A knowledge-based approach to medical records retrieval

Dina Demner-Fushman, Swapna Abhyankar, Antonio Jimeno-Yepes, Russell Loane, Bastien Rance, François Lang, Nicholas Ide, Emilia Apostolova, Alan R. Aronson

National Library of Medicine, Bethesda, Maryland

Abstract

The NLM LHC team approached the cohort selection task of the 2011 Medical Records Track as a question answering problem. We developed 60 training topics and then manually converted those topics to question frames. We started with the evidence-based medicine well-formed question frame and expanded it to explicitly capture temporal and causal relations. We then implemented a syntactic-semantic method for extracting the question frames from the free text topics.

Based on the clinical documentation standards and knowledge of the clinical documentation structure, we split each report type into sections corresponding to different categories of clinical content, with the result that each section contained a specific class of data. We then ranked each document section according to its likelihood of containing answers to specific question frame slots. For example, if a question concerns medications prior to admission, the answers should be found in the *Medications on Admission* and the *Medical History* sections. In addition, we split each section into *Positive* (containing asserted findings, problems, and interventions), *Negative* (in which findings are negated) and *Possible* (that includes all uncertain statements).

After structuring the questions and the documents, we developed algorithms for expressing question frames in the languages of the two search engines used for retrieval: Essie and Lucene. In addition to the UMLS synonymy-based query expansion built into Essie and implemented externally for Lucene, we expanded the terms in the documents with their ancestors and children from the MeSH hierarchy. We also expanded query terms for recognized drug names using RxNorm and Google searches.

In addition to the automatically generated baseline and expanded queries that we ran against the original and the structured documents, we used the Essie user interface for manual query generation. During this process, we determined that a third of the automatically generated question frames, although technically correct, needed significant modifications due to different sub-languages used in the documents and in the queries. The manually created queries were used to search the collection with each search engine.

Our manual queries submitted to Essie significantly outperformed all of our other runs (achieving 0.73 P@10, 0.66 bpref, and 0.49 R-prec). Interestingly, the best automatic run for Lucene was the baseline run (P@10 = 0.44, bpref = 0.47, R-prec = 0.33) that used the topics "as is" to search the original documents. The results for this run are not significantly different from the manual Lucene (P@10 = 0.51, bpref = 0.51, R-prec = 0.35) and the automatic Essie (P@10 = 0.49, bpref = 0.48, R-prec = 0.33) runs.

1. Introduction

The 2011 TREC Medical Records Track focused on finding patient cohorts based on short descriptions of the cohort inclusion criteria and clinical narrative documents generated during patients' hospital stays. For any given patient, all of the documents from one hospital stay were collated into a single visit.

Our previous efforts in clinical text processing showed that information in different document sections is more reliable for specific questions. For example, for a question regarding a patient's active medications, the medications listed in the allergy section should not be included in the results because they have the potential to cause adverse reactions and therefore are not given to the patient (Mork et al., 2010). Our approach to document segmentation is described in Section 2.

Another important issue in clinical text processing is distinguishing information that is negated from that which is asserted. For example, if we need to find radiology reports for patients with pneumonia, we don't want to see the reports for patients that had a chest x-ray done to exclude pneumonia and who, in fact, did not have evidence of pneumonia. Our approach to identifying assertions and negation is described in Section 3.

We believe that the evidence-based medicine approach to building a well-formed clinical question provides a good framework for simple questions, but we also know that this framework is not capable of capturing some nuances (for example, temporal relations) that might be very important for cohort identification (Huang et al., 2006). Our extensions to the basic clinical question framework are presented in Section 4.

To expand the question framework, test our query translation algorithms, and validate designating specific document sections as most likely to contain answers to specific question frame slots, SA (a clinical informatics fellow and a practicing pediatrician) generated 60 training topics that we shared with other track participants. While training the system, we realized that the levels of granularity in the questions and in the documents are often different: whereas the questions often contain drug and disease classes, the documents mostly contain specific disease codes and drug names. To compensate for the differences, we expanded the disease and drug terms in the documents and the drug and drug class names in the queries. This work is discussed in Section 5.

For retrieval, we used Essie (Ide et al., 2007) and Lucene.[1] We briefly describe these two search engines in Section 6, along with our query formulation and retrieval strategies. We conclude the report with a preliminary analysis of our experiments and results.

