Siena's Clinical Decision Assistant

Michael Ippolito, Katherine Small, Clayton Marr,
Steven Gassert, Kylie Small and Sharon Gower Small

Siena College Institute for Artificial Intelligence
515 Loudon Road
Loudonville, NY 12211
mp08ippo@siena.edu, smallk1@hawkmail.newpaltz.edu, clmarr@vassar.edu,
gasserts1@hawkmail.newpaltz.edu, ka12smal@siena.edu, ssmall@siena.edu

Abstract

This paper discusses Siena's Clinical Decision Assistant's (SCDA) system and its participation in the Text Retrieval Conference (TREC) Clinical Decision Support Track (CDST) of 2015. The overall goal of the 2015 track is to link medical cases to information that is pertinent to patient care. Participants were given a set of 30 topics in the form of medical case narratives and a snapshot[1] of 733,138 articles from PubMed[2] Central (PMC). The 30 topics were annotated into three major subsets: diagnosis, test and treatment, with ten of each type. Each topic serves as an idealized representation of actual medical records and includes both a *description*, which contains a complete account of the patient visit, and a *summary,* which is typically a one or two sentence summary of the main points in the *description*. SCDA used several methods to attempt improve the accuracy of medical cases retrieved. SCDA used the metathesaurus Unified Medical Language System (UMLS)[3] that was implemented using MetaMap (NIH, 2013), query and document framing (Small and Stzalkowski 2004), a ranked fusion of document lists and Lucene for initial document indexing and retrieval. The track received a total of 178 runs from 36 different groups. We submitted three task A runs where our highest P(10) run was 0.3767 and two task B runs where our highest P(10) run was 0.4167. The highest P(10) from CDST TREC 2014[4] was 0.39. The word described here was performed by, and the entire SCDA system built by a team of undergraduate researchers working together for just ten weeks during the summer of 2015. The team was funded under the Siena College Institute for Artificial Intelligence's National Science Foundation's Research Experience for Undergraduates Grant.

1. Introduction

The Clinical Decision Support Track (Simpson et al., 2014) is a program in the Text Retrieval Conference (TREC) (Voorhees, 2007). TREC is a program co-sponsored by the National Institute of Standards and Technology (NIST) and the U.S. Department of Defense. It focuses on supporting research in information retrieval and extraction, and

[1] http://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/
[2] http://www.ncbi.nlm.nih.gov/pmc/
[3] http://www.nlm.nih.gov/research/umls/
[4] At this point we can only compare our 2015 results to last year's as all 2015 results have not been released. The 2014 track only had Task B runs.

increasing the availability of appropriate evaluation techniques. The Clinical Decision Support Track was run for the second time in 2015. There were two defined tasks for 2015 and participants were allowed to participate in either one or both. Task A required participants to retrieve documents from the PMC corpus that were relevant to the medical case narratives; this task is identical to the 2014 TREC track. Task B was new in 2015 and also required the retrieval of relevant documents but the treatment and test topics were further annotated with a "*diagnosis*" field.

The highest ranked articles for each topic submitted by the participants were pooled and judged by medical librarians and physicians trained in medical informatics. In particular, the judgment sets were created using two strata: all documents retrieved in ranks 1-20 by any run in union with a 20% sample of documents not retrieved in the first set that were retrieved in ranks 21-100 by some run. Assessors were instructed to judge articles as either "definitely relevant" for answering questions of the specified type about the given case report, "definitely not relevant," or "possibly relevant." The latter judgment may be used if an article is not immediately informative on its own, but the assessor believes it may be relevant in the context of a broader literature review.

2. TREC 2014 Literature Review

While designing the experimental procedure for this year's clinical support track the team reviewed a significant amount of literature from the previous year's track. The University of California, Los Angeles (UCLA) implemented the use of a manual run (Garcia-Gathright, et al., 2014). Their manual run utilized domain experts for query expansion. Our work utilized domain experts to annotate last year's queries to improve the performance of framing for our automatic runs. Similarly to UCLA, we also utilized MetaMap, UMLS and Lucene (McCandless et al., 2010). MetaMap is used to both relate biomedical text to the UMLS Metathesaurus and to flag Metathesaurus concepts that are present within biomedical texts. Lucene is a full text search engine library that is composed entirely in Java and is used to build the initial indices on the document corpus.

