Title
A Web-Based Annotation System for Lung Cancer Radiology Reports

Author
Chen, Xiang

Publication Date
2012

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

A Web-Based Annotation System for Lung Cancer Radiology Reports

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Biomedical Engineering

by

Xiang Chen

2012

ABSTRACT OF THE THESIS

A Web-Based Annotation System for

Lung Cancer Radiology Reports

by

Xiang Chen

Master of Science in Biomedical Engineering

University of California, Los Angeles, 2012

Professor Denise R Aberle, Chair

Registries provide a valuable tool for cancer research and for enabling decision support systems. However, populating cancer registries with information from medical records can be a tedious and bottlenecking process. This thesis presents an annotator system that automatically extracts data elements from lung cancer radiology reports to populate a lung cancer registry. Annotators systems such as this utilize natural language processing (NLP) techniques to locate concepts from a text source. A web-based framework that wraps the annotator system into a graphical user interface for researchers and clinicians is also discussed.

The thesis of Xiang Chen is approved.

Corey Wells Arnold

Alex Anh-Tuan Bui

Ricky Kiyotaka Taira

Denise R Aberle, Committee Chair

University of California, Los Angeles

2012

TABLE OF CONENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

In the medical world, patient reports are moving from clipboards to an electronic medium. Due to evolving reimbursement requirements, benefits to research, and increased efficiency in healthcare delivery, the number of electronic health records (EHRs) is growing quickly [5]. Currently, most EHR content are written by physicians in unstructured, narrative text. While being in an electronic format improves access to information [6], this kind of data is difficult to use for searching, summarization, decision-support, or statistical analysis [7]. Reviewing narrative text requires manual effort and is inefficient compared to structured, coded data that can be processed programmatically by a computer. Natural language processing (NLP) helps to alleviate this problem by extracting key concepts from narrative text and allows a user to organize these concepts into structured data fields. An application of NLP is in the creation and maintenance of a cancer registry, in which data entry is currently done manually by reviewing patient reports. A functioning and up-to-date cancer registry can be a valuable tool in cancer research and future applications in decision support systems. Automating the process of data entry may expedite data acquisition rate and increase data availability for research. However, average computer users are not familiar with the technical details of NLP and thus may not have access to existing NLP tools or the required knowledge to apply them appropriately. This project addresses these issues by first creating an NLP annotator aimed at extracting data elements from lung cancer radiology reports. This aspect of the project is demonstrated on a corpus of radiology reports for the development of a lung cancer registry. Next, a web application has been constructed to wrap the lung cancer NLP system and any other NLP systems that are built

on the same framework. This application provides average computer users the chance to upload medical text reports for annotation work without needing any prior technical knowledge of the underlying NLP systems.

CHAPTER 2

Background

In medicine today, free-text remains the most common form of data representation. Despite the benefits that structured medical reports can bring to information retrieval (IR) such as easy-to-interpret data and quicker data access, medical professionals generally still use narrative text because it is the most convenient and efficient way to express concepts and events. The application of NLP techniques has the potential to make accessible data that are embedded within medical reports without disrupting a physician's workflow. Making NLP techniques accessible to the lay user is beneficial towards obtaining more data but has remained a hurdle due to the technical knowledge required to implement the requisite algorithms. The ensuing sections describe relevant research done in the area of NLP and how is has been applied in the medical domain.

2.1 NLP Methods

Current NLP methods used for information extraction are primarily divided into two categories: statistical and rule-based. Statistical methods extract information based on probabilities acquired from manual annotation of a set of training reports. Some common statistical models used in NLP systems include hidden Markov models, Bayesian networks, and support vector machines. Similarly, rule-based methods also require manual effort in the creation of hard-coded rules based on the review of large numbers of training reports. The main difference between the two categories is that in statistical methods, the resulting statistical model is used to extract information, while the hard-coded rules are used in rule-based methods. In applications, the situation will dictate which method should be chosen for a particular task. For more general IR

tasks in which the system is meant to be applied to a wide variety of medical reports, statistical methods are often chosen. These systems are capable of making inferences during situations in which a test case may not have been seen during the training process. For narrower tasks where a specific type of medical report is the primary focus, rule-based approaches are commonly utilized. The few patterns that are exhibited by the target reports can be captured almost entirely with a combination of rules. Most NLP systems also have several components in common including a tokenizer, sentence-boundary detector, part-of-speech tagger, morphological analyzer, shallow parser, named entity recognizer, and negation detector. Each of these components is explained in more detail during the discussion of existing NLP systems.

2.2 Information Retrieval and Text Mining

A primary motivation for many NLP efforts is IR and text mining. IR is an area of study centered on searching for documents and information within documents, while text mining is focused on examining the relationship between specific kinds of information [8]. Current work may be categorized into several common themes including named entity recognition (NER), relationship extraction, and text classification. The goal of NER is to identify all instances of a specific semantic type within a collection of text. For instance, disease mentions or tumor characteristics in medical reports are all considered to be named entities. NLP systems that are dedicated to NER are discussed in section 2.3.

2.2.1 Relationship Extraction

Relationship extraction tasks aim to detect predetermined types of relationship between entities in a document. Relationship extraction systems often include a NER component to find specific types of entities for possible relationship matching. The types of relationships, unlike entities,

can be either very general such as parent-child is-a relationships, or very specific such as drug interactions. Attempted approaches for relationship extraction include manually and auto-generated template-based methods, statistical methods, and NLP-methods. Manually generated template-based methods are pattern-based and require manual inspection by domain experts to create the templates [9]. In contrast, patterns in automatic template methods are created through generalizing text patterns around concept pairs known to have the desired relationship [10]. Statistical methods detect relationships by finding concepts that are found with each other more often than would be predicted by random chance [11]. Lastly, NLP-methods decompose text through sentence parsing components such as part-of-speech tagging and shallow parsing to detect relationships based on structural features [12].

2.2.2 Text Classification

Text classification efforts are focused on identifying documents in a corpus that have certain characteristics of interest. An example of a text classification attempt is one from Lakhani et al. [13] Similar to the objectives of this project, this group developed text-mining algorithms to identify radiology reports that contain critical results using a rule-based technique. During development, domain experts classified a list of symptoms as critical results. These symptoms included tension or increasing/new large pneumothorax, acute pulmonary embolism, acute cholecystitis, acute appendicitis, ectopic pregnancy, scrotal torsion, unexplained free intraperitoneal air, new or increasing intracranial hemorrhage, and malpositioned tubes and lines. Their algorithm identified common words and phrases to find each critical result. Further refinement of the algorithm included the use of synonyms for locating keywords, proximity searching for confirming matches by checking the distance between keywords, and negation detection to exclude results that had been negated by certain negative modifiers. As most of the

results appeared in the “Impressions” section of a radiology report, narrowing the search to only that section showed further improvement in the algorithm’s accuracy. Except for one, all of the algorithms for each critical result displayed overall f-measure scores of greater than 90%. The authors concluded that for text classification tasks such as the rule-based approaches are still the most accurate and reliable methods over traditional statistical or machine-learning NLP methods.

2.3 NLP Systems

While individual institutions have developed NLP systems for IR and text-mining, most of these systems are mainly used for in-house purposes and are highly specialized. Examples of a few developed NLP systems for general medical purposes include the Medical Language Extraction and Encoding system (MedLEE) from Friedman et al. [14], the clinical Text Analyzsis and Knowledge Extraction System (cTAKES) from the Savova et al. [15], and the Medical Knowledge Analysis Tool (medKAT/P) from Coden et al. [16]. Each of the systems will now be discussed in more detail.

2.3.1 MedLEE

MedLEE is an NLP system that extracts information from medical reports and translates it to terms in a controlled vocabulary for automated decision-support system and natural language queries. MedLEE is composed of different modules, each tasked with processing and transforming the text in accordance with a specific aspect of language; modules are executed sequentially until a final transformed text is obtained. The first of these components, the preprocessor, recognizes and categorizes words and phrases into an input format useable by subsequent steps of the system. For instance, the sentence *possible left ventricular hypertrophy* is transformed into: [possible,[left,ventricular,hypertrophy],]. Because *left ventricular*

hypertophy is bracketed, it was recognized as a phrase and will be treated as a single entity by the other components. Next, the parser uses grammar rules to identify and generate an intermediate text structure that includes primary findings and different types of accompanying modifiers. The compositional regularizer is then used to compose individual words into phrases by using a table of structural mappings. Finally, the encoder maps words and phrases into codes based on a table. If any of the initial parser efforts fail, the recovery component attempts to increase sensitivity by using alternative strategies to structure the text [17]. For several different applications, MedLEE is used as a base while extensions are designed around the system depending on the domain and goal of the application.

2.3.2 cTAKES

cTAKES is an NLP system primarily focused on retrieving medical named-entities from clinical documents. Similar to other NLP systems, the cTAKES system is a modular system that contains a pipeline of components combining both rule-based and machine learning techniques. Each of the components is executed in sequence with each step incrementally contributing to the overall annotation dataset. The current components within cTAKES include a sentence boundary detector, tokenizer, normalizer, part-of-speech (POS) tagger, shallow parser, and named entity recognition (NER) annotator that contains status and negation annotators.

The sentence detector is an extension of OpenNLP's sentence detector tool that predicts whether a period, question mark, or exclamation mark is the end of a sentence [18]. Next, the tokenizer first splits sentences based on spaces and punctuations and then merges the resulting tokens into date, fraction, measurement, person title, range, Roman numeral, and time annotations. After the text is tokenized, the normalizer finds a representation for each word that is

normalized according to alphabetic case, inflection, spelling variants, punctuation, genitive markers, stop words, diacritics, symbols, and ligatures [19]. This step makes it possible to later map multiple mentions of the same word to the same concept even if they do not have the same string representation in the input text. The POS tagger and shallow parser are again from the OpenNLP package and are responsible for tagging words and phrases with part-of-speech labels. Lastly, the NER component implements a dictionary lookup algorithm within a noun-phrase lookup window to extract named entities. Each entry in the dictionary has a designated word that is used in the lookup process. When a possible match is found, the rest of the entry is checked for in the lookup window, and if the entire entry matches, a named entity annotation is created with any associated Unified Medical Language System (UMLS) concept [15]. Besides these main components, negation detection is also implemented in the form of the NegEx algorithm. NegEx is a pattern-based approach that finds words and phrases indicating negation near an extracted named entity [20]. Similar to MedLEE, the named entity results from cTAKES are used as a foundation for more specialized NLP tasks.

