High Throughput Modularized NLP System for Clinical Text

Serguei Pakhomov
Mayo College of Medicine

Mayo Clinic
Rochester, MN, 55905
pakhomov@mayo.edu

James Buntrock
Division of Biomedical Informatics
Mayo Clinic
Rochester, MN, 55905
Buntrock@mayo.edu

Patrick Duffy
Division of Biomedical Informatics
Mayo Clinic
Rochester, MN, 55905
duffp@mayo.edu

Abstract

This paper presents the results of the development of a high throughput, real time modularized text analysis and information retrieval system that identifies clinically relevant entities in clinical notes, maps the entities to several standardized nomenclatures and makes them available for subsequent information retrieval and data mining. The performance of the system was validated on a small collection of 351 documents partitioned into 4 query topics and manually examined by 3 physicians and 3 nurse abstractors for relevance to the query topics. We find that simple key phrase searching results in 73% recall and 77% precision. A combination of NLP approaches to indexing improve the recall to 92%, while lowering the precision to 67%.

1 Introduction

Until recently the NLP systems developed for processing clinical texts have been narrowly focused on a specific type of document such as radiology reports [1], discharge summaries [2], medline abstracts [3], pathology reports [4]. In addition to being developed for a specific task, these systems tend to fairly monolithic in that their components have fairly strict dependencies on each other, which make plug-and-play functionality difficult. NLP researchers and systems developers in the field realize that modularized approaches are beneficial for component reuse and more rapid development and advancement of NLP technology. In addition to the issue of modularity, the NLP systems development efforts are starting to take scalability into account. The Mayo Clinic's repository of clinical notes contains over 16 million documents growing at the rate of 50K documents per week. The time and space required for processing these large amounts of data impose constraints on the complexity of NLP systems.

Another engineering challenge is to make the NLP systems work in real time. This is particularly important in a clinical environment for patient recruitment or patient identification for clinical research use cases. In order to satisfy this requirement, a text processing system has to interface with the Electronic Health Record (EHR) system in real time and process documents immediately after they become available electronically. All of these are non-trivial issues and are currently being addressed in the community. In this poster we present the design and architecture of a large-scale, highly modularized, real-time enabled text analysis system as well as experimental validation results.

2 System Description

Mayo Clinic and IBM have collaborated on a Text Analytics project as part of a strategic Life Sciences and Computational Biology partnership. The goal of the Text Analytics collaboration was to provide a text analysis system that would index and retrieve clinical documents at the Mayo Clinic.

The Text Analytics architecture leveraged existing interface feeds for clinical documents by routing them to the warehouse. A work manager was written using messaging queues to distribute work for text analysis for real-time and bulk processing (see Figure 1). Additional text analysis engines can be configured and added with appropriate hardware to increase document throughput of the system.

Proceedings of the ACL Interactive Poster and Demonstration Sessions,
pages 25–28, Ann Arbor, June 2005.

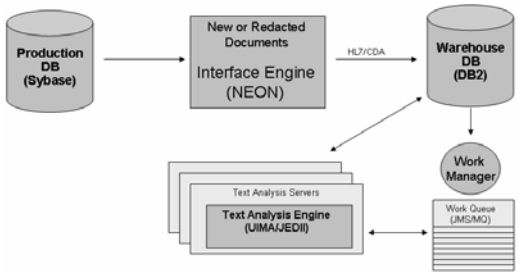

Figure 1- Text Analysis Process Flow

For deployment of text analysis engines we tested two configurations. During the development phase we used synchronous messaging using Apache Web Server with Tomcat/Axis. The Apache Web server provided a round robin mechanism to distributed SOAP requests for text analysis. This testing was deployed on a 20 CPU Beowulf cluster using AMD Athlon™ processors running Linux operating system. For production deployment we used Message Driven Beans (MDBs)using IBM Websphere Application Server™ (WAS) and IBM Websphere Message Queue™. The text engines were deployed on 2-CPU blade servers with 4Gb RAM. Each WAS instance had two MDBs with text analysis engines.

Work was distributed using message queues. Each text analysis engine was deployed to function independent of other engines. A total of 20 blade servers were configured for text processing. The average document throughput for each blade was 20 documents per minute.

The text analysis engine was designed by conceptually breaking up the task into granular functions that could be implemented as components to be assembled into a text processing system.

To implement the components we used an IBM AlphaWorks package called Unstructured Information Management Architecture (UIMA). UIMA is a software architecture that defines roles, interface, and communications of components for natural language processing. The four main UIMA services include: acquisition, unstructured information analysis, structured information access, and component discovery. For the Mayo project we used the first three services. The ability to customize annotator sequences was advantageous during the design process. Also, the ability to add annotators for specific dictionaries amounted only in minor work. Once annotators are written to conformance, UIMA provides pipeline development and permits the developer to quickly customize processing to a specific task. The final annotator layout is depicted in Figure 2.

The *context free tokenizer* is a finite state transducer that parses the document text into the smallest meaningful spans of text. A token is a set of characters that can be classified into one of these categories: word, punctuation, number, contraction, possessive, symbol without taking into account any additional context.