The NovaSearch (Mourão, et al., 2014) team explored both Ranked Fusion and utilizing the prestige of the retrieved journal to re-rank their results. The prestige of the journal article was used to increase relevance because they believed that a journal that was highly recognized for accurate information would be more likely to contain a document relevant to the query. Term frequency was developed by their domain experts in order to establish the relevance of different MetaMap semantic types and articles that displayed high frequency of relevant terms were ranked higher among articles that had lower frequencies. We utilized a similar methodology in SCDA.

San Francisco State University (Bhandari et al., 2014) also used MetaMap but they translated their case reports into a list of structured medical concepts. Instead of using this method, we utilized our framing technique to add structure to the first five paragraphs of each case report to automatically score the retrieved documents relative to our query.

3. The SCDA System Main Components

The main focus of the SCDA system was to use framing to create the simplest and most accurate query to provide to Lucene for a full-text search of the PubMed corpus. This meant initial manual analysis of the 2014 data by our domain expert to identify what aspects of the medical case reports were imperative to forming a query to return the highest quantity of relevant documents. This analysis was utilized to determine what aspects of the query we should automatically frame for the 2015 task. The remainder of this paper will discuss the modules of our SCDA system in detail as well as the results of our NIST evaluation.

3.1 Lucene Baseline

In order to run the initial retrieval on the corpus documents, Apache Lucene 4.0.0 was utilized to create an index for the corpus. Lucene is an open source search engine, written in Java, designed to function as a text search engine library.

Lucene was used to generate the baseline run of our system. Lucene has many built-in querying functions. During the indexing process, each document in the corpus was broken into four fields: title (including authors), abstract (null if none), body, and contents (the abstract and the body). When querying Lucene one can search the entire document or restrict its query to specific fields. Based on results obtained from the 2014 topics, the contents field provided the best results for our queries. Therefore, in our 2015 Task A run, the topic summaries alone were passed as individual queries. The search was restricted to the contents field and the top 20 documents were used in our baseline run.

In Task B, diagnoses were added to the query. Lucene allows multi-field queries, so a two-field query was passed. The first part contained the diagnosis and the second contained the topic summary. These fields can be weighted by certain degrees, but testing this on the 2014 topics did not change the documents returned, but only their scores. Likewise, changing the order of the fields in the query did not affect the documents returned. The Task B queries were not released until after our 10 week program completed. Therefore our domain expert added the diagnosis field manually and this is why we tagged that run as manual. It is important to note that the addition of the diagnosis filed was the only manual interaction in our Task B run.

3.2 The Framing Component

We added structure to our queries and our text passages in our framing component as can be seen in the example query frame and document passage frame below. Our frame attributes included: age, gender, time and symptoms. The diagnosis attribute was added for Task B only. In Figures 1-3 below we show a sample query frame for topic #22, where our P(10) = 1.0 as well as two data frames, one with a high score and one with a low score.

```
▼<topic number="22" type="treatment">
 ▼<description>
    A 65-year-old male with a history of tuberculosis has started to complain of productive cough with
    tinges of blood. Chest X-ray reveals a round opaque mass within a cavity in his left upper lobe. The
    spherical mass moved in the cavity during supine and prone CT imaging. Culture of the sputum revealed
    an organism with septated, low-angle branching hyphae that had straight, parallel walls.
  </description>
 ▼<summary>
    A 65-year-old male complains of productive cough with tinges of blood. Chest X-ray reveals a round
    opaque mass within a cavity in his lung. Culture of the sputum revealed fungal elements.
  </summary>
</topic>
```

Query Frame:

Topic Number	22
Age	Aged
Gender	Male
Time	None
Symptoms	productive cough, round opaque mass, cavity, (Coughing up phlegm) or (productive cough NOS), Sputum production, Coughing up phlegm, cavity, Dental caries-free, Nursing caries, Caries (morphologic abnormality), Dental caries extending into dentine, Gastrointestinal Diseases

Figure 1: Topic #22 and its corresponding frame – note the error made in keeping *cavity* as a symptom

Document Passage = A 22-year-old unmarried man presented to the chest outpatient department with a history of productive cough of two-month duration. He also complained of pain and swelling on the anterior aspect of right side of chest of one-month duration. Imaging studies of the thorax, including chest roentgenography and computerized tomography, revealed an unruptured lung abscess which had herniated into the chest wall. Culture of pus aspirated from the chest wall swelling grew Mycobacterium tuberculosis. He was diagnosed to have a tuberculous lung abscess which had extended into the chest wall, without spillage into the pleural cavity or the bronchial tree. Antituberculosis drugs were prescribed, and he responded to the treatment with complete resolution of the lesion.