2. Document Segmentation

We developed rules to segment all documents in the collection into unique sections containing specific clinical content based on manual examination of a random sample of documents. We then automatically segmented all of the documents based on these rules and manually evaluated the segmentation. Specifically, we iteratively: 1) reviewed a selection of documents, 2) created section headings for each document type, 3) created rules for assigning the section headers based on specific text indicators in the documents, 4) automatically segmented the documents, and 5) went back to step one and manually reviewed a new selection of segmented documents.

2.1 Manual rule creation

In the first step of the manual review, we examined the content and structure of sample documents from each of the collection's nine different document types. Some, such as SP (Surgical pathology) and ECHO (Echocardiogram), had a structured format and contained limited, specific content in a predictable sequence within that document type. Others, such as RAD (Radiology), had document subtypes (e.g., Chest, Angio), each with their own uniform structure and content. Other document types were far less structured, such as HP (History and physical) and DS (Discharge summary); in some cases, these documents contained similar data, while in others, the content was disparate in terms of the depth and breadth of the information as well as its sequence and length.

Second, we created section headings to encompass the specific clinical content contained in each document type. We wanted to create enough sections to appropriately segment different types of clinical data within each document; however, we did not want to make the sections so granular as to become unmanageable. As mentioned above, ECHO reports were very structured and only contained echocardiogram results, so we only needed to assign three unique section headings to encompass all of the relevant data: reason for the study, procedure details, and final diagnosis. Document types such as DS and HP contained a wide range of clinical content, including the reason for admission, past medical history, home medications, physical examination, lab and radiology results, code status, and discharge information. Up to sixteen different section headings were used to cover the diverse types of data contained in these more complex documents.

Third, we created detailed rules for the automatic report annotator to assign appropriate section headings based on specific indicators in the document text. Most of these rules were based on variations in the indicators used to indicate different types of information in the original text. The documents that originally had the most structure also had the least variability in how the different clinical content was named. For example, the admitting diagnosis (i.e. reason for the study) in the ECHO reports was identified by *referring diagnosis* (case-insensitive) either followed by a colon or occurring at the end of a line. However, in DS documents, the admitting diagnosis was identified by *admission* or *admitting* plus *diagnosis* or *diagnoses* plus either a colon or the end of a line.

In some instances, important types of clinical data, such as drug allergies or family history, did not have any indicators that signaled what information was following. For example, *allergies:* would sometimes indicate the start of the drug allergies section; however, many times the text would simply say *allergies are to…* or *penicillin allergy* without any introduction. Similarly, a patient's family history often started on a new line of the text without any indication of what was following. For these, we created more complicated rules that not only included information about a colon or the end of the line, but also about specific words to precede or follow potential section indicators. For example, *allergy* could be preceded by *penicillin* or followed by *to* or *are to*; *family history* could be followed by a colon, dash, or end of the line or *of*, *to*, *is*, or *shows* followed by a colon, dash, or end of the line.

[1] http://lucene.apache.org/java/docs/index.html

In the fourth step, the automated report annotator segmented all of the documents according to the section indicator rules as described below. Following each annotation cycle, we reviewed a new subset of documents from each document type and adjusted the section indicator rules as necessary. We went through multiple iterations of this process and ultimately created 548 section indicators to segment the 9 document types into 3 to 16 sections, which does not include the variation in case or spacing, or the different punctuation marks (colon, dash) and words that could follow an indicator.

2.2 Automated Annotation of Reports

After the 548 indicator rules and the assertion data (described in Section 3) were created, we automatically annotated all documents in the collection using those rules and data. Our report annotator examined each report one line at a time. If a line in the report matched one or more indicators, the annotator selected the longest (case insensitive) matching indicator, and annotated the text with an XML tag indicating the beginning of the corresponding section type. For example, in a document of type RAD, the string *clinical history* occurring either at the end of a line or immediately followed by a colon indicates the beginning of a **history_of_present_illness** section. Accordingly, the tag was inserted when the annotator encountered the line: CLINICAL HISTORY: DIFFICULTY FEEDING, FEEDING TUBE. This indicator rule would not be applicable, however, for the line: WALL DEHISCENCE. GIVEN THE CLINICAL HISTORY OF CHANGE IN MENTAL

The end of a section was not explicitly detected; a section was deemed to end simply when the next section began. Finally, any text appearing in a report before the first matching indicator was tagged as the **preamble**.