The *context sensitive spell corrector annotator* is used for automatic spell correction on word tokens. This annotator uses a combination of isolated-word and context-sensitive statistical approaches to rank the possible suggestions [5]. The suggestion with the highest ranking is stored as a feature of a token.

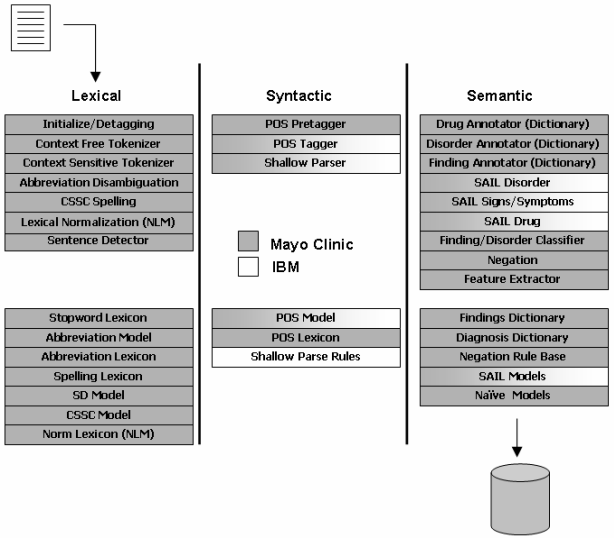

Figure 2 – Text Analysis Pipeline

The *lexical normalizer annotator* is applied only to words, possessives, and contractions. It generates a canonical form by using the National Library of Medicine UMLS Lexical Variant Generator (LVG) tool[1]. Apart from generating lexical variants and stemming optimized for the biomedical domain, it also generates a list of lemma entries with Penn Treebank tags as input for the POS tagger.

The *sentence detector annotator* parses the document text into sentences. The sentence detector is based on a Maximum Entropy classifier technology[2] and is trained to recognize sentence boundaries from hand annotated data.

[1] http://umlslex.nlm.nih.gov
[2] http://maxent.sourceforge.net/

The context dependent tokenizer uses context to detect complex tokens such as dates, times, and problem lists[3].

The *part of speech (POS) pre-tagger annotator* is intended to execute prior to the POS tagger annotator. The pre-tagger loads a list of words that are unambiguous with respect to POS and have predetermined Penn Treebank tags. Words in the document text are tagged with these predetermined tags. The POS tagger can ignore these words and focus on the remaining syntactically ambiguous words.

The *POS tagger annotator* attaches a part of speech tag to each token. The current version of the POS tagger is from IBM based on Hidden Markov models technology. This tagger has been trained on a combination of the Penn Treebank corpus of general English and a corpus of manually tagged clinical data developed at the Mayo Clinic [6], [7].

The *shallow parser annotator* makes higher level constructs at the phrase level. The Shallow Parser is from IBM. The shallow parser uses a set of rules operating on tokens and their part-of-speech category to identify linguistic phrases in the text such as noun phrases, verb phrases, and adjectival phrases.

The *dictionary named entity annotator* uses a set of enriched dictionaries (SNOMED-CT, MeSH, RxNorm and Mayo Synonym Clusters (MSC) to lookup named entities in the document text. These named entities include drugs, diagnoses, signs, and symptoms. The MSC database contains a set of clusters each consisting of diagnostic statements that are considered to be synonymous. Synonymy here is defined as two or more terms that have been manually classified to the same category in the Mayo Master Sheet repository, which contains over 20 million manually coded diagnostic statements. These diagnostic statements are used as entry terms for dictionary lookup. A set of Mayo compiled dictionaries are also used to detect abbreviations and hyphenated terms.

The *abbreviation disambiguation annotator* attempts to detect and expand abbreviations and acronyms based on Maximum Entropy classifiers trained on automatically generated data [8].

[3] Problem lists typically consist of numbered items in the Impression/Report/Plan section of the clinical notes

The *negation annotator* assigns a certainty attribute to each named entity with the exception of drugs. This annotator is based on a generalized version of Chapman's NegEx algorithm [9].

The *ML (Machine Learning) Named Entity annotator* is based on a Naïve Bayes classifier trained on a combination of the UMLS entry terms and the MCS where each diagnostic statement is represented as a bag-of-words and used as a training sample for generating a Naive Bayes classifier which assigns MCS id's to noun phrases identified in the text of clinical notes. The architecture of this component is given in Figure 3.

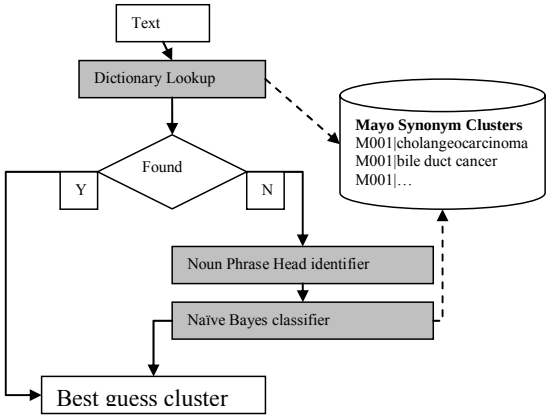

Figure 3. ML Named Entity Classifier

The text of a clinical note is first looked up in the MSC database using the *dictionary named entity annotator*. If a span of text matched something in the database, then the span is marked as a named entity annotation and the appropriate cluster ID is assigned to it. The portions of text where no match was found continue to be processed with a named entity identification algorithm that relies on the output of the *shallow parser annotator* to find noun phrases whose heads are on a list of nouns that exist in the MSC database as individual manually coded entries. For example, a noun phrase such as 'metastasized cholangiocarcinoma' will be identified as a named entity and subsequently automatically classified, but a noun phrase such as 'patient's father' will not.