Topic Number	22
Score	50.96
Document ID	3213720
Age	Adult
Gender	male
Time	Null
Symptoms	22-year-old unmarried man, history, productive cough, unruptured lung abscess, aspirated, chest wall swelling, diagnosed, tuberculous lung abscess, lesion, pain

Figure 2: Topic #22 high scoring data frame

Document Passage = This case series suggests that chronic candidal bronchitis is associated with significant morbidity and responds well to treatment. Such patients may benefit from extended antifungal therapy. Guidelines for the treatment of Candida in pulmonary secretions should be reevaluated.

Low Scoring Frame:

Topic Number	22
Score	0.0
Document ID	3527895
Age	ND
Gender	Null
Time	Chronic
Symptoms	candidal bronchitis

Figure 3: Topic #22 low scoring data frame

In order to create frames from queries and passages of text, the text was taken through a number of different steps. First, MetaMap was used on the text to generate a list of negated concepts. For example, upon processing the phrase *"cardiac arrest was ruled out"*, the function would add to the negated list any concept triggered in metamap for the frame *"cardiac arrest"*. Later, any concept in the candidate target concept list that matched a concept negated in the same phrase was removed. The text was further automatically modified to replace potentially problematic phrases, especially those that would cause problems for the parser (for example, the Latinate medical terminology "status post" was replaced with "after") based on a dictionary we generated from 2014 analysis.

The text was then run through the Stanford Parser, in order to detect semantic roles and relationships. The parser's output was stored as a set of hierarchical clauses. This clausal hierarchy was searched for words that triggered concepts using MetaMap. Using the typology of "semantic types" employed by MetaMap to categorize triggered concepts. If trigger concepts were found with one of eight designated types, the relevant concept was added to the symptom list variable for the frame of the larger given area of text. For example, the sentence *"64-year-old woman with uncontrolled diabetes, now with an oozing, painful skin lesion on her left lower leg"* would have, among its many triggered concept referents from Metamap's database, a concept referent for skin lesions, likely classed under the semantic class [anab] (Anatomical Abnormalities). Since [anab] is one of the designated semantic types for denoting symptoms, the noun clause containing it, *"oozing, painful skin lesion"* is added to the symptoms list.

Referring to the temporality typology suggested by the medical professionals employed by the UCLA team in 2014, our frame's time attribute functions to classify conditions into classes of "*acute*", "*progressive*" and "*chronic*". The text of each triggered symptom clause was searched for temporal wording describing the symptom, and if it was found, the appropriate time class was saved to the frame's time attribute.

Frame Scorer

After the Framing process was complete, SCDA had to rank each frame created by a document passage in order of its relevance to the query frame created by the topic. Our first scoring algorithm simply looked for equality of the contents of each frame attribute. The total score of the frame was then calculated as the average of the scores from each individual frame attribute.

Example of 1st Scoring algorithm:

Query Frame:

Gender	Female
Age	Child
Symptoms	Cough, Chest Pain, Left Lung Mass

Document Frame:

Gender	Undetected	Gender Score:	0
Age	Child	Age Score:	100
Symptoms	Cough, Chest Pain, Left Lung Mass	Symptoms Score:	100
		Total Score:	75

After several rounds of error analysis on the 2014 data we made a modification to our scoring algorithm. The improvement that we made to our scoring algorithm lies in the way we treated frames when certain data types were not populated. For example in the initial version of our scoring algorithm when the query frame detected the gender of the patient, and the document we were scoring it against did not mention a gender (or our frame builder failed to locate it), we would assign a score of 0 for the score for that frame attribute. In the updated version we did not assign a score of 0 to that data type but rather did not include that data type in the calculation of the final overall score for that frame.