Assertion tags were inserted using the output of the assertion extractor. For an example of assertion data, consider the text: *no erythema, no drainage*, which was identified as a negated statement. The section containing that string was annotated with: **<negation_assertion>**no erythema, no drainage **</negation_assertion>**

Our annotator ran multiple parallel processes on 24 3.3-GHz processors and completed the annotation of the 100,000+ reports in about ten minutes. After all the reports in the collection were annotated as described above, they were passed to both Essie and Lucene for indexing and retrieval, as described in the next sections.

3. Assertion Status Detection

Clinical texts are abundant in statements expressing the absence of or uncertainty associated with medical conditions. Thus clinical information retrieval systems need to accurately differentiate between the assertion status of statements, i.e. whether a statement is affirmed, negated, or uncertain (interchangeably called speculative). Similarly, medical conditions pertaining to the patient's family history need to be differentiated from ones pertaining to the patient.

Table 1 Sample corpus sentences demonstrating the assertion status associated with the medical condition liver disease: affirmed, negated, uncertain, and family history.

Assertion Status	**Example**
Affirmed	Patient with end-stage **liver disease**/ascites.
Negated	No **liver disease**.
Uncertain	Questionable **liver disease**.
Family history	Family history is significant for **liver disease** in the father.

Consider the sample corpus sentences shown in Table 1. While all four sentences mention the medical condition *liver disease*, the disease is affirmed and pertaining to the patient in only one of the examples. We determined that our system needs to be aware of the assertion status associated with the condition in order to accurately answer a query requesting, for example, patients with liver disease.

In our approach, we first detected the linguistic scope of negated and uncertain statements as well as those describing the patient's family history. Such statements (typically sentence clauses) were extracted from the text and annotated as described in Section 2.2.

The linguistic scope of negated, uncertain, and family history statements was detected utilizing a previously developed, open-source system – ScopeFinder.[2] The ScopeFinder system is a

[2]http://scopefinder.sourceforge.net/

linguistically motivated rule-based system for the detection of negation and speculation scopes (Apostolova et al., 2011). The system rule set consists of lexico-syntactic patterns. The lexico-syntactic patterns contain a combination of a lexical trigger (i.e. a cue word) and its associated syntactic scope expressed in Penn Treebank syntactic notation. For example, one of the negation scope rules matches the complement of the verb *denies.* When applied to the sentence *She also had cough but* ***denies fever***, the rule matches the sentence snippet shown in bold. The problem *fever* is then marked as negated.

The lexico-syntactic rules were initially automatically extracted from the BioScope corpus (Vincze et al., 2008), a biomedical corpus annotated with negation/speculation cues and their scopes. Additional lexico-syntactic rules were identified in the analysis of the 2011 Medical Records Track dataset and manually added to the rule set.

4. Automated Conversion of Topics to Question Frame

As mentioned above, SA developed a set of 60 training questions: 30 based on her patient encounters during the time of question generation as well as on interesting topics contained in recent issues of the General Medicine Journal Watch[3] and 30 based on the Institute of Medicine's priority topics.[4] These questions were used to create the frame structure and the algorithms for automatic structuring of the free-text topic into a frame structure.

4.1 Frame development

The original evidence-based medicine well-formed clinical question frames consist of four slots: Patient/Problem, Intervention, Comparison, and Outcome (Richardson et al., 1995). We refined the basic frame elements with syntactically related words; captured conjunction and prepositional attachment; and augmented the basic four-slot PICO frame (P (split into Patient, Problem and Anatomy), Intervention (merged with Comparison), Outcome) with relational slots that express question elements using predicate-argument structures ([concept]–(relation)–[concept]).

As we manually encoded our 60 test questions according to the expanded PICO framework, we further refined the frames as necessary to capture the intricacies contained in the test questions, such as temporal relations. For example, we defined three medication slots (medications before admission, on discharge, and the fall-back – medication for problem). These distinctions were needed to encode (and answer) temporal questions, such as *Find patients with HIV admitted for a secondary infection who were not on prophylaxis for opportunistic infection* and *Find patients with COPD who were discharged on inhaled steroids*. In the first example, only prophylactic drugs the patient was on prior to admission should have been considered, while in the second, an inhaled steroid was only relevant if the patient was discharged on that medication. The XML surface representation of our frame slots was chosen for the convenience of then automatically translating the frames to the query syntax of the search engines used for retrieval.

