

# Phrasal Queries with LingPipe and Lucene

Bob Carpenter

*Alias-i, Inc.*

`carp@alias-i.com`

October 25, 2004

## Abstract

The hypothesis we explored for the Ad Hoc task of the Genomics track for TREC 2004 was that phrase-level queries would increase precision over a baseline of token-level terms. We implemented our approach using two open source tools: the Apache Jakarta Lucene TF/IDF search engine (version 1.3) and the Alias-i LingPipe tokenizer and named-entity annotator (version 1.0.6). Contrary to our intuitions, the baseline system provided better performance in terms of recall and precision for almost every query at almost every precision/recall operating point.

## Thesis

We hypothesized that including phrasal queries would improve precision of ad-hoc queries. To explore this hypothesis, we submitted two systems for the ad hoc query task of the genomics track of the 2004 TREC Conference. The first was a baseline system implementing standard TF/IDF-based search over a tokenized document collection. The second included quoted phrase-level queries extracted automatically using statistical models.

## Topics and Documents

As with other ad hoc tasks in TREC, the input for evaluation was a list of topics of a fairly detailed variety. These were gathered from practicing research biologists and were meant to cover a range of topics. The first of the fifty used for the 2004 ad hoc genomics task is shown in Figure 1 (with line breaks inserted in place of some spaces and around XML text content segments for human readability).

The goal of the evaluation was to find relevant “documents”, in this case among a ten year subset of completed MEDLINE citations.

```

<TOPIC>
  <ID>1</ID>
  <TITLE>Ferroportin-1 in humans</TITLE>
  <NEED>
    Find articles about Ferroportin-1, an iron
    transporter, in humans.
  </NEED>
  <CONTEXT>
    Ferroportin1 (also known as SLC40A1; Ferroportin 1;
    FPN1; HFE4; IREG1; Iron regulated gene 1; Iron-regulated
    transporter 1; MTP1; SLC11A3; and Solute carrier family 11
    (proton-coupled divalent metal ion transporters), member 3)
    may play a role in iron transport.
  </CONTEXT>
</TOPIC>

```

Figure 1: Topic 1 for 2004 TREC Ad Hoc Genomics Task

These documents were down-converted to ASCII from the UTF-8 of the originals. Roughly 4.6 million completed citations were used as the data set, comprising roughly 9.6 million bytes of data. An example of a citation is shown in Figure 2.

The fields we indexed are shown in Figure 3.

Two of these fields were assigned by hand by curators at NIH before marking a citation as completed (only completed citations were used in this evaluation). The MeSH fields contain terms drawn from the Medical Subject Heading index, a standardized nomenclature for medicine. The MeSH terms preceded by asterisks are central topics for the article. The chemical identifiers and names are drawn from a couple of standardized sources.

Other available information that we did not index includes dates, journal titles and issues broken out (not shown), dates of labeling and status of labeling (not shown), names of authors, ISSN of journals, type of publication, place of publication, etc.

## Tokenization and Filters

Tokenization for biomedical literature is notoriously problematic, as illustrated nicely by the three forms taken by the gene that is the focus of topic 1: “Ferroportin-1”, “Ferroportin1”, and “Ferroportin 1”. We employed LingPipe’s default tokenizer for English, the `IndoEuropeanTokenizer`.

PMID- 14757427

TI - The ferroportin disease.

AB - A new inherited disorder of iron metabolism, hereafter called "the ferroportin disease," is increasingly recognized worldwide. The disorder is due to pathogenic mutations in the SLC40A1 gene encoding for a main iron export protein in mammals, ferroportin1/IREG1/MTP1, and it was originally identified as an autosomal-dominant form of iron overload not linked to the hemochromatosis (HFE) gene. It has distinctive clinical features such as early increase in serum ferritin in spite of low-normal transferrin saturation, progressive iron accumulation in organs, predominantly in reticuloendothelialmacrophages, marginal anemia with low tolerance to phlebotomy. Ferroportin mutations have been reported in many countries regardless of ethnicity. They may lead to a loss of protein function responsible for reduced iron export from cells, particularly reticuloendothelial cells. Now, the disorder appears to be the most common cause of hereditary iron overload beyond HFE hemochromatosis.