2.3.3 MedKAT/P

Lastly, MedKAT/P is a rule-based system designed to automatically instantiate a knowledge representation model from free-text pathology reports. This knowledge model is known as the Cancer Disease Knowledge Representation Model (CDKRM) and consists of a group of interlinked nodes. The nodes in the model are also referred to as classes, consisting of multiple attributes. Higher level classes may contain attributes that are nested classes, and these classes are known as container classes. Classes that have only values as their attributes are known as leaf classes. To automatically populate CDKRM, an annotation pipeline is developed that is broken down into several components. The first component, ingestion, takes in the complete

report and divides it into sections based on its structure. The next component is comprised of several general NLP techniques that include a tokenizer, sentence detector, part-of-speech tagger, and shallow parser. After text in the identified sections are tokenized and tagged, they are passed through the concept identification component. This component maps textual mentions of medical concepts to an International Classification of Diseases for Oncology (ICD-O) terminology list that is stored in a local Extensible Markup Language (XML) file format. The final step in the pipeline is to discover relationships between all the extracted concepts and populate the CDKRM. This step is done through a rule-based system that will generate the main container classes based on the sections that were initially extracted. It is assumed here that certain container classes will only appear in specific sections of a medical report. Because MedKAT/P is a rule-based system, it is designed around the structure of reports from the Mayo Clinic and does not perform nearly as well on reports from other institutions.

2.3.4 UIMA

Both the cTAKES and MedKAT/P NLP systems are built upon IBM's Unstructured Information Management Architecture (UIMA) framework. Software systems built on top of UIMA are designed to analyze large volumes of unstructured information in order to extract knowledge relevant to the user [1]. UIMA breaks applications down into components and each component provides metadata via XML descriptor files to the framework. The framework is then responsible for controlling the data that flows between the components (Figure 2.1). For instance, the tokenizer and POS tagger are a few of the components controlled by the UIMA framework in the cTAKES system. The components of an UIMA system do the actual work of analyzing the unstructured data, and they are the entry point for developers looking to create their own annotators. MedKAT/P and cTAKES are examples of UIMA software systems that

accept plain text, but other UIMA applications are also capable of analyzing images and audio files.

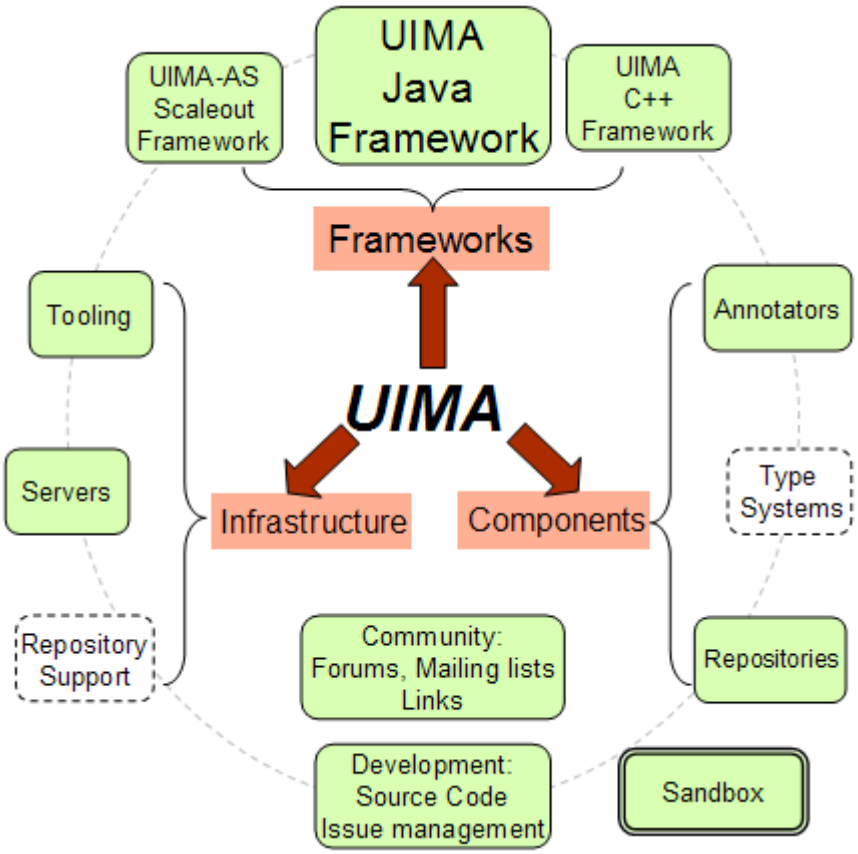

Figure 2.1: The UIMA Framework [1]

2.4 Web-Based Annotation Systems

While the abovementioned NLP tools are useful, their accessibility is only limited so far to researchers who are familiar with how they work. These tools are not easy to set up and use, so users who would like to utilize them in their work but lack the technical background are unable to do so by themselves. Web-based NLP systems are a way to increase user accessibility and several attempts are described below.

2.4.1 The Open Biomedical Annotator

One attempt at creating a more accessible annotation tool is the Open Biomedical Annotator (OBA) from Stanford University. The OBA is an ontology-based web service that annotates public datasets with biomedical ontology concepts [3]. The motivations behind OBA are similar to that of this project and can be summed as the following:

- Annotations often need to be done manually by domain experts or the authors of the data, and these tasks can be a very time and resource intensive.
- The number of available biomedical ontologies to use is large and they often change and overlap with each other. Moreover, these ontologies are also not always in the same formats or have APIs to allow users to access them.
- Users may not know the structure of the ontologies or have the technical background to use them for annotation purposes.
- Users often do not have the patience for making annotations without an immediate reward.

To overcome these issues, the OBA web service processes raw text by tagging it with relevant biomedical ontology concepts and returning to users the resulting annotations (Figure 2.2). OBA's workflow is broken down into two main steps. The first step feeds free-text into a concept recognition tool utilizing a dictionary containing a list of ontology concepts. This list of ontology concepts is comprised of multiple biomedical ontologies, and depending on what kind of text the user is interested in annotating, different ontologies will be chosen for the dictionary. The resulting annotations, known as direct annotations, are used as input for the second step of

the web service, the semantic expansion components. A few examples of these components include:

- An is-a transitive closure component that takes a direct annotation and traverses the parent-child hierarchy of the ontology to create new annotations that include the parent concepts.
- A semantic distance component that uses the idea of concept similarity to create annotations that include related concepts.
- An ontology-mapping component that will create new annotations based on existing mappings between different ontologies.

These components are just a few examples of the available semantic expansion components, and when the service is deployed, users are given options to customize the components depending on their needs. Once all annotations have been created, the results are returned to the user in the form of text, tab delimited, XML, or Web Ontology Language (OWL) files [3].

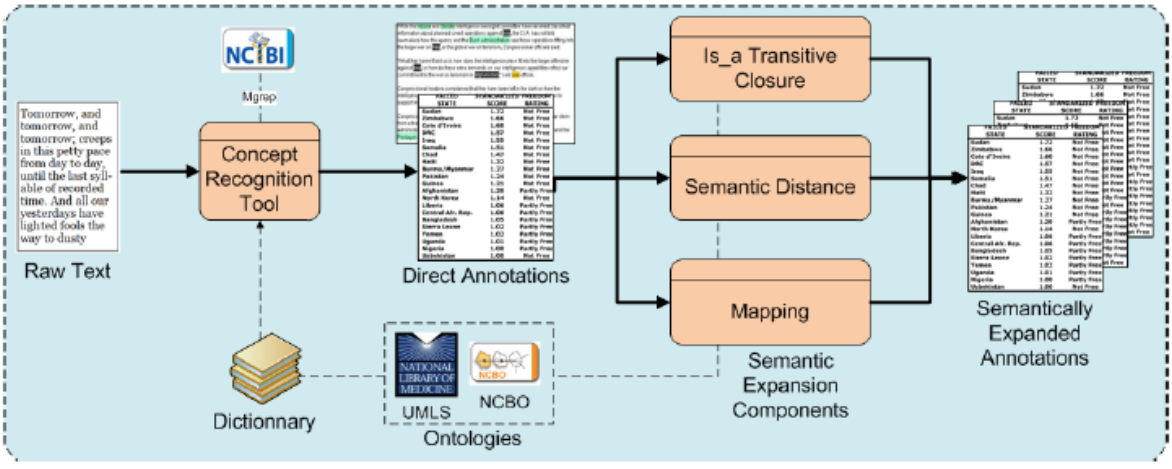

Figure 2.2: An overview of the OBA tool from Stanford [3]

2.4.2 CONANN

Another implementation of an online annotation system is the CONANN biomedical concept annotator from Drexel University [4]. The design of CONANN aims at achieving faster annotation times per phrase while still maintaining the same levels of accuracy as other biomedical annotation systems. CONANN incorporates an incremental filtering approach based around phrases to find the best-matching biomedical concept to a source phrase. Before any of the filters are applied, a candidate list of phrases is first generated from the UMLS that will represent the possible mapping matches to the source phrase. The subsequent incremental filtering system is divided into two approaches, coverage and coherence, and each successive filter becomes more computationally complex than the last (Figure 2.3).

The coverage filter measures the overlap of common words between candidate phrases and the source phrase. A score calculated by summing the inverse phrase frequency (IPF) values of all of the words in a phrase is given to each candidate and source phrase. A word's IPF is calculated with the formula, $\frac{N}{n_i}$, where N is the number of UMLS phrases and n_i is the number of phrases word i appears in the UMLS. All phrases that possess a score higher than a given threshold are passed on to the next component. As a compliment to the coverage filter, the coherence filter is designed to look at a phrase's word order rather than frequency. Instead of using IPF values, coherence is measured using skip-bigrams [21]. Skip-bigrams are all the possible word pairs that can be generated from the words of a given phrase. A score based on the amount of overlap between a candidate phrase and the source phrase's skip-bigram list is then generated for all candidate phrases. Similar to the coverage filter, all the phrases that have scored higher than a predetermined threshold are passed on to the next component. Once both

filters have been applied and a list of final candidate phrases has been gathered, the final concept mapping component maps all candidate phrases to UMLS concepts. The candidate phrases are then grouped according to their UMLS concept unique identifiers (CUIs), and the concept that has the most phrases grouped under it will be returned as the mapped concept for the given source phrase.

The online aspect of CONANN not only gives more people access to the annotation system, but also the advantage of supporting texts that are unknown to the system ahead of time. Users can submit any form of free text regardless of format. Furthermore, the system is capable of changing its concept resources (such as the UMLS and National Cancer Institute (NCI) Thesaurus). Because these concept resources are constantly expanding and different resources will be better suited for different types of users, a system needs to be dynamic in terms of its dictionary.

