The OHSU Biomedical Question Answering System Framework

A. M. Cohen[1], J. Yang[1], S. Fisher[2], B. Roark[2], and W.R. Hersh[1]

Department of Medical Informatics and Clinical Epidemiology[1],
Department of Computer Science and Electrical Engineering, OGI School of Science and Engineering[2]
Oregon Health & Science University, Portland, OR, USA

ABSTRACT

The Oregon Health & Science University submission to the TREC 2007 Genomics Track approached the entity list question answering task using a modular object oriented system framework. A system object coordinates a collection of processing objects into a pipe that constructs a set of queries, retrieves passage, and then processes those passages into a final output answer set. Using the framework we applied multiple levels of synonym expansion and a ranked series of topic queries with a range of specificities in order to retrieve all of the likely relevant passages with the most likely ranked higher. We then applied sentence pruning to the head and tail of each passage using both NLP and term-based techniques. Overall scores finished around the TREC Genomics mean for each of the four measures. Careful passage retrieval, including synonym expansion and multiple query construction, as well as sentence pruning was essential in achieving acceptable performance on this task.

1 INTRODUCTION

The 2007 Text Retrieval Conference (TREC) Genomics Track consisted of a single question answering task, which was scored by four measures. The question answering task presented a biological question where the expected answer was to include one or more entities of a given type. There were 36 separate question topics covering 14 different entity types. These topics were collected by surveying working biologists who were asked to provide questions from their own research. The topics were rephrased into questions which asked for a specific entity type. The challenge was to extract text passages specifically answering each topic question from a large corpus of over 160,000 biomedical full text articles selected from journals known to publish papers on genomics research. The answers for each topic were pooled and then evaluated by domain experts to create a gold standard. Each submission was scored and averaged over all of the topics for document mean average precision (MAP), character-based mean passage precision, and aspect precision.

2 BACKGROUND

With the ever-expanding knowledge base of science embedded in the biomedical literature, the importance of effective text processing tools to aid biomedical researchers continues to increase (Cohen and Hersh, 2005, Hunter and Cohen, 2006, Roberts, 2006). One way that the biomedical literature is used by scientists is to determine what is currently known about a subject related to their field of research. For automated systems to aid scientists with this task, the task must be framed in a formal manner amenable to machine processing. One formal, yet flexible way to represent these kinds of information needs is as *list entity* type questions.

In list entity questions, the answer to the information need includes one or more entities (or things) of a given type. The type is given as part of the question and can be used by the information system to enhance the ability of the system to address the question. Since the type of the concepts of interests that answer a given question are often obvious to the biologist from the question itself, this form of information need representation can be both useful to the biologist and convenient for the computer.

For example, suppose a biologist is interested in understanding what treatments have been tested for Alzheimer's disease in mouse models. Assuming that the treatments are pharmacological agents, the question can be represented as "What [DRUGS] have been tested in mouse models of Alzheimer's disease?". The list entity type is given in uppercase, surrounded by square brackets. This is both easy for the user to input and provides quite a bit of information and context for the question answering system to work with.

Useful answers to these types of questions will include not just a list of the relevant entities, but also the surrounding text passage. The user then reviews the passages, determine which provide substantiated entities and answers, and understand the meaning and relationships present within the answer to the question. A good set of results will contain both a high proportion of substantiated and correct answers, as well as covering a high proportion of the different entities that answer the question. For example, if the question is "What centrosomal [GENES] are implicated in diseases of brain development?" and there are ten genes involved, then ten correct passages all about the same gene are less desirable than ten passages each about one of the involved genes.

The TREC 2007 Genomics track task attempted to emulate this list entity question-based type of information need in a controlled, comparable form, using a specific

full text literature corpus. The literature corpus was the same as used in the 2006 Genomics track and consisted of 162,259 full text HTML articles, published between the years 1995 and 2006, and downloaded with the permission of the publishers from the Highwire Press web site (http://highwire.stanford.edu/). This literature corpus includes 49 journals that were known to publish articles on genomics subjects. However, the articles in the corpus included everything available from the web site, not just genomics articles.