3 Evaluation

The system performance was evaluated using a collection of 351 documents partitioned into 4 topics: pulmonary fibrosis, cholangiocarcinoma, diabetes mellitus and congestive heart failure. Each of

the topics contained approximately 90 documents that were manually examined by three nurse abstractors and three physicians. Each note was marked as either relevant or not relevant to a given topic. In order to establish the reliability of this test corpus, we used a standard weighted Kappa statistic [10]. The overall Kappa for the four topics were 0.59 for pulmonary fibrosis, 0.79 for cholangiocarcinoma, 0.79 for diabetes mellitus and 0.59 for congestive heart failure. We ran a set of queries for each of the 4 topics on the partition generated for that topic. Each query used the primary term that represented the topic. For example, for pulmonary fibrosis, only the term 'pulmonary fibrosis' was used while other closely related terms such as 'interstitial pneumonitis' were excluded. The baseline query was executed using the term as a key phrase on the original text of the documents. The rest of the queries were executed using the concept id's automatically generated for each primary term. On the back end, the text of the clinical notes was annotated with the Metamap program [3] for the UMLS concepts and the *ML Named Entity annotator* for MSC cluster id's. On the front end, the UMLS concept id's were generated via the UMLS Knowledge Server online and the MSC id's were generated using a combination of the same Naïve Bayes classifier and the same dictionary lookup mechanism as were used to annotate the clinical notes. We also tested a query that combined Metamap and MSC annotations and query parameters. Recall, precision and f-score (α=0.5) were calculated for each query. The results are summarized in Table 1.

	Precision	**Recall**	**F-score**
Key Phrase	0.77	0.73	0.749467
MSC cluster	0.67	0.89	0.764487
Metamap	0.71	0.84	0.769548
Metamap+MSC	0.67	**0.92**	0.775346

Table 1. Performance of different annotation methods.

The f-score results are fairly close for all methods; however, the recall is highest for the method that combines Metamap and the MSC methodology. This is particularly important for using this system in recruiting patients for epidemiological research for disease incidence or disease prevalence studies and clinical trials where recall is valued more than precision. A combination of Metamap and MSC annotations and queries produced the highest recall which shows that these systems are complementary. The modular design of our system makes it easy to incorporate complementary annotation systems like Metamap into the annotation process.

Acknowledgements

The authors wish to thank the Mayo Clinic Emeritus Staff Physicians and Nurse Abstractors who served as experts for this study. The authors also wish to thank Patrick Duffy for programming support and David Hodge for statistical analysis and interpretation.

References

1. Friedman, C., et al., *A general natural-language text processor for clinical radiology.* Journal of American Medical Informatics Association, 1994. **1**(2): p. 161-174.
2. Friedman, C. *Towards a Comprehensive Medical Language Processing System: Methods and Issues*. in *American Medical Informatics Association (AMIA)*. 1997.
3. Aronson, A. *Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program.* in *Proceedings of the 2001 AMIA Annual Symposium*. 2001. Washington, DC.
4. Mitchell, K. and R. Crowley. *GNegEx – Implementation and Evaluation of a Negation Tagger for the Shared Pathology Iinformatics Network*. in *Advancing Practice, Instruction and Innovation through Informatics (APIII)*. 2003.
5. Thompson-McInness, B., S. Pakhomov, and T. Pedersen. *Automating Spelling Correction Tools Using Bigram Statistics*. in *Medinfo Symposium*. 2004. San Francisco, CA, USA.
6. Coden, A., et al., *Domain-specific language models and lexicons for tagging.* In print in Journal of Biomedical Informatics, 2005.
7. Pakhomov, S., A. Coden, and C. Chute, *Developing a Corpus of Clinical Notes Manually Annotated for Part-of-Speech.* To appear in International Journal of Medical Informatics, 2005(Special Issue on Natural Language Processing in Biomedical Applications).
8. Pakhomov, S. *Semi-Supervised Maximum Entropy Based Approach to Acronym and Abbreviation Normalization in Medical Texts*. in *40th Meeting of the Association for Computational Linguistics (ACL 2002)*. 2002. Philadelohia, PA.
9. Chapman, W.W., et al. *Evaluation of Negation Phrases in Narrative Clinical Reports*. in *American Medical Informatics Association*. 2001. Washington, DC, USA.
10. Landis, J.R. and G.G. Koch, *The Measurement of Observer Agreement for Categorical Data.* Biometrics, 1977. **33**: p. 159-174.