SetSearch: A Set-Oriented and Entity-Aware Search System for Biomedical Literature

CS512 Project Report, Spring 2017

Jinfeng Xiao, Wenzhuang Chi, Peifeng Jing

ABSTRACT

Literature search is a fundamental step for many biomedical research projects. Despite the extensive studies of general document retrieval, finding highly relevant, properly ranked list of papers in scientific literature remains an open problem. In this demo, we present SetSearch, a set-oriented and entity-aware biomedical literature search system. SetSearch (i) integrates a data-driven text mining pipeline for entity detection and typing, (ii) adopts a novel entity-aware ranking algorithm which captures relations within *entity set*, and (iii) provides a clear interface for result visualization. Given an entity set query, SetSearch will first detect entities with types in that query, and return a ranked list of papers which are expected to capture the co-occurrence of entity pairs and the association between different entity types. Current version of SetSearch system has included over 26 millions paper abstracts in PubMed and 2.27 millions full-text papers in PubMed Central. In three multi-keyword search cases, SetSearch outperformed PubMed and Google Scholar in terms of the percentage of relevant papers in the top 20 results. In a systematic evaluation with 100 queries on a 5-million-paper subset of PubMed, SetSearch performed significantly better than BM25. We expect that the SetSearch philosophy can also benefit literature search in other domains.

1 INTRODUCTION

As a fundamental step in conducting research, literature search enables researchers to identify relevant papers about a topic and summarize essential claims about a particular issue. Considering the constantly-growing volume of scientific publications, a good literature search engine is essential to researchers, especially in the biomedical domain where the literature collection is so massive, diverse, and rapidly evolving—few people can master the state-of-the-art comprehensively and in-depth.

PubMed[1] [3] and Google Scholar[2] are probably the most popular biomedical literature search engines. PubMed supports keyword search of over 26 million biomedical and life sciences journal articles, and Google Scholar includes some additional resources beyond journal publications [9]. There are also other search systems available.

The existing systems work well on simple queries, but there is a specific type of queries that remains challenging. Those are **multi-entity partial-match queries**, as defined below.

[1]https://www.ncbi.nlm.nih.gov/pubmed

[2]https://scholar.google.com/

Definition 1. **Typed entities.** An **entity** is a word or phrase with a specific semantic meaning. Each entity has at least one **types**, describing the grouping of the semantic meaning of the entity. For example, "New York" can be an entity of the type "city", while "Donald Trump" can be an entity of the type "person". For simplicity, we assume each entity has exactly one type in this report. A **biomedical entity** is an entity whose type is more frequent in the biomedical corpus than in the general corpus. Examples of biomedical entity types are genes, diseases, species, chemicals, etc.

Definition 2. **Entity-set queries.** An **entity-set query**, or a **multi-entity query**, is a query containing multiple entities. Those entities may or may not share the same type. In this report, we will focus on those entity-set queries covering more than one biomedical entity types. This kind of queries is common in the biomedical domain.

Definition 3. **Information need.** We assume there is some specific information need behind each query. When users submit queries, they are trying to retrieve documents that can potentially help them answer some questions in mind. The need for information helpful in answering the questions is the **information need** behind the queries.

Definition 4. **Relevance.** Given a query, documents that provide information helpful in answering the question behind the query are considered as relevant to the query. In other words, relevant documents are those that meet the information need of the user behind a query.

Definition 5. **Partial-match queries.** A **partial-match query** is an entity-set query such that 1) there are few documents in the corpus containing all entities in the query, 2) some documents containing only part of the entity set are relevant, and 3) some documents containing only part of the entity set are irrelevant.

Here is an example of a biomedical partial-match query.

Example 1. A user may want to know how genes GABP, TERT, CD11b, as well as their interplay, are related to cancer. She submits a query with four keywords "GABP, TERT, CD11b, cancer". There is no any single paper covering all four keywords, but she is still interested in papers discussing cancer and at least one of the three genes. She is not interested in the interactions between those genes in contexts not related to cancer. This query is then a partial-match query.