4.2 Automatic frame extraction

Our system automatically extracted the frames in four steps. In the first step, the system submitted the question to MetaMap 2010 with the default settings to extract the UMLS® concepts (Aronson and Lang, 2010). For each concept, the system stored the lexical match with offset and length, negation and semantic types in a lookup table. Second, the system used regular expressions to extract Patient demographics and social history. The Population slot was limited to the occupations and ethnicities defined by the UMLS semantic types *Professional or Occupational Group* and *Population Group*, respectively. The patterns for social history were limited to identifying smoker status and alcohol and illicit drug use.

In the third step, the system processed the topic sentences using the Stanford dependency parser (de Marneffe et al., 2006). To prevent the parser from breaking up multi-word terms, the system concatenated the words in the terms prior to parsing. We focused on extracting a limited set of typed dependency relations, conjunctions, and modifiers. The frame slot was extracted only if the semantic and syntactic constraints were satisfied. If a rule was applied, the terms used in the rule were marked as *used* in the look-up table.

After completing iterations over the dependency paths, in the fourth and final step, all of the basic PICO elements not used in the previous steps were added to the frame. That is, if the concept lookup table for a given question contained concepts in the semantic groups Disorders (Problems), Interventions or Anatomy that were not marked as used in generating the question frame, the concepts were assigned to the traditional PICO frame slots.

[3] http://general-medicine.jwatch.org/

[4] http://www.iom.edu/Reports/2009/ComparativeEffectivenessResearchPriorities.aspx

4.3 Mapping frame elements to document sections

Once we had finalized our expanded PICO frame slots, we mapped each one to its corresponding document section(s) to enable the automatic retrieval runs by Essie and Lucene. For each slot, we created rules for: 1) which document sections should be searched, 2) how much weight should be given to the search results found in each particular section, and 3) which document sections should not be searched for that particular data element. For example, the *drug allergies* slot was mapped to the *allergies* section of the document with a weight of 1.0.

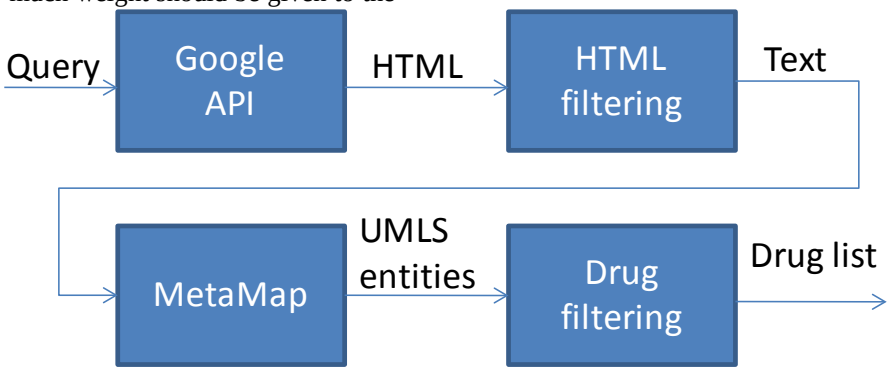

Figure 1 Web-based query expansion

Another rule explicitly stated that the *drug allergies* slot should not be mapped to *home medications*, *discharge medications*, or *hospital medications*. In another case, the *family history* frame slot was mapped to the *family history* document section with a weight of 1.0, and to other sections including the *history of present illness* and *past medical history* with a weight of 0.7.

5. Term Expansion

As mentioned above, the language and granularity of the *Problem* and *Intervention* terms in the documents were significantly different compared to those in our training questions. Therefore, we developed expansion techniques for terms found in the documents as well as those in the queries, and a specific process for drug expansion.

5.1 Document term expansion

We determined that the standard UMLS-based[5] synonymy expansion in Essie, which was provided to Lucene to create equal conditions for both search engines, was not sufficient to find good documents for the training topics. In general, the terms contained in the patient documents were much more granular than those in the queries. For example, a document might have *invasive ductal carcinoma* (a type of breast cancer) as a patient's diagnosis, whereas a query might ask for patients with *breast cancer*. Therefore, we decided to expand the terms identified in the documents with their parent and child terms in MeSH[6]. The expansion was based on finding a term that MeSH identified and then using the MeSH tree for expansion.