Example of Revised Scoring algorithm:

Query Frame:

Gender	Female
Age	Child
Symptoms	Cough, Chest Pain, Left Lung Mass

Document Frame:

Gender	Undetected	Gender Score:	null
Age	Child	Age Score:	100
Symptoms	Cough, Chest Pain, Left Lung Mass	Symptoms Score:	100
		Total Score:	100

3.3 Fusion

"Fusion" here refers to the creation of a new ranking of relevant documents by using multiple previous relevancy lists. If the elements of the latter set of lists were compiled using effective but diverse methods, it can be expected that (if done well), a fused result list would be at least more accurate than the average, and in some cases the list produced from fusion may in fact be more accurate than any of the lists component to its creation. This may occur due to the "chorus effect" (Mourão, et al., 2013): if a document is listed

as highly relevant by various lists that were compiled differently, it is highly likely that it is indeed much more relevant, compared to a document that was judged to be highly relevant by only one method, which is more likely to have been so judged in error.

There are many different methods to fuse relevancy lists. Among these, we chose to use Reciprocal Rank Fusion (henceforth RRF) and the log ISR Fusion method (Mourão, et al., 2013).

Reciprocal Rank Fusion (RRF)
RRF has the dual advantage of being both effective and simple, being an unsupervised fusion method not requiring any machine learning, complex voting algorithm or reference to global information. All one needs to perform a reciprocal rank fusion is a set of lists organized in descending order by relevance. At the same time, it has been shown to outperform most comparable fusion methods(Cormack, et al., 2009).

Before creating the final fused list, RRF assigns a score to each document involved. This score is calculated as the summation of that document's score for each participating list. The document's score for a given list is **1 / (k + r)** where k is a constant and r is the at document's rank on the given list. We set the value of k equal to 60.0, which has been previously found to be optimal (Cormack, et al., 2009).

Logarithm ISR Fusion (LISRF)
The ISR Fusion method, and its logISR variant, were tested by (Mourão, et al., 2013) in TREC 2013. While not as popular as RRF, it shares a lot of the same qualities, being both simple and effective, and is calculated similarly. LISRF's method of generating a document's score for a given list differs however, being calculated as **$\log(nHits) / r^2$,** where nHits is the total number of participating lists that include the given document, and r is the document's rank in the list currently being scored for.

Ultimately, in NovaSearch's performance at TREC 2013, while LISRF was slightly outperformed by RRF for P(10) .366 to RRF's .37, it consistently outperformed RRF in that paper with regards to discounted cumulative gain.

Fusion algorithms in application
Two of our runs employed fusion, one for Task A and one for Task B (where diagnosis was already given). In Task B, we employed a reciprocal rank fusion on the baselines for Task A and B to generate another run for Task B. For Task A, the fusion was a more complicated, three step fusion. First, two auxiliary runs were fused using RRF. Both of these runs used methods not used elsewhere: one was essentially a baseline run with a weighted query was generated by a framing method that rewarded rare symptoms and temporal info, and gave a lower weight to demographic information and more common symptoms, while the other was a similarly boosted query that used MetaMap instead of framing to produce the list of search items. The fusion of these two was then fused using the log ISR method with the baseline Lucene search results, and this fusion list was in turn fused using RRF with the results of the run that employed framing of document abstracts.

3.4 UMLS Synonym Finder

SCDA used UMLS to augment our symptoms field with their synonyms in both the query frame and our document frames. Additionally, our domain expert generated a list of common symptoms that we would not want to expand on after analyzing the 2014 data. UMLS offers an easy to use API that allowed our team to effectively retrieve the synonyms of symptoms. After inserting the synonyms our P@10 tests on the 2014 data raised slightly.

4. The SCDA Architecture

In a standard run, we used Lucene as described above to generate a list of the top 20 documents for a topic. This list, containing document id's and scores, is passed to the Framer. The topic is framed to create a Query Frame, and each returned document's abstract is framed and then scored against the Query Frame. The Framer returns a second re-ranked list of the highest scoring documents based on their frame's score. Finally, the baseline Lucene list and the Framed list are passed to the Ranked Fusion component to generate one more ranked list.