Traditional TF-IDF[3]-based literature retrieval algorithms, like BM25 which is used by PubMed, have the ranking

[3]TF: Term frequency. IDF: Inverse document frequency.

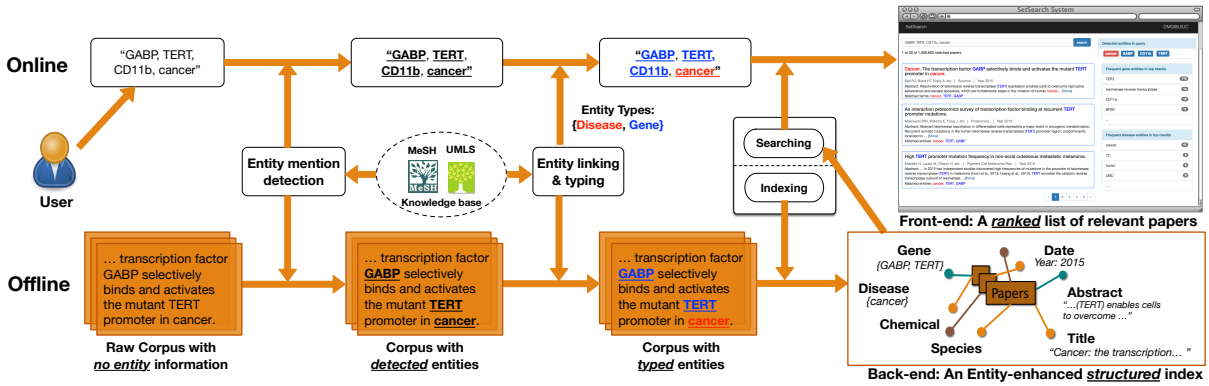

Figure 1: The architecture of the SetSearch framework.

methodology as **"the more words matched, the better"**: Documents containing more words and more occurrences of each word in the query are ranked higher. Unfortunately, applying such methodology to multi-entity partial-match queries can lead to rankings where documents irrelevant to users' information needs are ranked higher than relevant documents. Consider the following example where BM25 fails.

EXAMPLE 2. A user submits the following query to a TF-IDF-based search engine: "Compare the living expense in New York, Chicago, Seattle and San Francisco." The engine is so smart that it recognizes five phrases in the query with the highest IDF: "living expense", "New York", "Chicago", "Seattle", and "San Francisco". Suppose those five phrases have the same IDF, and the IDF values of the other tokens are so low that they do not contribute much to the ranking scores. Suppose we have these two documents in the corpus: Doc1) "Safety report of big U.S. cities: New York, Chicago, Seattle, San Francisco, etc." and Doc2) "IT elites are moving away from San Francisco due to the increasing living expense. Seattle serves as one of their destinations." Then the search engine will rank Doc1 higher than Doc2 because the TF score of Doc1 is about four thirds of the TF score of Doc2. However, the higher-ranked Doc1 is actually irrelevant to the user's information need, while the lower-ranked Doc2 is relevant.

The reason that traditional TF-IDF-based methods do not work well on partial-match queries is that their "the more words matched, the better" ranking methodology does not reflect users' information needs behind their queries. There have been efforts in developing better search engines for the biomedical domain. Unfortunately, none of the existing methods solved the partial-match challenge. An ideal search engine should be able to recognize the information need behind queries, and rank documents so that **"the more information need satisfied, the better"**.

We present SetSearch, a set-oriented and entity-aware search system, that can analyze users' information need behind their queries and rank documents based on how well they meet the information need. SetSearch is especially powerful for handling partial-match queries. Currently SetSearch is specialized for biomedical literature retrieval, but the methodology is extendable to other corpora. The major features of SetSearch system include the following:

(1) Integrate a data-driven text mining pipeline and external knowledge bases for entity mention detection and typing.

(2) Develop a new ranking function for entity set query which takes into consideration entity type information and pairwise relations among entities.