5.2 Query term expansion

We also found that the standard reference resources (e.g., RxNorm) were lacking the breadth of information necessary for query expansion. Even the UMLS, which contains information from multiple vocabularies, does not include all possible ways of classifying each medication or problem and available treatments. In addition, there is a lag time between when a particular drug is approved (or removed from the market) and when the updated information is incorporated into various resources. The same applies for treatments and problems.

However, there are an increasing number of websites not traditionally considered to be reference sites (e.g., Wikipedia) that contain lists of drugs and treatments relevant to problems that may not yet be curated into the existing standard databases and terminologies. We developed a process for extracting expansion terms from websites which we then combine with the drug expansion process described in 5.3. The approach consists of two main steps: identification of reference sites and the extraction of drugs or treatments from these sites. Figure 1 shows the flowchart for this approach.

In the first step, we queried Google's AJAX API based on the information need (e.g., a drug class, a drug class related to a problem or a problem) to retrieve relevant websites. The HTML code from the top 5 web pages for each query was retained for

[5] http://www.nlm.nih.gov/research/umls/

[6] http://www.nlm.nih.gov/mesh/

entity extraction. The websites provided enumerations of elements in well-structured HTML **<li>** tags. The HTML pages were filtered to keep the text delimited by the tags.

From the extracted text, we kept only the terms belonging to specific entities of interest. To do so, we used MetaMap 2011 to identify terms of the UMLS *Pharmacological substance* or the *Therapeutic or Preventive Procedure* semantic types. We further filtered the drugs identified by MetaMap by estimating term frequency and removing the terms with only one mention in order to further increase the quality of the extracted list. Since not all the *Pharmacological substances* in the UMLS are drugs, a stop word list was used to filter out false positives. The stop word list was prepared by close examination of the terms linked to concepts within the *Pharmacological substance* semantic type in the UMLS and includes terms like *water* (Humphrey, 1999).

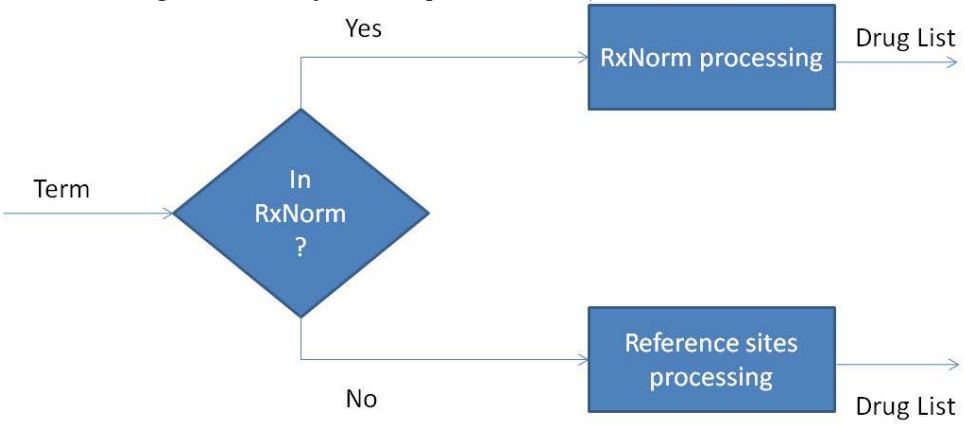

Figure 2 Drug expansion process

5.3 Drug expansion

We extended the website processing to include the expansion of brand names based on RxNorm[7]. Figure 2 shows the overall flowchart of the approach. When considering a candidate term for expansion, the term was first checked against RxNorm. If the term was in RxNorm, RxNorm was used for the expansion. If the term was not found in RxNorm, websites were used for the expansion.

In RxNorm, a brand name has at least one ingredient, and an ingredient may be related to several brand names. For example the brand name Ritalin is the trade name of the ingredient methylphenidate. Other brand names of methyphenidate include Metadate and Methylin. In this example, RxNorm would provide four additional brand names and a generic name. We used the RxNorm API (Peters and Bodenreider, 2008) to normalize and query drug names. The drug expansions were used by both search engines.

6. Indexing and Retrieval

The XML documents prepared as described in Section 2 were indexed using Essie and Lucene. Briefly, the original documents were annotated with additional tags and text to facilitate targeted searches. Extra XML tags delimited the recognized sections (*allergies*, *complications*, *discharge medications*, etc.) and assertions within them.