Using the given literature corpus, the task was set up to be a extraction-based question answering task with the additional requirement that each relevant passage must include at least one correct entity of the given type. Questions were given in the form as shown above, as sentences with the entity type given in square brackets. Answers were required to come from contiguous passages in the fixed literature corpus. Systems were to submit a ranked list of up to 1000 character offset passages for each of the 36 topic questions. The character based passage specified the PubMed ID (PMID), starting offset in characters, and length of passage corresponding to the passage within the HTML file being nominated as relevant to answering the question.

Submissions were free to use any size passage they desired, whether that be sentences, sentences fragments, paragraphs, etc. However, the maximum allowable passage was restricted by enforcing the rule that a passage could not include any HTML <P> or </P> tags. This was the same maximum passage criteria used in the 2006 Genomics track. This effectively limited submitted passages to around one paragraph of text, although entire reference sections from the end of papers sometimes were included by this rule. A file of the maximum legal spans was provided by the track administrators.

The submission for a system consisted of a ranked list of up to 1000 passages for each topic. A character-based gold standard set of answers was created from the submissions by using pooling, combined with expert judging. Submitted answers were mapped to their containing maximum legal span, and then were pooled for judging by taking the top ranked spans from each entry until 1000 spans were collected for each topic.

Human judges with expertise in biology were then asked to rate each pooled span for relevance, and select the relevant answer text from a plain text version from the pooled HTML span. These plain text selections were then mapped back to the original HTML file using a string alignment algorithm to create the gold standard set of passages. Each gold standard passage also had assigned to it by the judges one or more entity terms. The set of entity terms were chosen by the judges when reviewing the passages for relevance, and then assigned to each relevant passage in a second review pass. This ensured consistent application of the entities with each topic question.

Each system was scored four ways, each measure being a variation of mean average precision (MAP). The first measure was DOCUMENT MAP. This took the highest ranked passage for a document as the document rank. The document MAP was computed in the standard way from this ranked list of PMIDs

The second and third measures used were character-based passage MAP scores, which measured the cumulative overlap between characters in relevant and nominated passages at each point of correct passage recall. The PASSAGE measure was the same as used in the 2006 Genomics track, and measured the fraction of relevant characters averaged over each nominated passage. The PASSAGE2 measure was added this year. It measures the average fraction of relevant characters averaged over each nominated character. In essence, every character is a little document, and is relevant or not. This measure is somewhat more robust to passage length and was intended to replace PASSAGE as the primary character measure this year. These measures evaluate the proportion of retrieved text that is relevant to answering the user's question. For example, a passage score of 0.50 would imply that approximately half the characters in the retrieved passage was included with a correct answer to the topic questions.

The fourth measure was ASPECT MAP. This took the highest rank of a passage with a given assigned entity term as the recall rank for that entity term. This measures the completeness of the range of coverage of the system output. Systems that nominate passages that mention more distinct entities higher in the rankings score higher than those that nominate passages with little entity coverage. Note that since, for some questions, some entities may be mentioned very frequently and others hardly at all, the ASPECT MAP measures something distinct from DOCUMENT and PASSAGE MAPs, but is just as important to the user.

Having all of these separate measures allows a more detailed study of what algorithms and approaches are helpful in proving the different important qualities necessary for a good list entity answer extraction system. While good document-based MAP is essential, all by itself it does not aid the user much since full text papers can be long and time consuming to read. Character-based MAP allows users to focus their reading and time on only the sections most likely to be helpful. Finally, aspect-based MAP measures the ability of a system to provide broad coverage of a topic. Together these separate measures characterize the effectiveness of the various systems and techniques that can be applied to list entity-based for biomedical question answering.

3 SYSTEM AND METHODS

The Oregon Health & Science University submission to the TREC 2007 Genomics Track approached the question answer extraction task as an opportunity to create a modular, object oriented framework for building question answering systems. The system is build as an assemblage of modules, implemented as objects using the Python programming language, also using that language to interface to other systems or libraries as necessary.