(3) Support basic entity-aware text analytics based on top returned relevant papers.

The rest of this report is structured as follows. Section 2 describes our entity-aware graph-covering ranking principle. Section 3 introduces the system implementation of SetSearch. Section 4 introduces our user interface, and how to use SetSearch for real-world biomedical queries. In Section 5 we evaluate the performance of SetSearch with home-made queries and a standard dataset (TREC Genomics Tracks 2004-05), and compare to Google Scholar, PubMed and BM25. Section 6 summarizes this report and discusses our future plan.

2 GRAPH-COVERING RANKING PRINCIPLE

This section gives mathematical description about how SetSearch decodes users' information need behind multi-entity partial-match queries and ranks documents accordingly. The key is a graph-covering ranking principle which favors documents covering more entity types.

2.1 Two Assumptions

The development of SetSearch is based on the following two assumptions about information need behind queries. Readers are encouraged to check the two examples given in the Introduction section against these two assumptions.

ASSUMPTION 1. **Entity Type Coverage Assumption.** Given a partial-match query, if the set of entities shared between the query and a document covers more unique entity types, then

that document is more likely to be relevant to the information need.

ASSUMPTION 2. **Rare Entity Type Preference Assumption.** Given a partial-match query, documents containing more entities whose types are rather rare in the query is likely to be more relevant to the information need than documents containing more popular entity types in the query.

The intuition of Assumption 1 is that different entity types in an entity set query reflects different aspects of the information need. The need may not be satisfied if some aspects are completely missing in a document. The intuition of Assumption 2 is similar to IDF: If an entity type is rare in the query, it is more likely that its existence reveals more information than the popular entity types.

The reason why previous entity-based methods have not solved the partial-match challenge is that they did not recognize those two assumptions. The limitation of current entity-based methods comes from the following two aspects.

(1) Knowledge of entity types has been ignored. For the query in Example 1, a smart searcher should identify the entity mentions in the query and link them to a knowledge base so that a document discussing "*GABP*" and "*cancer*" should be better than the one describing "*GABP*" and "*TERT*", although both of them contain two query terms. However, current systems like PubMed do not have such intelligence.
(2) Relations among the entity set have been ignored. When submitting an entity set query, biomedical researchers usually prefer papers that cover more unique entities and reveal more inter-entity relations. In reality, however, the association or co-occurrence of multiple entities has not got enough attention. Papers where a few unique entities appear multiple times are sometimes ranked higher than papers containing more unique entities.

2.2 Entity Language Model for Document Representation

Given a query q, we want to rank a collection of documents $D = \{d_1, d_2, \cdots, d_n\}$ based the *relevance score* between document and query. There are many ways to define such *relevance*. For example, traditional vector space model directly calculates a relevance score based on $tf\text{-}idf$ information. The probabilistic retrieval model, such as BM25, uses the conditional probability distribution $Pr = 1|d_i, q$ as the relevance score. The language model based information retrieval (LMIR) uses the conditional probability distribution $Pq|d_i$ as the relevance score. The learning to rank framework (L2R) intends to directly learn a model for calculating such relevance score.

In this work, we use the LMIR framework because 1) it's easier to incorporate the entity (type) information, 2) it's more intuitive than L2R and thus easier for parameter tuning (either by manual adjustment or relevance feedback), 3) the performance of a well-tuned LMIR is comparable with L2R, 4) it does not require huge amount training data in model initialization, compared with L2R.

Table 1: Parameter Explanations

Name	Description
L_x	length of text sequence x, x can be document d_i, query q or co
$n_{t,x}$	number of token t in text sequence x
$n_{\phi t,x}$	number of tokens with type ϕt in text sequence x
α_x	type distribution of text sequence x
$\theta_{\phi t\|x}$	token distribution of type ϕt in text sequence x
μ_α, μ_θ	hyper-parameters for Dirichlet smoothing

The key idea of LMIR is that we assume each document d is generated by a document model M_d, and we rank documents based on their "ability" to generate the query q, namely, $Pq|M_d$. Another way to look at such LMIR is to assume that the document d is generated by a document model M_d, the query q is generated by a query model M_q, we rank each document based on the "distance" between M_d and M_q. One popular distance is KL-divergence [1].