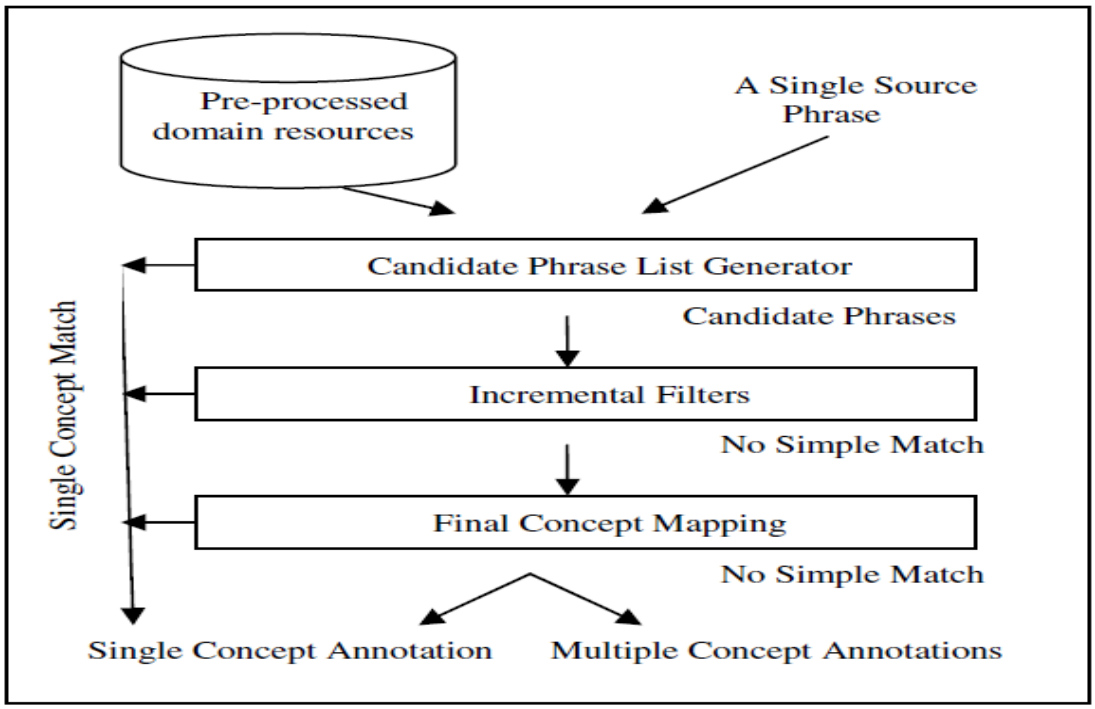

Figure 2.3: The CONANN biomedical concept annotator [4]

2.5 Cancer Registries

An area that can benefit from NLP efforts is data entry for a cancer registry. Cancer registries are information systems designed for the collection, management, and analysis of data on cancer patients. A well-maintained registry is a valuable research tool for those interested in the etiology, diagnosis, and treatment of cancer [22]. In the past, cancer registries were mostly used retrospectively to document care that had already been given [23]. However, more and more registries are now being used in a proactive fashion to influence individual cancer patient care. In one instance, the Commission on Cancer (CoC) utilized their cancer database, the National Cancer Data Base (NCDB), to create a set of care guidelines for several approved institutions. These guidelines, known as the Cancer Program Practice Profile Reports (CP^3R), were computed directly from the data reported by various cancer registries around the nation to the NCDB. A few examples of these guidelines included decisions to administer radiation therapy and certain drugs if the right criteria are met. A subsequent study that looked at the compliance rates of three different institutions utilizing registry-generated guidelines showed that rates across all guidelines were all moderately high. The lowest average compliance rate for a guideline was 68.2% while the highest was at 85.6% [23].

While the compliance rates were encouraging, there were still various administration issues that resulted in rate discrepancies between the institutions. One specific area of concern was the methods of data collection. When cancer registries were first established, all reports were done on paper. Accumulating reports for data collection became a very time-consuming process and thus were not usually done until months after the actual date of treatment [23]. As more institutions moved towards electronic medical records, relevant data became more abundant and readily available. However, most of these data were still recorded in the form of free text. So

while data had become easier to access, it did not necessarily result in faster data entry. The time that staff members used to retrieve reports were replaced by the increased amount of available data and the time it took to process all of them. The current method of manual review of patient reports for registry data entry is time-consuming and can lead to high administrative and staff costs. Automating this process can help reduce the time between treatment and data entry—leading to more relevant data that will improve the proactive benefits of a cancer registry.

CHAPTER 3

LungNLP

This thesis project is divided into two parts. The first part is based on developing an NLP system to extract key concepts from lung cancer radiology reports, while the second part centers on implementing a web application to provide access to this system. The lung NLP system portion of the project will first be discussed below.

3.1 The Lung Cancer NLP Annotator

Similar to cTAKES and medKAT/P, the lung cancer NLP annotator is constructed on top of the UIMA framework. A rule-based approach is chosen for the task because the scope of the application resides only within the lung cancer domain. With the expert knowledge available to identify data elements in a training set of reports and the structure that exists in some of the reports, a rule-based approach will be more reliable than a statistical-based approach.

3.1.1 The Data Elements

Prior to developing the annotation system, a domain expert created a tentative list of key data elements that are commonly expected to be found in a lung cancer radiology report. These elements are the data elements that will be used to populate a lung cancer registry and are listed in Appendix A. The focus of this work is to build an annotator capable of extracting diagnostic imaging elements. These diagnostic imaging elements include characteristics about a primary finding and any satellite nodules that are also discovered. The type of radiology reports that were found to include the most amount of diagnostic information was the Computed Tomography (CT) lung biopsy reports. These documents reported on the results of a lung biopsy

performed on a targeted mass and any other incidental findings. The information is conveyed in typical sections that include clinical history, findings, techniques, diagnostic considerations, and conclusions.

3.1.2 The Annotator

The lung cancer NLP portion of the project first required an inspection of the radiology reports that the system was to extract data from. As the system is rule-based, this inspection provides insight into the structure of the reports and what kind of rules should be applied to it. The first feature that all reports have in common is that they are composed of sections. The most common and informative sections include findings, diagnostic considerations, and conclusions. The types of information that are found in each section are also consistent. For instance, the characteristics of a tumor are usually found in the findings section while tumor staging information is found in the conclusions section. Therefore, the first step of the NLP pipeline is to create rules to detect the various sections and create annotation entries for each. The rules are based around looking for keywords that represent the headers of each section and punctuation and spacing patterns that signal the start of a new section. Examples of rules include looking for the words “conclusion” or “impressions” to signify the possibility of a conclusion section. Then after locating a possible match, formatting rules such as checking for a colon after the keyword and a blank line preceding the keyword are used to confirm the section match. A full list of the regular expression rules used in the annotator can be found in Table 3.1.

As different information resides in each section, it is inefficient to use only one annotator with one set of rules on all the sections. Thus separate annotators with distinct rule sets were created for each of the different sections. From all the inspected reports, there were two distinct

types of reports found, semi-structured and unstructured. In the semi-structured reports, information found in individual sections was often displayed in a structured manner. For instance, in most findings sections, tumor characteristics are listed in the format of "characteristic type: description" on individual lines. The types of tumor characteristics found are also consistent in each semi-structured report. In the unstructured reports, the information is written mostly in narrative text. While the types of information that are found such as which tumor characteristics is largely predictable in the more structured reports, the narrative reports often did not provide as much information or always include the same type of information from report to report. Therefore, to ensure the best results possible for information retrieval, different approaches are used for the semi-structured reports and the narrative reports.

For the semi-structured reports, an approach similar to that of the section detectors is used in retrieving the embedded information. Because information is listed in the format of "characteristic type: description," rules are based on detecting the characteristic type mentions that appear across reports. The expected characteristics to be found in most findings sections include primary finding, lesion laterality, lesion lung sublocation, lesion axial diameter, lesion perpendicular axial diameter, lesion consistency, lesion margins, lesion airway proximity, and lesion atelectasis. Once the mentions are located, formatting rules are used to confirm the presence of the characteristics. These rules include colons that are found after the type mentions and checks to ensure that the new characteristics always begin at the start of a new line. Examples in application of the rules are shown in Table 3.1. Prior to creating the characteristic annotations, each of the annotated sections is fed into the cTAKES pipeline to identify any existing medical concepts. After the characteristic annotations are created, they are cross

checked with the identified medical concepts to attach any possible associated medical CUIs to the annotations.

For the reports that are written in narrative text, the approach is based around locating terms that are often used in the description for the characteristic types. During the report inspection stage, a list of terms that frequently appeared as characteristic descriptions were accumulated along with their respective types. As the reports are first processed through the cTAKES pipeline, other NLP annotations such as part of speech tags, noun phrases, and negation flags are also created. The first step in the subsequent pipeline is to find any instances of a term from the list accumulated earlier. If a match is found, then the noun phrase that includes the matched term is inspected for additional matching criteria. These matching criteria are different between each of the characteristics. For instance, in looking for a lesion laterality element, the key words that are used are “left” and “right.” Normally, these words are very common in a report and not all of their mentions indicate lesion laterality. Therefore, in order to confirm a match, the containing noun phrase will be checked for the presence of a lesion lung sublocation such as “upper lobe” because the two characteristics are often mentioned in the same phrase such as “left upper lobe.” Once the characteristic annotations are created, they are cross checked with any identified medical concepts similar to the other approach in order to attach possible associated medical CUIs to the annotations.

Prior to creating “primary findings” annotations, candidate annotations are checked for possible negation. This check is implemented in the form of the NegEx algorithm from Chapman et al. [20]. The algorithm begins by taking the candidate entity and its encompassing sentence as initial inputs. Next, the sentence is parsed through to find all negation terms using three terms lists that include:

- Pseudo negation terms—terms that look like negation terms but do not negate the candidate entity.
- Pre-condition negation terms—negation terms that occur before the entity they are negating.
- Post-condition negation terms—negation terms that occur after the entity they are negating.

When the first non-pseudo negation term is found, one of two regular expression rules is used to determine the scope of the negation. If the term is a pre-condition negation term, the rule <negation term> <any number of words or punctuation> <termination term|end of sentence|another negation term> is used. Then if the term is a post-condition negation term, the rule <indexed term> <five words or medical phrases> <negation term> is used. This algorithm is then repeated for all detected non-pseudo negation terms in the sentence [24].