The framework consists of a ***QASystem*** object, which coordinates the actions of a group of processing modules in a pipeline. The ***QASystem*** object creates and maintains a ***Blob*** object that collects and stores all the accumulated results of the processing objects as *[name: value]* pairs, where the name is a string and the value can be any object. A processing object gets its input data from the ***Blob*** object and also stores any output in the blob. The final results for each application of the information extraction system to a topic are also stored in the blob.

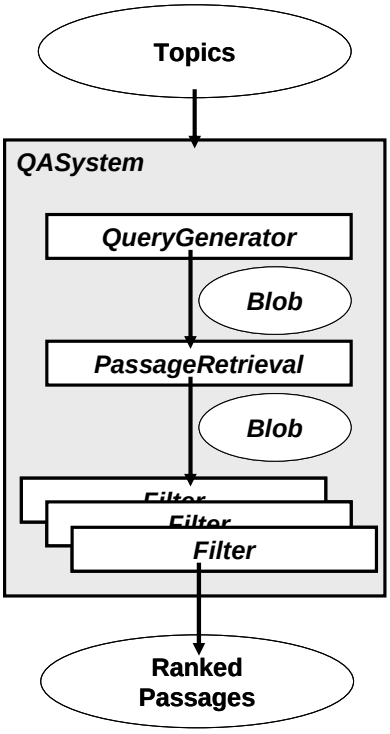

Figure 1. Question answering framework architecture.

The first object coordinated by the ***QASystem*** is the ***QueryGenerator***, which takes as input the topic question as given by the TREC Genomics Track topics file, and produces a series of one or more queries. The results of this are then passed to a ***PassageRetrieval*** object, which applies the queries and returns a set of retrieved passages. The ***QASystem*** then applies any number of ***Filter*** objects which can filter out passages, re-rank the passages, trim or edit passages, or in fact, manipulate the output of the ***PassageRetrieval*** object in any way necessary. The final result of the pipeline processing is a nominated set of ranked passages.

The overall architecture of the pipeline framework is shown in Figure 1. For the initial implementation of the system, there is only one filter module and each of the modules does its job in a fairly straightforward manner that will be described in the next section.

3.1 Query Generation

The goal of the ***QueryGenerator*** module is to transform the topic question into a Boolean expression for processing by ***PassageRetrieval*** module later in the pipeline. The ***QueryGenerator*** used in this version of the system actually creates two queries, a more specific primary query, and a less specific secondary general query. Each query is build off of term groups, where each term group is a list of terms generated from a word or phrase found in the topic question itself. The primary query essentially requires the search to match, at least one term from each term group (the term groups are ANDed), and the secondary query does not (the term groups are ORed).

Term groups are generated in several ways. First, a greedy search is done from the topic question to an indexed version of the MeSH terminology. The MeSH Substances database was included as a source of synonyms for one version of our system. Matching terms are expanded with MeSH synonyms into term groups. Second, gene names are matched and expanded using a dictionary extracted from Entrez and expanded using an orthographic variant generator (Cohen, 2005). Third, each entity type has a set of manually compiled associated keywords and these are used to generate a term group along with orthographic variations. Synonyms that included a parenthetical expression were separated as separate terms. Two part phrases separated by commas were split and put back together in inverted order as additional synonyms. Finally, stop words were removed.

3.2 Passage Retrieval

The passage retrieval phase was built upon the Python implementation of the open-source text retrieval engine Lucene (http://lucene.apache.org). The text corpus was parsed into maximum legal spans, and then each span was separately indexed by Lucene. Once this initial indexing was finished, we used a PassageRetrieval module in our framework to query the index and retrieve the potentially relevant passages. This step used the primary and secondary queries that were built by the ***QueryGenerator*** module. If the primary query did not return the maximum allowed 1000 passages per question, the secondary query was used as a back-off strategy. Passages returned by the

primary query were ranked in tf*idf order as computed and returned by Lucene. The tf*idf of passages returned by the secondary query were filter so that there were no duplication with the primary query and then scaled by the minimum tf*idf score from the primary query and then merged into a ranked list. This ensured that the passages returned by the primary query ranked higher than those of the secondary. Using this method we retrieved the maximum number of passages (1000) for each topic.