We assume the entity mention in each document is pre-extracted and typed. Also, we assume types have no overlaps, which is reasonable in bio-domain. We model the generative process of each document as following:

(1) For each token t in document d_i:
(2) (a) we generate the type of that token ϕt based on $P\phi t|\alpha_{d_i}$,
    (b) we generate the token t based on $Pt|\theta_{\phi t|d_i}$.

The parameters we need to infer for each document is its type distribution $P\phi t|\alpha_{d_i}$ and the token distributions for each type $Pt|\theta_{\phi t|d_i}$. Notice that the type of each token is pre-determined and thus there is no "latent variable" in this generative process.

We use Dirichlet smoothing for model parameter estimation and results are calculated as following:

$$P\phi t|\alpha_{d_i} = \frac{n_{\phi t,d} + \mu_\alpha \frac{n_{\phi t,D}}{L_D}}{L_d + \mu_\alpha}, \quad Pt|\theta_{\phi t|d_i} = \frac{n_{t,d} + \mu_\theta \frac{n_{t,D}}{n_{\phi t,D}}}{n_{\phi t,d} + \mu_\theta}, \tag{1}$$

where μ_α and μ_θ are two smoothing (hyper-)parameters and all the others are statistics of data.

2.3 Graph Model for Query Representation

We use a graph to represent a query, denoted as G_q. In this graph, each node represents a query term and each edge represents a latent relation between two query terms. To model the set property of each query, we use a complete graph to represent the query. For the other applications, there exist some other graph configurations.

Some other possible graph configurations include: 1) graph with only nodes (meaning all relations between query terms are not important, used in most traditional IR model), 2) near-complete graph with some edges missed (meaning some

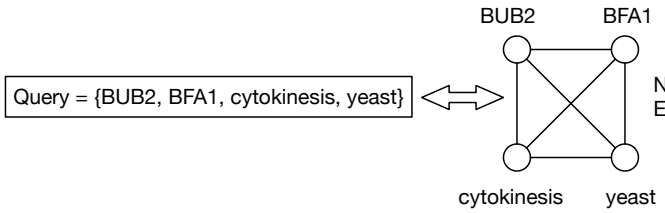

Figure 2: Graph Representation of Query

relations between query terms are not important), 3) hypergraph (each node represents a synonym set, used for query expansion).

2.4 Graph Covering Principle for Retrieving and Ranking Documents

For each term t in query, if it exists in document d_i, we say document d_i *covers* the node in G_q which corresponds to term t. For each pair of term t_a and t_b, if they both exist in document d_i, we say document d_i *covers* the edge in G_q which corresponds to latent interaction of term pair t_a and t_b. We use the notation $G_{q|d_i}$ to denote the subgraph of G_q that is covered by document d_i.

The subgraph $G_{q|d_i}$ represents the amount of latent information in query q that is explained by document d_i. We further view such subgraph as a Markov Network, and use it to define the joint probability of Pq, d as following:

$$Pq, d_i = \frac{1}{Z} \prod_{c \in G_{q|d_i}} \psi c, \tag{2}$$

where c is the clique in graph $G_{q|d_i}$ and Z is the normalization factor. Notice that given a graph $G_{q|d_i}$, the Gibbs distribution induced by it is not unique. Here, for the purpose of simplicity and modeling only second-order relation, we use a pairwise factorization schema to factorize $G_{q|d_i}$. Higher-order factorization can be later used to model more complex query term interactions.