In sections other than findings, most information is conveyed in the form of free text. The information that is consistently found in the diagnostic considerations and conclusions sections includes:

- Tumor T status—describes the size of the tumor whether it has invaded nearby tissue
- Tumor M status—describes the distant spread of cancer from one body part to another
- Tumor N status—describes regional lymph nodes that are involved
- Tumor stage—the overall staging characterization of the tumor on a 0-IV scale

Similar to the approach taken in detecting data elements in unstructured findings sections, the first step involves creating rules to locate terms that are frequently used to describe the

abovementioned characteristics. Once the terms are located, the surrounding text is then checked for additional criteria. For instance, the tumor TNM staging terms are usually mentioned in close proximity to one another. Therefore when one is found, the presence of the other statuses in the same sentence is checked to confirm the match. A diagram of the entire annotation process can be found in Figure 3.1.

Regular Expression	Explanation	Example
\n[\t\r]+\n\s+(Section Header)	Finds the start of sections based on frequently-occurring section headers	Findings: … Impressions: …
\s+M\.?\s*D	Finds the end of a medical report before the document signature begins	Impressions: … I, John Doe, M.D. has approved ...
\n\s+(Feature Header)	Finds the start of a feature subsection in semi-structured reports	Findings: Primary Finding: … Features: … Airway Proximity: …
\d{1,2}(\.\d)?\s*x	Detects the longest axial diameter of a lesion when both axial and perpendicular axial measurements are given	3.4 x 6.5 cm
\d{1,2}(\.\d)?	Detects the longest perpendicular axial diameter of a lesion when both axial and perpendicular axial measurements are given	3.4 x 6.5 cm
\s+(Concept)s?	Finds data elements in the text	… needle was advanced into the targeted mass within the left lower lobe …
(?i)M[0-1X][A-B]?	Detects mentions of a tumor M status from the TNM staging system	Clinical TNM classification T2A,N0,MX
(?i) +M+	Detects mentions of a tumor M status from the TNM staging system	M Status currently unknown
(?i)N ?[0-4]	Detects mentions of a tumor N status from the TNM staging system	Clinical TNM classification T2A,N0,MX
(?i)T[0-4][A-B]?	Detects mentions of a tumor T status from the TNM staging system	Clinical TNM classification T2A,N0,MX
(I{1,3}|[1-3]) ?[aAbB]	Detects mentions of a cancer stage that is between 1 and 3	Stage 3A
IV|4	Detects mentions of a cancer stage that is 4	Stage IV

Table 3.1: A list of the regular expressions used throughout the annotator and their respective purposes

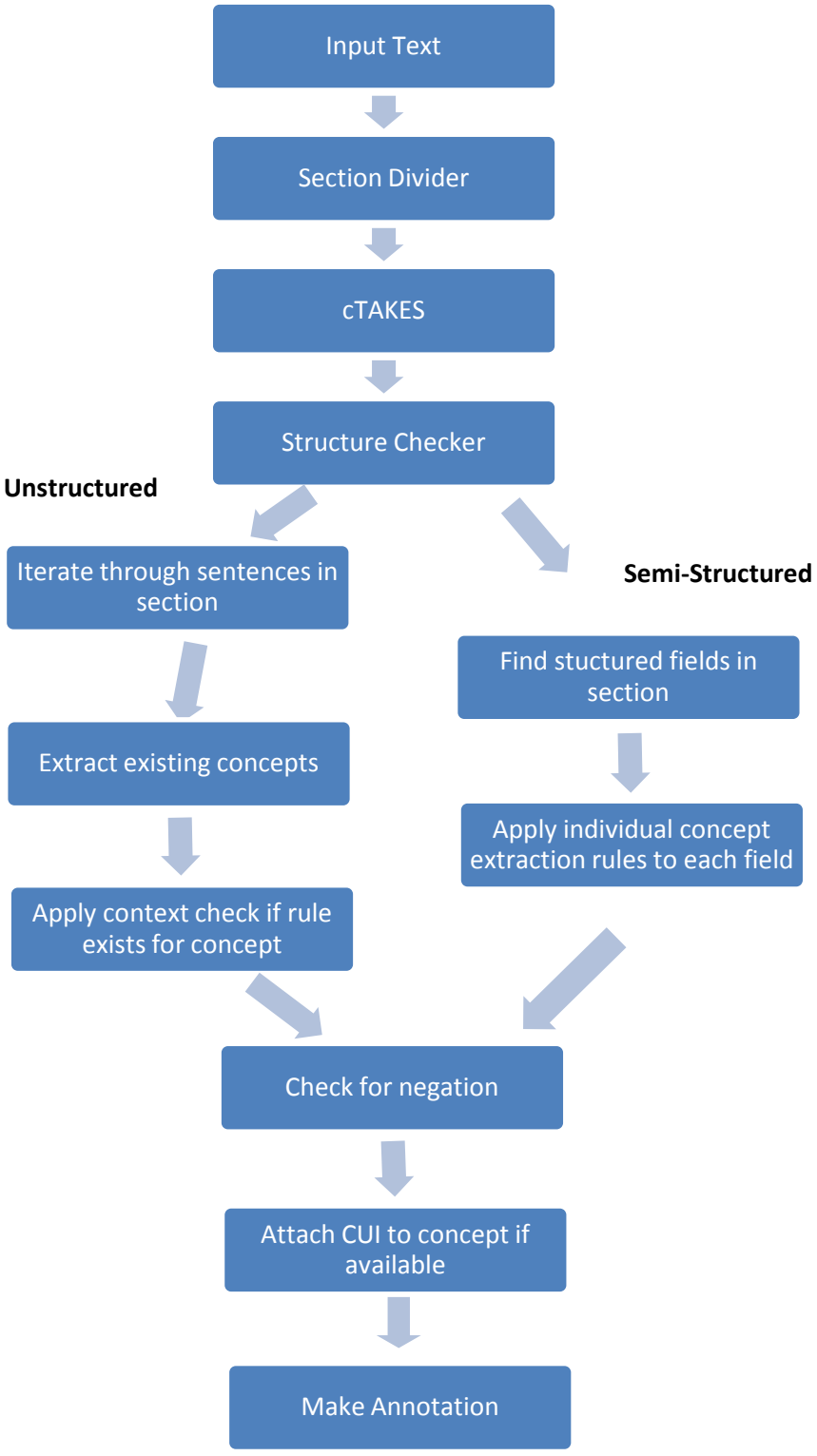

Figure 3.1: A diagram of the rule-based annotator process

CHAPTER 4

A Web-Based Annotation System for UIMA

4.1 Application Overview

While a UIMA-based annotator is a valuable tool for researchers, it is challenging for non-technical users, such as physicians, who would like to utilize the annotator for their research. The second part of this thesis centers on creating a web application to simplify the use of UIMA-based NLP systems such as cTAKES and the lung NLP system developed in this work. The web application allows users to upload reports that may then be annotated by any NLP system that is configured to run on the server. Users are also able to select which NLP system, cTAKES or the lung cancer system, to use for the annotations. Other options include which dictionaries to use for the medical concepts extraction step of the NLP pipeline. Once reports are annotated, users can either download the annotations in XML format or view them in an interactive viewer. The annotation viewer displays the uploaded text along with options to select each of the annotations and view their corresponding features. The web application also features an application programing interface (API) that allows other developers, who may not be familiar with UIMA-based annotators or may lack computing power, to integrate the developed tools with their code. The API adheres to Representational State Transfer (RESTful) standards for client-server communication and is constructed using Spring Model View Controller (MVC). The server is capable of receiving text uploads and returning XML-formatted results.

4.2 User Interface Focuses

The first portion of the web application, the user interface (Figure 4.1), focuses on allowing Java-based programs to be executed in a web-based environment. The main objective is to avoid requiring the user to understand how the underlying Java program is constructed and executed while maintaining the program's capabilities. Other requirements include an interface that resides on a singular page, fast annotation times, multiple ways to submit free-text reports, a convenient annotation viewer, and methods for users to export results

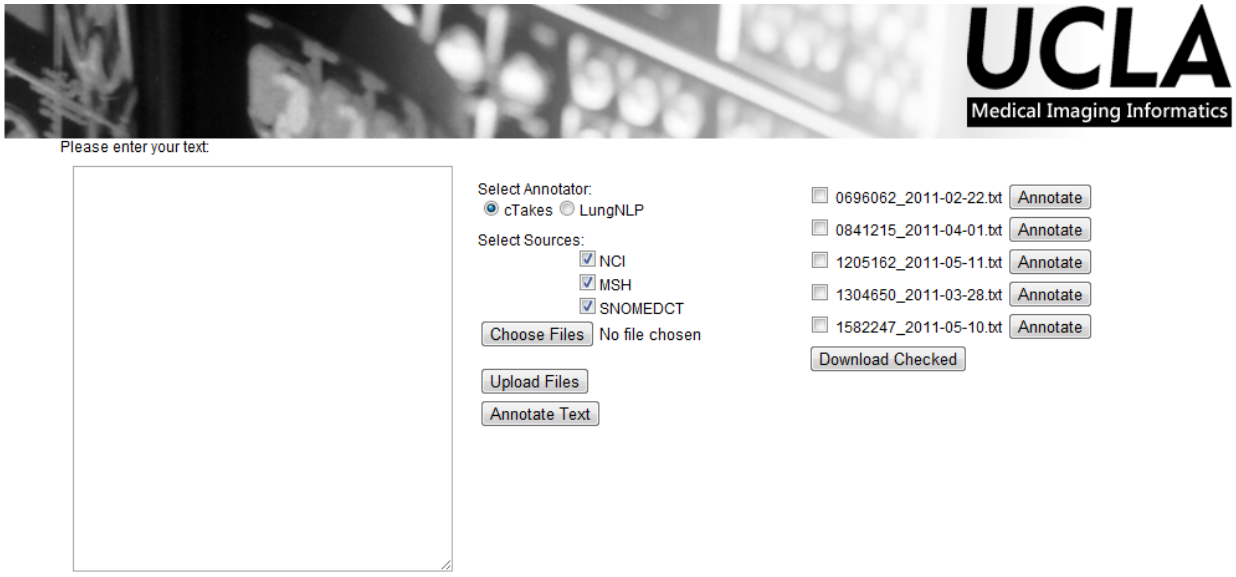

Figure 4.1: The main page of the web application displaying a text box on the right, user options in the middle, and a list of uploaded reports on the right.