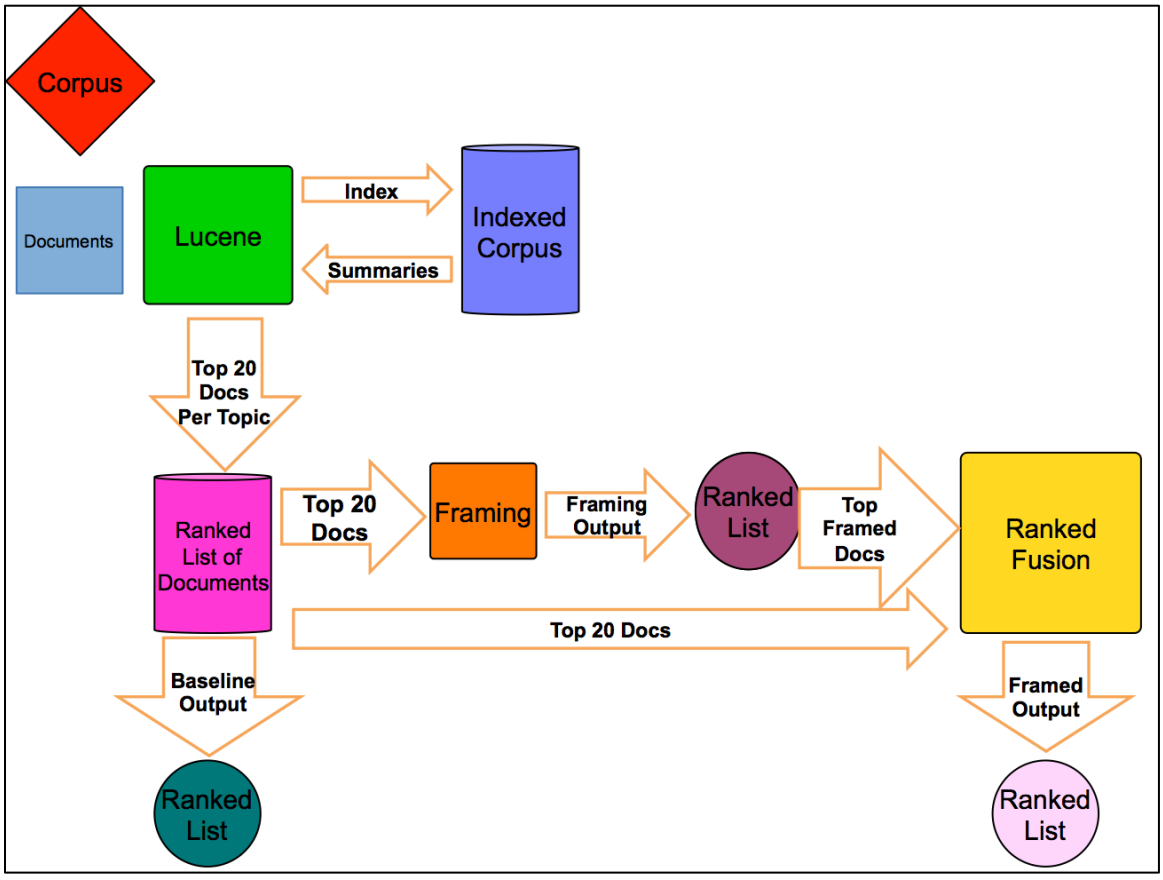

Figure4: SCDA Architecture

5. TREC Evaluations

The track received a total of 178 runs from 36 different groups. This set includes 92 automatic Task A runs, 11 manual Task A runs, 62 automatic Task B runs and 13 manual Task B runs. Task B runs used Task A topics (Figure 5) where the 20 test and treatment topics were augmented with a diagnosis field (Figure 6).

