Cengage Learning at TREC 2011 Medical Track

Benjamin King
University of Michigan
benjaminking@umich.edu

Lijun Wang
Wayne State University
ljwang@wayne.edu

Ivan Provalov
Cengage Learning
ivan.provalov@cengage.com

Jerry Zhou
Cengage Learning
jerry.zhou@cengage.com

ABSTRACT

This paper details Cengage Learning's submissions for this year's TREC medical track. The techniques we used fall roughly into two categories: information extraction and query expansion. From both the queries and the medical reports, we extracted limiting attributes, such as age, race, and gender, and labeled terms appearing in the Unified Medical Language System (UMLS). We also used three different techniques of query expansion: UMLS related terms, terms from a network built from UMLS, and terms from our medical reference encyclopedias. We submitted four different runs varying only in their methods of query expansion.

1. INTRODUCTION

Cengage Learning's approach to the retrieval task for the TREC 2011 medical track utilizes techniques for information extraction and query expansion in an aim to improve the precision of search results. We believe that one of the most effective ways to improve relevance when searching free text is to find structure in the text and extract information from it. The Unified Medical Language System (UMLS) produced by the National Library of Medicine is a large interconnected database of medical terms that we used to group and relate terms [Lindberg]. We also exploited the document structure in a number of other ways, such as removing irrelevant text, labeling negated and conditional text, and extracting indexable attributes.

In addition to exploiting the structure of the text, we also found that, as in other ad-hoc retrieval problems, the technique of query expansion was quite effective at improving the precision. We used two external sources for query expansion: UMLS and Cengage Learning's collection of medical reference encyclopedias.

In this paper, section 2 describes the background: the task description and related work. Section 3 describes methods for text processing, information extraction and for creating training data. Section 4 describes our indexing and retrieval methods. Section 5 lists the results of our work. Section 6 discusses our methods and their results and draws our conclusions.

2. BACKGROUND

2.1 Task Description

This year's TREC medical track was designed to emulate a common retrieval problem in the medical domain: searching medical records to try to find patients who may qualify for a research study. A set of approximately 100,000 de-identified medical records was provided by the University of Pittsburgh to be used as the corpus for the retrieval. The university also distributed a table that grouped reports together according to hospital visits, so that all the records for an individual patient's visit were grouped logically. The unit of retrieval for the task was the visit rather than the individual report.

TREC created 35 queries that were representative of researchers searching for subjects for a comparative effectiveness study. Participants were allowed to submit as many as four runs, containing up to 1,000 results for each of the 35 queries. These results were evaluated by several assessors who were experts on the subject matter. The primary evaluation measure for this track was mean average precision (MAP).

2.2 Related Work

Although this was the first year that TREC has hosted a medical track, there has been a great deal of research in medical information retrieval outside the conference. The work that we did was inspired by a number of earlier projects with similar aims.

One of the earliest, yet most comprehensive medical information retrieval systems was created by Carol Friedman et al. and was called MEDLEE. This system was capable of automatically extracting UMLS terms from a text and linking them with a UMLS concept, labeling the term as a finding, a procedure, a problem, or a treatment (among other labels). It could also extract a number of other attributes from each of these terms, such as the section in which they appeared, the term's certainty, and what adjectives or adverbs were used to describe the term [Friedman].

We were inspired to attempt to automatically detect negated and conditional text in our system after reading Wu et al.'s paper about a search engine for radiological reports that saw a significant increase in its retrieval precision after decreasing the relevance scores for findings that were negated or conditional [Wu]. The system itself used a modified version of NegEx, an algorithm for negation detection in medical reports, developed by Wendy Chapman et al. [Chapman].

Daniel T. Heinze et al. described a system capable of extracting a large number of features from free-text medical records, similar to attributes that medical coding experts might extract from a medical record for documentation and insurance purposes. Although they used very different techniques than our own (primarily vector analysis), this project inspired us to implement knowledge extraction in our own project.

In our reference terms extraction work, we developed an approach similar to Chinnakotla et al's [Chinnakotla] use of assisting languages as an external source for the pseudo relevance feedback. We applied these ideas by using an external authoritative source for the query terms expansion.

3. METHODS

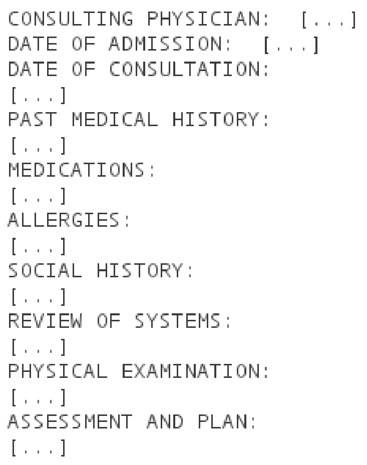
3.1 Training Data

As this was the first year of the TREC medical track, there were no judgments from prior years against which to develop and test our methods. To this end, we developed our own training dataset. Using the sample topics created by Swapna Abhyankar and posted to the TREC medical track Google group (see appendix for topics), we pooled the results of several different retrieval methods and evaluated approximately 190 reports for each of the topics. Except for the scoring returned by TREC, all the numbers reported in this paper were computed against these topics and judgments.

