Ranked Set Search in Medline Documents

Kristin Mittag[1], Alexander Hinneburg[2]

[1]OntoChem GmbH, Halle
[2]Informatik, Martin-Luther-University, Halle-Wittenberg, 06099 Halle
{mittag,hinneburg}@informatik.uni-halle.de

Abstract: Information needs like searching scientific literature that involve high recall rates are difficult to satisfy with ad hoc keyword search. We propose to state queries implicitly by the specification of a set of query documents. The result of such a query is a set of answer documents that are ranked within the answer set. We describe efficient techniques to process such queries. Preliminary experiments using data from the TREC Genomics track 2005 are reported.

1 Introduction

Searching literature in life sciences for relevant publications is a difficult task. The diversity of the scientific vocabulary and ambiguous naming conventions for entities make boolean keyword search less effective in retrieving all papers relevant to a given information need. Controlled vocabularies like MESH-terms and meta-information mitigate the problem. However, constructing a boolean query to retrieve for example all randomized clinical trials on a particular subject that are present in Medline still requires a lot of manual tuning by search experts. Thus, potential knowledge hidden in publication databases will be difficult to access for life science researchers in case search experts for the particular problem at hand are not available.

A typical situation in searching scientific literature is that some papers on a particular subject are easy to find. Those papers may serve as starting point for further searching. However, it is difficult to find all relevant publications on the subject. One strategy is to use the known relevant papers and search for similar papers for each of them, maybe by using the PubMed's related article feature [LW07]. It precalculates the probability that a user wants to see a document given interest in another through a probabilistic topic model based on Poisson distributions over term frequencies. This strategy is used by [LA98] where they use the related article feature to update a bibliography. But it still requires much manual effort – they run their algorithm several iterations and in every round the bibliography gets updated by user-selected relevant articles.

Searching a document collection based on a given set of relevant papers had been formulated as a two-class document classification problem[SA05, FBSS$^+$09, GvdL05]. The papers that are found in the beginning are used as examples to train a classifier. Random documents from the document collection pose as background class. The trained classifier is then applied to all documents in the collection to distinguish relevant documents from the background, thus, it is a computationally expensive operation. Such a classifier may be a set of discriminating words, derived by comparing probability distributions of training set and background collection. The problem is that many training documents (several thousands) and a well defined background are required. Thus, this technique is best applicable to large document classes and can for example be used to support document annotation with MESH-terms.

We propose a new search technique that does not require manual query construction and it needs a relatively small set of query documents (5 to 50). In addition, it works on top of standard information retrieval technology thus query processing is very efficient. Our new approach requires the user to specify the search query as a few query documents relevant to the information need at hand. Literally spoken, such type of query could be interpreted as the task: Read the given documents, find the subset of the document collection containing all related documents and rank them by relevance. Stating a query for a complex information need by picking a few example documents is convenient and relieves the user of difficult tasks like iteratively constructing lengthy boolean keyword queries. Instead, the information need is given implicitly by the combination of selected query documents. The result of the search is not only a ranking of all documents in the collection like in standard information retrieval systems but our method also decides how to limit the answer to a reasonable subset of documents. Thus, it delivers a ranked subset of documents as an answer. We call this problem *Ranked Set Search (RSS).*

Our contributions are that we first propose a formal definition of the new RSS problem. Second, we develop efficient techniques to compute a solution. Finally, we report preliminary experiments on real data from the TREC Genomics evaluation track 2005 that demonstrate superior effectiveness of our approach over several baselines. In the remainder of the paper, first, we develop the formal problem definition as well as efficient search techniques in Section 2. In Section 3, we present preliminary experiments and discuss the result.

2 Ranked Document Set Retrieval

In the sequel, we describe the new framework of ranked set search. The given data consists of a collection of documents $D = \{d_1, \ldots, d_N\}$. A query is a set of query documents $D_q = \{q_1, \ldots, q_R\}$ that may or may not be part of the document collection D. The output is a set of documents $D_a \subset D$ that is a subset of the document collection. The individual documents in D_a are ordered according to a distance function, called **document distance** $dist_{doc}$, that determines the ranking of the documents within the answer set. The document distance function defines a distance between a document $d \in D$ and the query D_q. Thus, the document distance also induces a ranking of documents over the entire collection D.

The answer set D_a is usually a relatively small subset of D. The answer set is chosen among the possible subsets of D such that a second distance function, called **set distance** $dist_{set}$, becomes minimal. In order to avoid to search among all possible subsets of D, which might be very time consuming in case of general set distances, we restrict the search for an answer set with minimal set distance to the top-k documents of the ranking induced by document distance. The definition of ranked set search summarizes our concept.