We retrieve and rank documents based on the joint distribution of query and document:

$$\begin{aligned} PM_{d_i}|q &= \frac{PM_{d_i}, q}{Pq} \\ &\overset{\text{rank}}{=} \log PM_{d_i}, q \\ &\overset{\text{rank}}{=} \sum_{c \in G_{q|d_i}} \log \psi c \\ &= \sum_{c \in G_{q|d_i}} fc. \end{aligned} \tag{3}$$

The last equation holds as we let $\psi c = \exp fc$, a common practice for non-negative potential function. Finally, since we use the pairwise factorization, each clique c can be either a single node t, or an edge t_a, t_b.

For a single node t, we define the function $f\cdot$ as following:

$$ft = \lambda_t \cdot \log Pt|M_{d_i}, \tag{4}$$

where λ_t represents the importance of a term t, we use the type importance (depends on the query) to represent the importance of that term itself, namely $\lambda_t = \lambda_{\phi t|q}$. Furthermore, we empirically set the type importance based on the inverse type frequency in the query as following:

$$\lambda_{\phi t|q} = \log 1 + \frac{L_q}{n_{\phi t,q}}. \tag{5}$$

For an edge t_a, t_b, we define the function $f\cdot$ as following:

$$fta, t_b = \lambda_{t_a,t_b} \cdot \log Pta, t_b|M_{d_i}, \tag{6}$$

where λ_{t_a,t_b} represents the importance of this pair. We use the type interaction strength to represent its importance, namely $\lambda_{t_a,t_b} = \beta_{\phi t_a \phi t_b}$.

There are many ways to define $Pta, t_b|M_{d_i}$. One simplest method is to assume they are independent, and thus $Pta, t_b|M_{d_i} = Pta|M_{d_i} Pt_b|M_{d_i}$. We can also directly estimate this probability under some constraints such as proximity constraints (two terms must co-occur within a context window) or ordinal constraints (two terms must present given a certain order).

Finally, by replacing eqs (1)(4)(6) into (3), we have the following ranking principle:

$$\begin{aligned} PM_{d_i}|q &\overset{\text{rank}}{=} \sum_{t \in G_{q|d_i}} St, d_i + \sum_{t_a, t_b \in G_{q|d_i}} \beta_{\phi t_a \phi t_b} \left(St_a, d_i + St_b, d_i \right) \\ &= \sum_{t \in G_{q|d_i}} \left(\lambda_{\phi t} + \sum_{\substack{t' \in G_{q|d_i} \\ t' \neq t}} \beta_{\phi t \phi t'} \right) St, d_i, \end{aligned} \tag{7}$$

where St, d_i is the term-wise score calculated based on previous entity language model as following:

$$St, d_i = \log P\phi t|\alpha_{d_i} + \log Pt|\theta_{\phi t|d_i}. \tag{8}$$

This is equivalent to the OR-pAND algorithm.

2.5 Offline construction of structured index

The input of SetSearch is a set of raw documents without any explicit entity information. To support entity-aware search, we integrate a data-driven text mining into SetSearch's offline phase, including entity mention detection [2] as well as entity linking and typing [7, 8].

A demonstrative example of this pipeline is shown in the lower row of Figure 1. Entity mentions, such as "GABP", "TERT", and "cancer", are first extracted from documents with the entity mention detection module, and their corresponding types are then identified and attached with the entity linking and typing module. With such information, a structured index structure is constructed offline, which contains (i) raw text information (*e.g.*, title, abstract, and full-text), (ii) entity mentions with their type information, and (iii) document metadata (*e.g.*, author list, journal name and publication date).

2.6 Online document retrieval and ranking

To better exploit the entity information and capture relations among a set of entities, we design a new ranking algorithm called SERank based on a variant of (i) *entity language model* [6] which captures entity information, and (ii) *graph covering ranking principle* which captures the relations among multiple entities in query.

Entity Language Model. We define a two-step generative process of each document as follows:

For each token t in document d_i:
(1) generate the token type ϕt based on $P\phi t|\alpha_{d_i}$,
(2) generate the token t based on $Pt|\theta_{\phi t|d_i}$.