Our evaluators were not medical experts, though we observed that in many cases such expertise was not necessary to produce a correct evaluation of a report, as the writing style tended to be fairly straightforward and clear regarding symptoms, diagnoses, and treatments. We expect that our TREC evaluations will not be grossly different from the results we obtained in our own testing.

3.2 Text Processing Pipeline

Although the medical records in the University of Pittsburgh data set are largely formatted as unstructured free text, inspection of the records reveals that there tends to be quite a bit of exploitable structure, as most of the individual reports seem to roughly conform to one of several conventional formats. Nearly all of the reports are broken into sections with headings that correspond to the semantic function of the section's text. Figure 1 shows an example of the headings present in one report.



```
CONSULTING PHYSICIAN:  [...]
DATE OF ADMISSION:  [...]
DATE OF CONSULTATION:
[...]
PAST MEDICAL HISTORY:
[...]
MEDICATIONS:
[...]
ALLERGIES:
[...]
SOCIAL HISTORY:
[...]
REVIEW OF SYSTEMS:
[...]
PHYSICAL EXAMINATION:
[...]
ASSESSMENT AND PLAN:
[...]
```

Figure 1: The headings for the sections of a sample medical report.

3.2.1 Form Text Removal

The first step in the report processing pipeline was to remove what we call "form text" from the reports. Form text is any text that appears verbatim in a large number of reports (150+) but does not contribute any meaningful content. Much of the form text is either legal in nature, such as disclaimers or statements qualifying a physician's signature, or structural, such as text that might delimit areas of the report or be used for filing purposes. In addition to keeping extraneous terms out of the index, this step also helped ensure that the length normalization for each report was computed more accurately.

To locate potential form text, we automatically extracted a list of all ten-word phrases that appear at least 150 times in the corpus. From this list, we manually constructed 88 regular expressions to remove instances of form text from the individual reports. On average, each report had about 45 words, or approximately 11% of its text, removed.

3.2.2 Aggregating by Visit ID

The second step in the pipeline was to group reports together according to their corresponding visit. The original corpus was composed of individual reports, many of which would correspond to a single hospital visit. A table of this mapping was also provided by the University of Pittsburgh. The retrieval task was judged on returning relevant visit ID codes instead of report IDs, so one easy way to retrieve visit codes while avoiding duplicates and other problems associated with indexing reports was to simply restructure the corpus so that each of its documents represents an individual visit.

3.2.3 GATE

For the remaining steps in the pipeline, we used the GATE (General Architecture for Text Engineering) text processing tool [Hamish]. GATE concretizes the idea of a text processing pipeline where each document undergoes the same processing steps, one after another. The end result of this is a set of GATE documents, which are text documents containing annotations that mark segments of the text as having specific properties and/or features.

A basic version of the GATE pipeline might include four steps: XML parsing/corpus creation, tokenization/sentence splitting, part of speech tagging, and lemmatizing. We built our final pipeline on this framework, though many of the stages in the GATE pipeline—the tokenizer, the sentence splitter, and the part-of-speech tagger—were modified specifically to perform more accurately in the domain of medical reports.

3.2.3.1 XML Preprocessing

When GATE loads an XML corpus into memory, it first processes the XML and marks up a document's text with annotations for each of the tag names that are present in the original markup.

3.2.3.2 Tokenizing

As mentioned previously (and explained in more detail below), it was necessary to modify the sentence-splitting stage of the pipeline to better suit our purposes. Because the tokenizer and sentence splitter are wrapped into a single GATE class, it was also necessary for us to re-implement the tokenizer, even though we were content with the performance of the GATE tokenizer. Our new implementation used the Java BreakIterator class operating on the token level, which uses punctuation and whitespace as context clues for separating tokens. Qualitatively, this seemed to work about as well the GATE tokenizer and we did not experience any issues related to tokenization later in the pipeline.

3.2.3.3 Sentence Splitting

The GATE sentence splitter frequently made errors when sentences were terminated with new line characters rather than with correct punctuation. We substituted our own sentence splitter that was based again on the BreakIterator class, operating this time on the sentence level. The BreakIterator tended to make some of the same mistakes that the GATE splitter made, so we also made additional splits to its output based on the presence of newline characters, blank lines, section headers, and a few other domain-specific features. Having sentences correctly segmented was important for later stages in the pipeline, as part-of-speech tagging and lemmatizing consequently depended on the sentence boundaries being correctly marked.

3.2.3.4 ICD Code Conversion

All diagnostic ICD codes were converted to their descriptions during the parsing step of the GATE workflow. These codes were then indexed in a separate field for further use.

3.2.3.5 Part of Speech Tagging

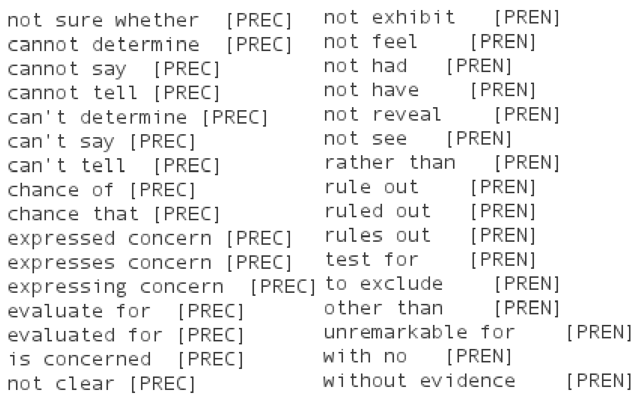
To tag parts of speech in the corpus, we used an unmodified version of the GATE ANNIE part of speech tagger. Even though we never explicitly used the part of speech that this stage provided in indexing the content, it was necessary for the GATE lemmatizer to know the correct tag in order to properly lemmatize the word.

