collect all patient related information created across multiple visits to healthcare facilities (beginning with TNs and physician notes, including lab tests and results, prescribed medications if any, diagnoses and discharge summaries), remove all redundant information from this collection and arrange all events on a timeline, thus making it useful for anyone who would be looking at such patient records in the future, especially during the patient's subsequent visits, when the healthcare provider can easily gain an understanding of the patient's history without having to go through multiple documents or having to confirm specific details with the patient due to lack of clarity in the documents.

Irvine (2007), whose work served as the motivation for this research, identifies the design and development of a system capable of doing end-to-end processing and interpretation of TNs based on TN-TIES as the future scope. Though this work involved re-designing the TN-TIES modules to suit this application, the process was made easier by the availability of a source of reference. I hope I have created a similar reference point for future researchers working in this area.

Appendix A

Classes of REs used by chunker and classifier to identify TEs

Relative Day/Date/Time (reldt) Class:

- (since)?morning|evening|night|today|tonight|yesterday|yest|tomorrow[^a-z]
  Ex : severe body ache since morning
- [^a-z][0-9]+[ ]*(day|week|hour|wk|hr|night)+[s]*[^a-z](ago|later)[^a-z]
  Ex : visited grandpa-4 days ago
- (last|past|within|in)?[^a-z][0-9]+[ ]*(day|week|hour|wk|hr|night)+[s]*[^a-z]
  Ex : vomiting x 3 in past 5 hrs
- (since)?[^a-z](this|last|)?[^a-z][a-z]{3,6}day[^a-z]
  Ex : having rash since last Friday

Day/Date (daydt) Class:

- [^a-z][0-9]{1,2}/[0-9]{1,2}/[0-9]{4}[^a-z]
  Ex : last visit:2/01/2006
- [^a-z][0-9]{4}/[0-9]{1,2}/[0-9]{1,2}[^a-z]
  Ex : saw pmd on 2007/05/05
- (on)?[^a-z][a-z]{3,6}day[^a-z]
  Ex : chills on Wednesday
- mon[^t]|tue[s]?|wed(nes)*|thur[s]?|fri
  Ex : tues took tylenol
- on sat
  Ex : attended camp on sat.

Time (timdt) Class:

- (since)?[^a-z][\d]{4}[ ]?([a|p]+m)?(today|yesterday|tonight|morning|evening)+
  Ex : last meal 0800 yesterday
- [^a-z][0-9]{1,2}:[0-9]{2}[ ]*[a|p]+m (today|yesterday)?
  Ex : woke up with headache 6:00 am today
- (since)?[^a-z](this|last|yesterday|today)? (morning|evening|night|a[.]?m|p[.]?m |(after)?noon)+
  Ex : crying since this evening

Duration (reldur) Class:

- [^a-z](for)|(x)+[ ]?[about|approx]? [0-9]+ (day|week|hour|wk|hr|night)+[s]?
  Ex : fever for about 2 nights now

Appendix B

Classification Approach

```
tnsegclass=””
part of next tnsegment=””
previous tnsegment=””
prevtnte=””
while tnsegment:
    tnsegclass=”othr”
    if tnsegment belongs to any TE class:
        tnte = tnsegment
        if previous tnsegment exists:
            if part of next tnsegment exists:
                tnseg = previous tnsegment + part of next tnsegment
            assign tnseg to tnte
        assign tnsegclass for tnte
        prevtnte = tnte
    else:
        if next tnsegment exists:
            if next tnsegment belongs to any TE class and not same as
prevtnte:
                if tnsegment contains any ‘keywords’:
                    split tnsegment at keyword
                    assign all except last part of the split to part of
                                        next tnsegment
                else:
                    assign tnsegment to previous tnsegment
            else:
                assign tnsegment to previous tnsegment
        else:
            assign tnsegment to previous tnsegment
```

References

Ahn, D., Fissaha Adafre, S., & de Rijke, M. (2005b). Towards task-based temporal extraction and recognition. *Proceedings of the Dagstuhl Workshop on Annotating, Extracting, and Reasoning about Time and Events,* Dagstuhl, Germany.

Bodenreider, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research,* 32.

Bramsen, P., Deshpande, P., Lee, Y. & Barzily, R. (2006). Finding temporal order in discharge summaries. *AMIA Annual Symposium Proceedings,* 81-85.

Chowdhury, G. (2003). Natural Language Processing. *Annual Review of Information Science and Technology (ARIST), Volume 37,* 51-89.

Combi, C. & Shahar, Y. (1997). Temporal reasoning and temporal data maintenance in medicine: Issues and challenges. *Computers in Biology and Medicine,* 27, 5, 349-.

Currie, A.M., Cohan, J. & Zlatic, L. (2001). Information Retrieval of Electronic Medical Records. *Proceedings of the Second International Conference on Computational Linguistics and Intelligent Text Processing (CICLing-2001),* edited by Alexander Gelbukh, pp. 460-471.*(Lecture Notes in Computer Science - 2004).* Berlin: Springer-Verlag.