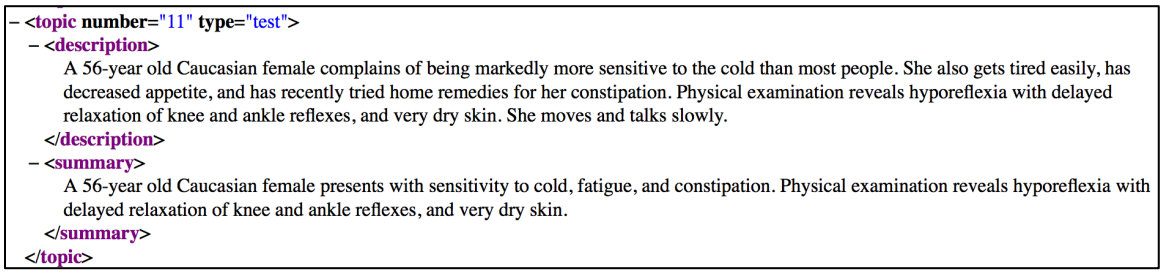

Figure 5: Topic 11 for Task A

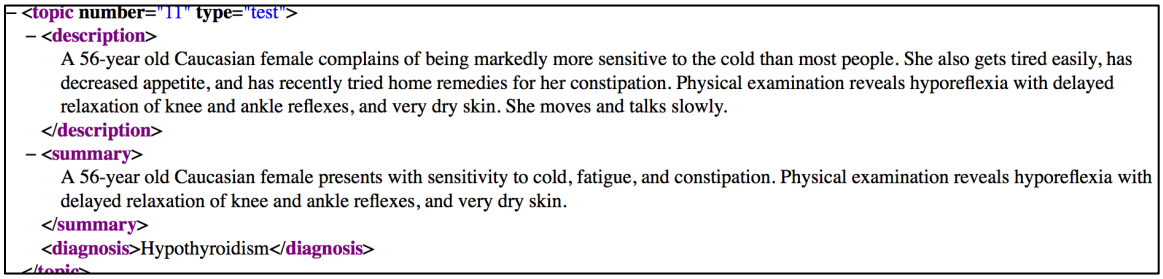

Figure 6: Topic 11 for Task B, augmented with the diagnosis field

SCDA Results

Our team submitted three automatic Task A runs and two manual Task B runs. The only feature of our Task B runs that were manual was the addition of the diagnosis field. We were required to add ours manually as our summer work ended prior to the official release of the Task B topics. Our domain expert augmented the Task A topics to add the diagnosis field. Our Task A runs consisted of our Lucene Baseline run: P(10) = 0.3767, our Framed Document Run: P(10) = .2667 and our Fusion Run: P(10) = .2900. Our Task B runs consisted of our Lucene Baseline run: P(10) = .4167 and our Fusion Run: P(10) = .4067.

Task A

Lucene Baseline:

1. Generate list of topic summaries
2. Run a Lucene search using each summary as a query to the Corpus index
3. Translate Lucene list of documents and scores into ranked list
4. Return ranked list as baseline

Framed Document Run:

Query frames were first built from each of the TREC topics. This frame would include age, gender, time and symptoms. The top 20 documents from the Lucene run that were retrieved for that topic would then also be processed. A frame would be created for each document based on its abstract. We would process the abstract and extract out the age,

gender, time and symptoms. Finally, the query frame and the document frame would be compared and the document frames scored as described above. The more similar the frames are the higher the score and the more relevant the document is to that particular topic. The documents were ranked based on their frame's score.

Fusion Run:
First, two auxiliary runs were fused using RRF. Both of these auxiliary runs used methods not used elsewhere: 1) a baseline run with a weighted query was generated by a framing method that rewarded rare symptoms and temporal info, and gave a lower weight to demographic information and more common symptoms, and 2) was a similarly boosted query that used MetaMap instead of framing to produce the list of search items. The fusion of these two was then created using the log ISR method with the results of the Lucene Baseline run, and this fusion list was in turn fused using RRF with the results of the Framed Document run.

Task B
Lucene Baseline:
The diagnosis field was added as a query to our summary. Both were weighted equally.

Fusion Run:
We used a reciprocal rank fusion to fuse the result lists of our respective Lucene Baseline runs for Task A and Task B, essentially using the fusion to assimilate the new information from the given diagnosis field into the old Lucene results list.

6. Conclusions

Framing
The complexity of this component and our ten-week time frame did not enable us to improve on this module following its initial implementation. We are currently performing a detailed error analysis on the 2015 data to determine how to improve on our framing component. This component should perform better than the baseline after our error analysis is complete and the component improved upon based on these findings.

Lucene Baseline
The Lucene queries over-performed based on the experiments we ran on the 2014 topics. Overall, the baseline list generated by the initial query was the best output of the system.

Five topics in the baseline did not return any relevant documents. Due to the nature of the system, in the situation where Lucene generates no relevant documents framing and ranked fusion cannot improve the output, (specifically, topics 18, 19, 20, 24, and 25).

In error analysis, topics with P(10) less than 0.3 were considered unsuccessful runs and anything with P(10) greater than 0.7 were considered successful. The five topics where P(10)=0.0 were examined first. A common thread between these topics was the use of semi-complex medical terminology: *dyspnea, bilateral edema, basilar crackles* (18), *dyspnea* (19), *myoclonic* (20), *dullness to percussion* (24), *osteolytic lesion, tachycardia, urinary incontinence* (25). When compared with topics 22 and 26 (P(10) = 1.0), the five

worst performing topics contain much more complex terminology. This is a common theme when examining the higher performance topics against the lower performing ones. For example, in topics 2 and 12(P(10) = 0.1), Lucene had to query with words such as *immunosuppressed, intranuclear, nuchal rigidity,* and *bronchoaveolar* with little to no other key terms. This is in heavy contrast to higher scoring topics such as 4 (p(10) = 0.7) and 19 (p(10) = 0.9) which include better querying terms like *human papilloma virus, hypertension,* and *acute stabbing chest pain* despite their included medical terms.