Notice that the type of each token is pre-determined. Therefore, for each document d_i, we only need to infer its type distribution $P\phi t|\alpha_{d_i}$ and the token distributions under each type $Pt|\theta_{\phi t|d_i}$. With Dirichlet smoothing, we can estimate those parameters based on the statistics of data [6].

Graph Covering Ranking Principle. We use a graph to represent query q, denoted as G_q, in which nodes represent query terms and edges represent latent relations between two query terms, as shown in Figure 3. Without any prior knowledge, we assume all pairwise relations between two terms are important and use a complete graph to represent this query. If query term t exists in document d_i, we say document d_i *covers* the node in G_q which corresponds to term t, and similarly, if a pair of term t_a and t_b exists in d_i, we say d_i *covers* the edge in G_q which corresponds to the latent relation of term pair (t_a, t_b). Finally, we use $G_{q|d_i}$ to denote the subgraph of G_q that is covered by document d_i.

The subgraph $G_{q|d_i}$ represents the latent information in query q that is explained by document d_i. By viewing it as a Markov network, we define the joint probability of Pq, d_i as the product of each clique's non-negative potential: ψc. For the purpose of simplicity, we model only the second-order relation between entities and thus each clique c is either a single node t or an edge t_a, t_b.

We retrieve and rank documents based on the joint distribution of query and document as follows.

$$PM_{d_i}|q \overset{\text{rank}}{=} \log PM_{d_i}, q \overset{\text{rank}}{=} \sum_{c \in G_{q|d_i}} fc. \tag{9}$$

The last equation holds as we let $\psi c = \exp fc$. For a single node t, we define $f\cdot$ to be $\lambda_t \log Pt|M_{d_i}$, where λ_t represents the importance of term t. Intuitively, we set the term importance based on the inverse type frequency (in query) of that token's type. Notice that $Pt|M_{d_i}$, the probability of token t generated by document d_i, is calculated based on previous entity language model.

For an edge t_a, t_b, we define the function $f\cdot$ as following:

$$ft_a, t_b = \lambda_{t_a,t_b} \cdot \log Pt_a, t_b|M_{d_i}, \tag{10}$$

where λ_{t_a,t_b} represents the importance of term pair. Without any prior knowledge, we assume all term pair importances are equal. As for $Pt_a, t_b|M_{d_i}$, the probability that document d_i "generates" this term pair, one simplest method is to assume they are independent and let $Pt_a, t_b|M_{d_i} = Pt_a|M_{d_i} Pt_b|M_{d_i}$.

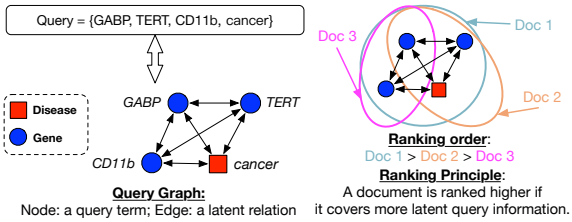

Figure 3: Graph Covering Ranking Principle

Table 2: Precision@20 in three types of queries.

	Google Scholar	PubMed	SetSearch
Case 1	0%	75%	**100%**
Case 2	5%	20%	**100%**
Case 3	5%	15%	**100%**

We can also estimate this probability under some constraints such as proximity constraints (two terms must co-occur within a context window) or ordinal constraints (two terms must present in a certain order).

3 USE CASE STUDIES

In this section, we use three multi-keyword biomedical literature search cases to demonstrate the advantages of SetSearch over PubMed and Google Scholar. All PubMed and Google Scholar searches were performed between Jan. 12 to 15, 2017, with the logic among the query terms set to all-OR[4] and other search options set to the default.

For each case study, we will first give the bio-medical question we would like to answer with the literature search. Then we will define the judging criteria: What kind of papers is considered relevant to the question? Among those relevant papers, what order do we prefer? We will compare the precision@20 achieved by SetSearch, PubMed and Google Scholar. We will also demonstrate that SetSearch results do satisfy our preference of ordering.