FAU - Pietrangelo, Antonello

LA - eng

PT - Journal Article

PT - Review

PT - Review, Tutorial

PL - United States

RN - 0 (Cation Transport Proteins)

RN - 0 (metal transporting protein 1)

RN - 9007-73-2 (Ferritin)

SB - IM

MH - Cation Transport Proteins/\*genetics

MH - Ferritin/blood

MH - Human

MH - Iron Metabolism Disorders/diagnosis/\*genetics/pathology

MH - Iron Overload/diagnosis/etiology/genetics/pathology

MH - Metal Metabolism, Inborn Errors/diagnosis/\*genetics/pathology

MH - Support, Non-U.S. Gov't

SO - Blood Cells Mol Dis 2004 Jan-Feb;32(1):131-8.

Figure 2: Relevant Fields of MEDLINE Citation 14757427

<i>Field</i>	<i>Desc</i>	<i>Boost</i>
PMID	PubMed Identifier	-
TI	Title	4
AB	Abstract	1
MH	MeSH	2
MH	*MeSH	4
RN	Chemical Identifiers and Names	1

Figure 3: Citation Fields

This tokenizer employs a fine-grained tokenization that breaks on just about any non-number-internal punctuation, but leaves alpha-numeric sequences intact. For instance, “Ferroportin-1” becomes three tokens, “Ferroportin”, “-”, and “1”. In contrast, “Ferroportin1” generates a single token, whereas “Ferroportin 1” generates two tokens. In retrospect, given the prevalence of this space/no-space problem, we should have broken on alpha-numeric boundaries, turning “Ferroportin1” into two tokens.

The Lucene search engine supports tokenization, stoplisting, etc. through extensions of the class `lucene.analysis.Analyzer`. An analyzer returns a token stream given a string and the name of a field. We used the same tokenization for every field of the citation, producing a single field of results. In so doing, we repeated the content the number of times specified by the boost column in Figure 3. This is a cheap-and-dirty way of massaging term frequencies; a cleaner implementation would’ve worked with a fielded index and fielded queries. The results would be identical up to some phrasal overlap at boundaries; to avoid such spurious phrases, we inserted dummy tokens between repetitions of content, and between the distinct MeSH and chemical terms.

We applied two filters to the token stream. First, we used the `LowerCaseFilterTokenizer` to convert all characters to lowercase in both queries and the index. Second, we applied LingPipe’s standard stop list using `EnglishStopListFilterTokenizer`. The full list of stop terms is available in the documentation.<sup>1</sup> We also filtered out query tokens consisting only of punctuation. But, we left the punctuation in the index and left the punctuation in compound terms.

---

<sup>1</sup>A good candidate for a stop list can be found on the NIH PubMed help page at: <http://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhhelp.html>

## Extracting Queries

The topics consist of three distinct sections: title, need and context. We tokenized each section, and boosting the scores of needs by a factor of 2 and of the title by a factor of 4. Like the boosting for documents, without the availability of training data, these numbers were chosen by means of a so-called “wild guess”. We have no idea if fiddling with the weights helped or hurt queries.

The Lucene query facility allows arbitrary terms to be boosted. These boost numbers are multiplied into TF-IDF scores before ranking outputs.

We constructed our Lucene queries programmatically, constructing a `Query` object out of `Term` objects. The terms were created by tokenization, and the queries were Lucene `BooleanQuery` objects, with the clauses being neither required or forbidden. This lack of positive/negative marking results in TF-IDF scoring being applied.

## Extracting Phrasal Query Terms

We used our own open source software, LingPipe, to extract so-called “named entities” from topics. For instance, for the topic listed in Figure 1, the term “metal ion transporter” is analyzed as a term denoting an atom.

LingPipe’s entity extraction is based on a Bayesian generative model that tags each token as being the beginning of a named entity, a continuation of a named entity, or not in a named entity. In our generative model, we break the entire sequence probability down using the chain rule, generating a token/tag pair based on the previous token/tag pairs. History is limited to a finite window of one previous tag and two previous tokens. The chain rule is used again to predict first the tag and then the token given the tag. Maximum likelihood estimates are generated using the labeled training data found in the GENIA corpus.<sup>2</sup>

A first-best hypothesis is extracted using dynamic programming (a slight generalization of the Viterbi decoder for HMMs to higher order models). Extraction throughput is roughly 100,000 tokens/second, making it by far the slowest part of the system.