With the current architecture design, some relevancy needs to be established for each topic in order to later improve the score. It is possible that the version of Lucene used in our system (version 4.0.0) is not suited to handle some of these terms as it is not the latest release and was built to run with Java 6. Using a more recent release of Lucene could yield better results. Furthermore, the use of other indexing/querying software such as Indri might be very helpful. If other querying software yielded different ranked lists, this could be extremely helpful for ranked fusion and would add diversity to framing. The cost of adding more documents to be framed, scored, and compared (as well as the time necessary to re-index the Corpus) needs to be weighed against the convenience of having the system work with one software. Finally, returning a larger list of ranked documents in Lucene could improve the framing score but would greatly increase the time needed to frame and score documents. Lucene can be set to return any number of documents for a query. Therefore, if time allowed for the framing and scoring of the top 100 documents we may have found relevance after framing for our lower scoring topics – especially those with P(10) = 0.000.

Fusion

Our fusion method underperformed expectations based on the experiments we ran on 2014 data, and only outperformed the baseline on one topic (#7).

Overall, the Task A Fusion run compared:
-significantly (p=0.0004) worse than the baseline Lucene run (1 topic better, 15 worse, 14 the same)
-significantly (p=0.016) better than the abstract-frame-scored run (8 topics better, 1 worse, and 21 the same)

Because the reciprocal ranked fusion did worse (0.29) than its P(10) score when experimentally tested on the 2014 data (0.34), while the Lucene baseline improved from (0.31 to 0.3767), would suggest that the three-step RRF algorithm may have been overfit to the 2014 data. In 2015, while it did worse than the average Task A run and worse than the baseline, it did (likely) do better than the average of its ingredients, suggesting it wasn't a complete failure. At the same time, though, it failed the central goal of list fusion, to perform better than all the ingredients. What went wrong is most likely that the fusion was not given enough ingredients, and that its ingredients were not diverse enough. Reciprocal rank fusions perform best when they are given a much larger set of lists, and when these lists are compiled using diverse (but effective) methods. All the methods fused except the baseline used very similar methods, centering around the extraction of medically relevant information from the topic summary, and the ranking of document relevancy by the occurrence of these terms. In addition, all three of these

methods employed Lucene at some point in their processes. For better performance, one could propose additional inclusion of lists compiled by methods that did not rely on Lucene and/or center on medical term occurrence.

Furthermore, that the baseline performed best suggests that Lucene may have done all the necessary term relevancy preening by itself. All of these methods employed Lucene and then modified its search or its results, meaning they were actually decreasing the score by doing so.

Two topics stand out for unusual results. The first is topic 21, arguably the worst performance of the fused list.