Definition 1 *Given a collection* $D = \{d_1, \ldots, d_N\}$ *and a set of query documents* $D_q =$

$\{q_1, \ldots, q_R\}$, ***Ranked Set Search (RSS)*** *determines an answer set* $D_a \subset D$ *such that*

$$dist_{set}(D_q, D_a) = \min_{1 \leq k \leq k_{\max}} \left\{ dist_{set}\big(D_q, top_k(D, D_q, dist_{doc})\big) \right\}$$

with $dist_{set}(\cdot, \cdot)$ *is a distance function between sets of documents,* $dist_{doc}(d, D_q)$ *is a distance function between a document* $d \in D$ *and the set of query documents and* $top_k(D, D_q, dist_{doc})$ *delivers the subset of the top-*k *documents of* D *with respect to the ranking induced by* $dist_{doc}(d, D_q)$*.*

Ranked set search does not require the specification of a parameter like k that limit a ranking to the subset of the top-k documents. Instead, the size of the answer set is controlled by the set distance function. How to choose a meaningful set distance? Computing set distance as the minimum, maximum or the average of the document distances of the top-k documents is not a promising way. The set distance would be either constant (minimum) or monotonically increasing with k (maximum, average). Thus, the minimization of such set distances would not yield interesting solutions.

A class of set distance functions, which does not lead to trivial solutions, is to compare the word probability distributions of the query set D_q and the top-k subsets of the ranking of D. A word probability distribution in the simplest form is a multinomial distribution over the vocabulary W, which assigns a probability to every word $w \in W$ and the probabilities of all words sum up to one.

Kullback-Leibler-Divergence between two probability distributions x and y over the vocabulary is a measure with sound information-theoretic basis:

$$KL(x||y) = -\sum_{w \in W} x(w) \ln y(w) - x(w) \ln x(w)$$

KL-divergence has a nice information-theoretic interpretation. It measures the additional average amount of bits required to code a document following a word distribution x with a coding scheme designed for documents following a distribution y instead of x. We denote the word probability distribution of the query document set D_q by P_q and the distribution of the answer set D_a by P_a. KL-divergence is zero, when both distributions are identical, otherwise it is larger zero. All word distributions are estimated as multinomial distributions. To compensate for the word burstiness phenomenon[1], we use idf-transformation of the multinomials [MKE05].

However, KL-divergence is not symmetric, i.e. $KL(P_q||P_a) \neq KL(P_a||P_q)$. What version should be used? To understand the impacts of both versions, we look at the active vocabularies W_q and W_a that belong to D_q and D_a respectively. $KL(P_q||P_a)$ is small when all words of W_q are also included in W_a. Words of W_q that are missing in W_a create large overhead, because they are considered very infrequent according to distribution P_a. Vice versa, words that are in W_a but not in W_q do not increase $KL(P_q||P_a)$ by much. Thus, minimizing $KL(P_q||P_a)$ favors answer sets that obey $W_q \subseteq W_a$. The opposite version $KL(P_a||P_q)$ does not care much about words of W_q that are missing in W_a. It is much more sensitive to new words that do not appear in W_q but in W_a and tries to avoid such new words. Thus, minimizing $KL(P_a||P_q)$ favors answer sets that obey $W_q \supseteq W_a$.

[1] Burstiness says that a word that occurs once in a document has higher probability to appear again [Kat96].

method	greedy	TF-IDF	KL 1	KL 2	KL 3	KL 4
MAP	0.92	0.45	0.39	0.56	0.58	0.59
R-Precision	0.81	0.44	0.49	0.59	0.60	0.60
size	77	24735	24	96	269	678

Table 1: Mean Average Precision (MAP), R-Precision and size of result-set for different methods using 100 samples in all experiments.

We make use of both versions. First, we minimize the set distance $KL(P_q||P_a)$. Thus, W_a contains as many as possible words of W_q and a few additional related words. We use the answer set as *pseudo relevance feedback* and expand the query with it $D_{q'} = D_q \cup D_a$ to generalize the vocabulary of the original query D_q. In a second step, we minimize the opposite version, the set distance $KL(P_a||P_{q'})$ with respect to the expanded query $D_{q'}$. Using the answer set of the second step as well as pseudo relevance feedback, whole two-step procedure can be repeated. In our experiments, we compare KL-based RSS methods that run the two steps one and two rounds respectively.

Finally, we discuss the document-distance that compares a single document with the whole set of query documents. A document $d \in D$ is close to the query set D_q, when d contains most of the words in W_q. Therefore, we use the first version of the KL-divergence $KL(P_q||P_d)$ as document distance that defines the ranking. P_d is the word distribution of the single document d. Because, standard information retrieval systems do not support that kind of distance by an index structure, we use sampling and standard TF-IDF rankings to approximate the KL-based ranking. For the approximation, we draw several samples from the word distribution P_q. Each sample consists of a set of randomly drawn words and is used as keyword query that is submitted to a standard retrieval system. The system delivers a TF-IDF ranking of documents for each keyword sample. The final KL-based ranking is computed by merging the TF-IDF rankings with respect to the document distance $KL(P_q||P_d)$. An example is given at the end of Section 3. Notice that the document distance changes after pseudo relevance because D_q does change. In our preliminary experiments, we did not recompute the keyword samples and the TF-IDF rankings but just continued the merging with the changed document distance.