3.2.3.6 Lemmatization

It was not uncommon to observe long passages of text written in all capital letters in the medical reports. The GATE part-of-speech tagger tended to tag most anything beginning with a capital letter as a proper noun. This hurt the performance of lemmatization later in the pipeline, as words in all-caps were incorrectly lemmatized because they carried the wrong part of speech. We modified the lemmatizer to lemmatize words that were written in all-caps and were tagged as NNP (proper noun) instead of skipping them, which is the usual behavior.

3.3 Negation and Uncertainty Detection

Following the techniques in the recent paper by Wu et al. on improving precision in radiology report retrieval by detecting negation and uncertainty [Wu], we implemented our own negation and uncertainty annotator based on the NegEx algorithm description [Chapman] and its open-source implementation.[1] We used 256 rules for detecting negation and 130 rules to detect uncertainty. Figure 2 shows a sample of some of these rules for both negation and uncertainty.



```
not sure whether  [PREC]       not exhibit   [PREN]
cannot determine  [PREC]       not feel    [PREN]
cannot say  [PREC]             not had   [PREN]
cannot tell [PREC]             not have    [PREN]
can't determine [PREC]         not reveal    [PREN]
can't say [PREC]               not see   [PREN]
can't tell  [PREC]             rather than   [PREN]
chance of [PREC]               rule out    [PREN]
chance that [PREC]             ruled out   [PREN]
expressed concern [PREC]       rules out   [PREN]
expresses concern [PREC]       test for    [PREN]
expressing concern  [PREC]     to exclude    [PREN]
evaluate for  [PREC]           other than    [PREN]
evaluated for [PREC]           unremarkable for    [PREN]
is concerned  [PREC]           with no   [PREN]
not clear [PREC]               without evidence    [PREN]
```

Figure 2: Examples of rules for conditionals (left) and negation (right). The text in brackets is functional and indicates the type of rule listed (see the NegEx documentation for more detail).

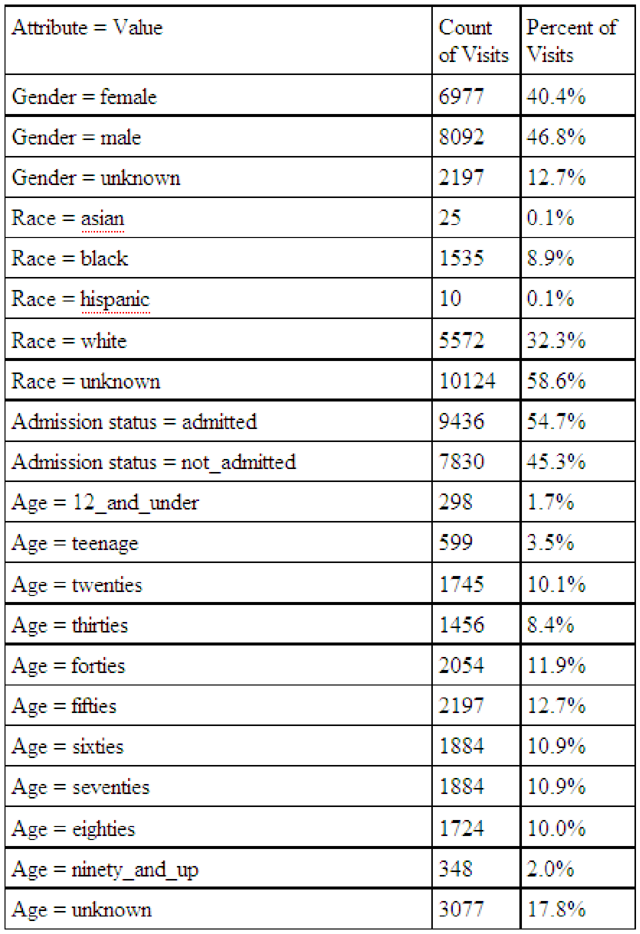
According to Mutalik, simple rule-based negation detection is capable of detecting more than 97% of negations. The results of our own ad-hoc evaluation also seemed to agree with this statistic, as the automated detection algorithm rarely produced either false negatives or false positives.

[1] http://code.google.com/p/negex

In our indexing stage, we excluded any token labeled as negative or uncertain from being indexed with the rest of the content. Quantitatively, we found this to improve the average precision by approximately 5%.

3.4 Information Extraction

We identified four major attributes of a patient and his/her visit that would be 1) easy to extract from the reports and 2) likely to be specified in the query. These were race, gender, admission status (admitted or not admitted to the hospital), and age. All four attributes were identified using regular expressions that were built manually according to patterns observed in the text of the visits. Table X lists the frequencies with which the different attributes were labeled in the visits.