The remaining step for the automatic retrieval experiments was to generate rules for translating question frames to queries. The rules and retrieval strategies are described next.

6.1 Essie indexing and retrieval

We wrote a translation utility to convert the results of question frame extraction into Essie query syntax.

Conceptually, translation is a simple process:

1. predicates map to search areas, defined as a weighted set of XML tags (for example, predicate <PMH> <Prblm> hepatitis </Prblm> <Cause> blood transfusion </Cause> </PMH>, extracted from the topic *patients with a history of hepatitis related to blood transfusion, now with liver cancer*, was mapped to the *past medical history* section)
2. predicate arguments (in the above example, *hepatitis* and *blood transfusion*) are terms and are searched with concept expansion, which includes UMLS term variants and synonyms

[7] http://www.nlm.nih.gov/research/umls/rxnorm/

3. multiple predicates produce multiple query clauses, which are combined with an AND operator

Several rounds of manually inspecting test results and modifying the translation utility produced a tangled, complicated algorithm with the following highlights:

Multiple arguments to a single predicate were searched three ways and the results were combined with an OR operator:

1. arguments searched and combined with a NEAR operator at weight 0.7 (requires arguments to be together)
2. arguments searched and combined with an AND operator at weight 0.7 (allows arguments to be separated)
3. arguments searched and combined with an OR operator at weight 0.1 (allows arguments to be missing)

Note that anything found by method 1 is also found by methods 2 and 3, resulting in a combined weight of >0.9.

Likewise, arguments with modifiers (for example, <Procedure>surgery<MOD>robotic-assisted </MOD></Procedure>) were searched two ways and the results were combined with an OR operator:

1. argument and modifier searched and combined with a NEAR operator at weight 0.7
2. argument searched without the modifier at weight 0.7

Modifiers were dropped entirely if they were too common (found in the corpus more than 200 times).

Some corpus-based ad hoc synonymy was added, including:
status post → status post OR s/p
male → male OR man OR mr. OR his OR he OR gentleman
female → female OR woman OR mrs. OR ms. OR her OR she OR lady
adult → in 20s OR in 30s OR in 40s OR in 50s OR in 60s

As a final fallback strategy, the original topic text was searched at weight 0.01 with *lossy expansion*, which finds documents with most of the query words. This Essie feature is known to perform poorly and is rarely used in practice (except as a last resort in TREC). Our final searches were restricted to positive text, defined as original text without negative and family assertions and with speculative assertions weighted at 0.25.

6.2 Lucene indexing and retrieval

Indexing of the visits was done based on a standard Lucene analyzer and stop word list removal. The section processing presented above provided a rich set of sections which were stored and retrieved based on Lucene fields. For each visit, we prepared one field that contained all the text in the report, and additional fields that segmented each section's text into positive, speculative and negative subsections.

The extracted question frames were reformulated based on Lucene's query language, which allows for field queries and in addition, weighting query expressions using the character ^ at the end of the search expression. In addition, in specific cases the query terms in a given expression were constrained to be found within a specific number of words using the character ~ followed by the maximum allowed length of the span of text.
The original query, with a lower weight (0.02), was combined with the reformulated query to retrieve missing documents. Finally, expansion of the PICO predicate Age was based on the ad hoc synonymy described above. Age was used to filter out visits that were not in the range specified by the query.

An example of the expansion of topic 114 is presented below:

```
((Adult patients discharged home with palliative care
/ home hospice)^0.02) OR
 (((assessment_and_plan_positive_text: home OR
hospice palliative care))^1.0
((addendum_positive_text: home OR hospice
palliative care ) (course_positive_text: home OR
hospice palliative care))^0.7
((assessment_and_plan_speculative_text: home OR
hospice palliative care ))^0.5
((addendum_speculative_text: home OR hospice
palliative care ) (course_speculative_text: home OR
hospice palliative care))^0.35) AND
((age_in_section_best: "in 50s") OR
(age_in_section_best: "in 60s") OR
(age_in_section_best: "in 40s") OR
(age_in_section_best: "in 30s") OR
(age_in_section_best: "in 20s"))
```

7. Experiments

Our experiments had three goals: 1) to establish if domain knowledge is absolutely necessary for clinical document retrieval, 2) to establish if a widely-used general purpose search engine would benefit from domain knowledge, and 3) to determine if the cohort identification task can be completely automated.