3 Experiments

We evaluate KL-based RSS on the TREC Genomics collection (ad hoc retrieval task 2004+2005) [HCY$^+$05]. That is a 10-year subset of Medline (1994-2003) with about 4.5 million documents. In preliminary experiments we used a single topic (ID 131) out of 50 topics, from TREC Genomics 2005 ad hoc retrieval task. It contains 42 documents, which are judged relevant by human evaluators. For each document, abstract and title were preprocessed using *Apache Lucene* (`http://lucene.apache.org`). In the preprocessing, we used stemming and filtered stopwords as well as infrequent words.

We evaluate our method against a baseline search method that uses TF-IDF rankings. The method merges the TF-IDF rankings of the sampled keyword queries using the TF-IDF score of the retrieved documents. For efficiency reasons we restricted the TF-IDF ranking

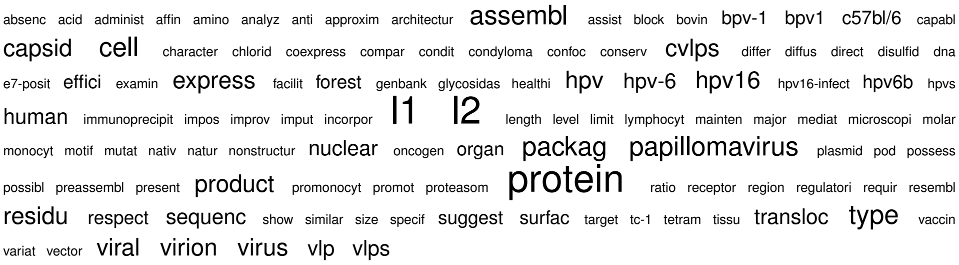
absenc acid administ affin amino analyz anti approxim architectur assembl assist block bovin bpv-1 bpv1 c57bl/6 capabl
capsid cell character chlorid coexpress compar condit condyloma confoc conserv cvlps differ diffus direct disulfid dna
e7-posit effici examin express facilit forest genbank glycosidas healthi hpv hpv-6 hpv16 hpv16-infect hpv6b hpvs
human immunoprecipit impos improv imput incorpor l1 l2 length level limit lymphocyt mainten major mediat microscopi molar
monocyt motif mutat nativ natur nonstructur nuclear oncogen organ packag papillomavirus plasmid pod possess
possibl preassembl present product promonocyt promot proteasom protein ratio receptor region regulatori requir resembl
residu respect sequenc show similar size specif suggest surfac target tc-1 tetram tissu transloc type vaccin
variat vector viral virion virus vlp vlps

Figure 1: Keyword cloud for our example run.

of each query to 1000 documents. When a document occurs in more than one TF-IDF ranking, the highest score is used. The result is a single ranking of all documents from the TF-IDF rankings.

We also developed an algorithm to compute an upper bound of the search performance to check the open potential of our method. This algorithm needs to know the relevant documents of a RSS query beforehand, thus, it can be run only on the evaluation data set. The problem is to cover all relevant documents in the TF-IDF rankings of the sample keyword queries by prefixes of minimal lengths. We map that problem to the set cover problem, a standard NP-complete problem, and compute an approximate solution by an greedy algorithm.

In each test run, we randomly selected 10 documents as query set from our test topic 131 and took the remaining 32 documents as relevant documents. Performance is measured by *Mean Average Precision* (MAP) and *R-Precision*, which are standard IR performance measures and produce consistent results [BV00]. MAP averages the precisions at each relevant document in the result set and takes the mean of the average precisions over all queries. R-Precision takes the precision at position $R = number\ of\ relevant\ documents$ (32 in our case) that is the first position where recall 1 is possible. All results are averages over 50 repetitions.

Table 1 shows results of the baselines and two KL-based RSS-algorithms. The KL-based RSS-algorithm that minimize the two versions of KL-Divergence one time (KL 1) has comparable R-Precision as the baseline (TF-IDF) but small MAP. The KL-based RSS-algorithm that repeat the minimization two times (KL 2), however, achieves a significant improvement over the baseline in both R-Precision as well as MAP. Additional repetitions (KL 3, KL 4) only marginally improve MAP and R-Precision. On the other hand, the result of the upper bound (optimal) shows that there is still room for improvement based on new techniques.