4.2.1 Singular Page Interface

The first issue of confining the interface to a singular page is solved by letting user decisions result in changes dynamically on the current page rather than a new page. JavaScript and Flash are a few of the common ways to accomplish this task. For this project, JavaScript is chosen

over Flash because it is not as resource-intensive and thus can be utilized on a wider range of computers. Furthermore, as UIMA annotators are written in Java, using JavaScript provides a natural platform for communication between the server and the client. A Java library called Direct Web Remoting (DWR) provides the necessary tools to accomplish this task. The main feature of DWR is to generate JavaScript code to allow web browsers to securely call into the server's Java code as if it is running locally. To accomplish this, DWR is divided into two main parts:

- A Java Servlet running on the server that process requests and sends responses back to the browser.
- JavaScript running on the browser that sends requests based on user interaction and dynamically updates the webpage from server responses.

DWR works by dynamically generating JavaScript code based on Java classes and then utilizes asynchronous JavsScript and XML (AJAX) techniques to make it feel like the Java code is being executed on the browser. However, the Java code is actually being executed on the server with DWR marshaling the data back-and-forth between server and client [2]. To take advantage of DWR, a helper class in Java was created to interact with UIMA's API and handle data requests that are being marshaled in. This helper class is used by DWR to expose JavaScript-callable functions to the client side.

4.2.2 Fast Annotation Results

Achieving fast annotation results is related to the way UIMA annotators are executed. When running an UIMA annotator, an XML file is first read to load the annotator into memory. This file, the descriptor, contains information such as which components are included in the annotator,

in what order will the components be executed, and what annotations will be outputted. Once the annotator is loaded into memory, documents are then processed in a fast manner. The bottleneck in this process is located at the initial loading of the annotator, which can vary depending on the complexity of the annotator. For a local machine, this is not as big an issue, because the annotator is executed in its native environment and all code and libraries are loaded into memory as the initial step. However, for a web environment, the problem of keeping the annotator in memory becomes an issue. In a JavaSerer Pages (JSP) implementation, Java code embedded within a web page is executed when the page is first loaded, but any Java variables created are freed once the page is generated. In this scenario, users can submit documents via the webpage, load the annotator, and then process the submitted documents. After processing the documents, results are returned to the user, but the annotator is freed from memory. As such, if the user decides to upload more documents to annotate after the initial batch, the annotator must be loaded again, resulting in delays. Ideally, the annotator should be loaded into memory upon opening the web page and stay there until the user ends the session. Similar to the last issue, DWR is also capable of providing a solution to this problem. When DWR is first initiated, an instance of the Java class that DWR exposes to the client side is created on the server (Figure 4.2). In a customizable setting, the developer can specify how long this instance stays active on the server side. Options for this setting include for the duration of the user session or just for the duration of a request. For the purpose of this project, the session duration is the ideal solution to allow for fast annotation results. When a user first loads the webpage, the session begins and DWR calls to instantiate the annotator object on the server side. The annotator and any associated Java classes then stay loaded in memory and process any text reports that DWR marshals in from the client. When the user is finished and closes the webpage, the session ends,

and the annotator is freed from the server's memory. A side benefit of this method is that different users will also be distinguished from each other and have their own sessions with a distinctive instance of the annotator. While it may be even faster to have just one instance of the annotator loaded on the server for all users, so each new user will not have to wait for the initial loading period, UIMA's API only allows for one document to be processed by the annotator at any given time. Thus if only one annotator instance is active for all users, larger delays will occur once multiple users are trying to use the webpage at the same time and the list of documents to annotate becomes a large queue. As the user base increases, the memory strain of having an annotator session loaded in memory for each user can become an issue. At that point, the tradeoff of the time saved from having one annotator for all users versus one annotator per user must be analyzed to determine the proper solution.

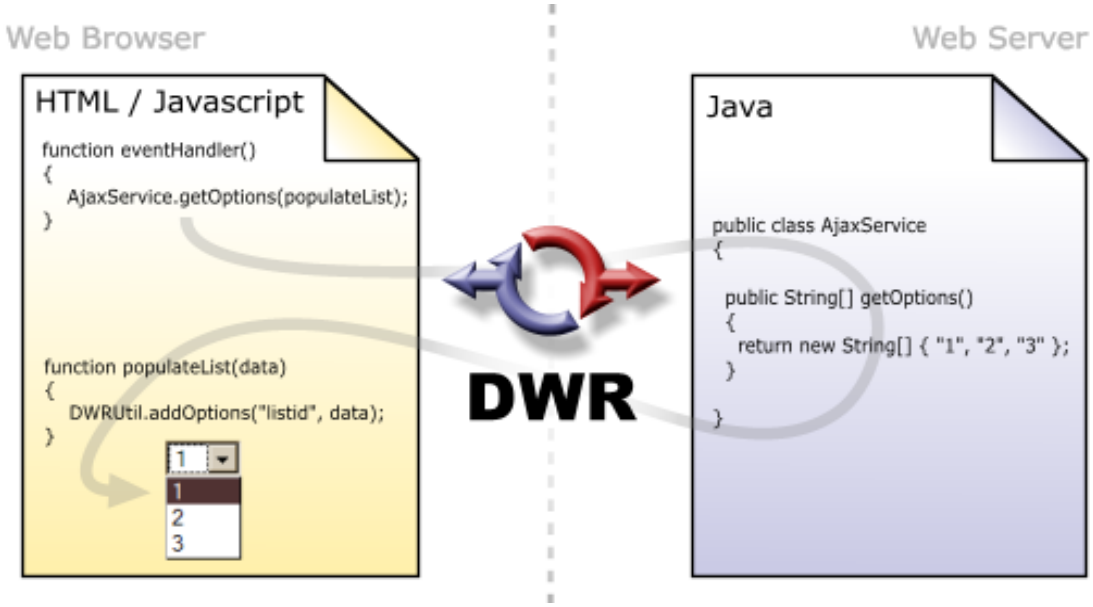

Figure 4.2: An example of how DWR facilitates communication between Java and JavaScript [2]

4.2.3 Data Input

The next focus of the web application is based around giving users multiple options on how they can submit text to be annotated. The first option is a text box that allows users to write their own or copy and paste text into the box to be submitted. While this method is straightforward and the simplest to use, pasting in any text that may contain a specific structure may result in the structure not being interpreted accurately by the annotator. Because of formatting limitations in an HTML text box, not all formatting is read correctly once it has been pasted into the text box. This issue can potentially lead to inaccurate annotation results, because some annotation systems are dependent on the formatting of the text. The second method of text submission allows users to directly upload one or multiple reports in text format to the server via an HTML input field. In HTML5, selecting and uploading multiple reports at the same time has become much easier and no longer relies on JavaScript packages.

4.2.4 Annotation Viewer

Once reports are uploaded to the server, the user needs a viewer to adequately examine the annotation results. For this, an annotation viewer was created in the form of a JavaScript popup (Figure 4.3). When a user opens the viewer, the original page is darkened, and the viewer becomes the center of focus. When the user clicks anywhere outside of the viewer, the viewer closes and the user can continue working with the main application page. In the viewer itself, there are radio buttons positioned on top displaying all the resulting annotation types such as medical entities, dates, and drug mentions. Selecting any of the radio buttons will highlight the corresponding annotations in the original text that is displayed under the selection buttons. Hovering over any of the highlighted annotations displays another JavaScript popup with the

annotation's attributes. Under this system, only one annotation can be selected at any time. This limitation is a result of the way annotations are stored in the UIMA framework. For every annotation, the start and end positions of the annotated words or phrases in the original text are used to distinguish one annotation from another. When a user selects a certain annotation type on the web page, the server adds HTML span tags around each of the annotations for highlighting purposes and returns the modified text for display. The drawback to this method is that when the text is modified with span tags, the original start and end positions of an annotation will no longer be accurate. The calculations for determining the new start and end positions can become increasingly complicated when multiple annotation types are involved. Furthermore, multiple annotation types can have annotations that overlap each other in the text. When this is translated over to HTML for display, the span tags will be incapable of distinguishing which tag belongs to which annotation type even if span attributes are introduced. The result will appear as if one annotation is completely encapsulating the other, making the other annotation indistinguishable.

4.2.5 Data Output

Besides being able to simply view the annotation results, users also need methods to export the results. Native to UIMA, methods already exist in its API to export annotation results into XML format. XML is a widely used format in the research domain to transfer data and has the capabilities to represent the majority of features in UIMA annotations. To take advantage of this API, the user is given the option of either downloading individual XML representations of the annotation results, or all of the annotation results in one zip file. The XML files are created on the server through UIMA's API and then marshaled to the client by DWR. One limitation with the API is that it is unable to translate UIMA annotation attributes that are multi-layered and

multi-featured. The API is only capable of translating the basic attributes that are single-valued from UIMA annotations to XML format. For instance, while cTAKES is able to generate several useful annotations, the main purpose of using cTAKES is its medical concept extraction capabilities. However, the attribute that contains the UMLS CUI for each extracted concept is contained within another multi-layered attribute. Therefore, the translated XML file will not contain information about a medical concept annotation's UMLS CUI code. To remedy this, a secondary export option is implemented to export only the medical concepts and their corresponding CUIs. Also instead of exporting in XML format, CSV files are introduced as an option because it is easier to import into other applications.