---

<sup>2</sup>The GENIA corpus contains 600,000 tokens of data drawn from MEDLINE abstracts. Sequences of tokens are labeled with tags representing about 40 types of biologically relevant entity. The GENIA corpus distinguishes among molecular types (DNA vs. RNA vs. protein), as well as molecular structure (molecule family vs. macro-molecular structure vs. molecule vs. sub-region, etc.). In addition, other entities are included such as cell lines, organisms, and a generic “other” category.

In addition to “metal ion transporter”, the phrase “iron transport” is properly labeled as “other name,” the type assigned to processes such as transportation. But there were errors in type, such as “Ferroportin1” being labeled as an other name rather than a gene. Unfortunately, the long phrase “SLC40A1; Ferroportin 1; FPN1; HFE4; IREG1; Iron regulated gene 1; Iron-regulated transporter 1; MTP1; SLC11A3” was misrecognized as a DNA region, and “Solute carrier family 11 (proton-coupled)” (with the mismatched parentheses) is labeled as a protein family. The first topic, with its word salad of terminology, presents a difficult case; other cases for which performance was better are listed below.

For each phrase found, we included an additional clause in the query with a cumulative boost of an additional factor of 4. On the high side, a phrasal query matching a core MeSH term or a term in the title would be boosted by a factor of 16; a simple token matching an abstract has a boost factor of 1.

Interestingly, most of the queries were fairly terminologically poor in that they did not include many interesting named entities. An example where the entity recognizer fared well was in the need “Find correlation between DNA repair pathways and oxidative stress.” for which LingPipe found the relevant terms “DNA repair pathways” and “oxidative stress”. Other examples of correct terms extracted are “DNA repair”, “skin-carcinogenesis”, “TOR signaling”, “UV-carcinogenesis”, “mouse kidney”, “morphological changes”, “gene expression”, “signal-transducing molecule”, “nerve growth factor pathway”, “*Saccharomyces cerevisiae*”, “BCL2-interacting molecules”, “anti-p53 monoclonal antibody DO1”, and even “Sleeping Beauty transposons”. Single token phrases are also extracted; good examples include “NEIL1”, “Smad4” and “TGFB”.

Incorrectly extracted terms that were too long included the ones listed above, as well as “Determine binding affinity”, which should just be “binding affinity” and “BUB2/BFA1”, which should be two items (we think). Common mistakes include over-running punctuation, as in “Mental Health Wellness 1 (MWH1”, which should be “Mental Health Wellness 1” and “MWH1” for the gene and its acronym.

Incorrectly extracted terms that were too short included “mice” as opposed to the correct “hairless mice”. In general, LingPipe tends to err on the side of increased length.

Missed terms included the “inhibitors” in “Human gene BCL-2 antagonists and inhibitors”, though “Human gene BCL-2 antagonists” was properly extracted. LingPipe also missed “double-stranded DNA breaks”, but this noun is not an entity in the GENIA ontology, so this isn’t a mistake per se.

A difficult case is posed by expressions such as “Glyphosate tolerance gene sequence”, which was properly extracted, but contains the

subterm “Glyphosate tolerance” that would also be nice to have in an ad hoc query.

Several of the topics were rather less entity-specific, such as “Risk factors for stroke”, from which no entities were extracted.

Held out analysis on the GENIA corpus, which only closely matches a subset of the MEDLINE abstracts, indicates a roughly 70 percent accuracy rate in finding appropriate terms. We do not care about labels here, but we do care about boundaries. Phrases that are too long wind up being too restrictive and those that are too short are too permissive.

## TF-IDF

We used the default TF-IDF weighting in Lucene, as implemented by the class `DefaultSimilarity`. Lucene provides a rich mechanism for monkeying with the TF-IDF scoring factors. The default term frequency (TF) implementation takes the square root of observed frequency. The default inverse document frequency (IDF) implementation takes  $\log(numDocs/(docFreq+1))+1$ . Documents are normalized to unit length by dividing each term weight by the square root of the sum of the squared weights. Boosts are implemented as multiplicative factors.