The task of topic 131 is to find documents about genes "*L1* and *L2* in the *HPV11* virus" and the "role of L2 in the viral capsid". HPV is an abbreviation for *human papillomavirus* and L1 and L2 are the names of two proteins of the viral capsid and their respective genes. Our method to specify queries by documents instead of keywords also helps to automatically identify selective keywords. The keyword cloud (Figure 1) illustrates the frequency of keywords in the keyword query samples that contribute to the final result. Words with bigger font size are found more often in those samples and could be relevant for the topic. For example *papillomavirus* is part of the acronym HPV11. HPV6 and HPV16 are dif-

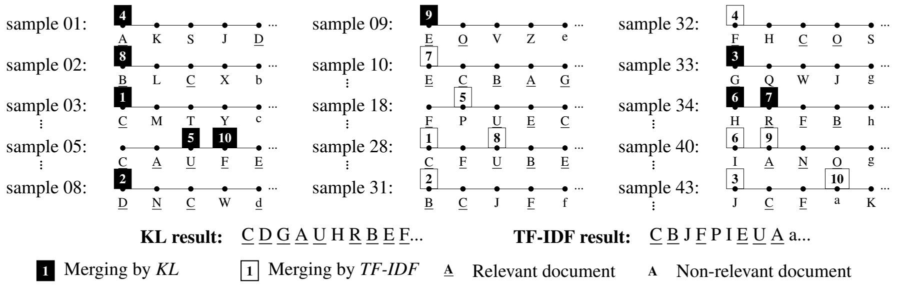
sample 01: 4 A K S J D ...
sample 02: 8 B L C X b ...
sample 03: 1 C M T Y c ...
sample 05: C A 5 U 10 F E ...
sample 08: 2 D N C W d ...
sample 09: 9 E O V Z e ...
sample 10: 7 E C B A G ...
sample 18: F 5 P U E C ...
sample 28: 1 C F 8 U B E ...
sample 31: 2 B C J F f ...
sample 32: 4 F H C O S ...
sample 33: 3 G Q W J g ...
sample 34: 6 H 7 R F B h ...
sample 40: 6 I 9 A N O g ...
sample 43: 3 J C F 10 a K ...
KL result: C D G A U H R B E F...
TF-IDF result: C B J F P I E U A a...
■ Merging by KL
□ Merging by TF-IDF
A Relevant document
A Non-relevant document

Figure 2: Top-5 documents of the TF-IDF rankings of the keyword query samples are shown that contribute to the final results of *KL* and *TF-IDF*. Documents are identified by letters.

ferent types of HPV, as well as HPV11. The two proteins L1 and L2 are expressed in the infected cell, in which new viruses are assembled.

Figure 2 illustrates the merging of the TF-IDF rankings for the first 10 documents of the final results of KL-based RSS and the baseline TF-IDF on a real example. Different rankings contribute to the results of KL-based RSS and TF-IDF. KL-based RSS has only a single non-relevant document among the top 10, while TF-IDF selected four non-relevant documents.

We conclude that stating a query implicitly by a set of query documents helps to take variations of terms as well as related terms into account. Our preliminary experiments show that RSS is a promising approach to searching life science literature.

References

[BV00] C.Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *SIGIR '00*, pp. 33–40, 2000.

[FBSS$^+$09] J.-F. Fontaine, A. Barbosa-Silva, M. Schaefer, M. R. Huska, E. M. Muro, and M. A. Andrade-Navarro. MedlineRanker: flexible ranking of biomedical literature. *Nucl. Acids Res.*, 37(suppl_2):W141–146, 2009.

[GvdL05] T. Goetz and C. W. von der Lieth. PubFinder: a tool for improving retrieval rate of relevant PubMed abstracts. *Nucleic Acids Res*, 33(Web Server issue), 2005.

[HCY$^+$05] W. Hersh, A. Cohen, J. Yang, R.T. Bhupatiraju, P. Roberts, and M. Hearst. TREC 2005 Genomics Track Overview. In *TREC 2005*, 2005.

[Kat96] S. M. Katz. Distribution of content words and phrases in text and language modelling. *Nat. Lang. Eng.*, 2(1):15–59, 1996.

[LA98] X. Liu and R. B. Altman. Updating a Bibliography Using the RELATED ARTICLES function within PubMed. In *Proc. AMIA Symp*, pp. 750–754, 1998.

[LW07] J. Lin and J. W. Wilbur. PubMed related articles: a probabilistic topic-based model for content similarity. *BMC Bioinformatics*, 8(1), 2007.

[MKE05] R. E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the Dirichlet distribution. In *ICML '05*, pp. 545–552, 2005.

[SA05] B. P. Suomela and M. A. Andrade. Ranking the whole MEDLINE database according to a large training set using text indexing. *BMC Bioinformatics*, 6(1), 2005.