Attribute = Value	Count of Visits	Percent of Visits
Gender = female	6977	40.4%
Gender = male	8092	46.8%
Gender = unknown	2197	12.7%
Race = asian	25	0.1%
Race = black	1535	8.9%
Race = hispanic	10	0.1%
Race = white	5572	32.3%
Race = unknown	10124	58.6%
Admission status = admitted	9436	54.7%
Admission status = not_admitted	7830	45.3%
Age = 12_and_under	298	1.7%
Age = teenage	599	3.5%
Age = twenties	1745	10.1%
Age = thirties	1456	8.4%
Age = forties	2054	11.9%
Age = fifties	2197	12.7%
Age = sixties	1884	10.9%
Age = seventies	1884	10.9%
Age = eighties	1724	10.0%
Age = ninety_and_up	348	2.0%
Age = unknown	3077	17.8%

Table 1: This table shows the frequencies with which each of the different labels for the four extracted attributes were applied.

3.5 UMLS Term Labeling

Building on the work of Nadkarni, we also identified terms from the Universal Medical Language System (UMLS) that appeared in the reports. To do so, we extracted a subset of UMLS term names from the UMLS database and used these to populate a trie, which was used to efficiently mark any instance of one of these UMLS terms in the report text.

In selecting a subset of UMLS term names and concept IDs, we excluded all non-English UMLS strings and strings with a length of more than 50 characters (since terms this long would not likely be specified in a query, and therefore would not be useful to store in the index). To combat the problem of ambiguous strings, any string that pointed to multiple concept IDs was changed to point only to the concept with the lowest numerical value, with the intuition that lower concept IDs tend to refer to the common usage of a word or phrase. Together these techniques yielded better performance empirically than any other scheme we tested for defining an effective subset of UMLS strings and concept IDs.

If a phrase that contained a number of UMLS strings was to appear in the report text, such as “paroxysmal atrial fibrillation,” it would be tagged (in this case) as containing five different UMLS concepts: “paroxysmal atrial fibrillation.” “atrial fibrillation,” “paroxysmal,” “atrial,” and “fibrillation,” all of which have different concept IDs and would be indexed as separate terms.

A UMLS term was considered to be negated [or uncertain] if it contained at least one negated [or uncertain] token, though in practice, all the term's tokens usually had the same value for the label in question.

After lemmatizing the UMLS strings (using the GATE stemmer), we used a trie containing all the UMLS strings with their associated concept IDs to identify exact matches of these strings in the lemmatized corpus and to store the concept IDs for indexing.

Perhaps the greatest advantage that indexing UMLS terms provides is that it helps, in this specific domain, to solve the old problem of synonymy in information retrieval. For example, given the concept of “Atrial Fibrillation,” the following strings are all mapped to the same UMLS concept ID, C0004238:
-- Atrial Fibrillation
-- Auricular Fibrillation
-- AFib
-- AF
-- and a number of others …

Thus if a topic contains “atrial fibrillation” and the UMLS term C0004238 is added to the query, it will match additional documents containing not only the words atrial or fibrillation but also any number of the semantic and morphological variants also associated with the concept C0004238. This helps to improve both recall, by retrieving documents that would otherwise have had a vocabulary mismatch, and precision, by ranking documents according to the frequency of concept, which should be more accurate than one of its plain text strings.

4. INDEXING AND RETRIEVAL

4.1 Indexing

Our index had a number of fields, not all of which ended up being used in a scored retrieval run:

- Report Text
- UMLS Terms
- Age
- Race
- Gender
- Admission status (admitted vs. not admitted)
- Chief Complaint
- Discharge Diagnosis (with ICD-9 names substituted for the codes)
- Negated Report Text (tokens in the report text that were negated)
- Negated UMLS Terms

The Report Text, Chief Complaint, Discharge Diagnosis, and Negated Report Text were lemmatized by GATE, stemmed with the Porter stemmer, and filtered against a list of medical stopwords before being indexed. The rest of the fields were not analyzed in any way other than tokenization. As alluded to earlier in the paper, when indexing the report text and the UMLS terms, words or terms that were tagged as negated or conditional were not added to the index.

4.2 Retrieval Platform

In choosing a retrieval platform for these experiments, we initially considered tools such as Terrier or Lemur that have built-in facilities for running TREC experiments. We finally settled on using Lucene, for its ease-of-use, our familiarity with it, and for the fact that it provides for flexible searching over multiple index fields, which was crucial to our approach.

4.3 Queries

Queries were constructed for Lucene to search different text against the various indexed fields. The queries were lemmatized, stemmed with the Porter stemmer, and filtered against the list of medical stopwords just as the indexed text was. The overall structure for a query consisted of a collection of filters and clauses containing subqueries. Each subquery contained search terms for only a single field, and many of the subqueries had a manually adjusted boost applied to them in order to improve precision by keeping certain clauses from dominating the scoring algorithm. Figure 3 shows an example of a query that demonstrates these features.