Accordingly, we used Essie, a domain specific search engine that cannot be easily decoupled from its knowledge, and Lucene, to which we added the same knowledge embedded in the document structure and query formulation and expansion. Towards our second goal, we compared the "off-the-shelf" Lucene runs with the Lucene knowledge augmented runs.

Finally, we focused on manually modifying queries until the top 10 visits looked relevant for the most part. The final queries and top ten results (without eliminating the obviously irrelevant documents that could not be eliminated with query modifications) were reviewed by two MDs (SA and DDF). In total we submitted six runs as described in Table 2.

Table 2 NLM runs submitted to the Medical Records Track

Run	Description
NLMManual (judged)	Manual queries generated using the Essie user interface, padded with the lossy expansion of the original topics
NLMManualLuc (judged)	Manual queries translated to the Lucene query language
EssieAuto (not judged)	Automatic Essie queries described in Section 6.1
NLMAutoLuc (not judged)	Automatic Lucene knowledge-based queries described in Section 6.2
NLMLucene (not judged)	Baseline 'out-of-the-box' Lucene retrieval over original documents
NLMLucenePS (not judged)	Baseline 'out-of-the-box' Lucene retrieval over non-negated sections

8. Results

To our surprise, our baseline Lucene run was statistically (Wilcoxon signed rank test) as good as our Lucene manual run and the Essie automatic run. See Table 3.

Table 3 Evaluation results

Run	P@10	bpref	R-prec
NLMManual	0.7265	0.6583	0.4999
NLMManualLuc	0.5147	0.5126	0.3567
EssieAuto	0.4971	0.4822	0.3369
NLMAutoLuc	0.2294	0.3671	0.1911
NLMLucene	0.4382	0.4781	0.3367
NLMLucenePS	0.1735	0.3317	0.1285

Our manual Essie run was significantly better than all our other runs on all three reported metrics. The Essie automatic, Lucene manual and Lucene baseline runs were significantly better than the two automatic knowledge-enhanced Lucene runs, with the NLMLucenePS run being the worst compared to all other conditions.

Although we manually verified that most of the top ten documents in the NLMManual run were relevant, on three topics (110, 123, and 128) this run performed worse than the median (See Figure 3). An in-depth analysis of whether this is caused by the differences in the evaluators' opinions or technical errors is underway.