3.3 Passage Filtering

The final processing module trimmed the retrieved passages using MMTx Metamap (Aronson, 2001) as named entity recognition engine. Similar techniques were used by other participants in the 2006 TREC genomics track (Demner-Fushman *et al.*, 2006). Each passage retrieved by the previous stage was split into sentences, and each sentence was scored for matching entities. The UMLS semantic types of entities that were scored varied based on the entity type given in the initial topic question. This list was assembled by hand and represents just an initial implementation. Certainly once the TREC results are available as training data, each individual semantic type could be evaluated as to its usefulness for each entity question type.

MMTx natural language processing is complex and runs rather slowly. While the earlier stages of the system ran quickly and essentially in real-time, the MMTx based filtering slowed down the system dramatically, requiring overnight runs. Because of this, the system implementation made two approximations when performing sentence scoring. First, each passage was processed by MMTx and the found entity types and phrases returned. Then the passage was split into sentences and these sentences were matched against the extracted phrases to count how many times entity phrases occurred in each sentence. This count was used as the sentence score.

The second approximation was to only process the first 100 ranked passages. This is because the pooling used by TREC would not likely reach below the 30^{th} or 40^{th} passage, and therefore there was reduced incentive to spend a lot of time processing passages that were less likely to be relevant and unlikely to be scored.

After sentences were scored, the module trimmed the passage using the following method. The average count over all the sentences was computed and any first or last sentence in the current passage that had a count less than half the average was trimmed off. This process was repeated on the newly trimmed passage until no more sentences could be trimmed.

Trimmed passages were not re-ranked by this phase, that is, passages were ranked by their original tf*idf score as for the full untrimmed passage. The only exceptions to this are for passages that had a total count score of zero, these passages were demoted to the end of the ranking. The EXCLUSIONS version of our system removed passages that appeared to be keyword or abbreviations sections. All of our submitted runs were automatic, with no manual query generation, and no tuning or modifications to the system based on the results or retrieval of initial runs on the official topics. We performed some basic system debugging and sanity-checking using the training topics given with the track protocol description.

4 RESULTS

Performance was determined by the official scoring program for the track, trecgen2007_eval. The results of our three official runs and several subsequent evaluation runs are presented in Table 1. For comparison, the high, low, and mean scores for all runs submitted to the track are shown for each of the three measures.

The runs labeled OHSUQA, OHSUQASUB, and OHSUQASUBEX are our officially submitted runs. These runs were output from the system as described above, with a few small variations. The OHSUQASUB and OHSUQASUBEX runs include the MeSH substances database in the synonym generation step, while the OHSUQA run does not. The OHSUQASUBEX run includes a few simple pattern matching rules to filter out passages labeled “keywords” and “abbreviations”. This was found to improve performance a bit on some training data we created based on last years Genomics track task.

Table 1. System performance on all measures averaged across all topics.

RUN	DOC MAP	PASSAGE MAP	PASSAGE2 MAP	ASPECT MAP
OHSUQA	0.1719	0.0403	0.0440	0.1075
OHSUQASUB	0.1684	0.0388	0.0434	0.1080
OHSUQASUBEX	0.1695	0.0392	0.0439	0.1104
LuceneOnly	0.1450	0.0263	0.0317	0.1003
Lucene+Backoff	0.1602	0.0294	0.0354	0.1088
3StageBackoff	0.1900	0.0364	0.0432	0.1291
3StageBackoff+TermTrim	0.1897	0.0415	0.0460	0.1286
TREC MIN	0.0329	0.0029	0.0008	0.0197
TREC MEDIAN	0.1897	0.0565	0.0377	0.1311
TREC MEAN	0.1862	0.0560	0.0398	0.1326
TREC MAX	0.3286	0.0976	0.1148	0.2631