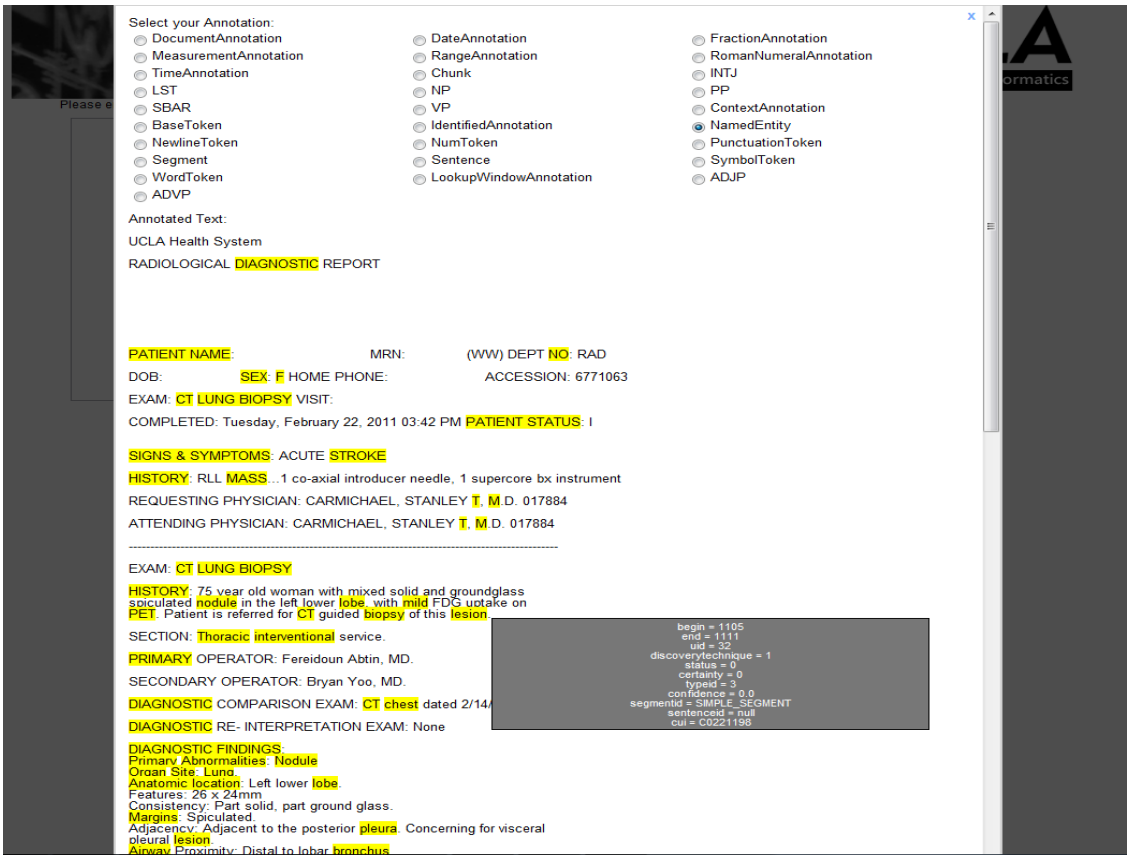

Figure 4.3: The annotation viewer with options to select different annotation types on top and mouseover display of attributes for each highlighted annotation.

4.2.6 Additional Features

For cTAKES and any other annotators that rely on cTAKES, a feature for the web application was created to allow users to choose between different dictionaries when the annotations are created. Because each dictionary is more relevant to a specific kind of text, not all dictionaries need to be used during runtime. Some of these different dictionaries include the Systemized Nomenclature of Medicine—Clinical Terms (SNOMED-CT), Medical Subject Headings (MeSH), and the NCI Thesaurus. To implement this feature, the user selection is first collected from the checkboxes on the web page and sent to the server along with the text to be annotated. Once on the server, the structured query language (SQL) query that cTAKES uses for dictionary lookup is modified for constraints on which dictionary(s) to use.

4.3 API Access

Other users who may be interested in UIMA annotators include other software developers. However, for most developers, working with a Graphical User Interface (GUI) is not the most efficient route. Some developers may directly want to take the resulting annotations and use them as input to other parts of a program. Having to go through a GUI for annotation results will slow down a program because the developer needs to be present to obtain the results manually. Two distinctive groups of developers include those that are not familiar with the UIMA package and those that are but lack the processing power locally to conduct large annotation projects. For these users, an API adhering to REST constraints was created for developers to bypass the GUI and obtain results from UIMA annotators programmatically.

The first step in creating this API is to create a package on the client side that developers can use and interact with instead of having to deal directly with the REST services. This

package deals with all protocols involved in calling REST services, so developers do not need technical knowledge in interacting with RESTful web services. For users who are not familiar with UIMA, the client package has methods to take in text reports as inputs and return XML or CSV files as the result. The service is also capable of returning single annotations such as medical entities or drug mentions. For the other users who are familiar with UIMA but prefer the processing done on another system, a UIMA specific Java object, the JCas, is returned as the result. The JCas is a Java class used in UIMA annotators that contains information about the input text and all resulting annotations. Users familiar with UIMA are able to iterate through annotations via the JCas and perform further analysis.

On the server side, a web service adhering to REST standards was built utilizing Spring's Web MVC framework. Because this is a RESTful service, all entities the server works with will be stateless. Any classes created during a request call by the client will be deleted from memory once the request is over and data has been returned. Therefore, all results will be managed by the client-side package. However, as the client package does not require the user to have knowledge of UIMA, it does not have any dependencies on UIMA libraries and cannot return individual annotations to the user. Full annotation results from the web service are returned to the client in the form of serialized Java objects. When the user requests to see an individual annotation from a report, the serialized object is passed back to the server, becomes deserialized into a JCas object, and the desired annotation is returned back to the client in XML format.

CHAPTER 5

Results

5.1 Results

116 total CT lung biopsy radiology reports from UCLA were used as a training set during the development of the NLP system. Of these 116 reports, 18 were in a semi-structured format that had certain groups of information laid out in a more consistent and predictive manner. The remaining 98 reports conveyed information in the more traditional narrative text format and had little to no structure outside of the individual sections. The annotator's regular expression rules were tested and refined over these 116 training reports. A separate set of 25 with four in semi-structured format and 21 in unstructured format were set aside for testing. Using this test set, a gold standard was created by manually labeling data elements under the supervision of a domain expert. From the semi-structured reports, 30 data elements were manually labeled. Of these 30 annotations, 28 were correctly identified by the NLP system, resulting in a recall of 0.96 and precision of one. From 21 unstructured reports, 91 data elements were discovered during manual annotation. A total of 61 data elements were identified by the NLP system with 60 of them being correct—resulting in a recall of 0.66 and precision of 0.98. The total recall and precision for all of the reports combined were 0.73 and 0.98, respectively (Table 5.1). The results are further broken down into individual data element types and are listed in Table 5.2.

	Semi-structured (N=4)	Unstructured (N=21)	Total
Total elements	30	91	121
Total detected elements	28	61	89
Correctly detected elements	28	60	88
Precision	1	0.98	0.98
Recall	0.93	0.66	0.73

Table 5.1: Results for annotation of both semi-structured and unstructured reports

	Total Elements	Detected Elements	Correct Elements	Precision	Recall
Lesion Consistency	5	5	5	1	1
Lesion Margins	4	4	4	1	1
Longest Perpendicular Axial Diameter	17	2	2	1	0.12
Longest Axial Diameter	18	3	3	1	0.17
Lesion Atelectasis	2	2	2	1	1
Lesion Airway Proximity	2	2	2	1	1
Primary Finding	25	26	25	0.96	1
Lesion Lung Sublocation	23	23	23	1	1
Lesion Laterality	25	22	22	1	0.88
Lung Cancer Stage	0	0	0	NA	NA
N Status	0	0	0	NA	NA
M Status	0	0	0	NA	NA
T Status	0	0	0	NA	NA

Table 5.2: Annotation results broken down into data element types. Although rules were created to detect lung cancer stages and TNM statuses because of their prevalence in the training set, none were observed in the testing set.

For comparison, a conditional random field (CRF) classifier was created using the manual annotations in the training set. CRFs are a group of statistical modeling methods that, unlike ordinary classifiers, take context into account when predicting labels for sequential data. A CRF is defined as a graph whose nodes are divided into two disjoint sets: observed (X) and output (Y) variables. The conditional distribution $p(Y|X)$ is then represented by this model [25].

In a given application, three main steps represent the NLP process: model training, inference, and decoding. The model training step determines the conditional distributions between a given output variable Y and its associated features functions that are obtained from a corpus of training data. The feature functions are measurements on the input sequence that partially determine the likelihood of each possible Y value. The model then assigns a weight to each feature and combines them to determine the probability of a value of Y. The inference step uses these features to determine the probability of a label sequence Y given the observed variable X. Lastly, the decoding step determines the most likely label sequence Y given X.

The CRF implemented used for this work is part of the machine learning for language toolkit (MALLET) from McCallum et al. [26]. Training files are structured in the format of "feature1 feature2 … featureN label" with feature1 corresponding to a token from the source text. Each subsequent token in the text is placed on the next new line with its own features. The features used to train the CRF consisted of part of speech, the word's capitalization, and a UMLS CUI, if available. These features are obtained from the cTAKES annotator. The labels given were determined by manual annotation, with a total of 13 distinct labels assigned. Test sets are structured in the same format as the train sets, except the label is withheld. A tenfold cross-validation was performed on the complete set of 141 available reports with each round consisting of 90% of the reports for training and 10% for testing. The results were broken down by the type of data element and averaged over the ten rounds of cross-validation. For comparison, the CRF was also applied to versions of the original report in which only relevant sections of the report are used as input. Extracting the relevant sections is done by the lung cancer annotation system's section divider, and the results are compared in Table 5.3.

	Complete Document		Sectioned Document	
	Precision	Recall	Precision	Recall
Lesion Consistency (N=42)	0.91±0.09	0.42±0.23	0.96±0.06	0.43±0.23
Lesion Margins (N=30)	0.96±0.05	0.60±0.23	0.98±0.03	0.56±0.24
Longest Perpendicular Axial Diameter (N=77)	0.75±0.14	0.63±0.16	0.77±0.12	0.59±0.15
Longest Axial Diameter (N=87)	0.80±0.10	0.66±0.11	0.81±0.07	0.69±0.10
Lesion Atelectasis (N=24)	0.88±0.20	0.77±0.21	0.88±0.20	0.77±0.21
Lesion Airway Proximity (N=28)	1.00±0.00	0.74±0.24	0.98±0.04	0.74±0.27
Primary Finding (N=141)	0.52±0.19	0.18±0.08	0.52±0.17	0.23±0.10
Lesion Lung Sublocation (N=139)	0.58±0.20	0.31±0.14	0.53±0.12	0.35±0.10
Lesion Laterality (N=141)	0.58±0.19	0.41±0.13	0.52±0.12	0.35±0.10
Lung Cancer Stage (N=14)	NA	0.00	0.17±0.17	0.06±0.08
N Status (N=16)	NA	0.00	0.25±0.25	0.04±0.06
M Status (N=15)	NA	0.00	0.25±0.25	0.07±0.11
T Status (N=16)	NA	0.00	0.25±0.25	0.02±0.04
Total	0.75±0.08	0.40±0.09	0.70±0.05	0.43±0.08

Table 5.3: Results for the CRF are broken down into label type where N is the total number of elements in each category and compared between the original documents and versions that only contain the relevant sections.