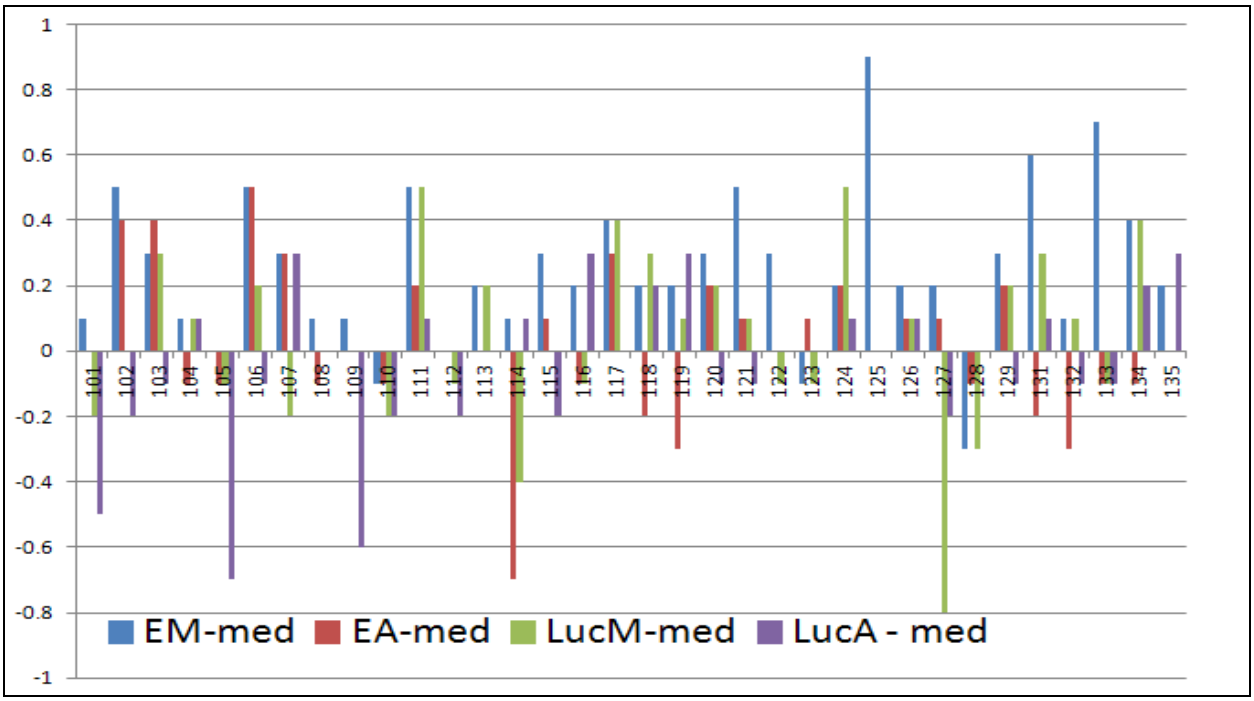

Figure 3. Differences in P@10 between the NLM runs and the judged median results per topic (EM = NLMManual, EA = EssieAuto, LucM = NLMManualLuc**,** LucA = NLMLucene)

9. Discussion

The preliminary answers to our questions indicate that depending on the nature of the task, an "out of the box" search engine might be quite sufficient for clinical record retrieval. One such task would amount to finding enough patients for a study in a very large clinical database – in this case, relatively high precision demonstrated by the automatic runs will ensure that a sufficient number of the found patients are eligible for inclusion, and the size of the database will ensure the needed number of patients is found. Unfortunately, we cannot judge the quality of the recall in this evaluation.

For our knowledge question, the answer seems to be that blindly adding knowledge to a general-purpose search engine significantly hurts its performance, but a domain specific engine is more powerful, especially when used by domain experts. We have to note that the significantly weaker performance of the Lucene manual run might be partially due to running queries developed specifically for Essie. As much as we tried to preserve the gist of the queries, some of it might have been lost in translation.

Finally, the third of the original cohort identification topics needed significant modifications and, in some cases, significant time spent on finding the right terms by domain experts. So the answer to our complete automation question is probably no, and maybe we should next focus on better presentation of the results for quick evaluation and simplification of the query syntax.

References

Apostolova E, Tomuro N, Demner-Fushman D. Automatic extraction of lexico-syntactic patterns for detection of negation and speculation scopes. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011, Volume 2, 283-287.

Aronson AR, Lang FM. An overview of MetaMap: historical perspective and recent advances. J Am Med Inform Assoc. 2010 May-Jun;17(3):229-36.

Huang X, Lin J, Demner-Fushman D. Evaluation of PICO as a knowledge representation for clinical questions. AMIA Annu Symp Proc. 2006:359-63.

Ide NC, Loane RF, Demner-Fushman D. Essie: A Concept Based Search Engine for Structured Biomedical Text. J Am Med Inform Assoc. 2007 May-June;14(3):253-263. .

Humphrey SM. Automatic Indexing of Documents from Journal Descriptors: A Preliminary Investigation. J Am Soc Inf Sci. 1999;50(8):661-674.

de Marneffe M-C, MacCartney B, Manning CD. Generating Typed Dependency Parses from Phrase Structure Parses. In LREC 2006 http://nlp.stanford.edu/pubs/LREC06_dependencies.pdf Accessed August 16, 2011

Mork JG, Bodenreider O, Demner-Fushman D, Dogan RI, Lang FM, Lu Z, Névéol A, Peters L, Shooshan SE, Aronson AR. Extracting Rx information from clinical narrative. J Am Med Inform Assoc. 2010 Sep-Oct;17(5):536-9.

Peters L, Bodenreider O. Using the RxNorm web services API for quality assurance purposes. AMIA Annu Symp Proc. 2008 Nov 6:591-5.

Richardson WS, Wilson MC, Nishikawa J, Hayward RS. The well-built clinical question: a key to evidence-based decisions. ACP J Club. 1995 Nov–Dec;123(3):A12–3.

Vincze, G. Szarvas, R. Farkas, G. Mora, and J. Csirik. 2008. The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. BMC bioinformatics , 9(Suppl 11):S9.