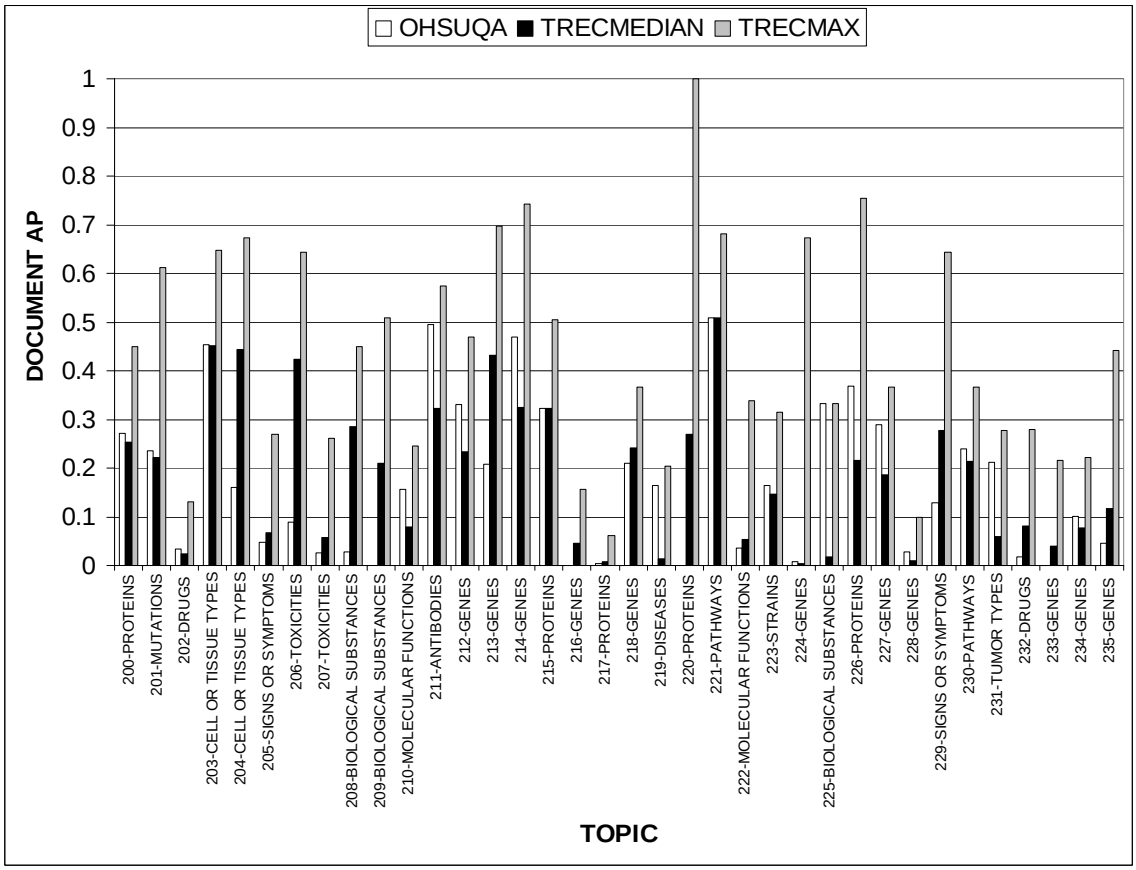

Figure 1. DOCUMENT average precision on each topic for the OHSUQA system, compared to TREC MEDIAN and MAX for automatic runs. The x-axis shows both the topic number and the required list entity type for that topic.