The label types that have the highest precision and recall rates for the CRF tagger are "Lesion Airway Proximity" and "Lesion Atelectasis", respectively for both types of documents. The label with the lowest precision is "Primary Finding" for the complete document and "Lung Cancer Stage" for the sectioned document. No samples in the complete documents are labeled by the CRF as a "Lung Cancer Stage" or any of the TNM statuses, resulting in the lowest recall rates of 0 and a lack of precision scores. Similarly, the T status recorded the lowest recall rate for the sectioned documents. All results from the regular-expression based annotator performed better than the CRF with the exception of the "Longest Perpendicular Axial Diameter" and "Longest Axial Diameter." Although the number of test samples is limited, results indicate the superiority of the knowledge-based regular expression annotator.

CHAPTER 6

Discussion and Conclusion

6.1 The Annotator

While not exemplified in the testing results due to the limited sample size, there are several areas of concern that appeared during the development process. One source of error in semi-structured reports is multiple data elements appearing for a given data type. For instance, several reports have inconclusive tumor TNM statuses and will report multiple statuses such as multiple M values. The algorithm currently only looks for one status mention per report, so any additional mentions are ignored. In the unstructured reports, the main sources of recall error are from data elements not being under an expected section and phrases divided onto separate lines. For example, while tumor sizes are most often found in findings sections, they occasionally appear in the techniques section or just in the middle of a report under no particular section. When phrases are separated onto different lines because of spacing constraints, they are no longer recognized as a single phrase and instead are considered as separate phrases. For key phrases such as “lower lobe” that are divided onto separate lines, the look-up process does not pick up the words as a phrase and misses the possible match. Furthermore, sentences and noun phrases are used as look-up windows for certain entities to confirm matches. When sentences and noun phrases are split due to spacing constraints instead of naturally, look-up windows become incomplete and a possible match may be incorrectly skipped. A potential solution to this problem is to preprocess reports and eliminate all newline characters. However, doing this would ruin the structure of the reports that most of the regular expression rules depend on. The precision errors occur mostly from mentions of nodules and masses that are not considered to be a primary finding. When a

primary finding nodule is mentioned in the same section along with several other satellite nodules, it becomes difficult to distinguish which one is the primary finding. If the nodule features are also presented along with each nodule, then it is even harder to determine which features belong to which nodule mention. The problem with accurately determining lookup windows also adds to the issue because context cannot be reliably used to separate out the different nodules.

With the limited amount of reports that are available, the lung NLP system is effective at extracting most data elements that are present in the reports. Regular expression rules show a greater recall and precision rate with semi-structured reports because the presence of data elements is more predictable when a structure is in place. Compared to free-text, creating rules for an information retrieval task becomes increasingly easier with increasing levels of report format. In a structured report, the format dictates what kind of information is placed in which sections. Because rules no longer need to be applied to the whole report and instead only to specific sections, they can be narrower and more refined to fit the specific tasks. Once the report format becomes more integrated into the UCLA system, more reports will be available to refine the rules and create an even more precise system.

6.2 The CRF Classifier

In comparison to the CRF classifier, the regular-expression based annotator performed noticeably better. This is partially due to the fact that there were only 141 reports available, and therefore the training algorithm was unable to learn the patterns of expression for each data element. This is especially true for several labels such as the tumor TNM statuses in which less than 20 cases each are available for training. In comparing the performance between the full and

sectioned reports, the sectioned reports have an overall greater average recall level, but lower average precision score. However, none of the scores are significantly different from one another given the standard deviations. Possible explanations for the existing discrepancies are that in the complete documents, there are more words that are provided to the classifier as non-labels to train on—leading to higher precision. Conversely, the sectioned reports limit the testing set to only sections of reports that are known to contain potential concepts, leading to a higher recall rate.

6.3 Structuring Reports vs. NLP Development

It is evident and not unexpected that information retrieval tasks perform better on structured reports compared to unstructured reports. Whether the method is rule-based or statistically-based, structured reports allow these methods to be narrower in scope and more accurate as a result. Furthermore, the amount of information found within a semi-structured report is also significantly greater than any unstructured report. Each semi-structured report averages 10.6 data elements while the unstructured reports averages 4.4 data elements per report. While NLP methods are constantly improving, changing medical formats and different standards across institutions have made it difficult for any specific NLP method to be effective in the general case. Although a method may perform well on one type of report, it can be ineffective on another type. Much research is focused on developing and refining NLP techniques; however developing methods that allow a radiologist to efficiently create structured data are equally important. The task of information retrieval relies not only on the method of retrieval, but also the information itself. Clinicians, researchers, and should increasingly investigate capture systems that standardize medical reports to enable more effective information extraction. Similar to the benefits that moving from paper to electronic health records have had on information retrieval,

creating more structure in EHRs will result in even further benefits. Ultimately, it is ideal to have a standard format across all institutions, but such advancements may not be feasible in the near future without overcoming major administrative hurdles. A more realistic approach is to support a more semi-structured format, such as that of the reports from UCLA.

6.4 Web-Based System Contributions

The main contributions of the web-based system are distributed as such:

- The web application gives researchers who may not be familiar with UIMA an opportunity to use UIMA-based annotators.
- The web system is capable of supporting any UIMA-based annotator and the user is given options to switch between annotators on the fly.
- Developers are also capable of accessing the web system programmatically through an API regardless of their technical knowledge of UIMA.

The project's web application demonstrates a beneficial approach to extending the usefulness of UIMA-based annotators to a larger audience. Currently, UIMA is largely an unknown entity to those outside of the research circle. Despite its potential and value of the results that it can produce, UIMA-based annotators can be overwhelming to setup and used properly for average computer users. With the web application, both the lung NLP system and cTAKES are wrapped into a user-friendly interface. Users need to only concern themselves with submitting text reports for annotation work and how to analyze the annotation results. By outputting annotator results into more common formats such as XMLs or CSVs, more people are able to utilize the processed annotations without any technical knowledge of UIMA.

Most current web-based annotators such as CONANN and the OBA are built in a way where the annotator pipeline and the web-based interface are packaged and constructed as one system. The resulting web system is often tailored closely and very specific with respect to the inputs and outputs of the annotator. Therefore, if developers want to create another annotator in a different domain, or if another party likes the web interface and wants to incorporate it into their annotator, the web system will likely need to be rebuilt because of how closely it is tailored to the original annotator. The web application presented in this project is built around a framework rather than a specific annotator system. All aspects of the system are created to be as general as possible, so that any annotator systems built on the framework will be compatible with the web application.

Initially, the web application was constructed with only the cTAKES annotator available for use. While a custom option specific to cTAKES was implemented in the system to allow users to select their choice of lookup dictionaries, all other aspects of the system were made to be compatible with other UIMA text annotators. When the lung NLP system was completed, it was incorporated into the web application with minimal effort. The only tasks that needed to be done were to specify the location of the new annotator's descriptor file and create an additional check box on the webpage to give users the option of selecting the annotator. Further testing was done by including MedKAT/P, another UIMA-based annotator system and one that was completely independent of cTAKES, into the web application. Similar to the lung NLP system, MedKAT/P had no compatibility issues during the process and was fully functional in the web environment.

For a sophisticated NLP system such as cTAKES, processing a large number of reports requires significant amounts of computing power. The API aspect of the web application gives developers the opportunity to conduct their annotation work on our servers. Returning users a

serialized Java object allows them to continue working with UIMA without the need for extra processing power. Two systems, one as a server and one as a client, are set up to test this concept. The client, installed with cTAKES, sends a report to the server for annotation work and obtains a serialized Java object in return. Because the client has cTAKES installed, the Java object is deserialized into a JCas and becomes usable as if the annotations were created locally.

6.5 Limitations

6.5.1 Lung NLP System

While the lung NLP system works well on medical reports from UCLA, the same results may not translate to report collections from other institutions. This is a known limitation of rule-based systems as every institution has their own ways of formatting and writing medical reports. Without administrative changes to enforce more standardized medical report formats, rule-based systems are limited to narrow scopes and can only be evaluated on a case-by-case basis.

6.5.2 Web Application

The main limitation of the web application lies in the need of a person with the necessary technical skills to implement new annotators into the system if the default annotators are not adequate for a desired task. While the process may seem trivial to someone with the appropriate background, researchers who are only interested in the annotations may still be unable to do so. For example, a researcher may come across a UIMA-based annotator on the web that he or she would like to use to process certain documents. Without any technical knowledge, the researcher will not be able to implement the annotator into the web application without external help. A person familiar with programming will need to implement the annotator for the researcher to use.

This need is largely unavoidable and argues for the creation of a university service to aid non-technical researchers to create complex document annotators.

6.6 Conclusions

At the rate at which EHRs are being integrated and used today, NLP techniques must continue to evolve to take advantage of the increased data influx. While researchers are primarily tasked with NLP work, to enable improved systems, equal focus should be given to methods for structuring medical reports at the time of creation. Subsequently, making improved NLP results accessible to a larger number of researchers requires interfaces that cater to individuals who may not have a technical background. The web application in this project has demonstrated a simple interface that has the capabilities of a complex NLP system, but requires none of the knowledge to operate it. Efforts such as this may be used to illustrate the potential benefits of NLP and may even lead to increased support from clinicians.

6.7 Future Work

An immediate goal for the lung NLP system is to extract patient demographic information. Alongside diagnostic data elements, demographic information is crucial to a cancer registry for research and proactive treatment purposes to influence individual patient care. Next, the usability of the lung NLP system can be expanded by extending into other radiology reports outside of the thoracic domain, or to include compatibility for lung cancer reports from other institutions. However, before any of these advancements can be made, more patient reports should be obtained for training and evaluation. With more data, it is possible that a statistical approach could be taken to expand the scope of the application. Although it is likely that rule-

based approaches will still play an important role and that a successful system will be a hybrid of rule-based and statistical techniques.

As mentioned earlier in the web application's limitations, the need of programmers to implement new annotators can be a hurdle for researchers, and future implementations will attempt to eliminate this need. Ideally, average users will be able to take any UIMA-based annotator they find and use it through the web application via an import feature. For more frequently used annotators, customized options similar to the dictionary option for cTAKES can also be implemented. Overall, future work will primarily focus on improving user experience by adding new features and revising old ones to be more user-friendly.

Appendix A

A complete list of the diagnostic imaging data elements that are to be used in a lung cancer registry. Only 13 of the elements listed below were consistently found throughout the available medical records and were thus the only ones to be extracted by the annotator. They are listed in bold in the following list.