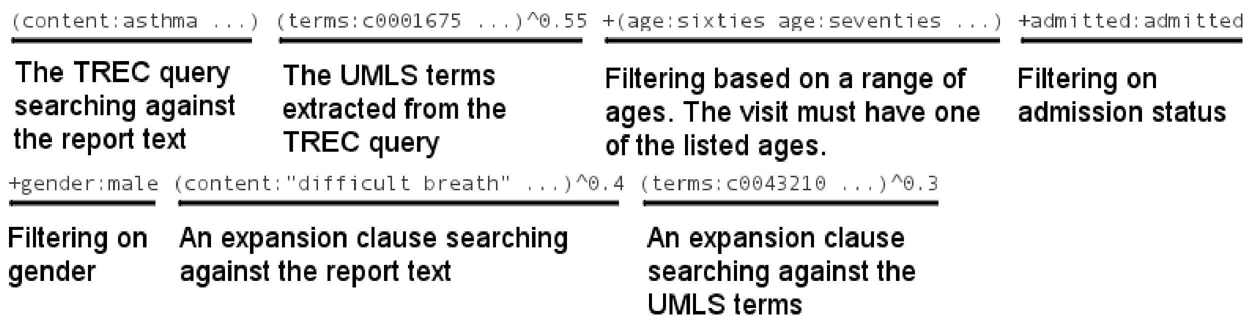

(content:asthma ...)
The TREC query searching against the report text
(terms:c0001675 ...)^0.55
The UMLS terms extracted from the TREC query
+(age:sixties age:seventies ...)
Filtering based on a range of ages. The visit must have one of the listed ages.
+admitted:admitted
Filtering on admission status
+gender:male
Filtering on gender
(content:"difficult breath" ...)^0.4
An expansion clause searching against the report text
(terms:c0043210 ...)^0.3
An expansion clause searching against the UMLS terms

Figure 3: This figure shows an annotated sample query. Some of the subqueries are abbreviated by an ellipsis in order to save space in this diagram.

4.4 Query Expansion

4.4.1 UMLS Related Terms Query Expansion

The UMLS metathesaurus contains a couple of large tables that relate concepts to one another. One of these tables, which we did not end up using in our runs, contains terms that are collocated in medical literature. Expanding with terms from this table did offer some improvement, but not nearly as much as the other table.

The second table contains pairs of terms that exhibit one of several types of semantic relationships. Because we had the ability to filter out certain types of relationships that did not work well for query expansion, we were able to tune this expansion's performance, making it more successful than the results achieved with the collocation table. The types of relationships that we used to choose expansion terms are as follows:

- AQ (allowed qualifier for the term)
- RN (has a narrowing relationship to term)
- RQ (related to and possibly synonymous with the term)
- SY (asserted as synonymous by the medical source)

Because all of the expansion terms came from the UMLS database, we were able to use both the UMLS concept IDs and the plain text name of the concept in this query expansion.

4.4.2 Medical Reference Query Expansion Using External Authoritative Source

Medical reference query expansion was based on the idea that medical encyclopedias may be able to suggest effective expansion terms for a query. Cengage Learning produces a number of medical reference encyclopedias. We lemmatized, stemmed, and indexed articles from these encyclopedias and searched each query against them, choosing frequent terms from the highest scoring article to augment the query.

In a way, this method acts like Pseudo Relevance Feedback [Rocchio], in that a document that is judged relevant by the scoring algorithm is used to suggest expansion terms for the query. The major difference, however, is that the pseudo-relevant document is drawn from a different collection than the one being searched against. In spite of this corpus mismatch and potential incompatibility, this technique resulted in a great improvement in average precision.

We used the vector space–based pseudo relevance approach implemented by Lucene's "More Like This" class. The idea is to take top k documents from the search results, assume they are relevant to the original query, and find the documents similar to those top documents. In the next step, the top terms that categorize those documents are extracted by using the best TF/IDF scores.

This approach extracts the most discriminative terms from each document and makes the expansion an effective technique. A total of 24 medical reference titles were tested separately and combined (up to three in a combined approach). Each title was indexed separately and then searched for the query expansion term extraction. When a reference source was searched, a single top document was retrieved and analyzed for the top TF/IDF scored terms. A minimum term frequency of one (tf=1) and document frequency of five (df=5) were used. The top 100 terms were then added for the query expansion. A boost proportionate to each term's TF/IDF score was used for the expanded query. In a combined approach, these steps were repeated and all terms were combined for the expansion.

We didn't observe issues typical for non-authoritative, web-based content, such as topic drift [Lv]. Gale medical reference articles are cohesive, well-structured documents describing conditions, symptoms, diagnosis, and treatment. Some titles performed better than others and the combined approaches produced better results than the best titles alone. Three titles performed best in the combined runs:

1. *Complete Human Diseases and Conditions*, 2008, Charles Scribner's Sons (part of Cengage), 978-0-684-31541-6

2. *The Gale Encyclopedia of Senior Health: A Guide for Seniors and Their Caregivers*, 2009, Gale (part of Cengage), 978-1-4144-4855-8

3. *The Gale Encyclopedia of Medicine*, 2011, Gale (part of Cengage), 978-1-4144-8691-8

We think that the three titles we used in the submission did well because of the difference in language used in each title and the complimentary nature of these materials. *The Gale Encyclopedia of Medicine* is an all encompassing reference source consisting of very detailed articles written for the general public. It includes information on more than 1,700 medical disorders and concepts. Each article includes in-depth discussion of causes, symptoms, diagnosis, treatments, procedures, and other related topics.

The Gale Encyclopedia of Senior Health focuses on senior's health and is targeted toward senior readers and their caregivers. It covers various issues related to aging, including diseases, treatments, tests, and medication specific to this population. The encyclopedia also includes topics specific to the aging population but beyond descriptions of medical diagnoses.