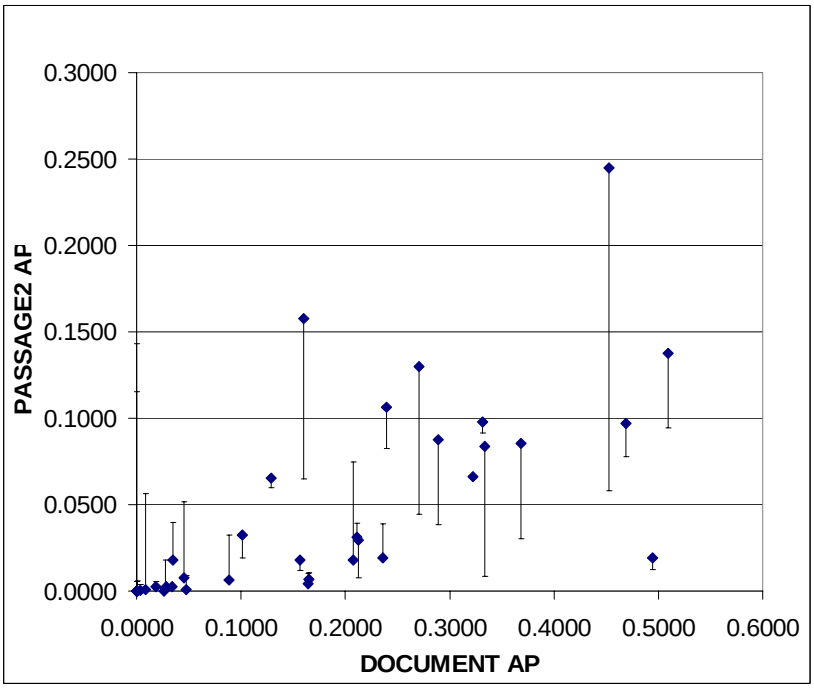

Figure 2. PASSAGE2 average precision versus DOCUMENT average precision on each topic for the OHSUQA system shown as diamonds, and also showing the relationship between document and passage performance, with the mean average PASSAGE2 for all submitted runs shown as a bar with a connecting line to the corresponding topic score.

In the table, below the official submitted runs, we show results for several variations of our system that remove specific features and implementation decisions. The row labeled LuceneOnly implements a ***QueryGenerator*** that only produces a primary query, and uses the ***PassageRetrieval*** module, again with no passage filter modules. The row labeled Lucene+Backoff includes the ***QueryGenerator*** and ***PassageRetrieval*** modules including the primary and secondary queries, but does not include any passage filter modules. For comparison, the bottom four rows of the table show overall summary results for all runs officially submitted to the TREC Genomics track this year.

Figure 1 shows the Average Precision (AP) for each topic comparing the results of the OHSUQA system to the TREC MEAN and MAX runs. The x-axis shows both the topic number and the required list entity type for that topic. In general the document AP performance is somewhat less than the best systems, but usually at least as good as or better than the median, but with many exceptions, such as topics 209, 216 and 220.

Figure 2 shows the relationship between document and passage performance for our OHSUQA system. For the most part the per-topic PASSAGE2 average precision of our system is better than the average of submitted TREC runs. However, for about eight topics the document retrieval performance is very poor, and this is directly reflected in the passage retrieval score. This is especially true for topics 209 and 220, where the average submitted PASSAGE2 scores are 0.1430 and 0.1156 respectively, but our system failed to retrieve any documents.

Because of this strong correlation between PASSAGE2 and DOCUMENT performance, it seemed likely that the best way to improve overall performance would be to refine our initial passage retrieval strategy. While our originally submitted runs used a simple two-stage primary query and secondary backoff strategy, it was clear that for some topics the primary query was too specific, and the secondary too general. Furthermore, by inspecting the generated queries for topics such as 200, it was clear that sometimes our synonym expansion system did not find any synonyms for important topic concepts, especially for genes and proteins.

In our submitted system runs we assumed that we could increase precision by limiting the source of gene and protein synonyms to a manually selected set of species. This assumption turned out to be incorrect, and the system failed to generate enough gene and protein synonyms. We improved the synonym generation of our system by including an expanded synonym dictionary generated from all species included in the Entrez gene file.

We also extended the query generation and passage retrieval to use a three stage (instead of two stage) query system. The primary query was generated as in our submitted system, but a new secondary query was built by requiring at least one term from the term group with the most members. The tertiary query was identical to the secondary query in the previous system. Therefore we have three queries with a range of specificity going from very specific (requiring at least one synonym from each term group), to less specific (requiring at least one synonym from the largest term group), to very general (no terms required). All passages were ranked by tf*idf score as before.

The results of this extended system are shown in Table 1 as the run labeled ThreeStageBackoff. The DOCUMENT and ASPECT scores are improved over the baseline OHSUQA system. The PASSAGE score are worse, as would be expected since no passage trimming is used in the ThreeStageBackoff system.