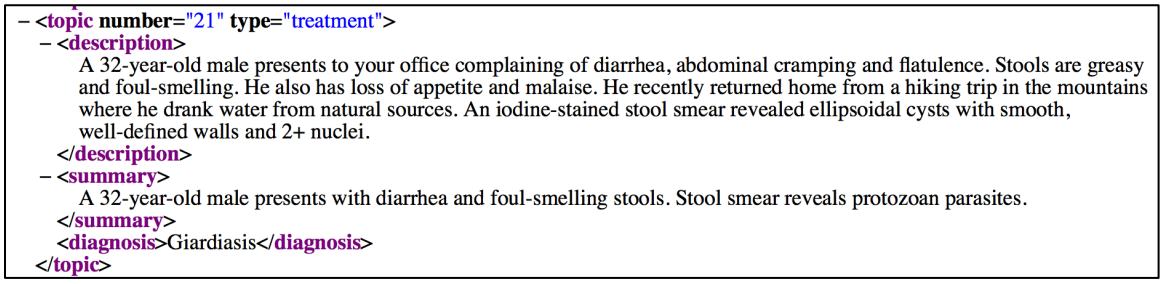

Figure 7: Topic 21 Task B

This is the only topic where the fused list scored worse (P(10) = 0.0) than the abstract-frame-scored list (P(10)= 0.1), both being under the baseline (0.4). The fused list scoring 0 here would suggest that the relevant documents returned by each of the other two were not the same ones and were also absent from the auxiliary ingredient lists FrameBoostedQuery and MetaBoostedQuery, as otherwise they would have made into the fused list. This would be the worst-case scenario for an RRF, which is built based on ingredient consensus: all the ingredient lists are so different they have no consensus at all (as opposed to being significantly different but having notable points of consensus, the best-case-scenario), causing the RRF to fail to make a list that is better than its ingredients. This analysis would also be consistent with the high variance of responses among different TREC participants for topic 21. In topic 7, meanwhile, the RRF outperforms the baseline.

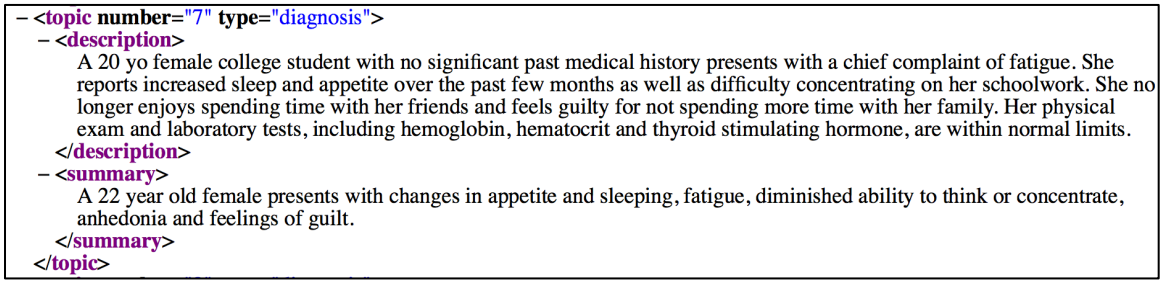

Figure 8: Topic 7 Task A

This was also the only topic where the framed list outperformed the baseline. Whether this is to be seen as a success of the RRF or not depends on the scores of the auxiliary

ingredients: if they were similarly better than the baseline, this would be a "success" attributable to the semantic framing theories employed by the three modified ingredient runs, whereas if they are not, it can be seen as a case where the RRF optimally fused its ingredients. Further analysis needs to be done.

7. References

Bhandari, Aayush, James Klinkhaer and Anagha Kulkarni. 2014. San Francisco State University at TREC 2014: Clinical Decision Support Track and Microblog Track. In Proceedings of The Twenty-Third Text Retrieval Conference (TREC 2014).

Cormack, G. V., Clarke, C. L. A., and Buttcher, S. 2009. Reciprocal Rank Fusion outperforms Condorcet and Individual Rank Learning Methods.

Garcia-Gathright, Jean I., Frank Meng and William Hsu. 2014. UCLA at TREC 2014 Clinical Decision Support Track: Exploring Language Models, Query Expansion, and Boosting. In Proceedings of The Twenty-Third Text Retrieval Conference (TREC 2014).

Mourão, André, Flávio Martins and João Magalhães. 2014. NovaSearch at TREC 2014 Clinical Decision Support Track. In Proceedings of The Twenty-Third Text Retrieval Conference (TREC 2014).

Mourão, André, Flávio Martins, and João Magalhães. 2013. NovaSearch at TREC 2013 Federated Web Search Track: Experiments with rank fusion. In Proceedings of The Twenty-Second Text Retrieval Conference (TREC 2013).

McCandless Michael, Erik Hatcher and Otis Gospodnetic. 2010. Lucene in Action. Second Edition. Manning Publications.

National Library of Medicine (NLM). 2013. MetaMap- A Tool For Recognizing UMLS Concepts in Text. Software.

Simpson, Matthew S., Ellen M. Voorhees and William Hersh. 2014. Overview of the TREC 2014 Clinical Decision Support Track. In Proceedings of The Twenty-Third Text Retrieval Conference (TREC 2014).

Small, Sharon and Tomek Strzalkowski. *2004. HITIQA: A Data Driven Approach to Interactive Analytical Question Answering.* Proceedings of Human Language Technology Conference. Boston, Massachusetts.

Voorhees, Ellen M. 2007. Overview of TREC 2007. In Proceedings of The Sixteenth Text Retrieval Conference (TREC 2007).