Complete Human Diseases and Conditions has yet a different audience, as it is published for younger readers (teenagers). It presents information on numerous diseases and conditions. Articles include a definition of the disease or condition; an explanation of how it works in the body; information on causes, symptoms, and diagnosis; descriptions of treatments or cures; and how lifestyles affect one's health.

Since these medical reference sources are separate from UMLS, it was not possible for the names of terms between the two sources to be properly aligned. In many cases, the reference terms that were suggested had different names than those used in UMLS. Additionally, the reference content was stemmed, which made it even more difficult to reconcile with an unstemmed UMLS. For these reasons, it was not possible to search these expansion terms against the UMLS terms index field, but only against the report text field.

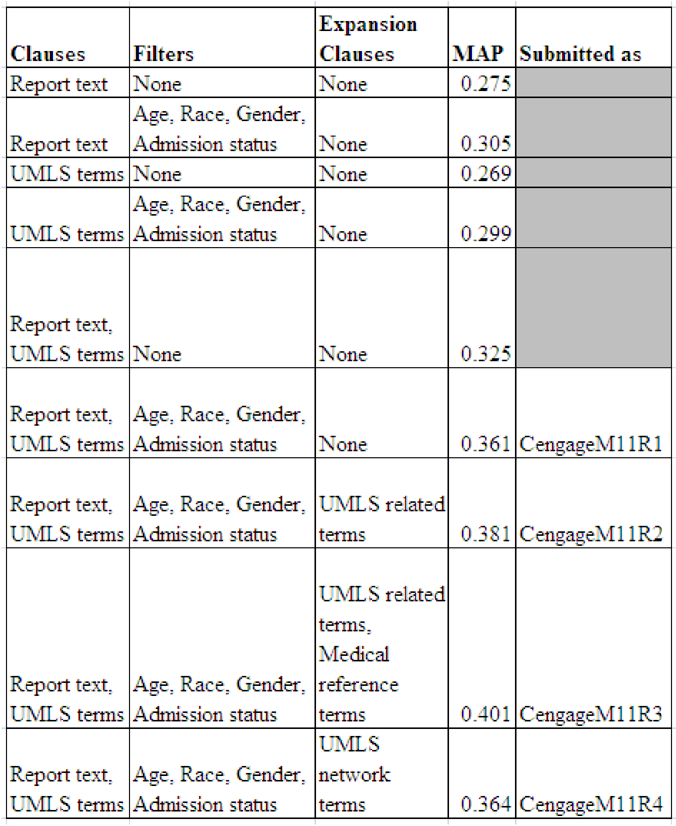
4.4.3 UMLS Network Query Expansion

This technique represents perhaps our most unique approach to this problem. Using Neo4j, a graph building API for Java, we constructed a graph of UMLS, where the nodes were concepts and the edges were relationships from the UMLS related terms table. However, unlike the UMLS related term expansion, we did not exclude any type of relationship in building the network.

We first extracted all of the UMLS terms that appeared in the query. The next step was to find the smallest subgraph of the UMLS network that contained all of the query terms. The terms from this smallest subgraph were then used as expansion terms for the new query. Because this UMLS network contained both UMLS concepts and names, we were able to search against both the report text and the terms fields.

5. RESULTS

As previously stated, all the results produced prior to submission were evaluated against our own qrels on a set of sample topics. The following table is constructed in such a way as to attempt to demonstrate the relative effectiveness of different techniques we attempted.



Clauses	Filters	Expansion Clauses	MAP	Submitted as
Report text	None	None	0.275	
Report text	Age, Race, Gender, Admission status	None	0.305	
UMLS terms	None	None	0.269	
UMLS terms	Age, Race, Gender, Admission status	None	0.299	
Report text, UMLS terms	None	None	0.325	
Report text, UMLS terms	Age, Race, Gender, Admission status	None	0.361	CengageM11R1
Report text, UMLS terms	Age, Race, Gender, Admission status	UMLS related terms	0.381	CengageM11R2
Report text, UMLS terms	Age, Race, Gender, Admission status	UMLS related terms, Medical reference terms	0.401	CengageM11R3
Report text, UMLS terms	Age, Race, Gender, Admission status	UMLS network terms	0.364	CengageM11R4

Table 2: Testing dataset results – MAP

In selecting the runs to submit, we attempted to choose techniques that produced high average precision in our own testing but that also demonstrated interesting approaches. Rather than submitting baseline runs for comparison purposes, we instead report our own evaluations on the baseline for comparison with our evaluations of the submitted runs, a method we believe will be suitable for the purposes of comparison.

5.1 Comparison with TREC Evaluation Results

Evaluation results for Cengage runs show consistency with our testing data MAP measurements, except between submissions CengageM11R1 and CengageM11R4 (Table 3). The MAP difference from the evaluated submission runs shows no statistical significance between these two runs (Table 4). CengageM11R2 had a statistically significant difference with CengageM11R1 but not with CengageM11R4. Finally, CengageM11R3 showed statistically significant improvements when compared to all other submissions. The best overall submission was CengageM11R3, which included report text and UMLS terms clauses, filters (age, race, gender, admission status), and expansions with UMLS related terms and medical reference terms.