Day, D., McHenry, C., Kozierok, R., & Riek, L. (2004). Callisto: A configurable annotation workbench. *International Conference on Language Resources and Evaluation.*

Ferro, L., Gerber, L., Mani, I., Sundheim, B. and Wilson G. (2005) "TIDES 2005 Standard for the Annotation of Temporal Expressions" April 2005, Updated September 2005 -- *the September version updates all references to the ISO 8601 standard to incorporate the latest edition (*8601:2004, Third Edition*).*

Gaizauskas, R., Harkema, H., Hepple, M. & Setzer, A. (2006). Task-oriented extraction of temporal information: The case of clinical narratives. *Thirteenths International Symposium on Temporal Representation and Reasoning (TIME'06),* 188-195.

Haas, S.W., Travers, D.A., Waller, A., & Kramer-Duffield, J. (2007) What is an event?: Domain constraints for temporal analysis of chief complaints and triage notes. *Proceedings of the American Society for Information Science and Technology,* 44(1).

Harkema, H., Stezer, A., Gaizauskas, R. & Hepple, M (2005). Mining and Modelling Temporal Clinical Data. *Proceedings of the UK e-Science All Hands Meeting,* 2005:507-14.

Harkema, H., Dowling, J.N., Thornblade, T. & Chapman, W.W. (2010). Context: An Algorithm for Determining Negation, Experiencer, and Temporal Status from Clinical Reports. *Journal of Biomedical Informatics*, 42(5):839-51.

Irvine, A.K., Haas, S.W., & Sullivan, T. (2008). TN-TIES: A system for Extracting Temporal Information from emergency department triage notes. *AMIA Annual Symposium proceedings*, 328-32.

Johansson, R., Berglund, A., Danielsson, M., & Nugues, P. (2005). Automatic text-to-scene conversion in the traffic accident domain. *International Joint Conference on Artificial Intelligence, 19*, 1073-1078.

Kind, A.J.H. & Smith, M.A. (2008). Documentation of Mandated Discharge Summary Components in Transactions from Acute to Subacute Care. *Advances in Patient Safety: New Directions and Alternative Approaches. Vol. 2. Culture and Redesign,* 2:179-188.

Mani, I. and Wilson, G. (2000). Robust Temporal Processing of News. *In Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL'2000),* 69-76.

Mani, I., Wilson, G., Sundheim, B., and Ferro, L. (2001). Guidelines for Annotating Temporal Information. *In Proceedings of HLT 2001, First International Conference on Human Language Technology Research, J. Allan, ed.*, Morgan Kaufmann, San Francisco.

Osborne, J.D., Lin, S., Zhu, L., Kibbe, W.A. (2007). Mining biomedical data using MetaMap Transfer (MMtx) and the Unified Medical Language System (UMLS). *Methods In Molecular Biology*.

Pustejovsky, J., Castano, J., Ingria, R., Gaizauskas, R., Setzer, A., Saurí, R. & Setzer, A. (2003). TimeML: Robust Specification of Event and Temporal Expressions in Text. *AAAI Spring Symposium on New Directions in Question Answering.* Stanford, CA. 28-34.

Ruch, P., Baud, R. & Hilaro, M. (2002). Text Mining and Information Retrieval in Medical Records: an Inquiry into Automatic Spelling Correction. *Workshop on Natural Language Processing in Biomedical Applications*

Sullivan T, Irvine A, Haas SW. (2008). It's all relative: usage of relative temporal expressions in triage notes. *ASIS&T. Annual Meet 2008*, (AM08: Oct.; Columbus, OH).

Suominen, H., Lehtikunnas, T., Back, B., Karsten, H., Salakoski, T. & Salantera, S. (2007). Applying language technology to nursing documents: Pros and cons with a focus on ethics. *International Journal of Medical Informatics,* 76(S), S293-S301.

Travers D, Wu S, Scholer M, Westlake M, Waller A, McCalla A. (2007). Evaluation of a chief complaint processor for biosurveillance. Proceedings of the 2007 AMIA Symposium, 736-740

Waller, AE, Ising, AI, Deneka, L. North Carolina Biosurveillance System. 2008. In: Wiley Handbook for Science and Technology for Homeland Security. Hoboken, NJ: Wiley.

Yousefi, A., Mastouri, N. & Sartipi, K. (2009). Scenario-Oriented Information Extraction from Electronic Health Records. *IEEE International Symposium on Computer-Based Medical Systems,* 1-5

Zhou, L., Friedman, C., Parsons, S. & Hripcsak, G. (2005). System architecture for temporal information extraction, representation and reasoning in clinical narrative reports. *Proc AMIA Symposium,* 869-73.

Zhou, L., Melton, G.B., Parsons, S. & Hripcsak, G. (2006). A temporal constraint structure for extracting temporal information from clinical narrative. *Journal of Biomedical Informatics,* 39(4): 424-39.

Zhou, L. & Hripcsak, G. (2007). Temporal Reasoning with medical data – A review with emphasis on medical natural language processing. *Journal of Biomedical Informatics*, 40, 183-202.

Zhou, L., Parsons, S. & Hripcsak, G. (2008). The evaluation of a temporal reasoning system in processing clinical discharge summaries. *JAMIA*, 99-106.

Zhou W, Torvik V, Smalheiser N (2006). ADAM: another database of abbreviations in MEDLINE. *Bioinformatics*, 22:2813-2818.