The upshot of using a standard TF-IDF setup is that misrecognized phrases will simply drop out of the computation. Properly recognized phrases should provide a solid boost to results.

## Execution Speed and Size

The same index was used for both experiments. It took roughly five hours to produce the index, which also stored the original titles, abstracts, MeSH terms, and chemical names. The resulting size of the index was approximate 9GB.

Evaluating queries took about 15 minutes to return 1000 documents per 50 topics. The time for phrase extraction was an insignificant fraction of this time, taking less than a second.

## Results

The baseline system scored above the median result for almost every query, and above the phrase-based system for every query. The collective results are reported in Figure 4, Figure 5, and Figure ??..

Eval	Base	With Phrases
Retrieved	50000	50000
Relevant	8268	8268
Rel-ret	4635	4140

Figure 4: Unranked Performance

Prec at Recall	Base	With Phrases
at 0.00	0.8013	0.7387
at 0.10	0.5561	0.5016
at 0.20	0.4886	0.4155
at 0.30	0.4314	0.3594
at 0.40	0.3548	0.2771
at 0.50	0.2792	0.2277
at 0.60	0.2324	0.1956
at 0.70	0.1929	0.1552
at 0.80	0.1478	0.1230
at 0.90	0.0954	0.0943
at 1.00	0.0536	0.616
Average	<b>0.3094</b>	<b>0.2656</b>

Figure 5: Interpolated Recall-Precision and Average Precision

Documents	Baseline	Phrases
5 docs	0.5800	0.5240
10 docs	0.5380	0.4800
15 docs	0.5293	0.4560
20 docs	0.4960	0.4330
30 docs	0.4627	0.4087
100 docs	0.3458	0.3030
200 docs	0.2572	0.2240
500 docs	0.1515	0.1320
1000 docs	0.0927	0.0828
R-Precision	<b>0.3515</b>	<b>0.3187</b>

Figure 6: Precision at Doc Counts and R-Precision

## Discussion

Our initial hypothesis, that phrasal terms would help with precision, was fairly thoroughly disproved in the context of the TREC ad hoc genomics evaluation. Although there are a handful of queries for which the phrases helped, their impact was otherwise negative.

We are still uncertain why our results were negative. The score for documents matching a phrasal query are boosted, and those not matching are left unchanged. Thus the only effect we anticipated was higher precision for documents containing relevant terms.

We see several possible explanations. Although term extraction is performing at state-of-the-art levels, this may not be good enough. But the only way it would have a negative effect is if ill-formed terms were actually found in some documents and it boosted their scores. Another explanation is that the short lengths of MEDLINE citations leaves little chance for the terms to disambiguate.

Our qualitative evaluation of term extraction shows that it's performing as well as can be expected. One possible explanation is the relative shortness of the MEDLINE citations. We believe that with full documents, the chance occurrence of phrase fragments without the entire phrase showing up will be much higher. Even so, this does not explain why the phrases didn't provide some boost for citations in which they were found. Another possibility is that the phrases boosted precision for finding that phrase, but the true distinguishing features were elsewhere in the abstracts.

## Resources

All of the resources used for carrying out these experiments are available from the following web sites:

TREC	<a href="http://trec.nist.gov">trec.nist.gov</a>
TREC Genomics	<a href="http://medir.ohsu.edu/~genomics">medir.ohsu.edu/~genomics</a>
LingPipe	<a href="http://www.aliasi.com/lingpipe">www.aliasi.com/lingpipe</a>
Lucene	<a href="http://jakarta.apache.org/lucene">jakarta.apache.org/lucene</a>
GENIA	<a href="http://www-tsujii.is.s.u-tokyo.ac.jp/~genia">www-tsujii.is.s.u-tokyo.ac.jp/~genia</a>
MEDLINE	<a href="http://www.nlm.nih.gov/databases/databases_medline.html">www.nlm.nih.gov/databases/databases_medline.html</a>
PubMed	<a href="http://www.ncbi.nlm.nih.gov/entrez">www.ncbi.nlm.nih.gov/entrez</a>