	BPREF	R-PREC	P@10	MAP	MAP (Testing Dataset)
CengageM11R1	0.5308	0.4318	0.6176	0.4186	0.361
CengageM11R2	0.5285	0.4299	0.6265	0.4311	0.381
CengageM11R3	0.5523	0.44	0.6559	0.457	0.401
CengageM11R4	0.5267	0.4271	0.5941	0.4162	0.364

Table 3: Evaluation results, compared with testing dataset MAP

P-Values	M11R1	M11R2	M11R3	M11R4
M11R1				
M11R2	0.020			
M11R3	0.019	0.018		
M11R4	0.825	0.110	0.004	

Table 4: P-values for evaluated submission runs MAP scores

6. CONCLUSION

In summary, we used text processing, information extraction, and query expansion to produce our runs for this year's track. We have found these techniques to be useful in improving the precision of the search results. We feel that information extraction especially holds a lot of promise for further research.

The detection of negation and conditionals was important for our submissions. Versus an index that did not have negations and conditionals excluded, there was an improvement in average precision of about 5% gained by leaving such terms out of the index with no significant reduction of recall. We believe that negation and conditional detection should be a part of any production medical information retrieval system.

As Nadkarni et al. also found, we have concluded that UMLS concept indexing is a useful technique when used in conjunction with other techniques, but it is still not ready to be used on its own without further refinement [Nadkarni]. The performance of UMLS terms alone was approximately 2% worse than searching the plain text query (MAP of 0.269 vs. 0.275), but when used in conjunction with the plain text query, the two performed about 18% better than the plain text alone (MAP of 0.325 vs. 0.275).

Using an external authoritative source for pseudo relevance feedback resulted in additional improvement, even combined with other query expansions (6% MAP improvement). This method seems to perform very well when the external source has well-structured, single topic documents. Multi-source expansion worked best when the language in the reference sources differed (e.g. was geared toward different target audiences or age groups).

If we were to participate in this track again, we would try to focus more effort on information extraction. It should be possible to extract many more attributes from the reports than the few that we did. For example, it should be possible to extract a list of all symptoms the patient exhibits, all the diagnoses received, all the medications being taken, all the procedures performed, and so on. Having the ability to search against such specific index fields should only improve the precision of the results if properly utilized.

7. ACKNOWLEDGEMENTS

We thank Drew Koszewnik, Duane May, Eugene Kiel, Paul Tunney, Rohit Laungani, Pete Pfeiffer, John Nader, Dr. James McKinley, Alvin Cabansag, Bernard Grunow, Susan Kelsch, Matt Green, Sarah Krot, John Magee, Denise DiGregorio, Ben Katt, Bridget Travers, Shirelle Phelps, Kristin Key for their ideas and support.

8. REFERENCES

[1] W.W. Chapman, W. Bridewell, P. Hanbury, G.F. Cooper, B.G. Buchanan. A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics, 34(1): 301-310, February 2001.

[2] H. Cunningham, A. Hanbury, and S. Rüger. Scaling up high-value retrieval to medium-volume data. In H. Cunningham, A. Hanbury, and S. Rüger, editors, Advances in Multidisciplinary Retrieval (the 1st Information Retrieval Facility Conference). LNCS volume number: 6107, Lecture Notes in Computer Science, Vienna, Austria, May 2010. Springer

[3] C. Friedman, G. Hripcsak, W. DuMouchel, S.B. Johnson, P.D. Clayton. Natural language processing in an operation clinical information system. Natural Language Engineering, 1(1): 83-108, 1995.

[4] D.T. Heinze, M.L. Morsch, R.E. Scheffer, M.A. Jimmink, M.A. Jennings, W.C. Morris, A.E.W. Morsch. LIFECODE: a deployed application for automated medical coding. AI Magazine, 2(2): 76-88, Summer 2001.

[5] D.A. Lindberg, B.L. Humphreys, A.T. McCray. The Unified Medical Language System. Methods of Information in Medicine, 32(4): 281-291, August 1993.

[6] P.G. Mutalik, A. Deshpande, P.M. Nadkarni. Use of general-purpose negation detection to augment concept indexing of medical documents: a quantitative study using the UMLS. Journal of the American Medical Informatics Association, 8(6): 598-609, November 2001.

[7] P. Nadkarni, R. Chen, C. Brandt. UMLS concept indexing for production databases. Journal of the American Medical Informatics Association, 8(1): 80-91, January 2001.

[8] A.S. Wu, B.H. Do, J. Kim, D.L. Rubin. Evaluation of Negation and Uncertainty Detection and its Impact on Precision and Recall in Searching. Journal of Digital Imaging, 24(2): 234-242, April 2011.

[9] J. J. Rocchio. Relevance feedback in information retrieval. In The SMART Retrieval System: Experiments in Automatic Document Processing, pages 313-323. Prentice-Hall Inc., 1971.