Table: Diagnostic Imaging				
Key	**Field**	**Req**	**Values | Constraints**	**Description**
PK	ImagingFindings	R	Consecutive numbers	Unique identifier of Imaging Findings (01, 02, 03, 04, etc.)
FK	ImagingExamAccession	R	7 digits	Unique UCLA accession number for each imaging study entered into RIS
	PrimaryFinding	R	Lung Nodule Lung Mass Other, specify_____	What is the principal abnormal finding on this exam that may be related to lung cancer?
	LesionLaterality	R	Right Left Other	Indicates which side of the chest the lesion involves
	LesionLungSubLocation	R	UL ML LL Lingula Hilum Mediastinum	Indicates the anatomic location of the primary abnormality to greater specificity. This field appears *only if* **Primary Finding = Lung nodule or mass**
	Longest axial diameter	R	Digits (mm)	Record the longest diameter of the lung lesion at its axial equator (widest axial level on CT).
	Longest perpendicular axial diameter	R	Digits (mm)	Record the longer perpendicular diameter of lesion at its axial equator
	Series-Image	R	(S#-Im#)	Record the specific DICOM series # and

				image # on which the measures were made. Example (3-21)
	LesionConsistency	R	Solid Ground glass Part solid (mixed ground glass and solid) Cavitary Liquefaction necrosis	Describes the attenuation (consistency) of the nodule?
	LesionMargins	R	Smooth Spiculated Lobulated	
	LesionPriorComparison	R	Yes No	
	LesionComparisonExamDate	C	YYYY-MM-DD	
	LesionEvolution	C	Stable in size and consistency Possible growth Definite growth Indeterminate growth Possible change in consistency (attenuation) Definite change in consistency Indeterminate change in consistency	Conditional field ONLY if prior images available for comparison. Record any change in nodule growth or consistency relative to prior exams.
	LesionAirwayProximity	R	Distal to lobar bronchus (T1) Lobar bronchus (T1) Main bronchus ≥ cm from carina (T2) Main bronchus < 2 cm from carina (T3) Invasion of carina or trachea (T4) Indeterminate	
	LesionInvasion	R	None Visceral pleura (includes invasion of adjacent lobe) = T2 Chest wall = T3 Diaphragm = T3 Mediastinal pleura = T3 Parietal pericardium = T3 Mediastinum = t4 Heart = T4 Trachea or carina = T4 Esophagus = T4 Vertebra = T4	
	LesionInvasionProbability	C	Low	What is the

			Possible Indeterminate Probable Definite	likelihood of invasion of the tumor into adjacent structures? For each potential site of invasion, list likelihood of invasion.
	LesionAtelectasis	R	None Obstructive atelectasis extends to hilum but involves less than lung = T2 Atelectasis entire lung = T3	
	SatelliteNodule	R	None Satellite nodule(s) same lobe = T3 Satellite nodule(s) ipsilateral lung = T4 Satellite nodule(s) contralateral lung = M1a Second primary lesion possible or probable	
	SatelliteNoduleLaterality	C	Right Left	
	SatelliteNoduleAnatomicLocation	C	UL ML LL Lingula	
	SatelliteNoduleSeriesImage	C	(S-I)	
	SatelliteMetastasisProbability	C	Low Possible Indeterminate Probable Definite Second primary lesion possible/probable	What is the likelihood that the satellite nodule represents malignancy?
	TStatus	R	T0 T1 T2 T3 T4 Unknown or Indeterminate	
	LymphNodeAbnormalities	R	None 1R = Right supraclavicular 1L = Left supraclavicular 2R = Right upper paratracheal 2L = Left upper paratracheal	List all lymph nodes affected on imaging exam. Include all that apply.

			3a = Prevascular 3b = Retrotracheal 4R = Right lower paratracheal 4L = Left lower paratracheal 5 = Subaortic 6 = Paraaortic 7 = Subcarinal 8R = Right paraesophageal 8L = Left paraesophageal 9R = right pulmonary ligament 9L = left pulmonary ligament 10R = Right hilar 10L = Left hilar 11R = Right interlobar 11L = Left interlobar 12R = Right lobar 12L = Left Lobar 13R = Right segmental 13L = Left segmental 14R = Right sub-segmental 14L = Left sub-segmental Indeterminate	
	LNInvolvementProbability	C	Low Possible Indeterminate Probable Definite	What is the likelihood that a specific LN represents metastatic disease? Probability should be assigned for each individual nodal region affected
	NStatus	R	N0 N1 N2 N3 Unknown or Indeterminate	
	NodalBurden	C	Single compartment Bulky single compartment (> 20 mm) Multiple compartment Indeterminate	
	Metastases	R	None Lung nodule(s) contralateral lung = M1a Pleural effusion = M1a Adrenal = M1b	List all metastases affected on imaging exam

			Brain = M1b Liver = M1b Osseous = M1b Other = M1b Indeterminate	
	MStatus	R	M0 M1a M1b Unknown or Indeterminate	
	OtherImagingFindings	R	None or none significant Mild Emphysema Moderate emphysema Severe emphysema Post-inflammatory scarring-non-specific Apical scarring UIP-type fibrosis NSIP-type fibrosis Other diffuse lung disease, specify___ Large airways disease Small airways disease Respiratory bronchiolitis | RBILD Aspiration-related disease Coronary calcific atherosclerosis Aortic | systemic calcific atherosclerosis Aortic aneurysm (thoracic) Aortic aneurysm (abdominal) Aortic dissection-chronic Pulmonary embolism-acute Pulmonary embolism-chronic Pulmonary hypertension (possible, probable) Goiter | Multi-nodular goiter Hiatal hernia Liver disease (unrelated to lung cancer) Renal disease (unrelated to lung cancer) Adrenal disease (unrelated to cancer)	Record all other notable pathology visible on imaging that is UNRELATED to lung cancer

			Etc, etc.	
	PrimaryDiagnosis	Y	Lung cancer-consider NSCLC Lung cancer-consider SCLC Lung cancer-consider carcinoid	
	LungCancerStage	Y	Not applicable Stage IA Stage IB Stage IIA Stage IIB Stage IIIA Stage IIIB Stage IV Unknown or indeterminate	
	AlternateLungCancerStage	C	Stage IA Stage IB Stage IIA Stage IIB Stage IIIA Stage IIIB Stage IV	

REFERENCES

[1] Apache UIMA. Available at uima.apache.org

[2] Direct Web Remoting. Available at http://directwebremoting.org/dwr/introduction/index.html

[3] Jonquet C, Shah N, Musen M. The Open Biomedical Annotator. Summit on Translational Bioinformatics l2009: 56-60.

[4] Reeve L, Han H, Cohen-Boulakia S, Tannen V. CONANN: An Online Biomedical Concept Annotator In: Data Integration in the Life Sciences: Springer Berlin / Heidelberg; 2007, p. 264-279.

[5] Schuemie MJ, Sen E, Jong GWt, Soest EMv, Sturkenboom MC, Kors JA. Automating classification of free-text electronic health records for epidemiological studies. Pharmacoepidemiology and Drug Safety l2012: 8.

[6] Corrigan JM, Donaldson MS, Kohn LT, Maguire Sk, Pike KC. Crossing the Quality Chasm: A New Health System for the 21st Century: National Academy Press; 2001.

[7] Meystre SM, Savova GK, Kipper-Schuler KC, Hurdle JF. Extracting information from textual documents in the eletronic health record: A review of recent research. IMIA Yearbook of Medical Informatics l2008;47: 128-144.

[8] Cohen AM, Hersh WR. A Survey of Current Work in Biomedical Text Mining. Briefings in Bioinformatics l2005;6: 57-71.

[9] Yu H, Hatzivassiloglou V, Friedman C, al. e. Automatic Extraction of Gene and Protein Synonyms from MEDLINE and Journal Articles. In: AMIA Symposium. San Antonio, TX.; 2002. p. 919-923.

[10] Yu H, Agichtein E. Extracting Synonymous Gene and Protein Terms from Biological Literature Bioinformatics l2003;19: i340-349.

[11] Lindsay RK, Gordon MD. Literature-based Discovery by Lexical Statistics. J. Amer. Soc. Information Sci. l1999;50: 574-587.

[12] Friedman C, Kra P, Yu H, al. e. GENIES: A Natural-language Processing System for the Extraction of Molecular Pathways from Journal Articles. Bioinformatics l2001;17: S74-82.

[13] Lakhani P, Kim W, Langlotz CP. Automated Detection of Critical Results in Radiology Reports. Journal of Digital Imaging l2011;25: 30-36.

[14] Friedman C, Johnson S, Forman B, Starren J. Architectural requirements for a multipurpose natural language processor in the clinical environment. Proc Annu Symp Comput Appl Med Care l1995: 147-151.

[15] Savova GK, Masanz JJ, Ogren PV, Zheng J, Sohn S, Kipper-Schuler KC, Chute CG. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evalutation and applications. Journal of the American Medical Informatics Association l2010;17: 507-513.

[16] Coden A, Savova G, Sominsky I, Tanenblatt M, Masanz J, Schuler K, Cooper J, Guan W, Groen PCd. Automatically extracting cancer disease characteristics from pathology reports into a Disease Knowledge Representation Model. Journal of Biomedical Informatics l2009;42: 937-949.

[17] Friedman C. A Broad-Coverage Natural Language Processing System. In: AMIA; 2000. p. 270-274.

[18] Apache OpenNLP. Available at http://opennlp.apache.org/index.html

[19] LVG User Guide. Available at http://www.lexsrv3.nlm.nih.gov/SPECIALIST/Projects/lvg/current/docs/userDoc/index.html.

[20] Chapman W, Bridewell W, Hanbury P, al e. A simple algorithm for identifying negated findings and diseases in discharge summaries. J Biomed Inform l2001;34: 301-310.

[21] Lin C-Y, Och FJ. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics. Barcelona, Spain: Association for Computational Linguistics; 2004.

[22] The Cancer Registry and the Registrar. In: Association NCR, editor. Alexandria, VA; 2012.

[23] Beatty JD, Adachi M, Bonham C, Atwood M, Potts MS, Hafterson JL, Aye RW. Utilization of Cancer Registry Data for Monitoring Quality of Care. The American Journal of Surgery l2011;201: 645-649.

[24] Negation Identification for Clinical Conditions. Available at http://code.google.com/p/negex/wiki/NegExAlgorithmDescription

[25] Lafferty J, McCallum A, Pereira F. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proc. 18th International Conf. on Machine Learning: Morgan Kaufmann; 2001. p. 282-289.

[26] MALLET: A Machine Learning for Language Toolkit. Available at http://mallet.cs.umass.edu