Finally we wanted to determine whether the time consuming MMTx-based NLP was worth the cost. To the ThreeStageBackoff system we added a simple sentence trimmer that works much like the MMTx based trimmer above, but instead of counting indentified UMLS concepts this system simply counts the number of identified query terms in each sentence. Sentences with less than half the average number of query terms are pruned from the head and tail of the passage as with the MMTx-based approach. These results are shown in the table as the run labeled “3StageBackoff+TermTrim” This system produced PASSAGE scores equal to or better than the MMTx-based system.

5 DISCUSSION

Several things are made clear from Table 1. A single automatically generated specific query was not sufficient to retrieve all of the relevant passages. The two-stage strategy was better than a single query, and the three-phase query was better than the two stage. However, these staged queries are dependent upon having good synonym dictionaries in order to recognize concepts of interest and also to require terms for those concepts in the query. Our synonym dictionaries were automatically generated from MeSH and Entrez with no manual editing except for a small stop list of excessively common words (e.g. “protein”). For some of the topics, the synonym dictionary “misses” resulting in very poor DOCUMENT scores, and therefore poor PASSAGE and ASPECT scores. Since our indexing using was composed of the maximum legal spans, poor DOCUMENT scores are approximately equivalent to poor paragraph retrieval in our system. Studying the effect of the indexing passage size on document performance is an area that needs further study. It would be useful to understand how performance is affected by single sentence indexing, as well as by including leading/trailing paragraphs or MeSH

terms when indexing maximal length spans. Single sentence indexing would have the advantage of not requiring subsequent passage trimming..

The passage trimming does help somewhat, as evidenced by the PASSAGE and PASSAGE2 scores for the 3StageBackoff+TermTrim system. It is not clear that NLP-based passage trimming offers better potential than simple synonym term based trimming. However, the configuration and tuning of the NLP-based passage trimming is complex, and will require much further work to determine which UMLS semantic types are most informative about sentence relevance for each entity type. The semantic types used in the current system were determined entirely by inspection. Leave one-out comparisons for each query and each semantic type could be used to determine which semantic types are appropriate for each entity type.

Finally, Figure 1 makes it clear that our passage retrieval performance is well below the top retrieval for all automatic systems. This is an area where there is a major opportunity to improve our system. Modifications to our approach that could be advantageous include the use of MeSH terms when indexing the document corpus and performing the queries, as well as a more sophisticated means of generating synonyms. The multi-stage query approach could also incorporate the use of hypernyms and hyponyms into the search strategy.

6 CONCLUSIONS

The TREC Genomics track list entity question answering extraction task was a new one for this year, modified from last year by providing the entity type. Like last year, this task pushed the envelope of genomics biomedical information retrieval, but this year models a real-world use case more closely, and covers a broader range of question and entity types than ever before. The current gold standard collection will now enable further research and tuning of genomics list entity question answering systems. This will likely result in further system performance improvements that will motivate both task oriented studies and motivate actual use of these systems by biomedical researchers as part of their regular workflow.

ACKNOWLEDGEMENTS

This work was supported in part by Grant ITR-0325160 from the National Science Foundation.

REFERENCES

Aronson, A. R. (2001) Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proc AMIA Symp*, 17-21.

Cohen, A. M. (2005) Unsupervised gene/protein entity normalization using automatically extracted dictionaries. In *Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, Proceedings of the BioLINK2005 Workshop*.

Cohen, A. M. and Hersh, W. (2005) A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, **6**, 57-71.

Demner-Fushman, D., Humphrey, S. M., Ide, N. C., Loane, R. F., Ruch, P., Ruiz, M. E., Smith, L. H., Tanabe, L. K., Wilbur, W. J. and Aronson, A. R. (2006) LHNCBC-2006-072 Finding Relevant Passages in Scientific Articles: Fusion of Automatic Approaches vs. an Interactive Team Effort. *Proc TREC*, **569**, 76.

Hunter, L. and Cohen, K. B. (2006) Biomedical language processing: what's beyond PubMed. *Mol Cell*, **21**, 589-94.

Roberts, P. M. (2006) Mining literature for systems biology. *Briefings in Bioinformatics*, **7**, 399.