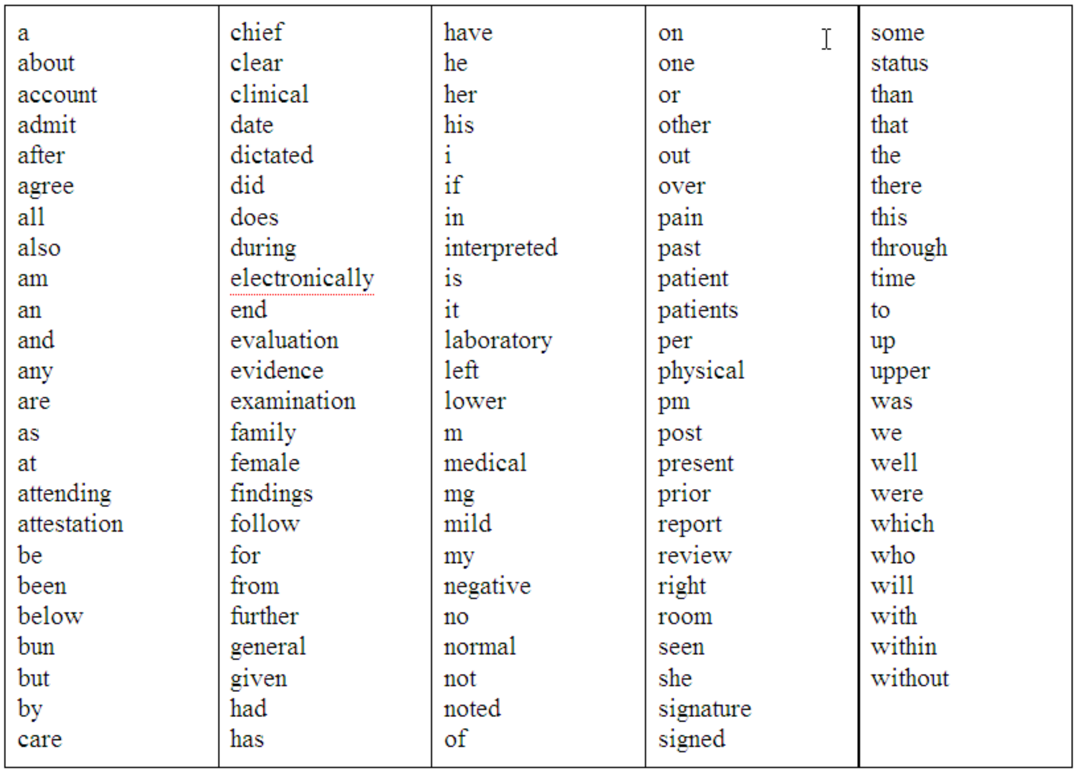
[10] Yuanhua Lv, ChengXiang Zhai. "Positional Relevance Model for Pseudo-Relevance Feedback". Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'10), pages 579-586, 2010.

[11] Manoj K. Chinnakotla Karthik Raman Pushpak Bhattacharyya, Multilingual Pseudo-Relevance Feedback: Performance Study of Assisting Languages, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1346–1356, Uppsala, Sweden, 11-16 July 2010.

Appendix A: Sample Topics Used for Training

1. patients with atrial fibrillation treated with ablation
2. patients with atrial fibrillation treated pharmacologically
3. patients younger than 50 with hearing loss
4. patients admitted for injuries s/p fall
5. patients with GERD who had esophageal adenocarcinoma diagnosed by endoscopy
6. patients on monoclonal antibody treatment for inflammatory bowel disease or arthritis
7. patients treated for MRSA infection during this visit
8. men with prostate cancer treated with surgery or radiotherapy
9. patients seen in the ER for low back pain who were not admitted to the hospital
10. patients admitted for complications from dementia
11. patients with dementia who were discharged to a skilled nursing facility or other institutional setting
12. patients with a BMI > 40 without hypertension or diabetes
13. patients with a history of treatment for ductal carcinoma in situ
14. patients who had a CT scan after which follow-up PET scan was recommended for possible malignancy
15. patients who had a PET or MRI scan for cancer staging
16. patients with CEA+ cancer
17. patients younger than 30 admitted for dental abscess
18. patients admitted for psychiatric treatment who were discharged to residential treatment
19. encounters requiring a translator where the patient's language is not specified in the notes
20. women who are currently pregnant and have been smoking and/or drinking during the pregnancy
21. teenagers who have taken or plan to take Plan B
22. patients admitted for treatment of vascular claudication
23. women with hip or vertebral fracture despite being on medication for osteoporosis/osteopenia
24. patients admitted for a reason directly related to medication non-compliance
25. patients who had robotic assisted surgery
26. patients who were homeless but were placed/discharged to a facility other than a homeless shelter
27. patients admitted for complications due to renal failure despite being on dialysis
28. patients with depression who are receiving medical treatment as well as therapy
29. patients admitted for complications related to psoriasis
30. all patients who left the hospital AMA

Appendix B: List of Medical Stopwords



a	chief	have	on	some
about	clear	he	one	status
account	clinical	her	or	than
admit	date	his	other	that
after	dictated	i	out	the
agree	did	if	over	there
all	does	in	pain	this
also	during	interpreted	past	through
am	electronically	is	patient	time
an	end	it	patients	to
and	evaluation	laboratory	per	up
any	evidence	left	physical	upper
are	examination	lower	pm	was
as	family	m	post	we
at	female	medical	present	well
attending	findings	mg	prior	were
attestation	follow	mild	report	which
be	for	my	review	who
been	from	negative	right	will
below	further	no	room	with
bun	general	normal	seen	within
but	given	not	she	without
by	had	noted	signature	
care	has	of	signed	