3.1 Case 1: Small-Size Gene-Disease Query

Question: How are the genes *GABP*, *TERT*, *CD11b* and *FOXP2*, as well as their interplay, associated with *cancer*?
Query: "*GABP, TERT, CD11b, FOXP2, cancer*".
Relevance Criterion: A retrieved paper is judged as relevant if and only if it provides information about the relation between *cancer* and at least one gene among *GABP*, *TERT*, *CD11b* and *FOXP2*.
Order Criterion: Among relevant papers, we prefer to rank higher those covering more *unique* genes.

Under the relevance criterion, all of the top 20 papers returned by SetSearch are relevant, while 15 papers among the top 20 PubMed results are relevant, as listed in Table 2. All the top 20 papers retrieved by Google scholar are irrelevant because they contain "cancer" only but none of the genes among the four.

[4]The default all-AND logic did not return any results.

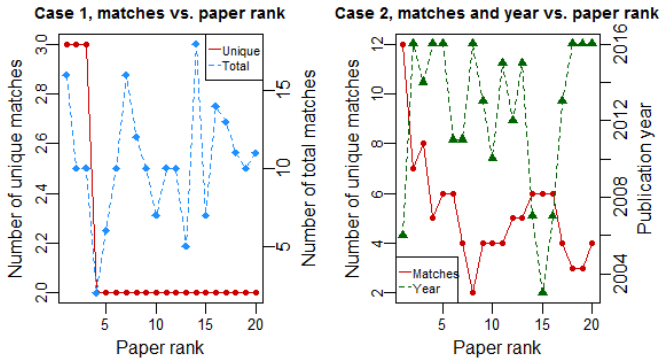

Figure 4: Order criteria check of SetSearch for Cases 1 & 2.

As shown in the left hand side of Figure 4, SetSearch tends to rank papers covering more unique entities higher, even if they contain fewer total matches. Therefore, SetSearch results satisfy the order criterion.

3.2 Case 2: Medium-Size Gene-Disease Query

Question: Given a list of 38 genes (as in the query below) that are known to be associated with prostate cancer according to Genetics Home Reference [4], what are the details about the associations?

Query: "*AR, BRCA1, BRCA2, CD82, CDH1, CHEK2, EHBP1, ELAC2, EP300, EPHB2, EZH2, FGFR2, FGFR4, GNMT, HNF1B, HOXB13, HPCX, IGF2, ITGA6, KLF6, LRP2, MAD1L1, MED12, MSMB, MSR1, MXI1, NBN, PCAP, PCNT, PLXNB1, PTEN, RNASEL, SRD5A2, STAT3, TGFBR1, WRN, WT1, ZFHX3, prostate cancer*".

Relevance Criterion: A retrieved paper is relevant if and only if it provides information about the relation between prostate cancer and at least one gene in the list of 38 genes.

Order Criterion: Among relevant papers, we prefer those covering more unique genes and published more recently[5].

The precision@20 values for this query is given in Table 2. Comparison to Case 1 shows that PubMed's precision@20 decreases as the query became longer, while SetSearch keeps its high precision. Note that we had to evaluate Google Scholar with a smaller query consisting of "prostate cancer" and the first 28 genes only, because it could not take longer queries. As shown in the right hand side of Figure 4, the papers ranked higher by SetSearch are those covering more query entities and more recent. Therefore, SetSearch satisfies the order criterion.

3.3 Case 3: Hidden Entity Discovery

Question: Given a list of 10 genes (as in the query below), what is the most relevant biomedical challenge?

Query: "*APP, APOE, PSEN1, SORL1, PSEN2, ACE, CLU, BDNF, IL1B, MAPT*".

[5]Users can choose whether to include a time factor in the ranking function to favor more recent papers.

Relevance Criterion: Those genes are actually the top genes associated with Alzheimer's disease according to DisGeNET [5]. Therefore, a retrieved paper is relevant if and only if it is discussing at least one of the query genes in the context of Alzheimer's disease.

Order Criterion: Same as Case 2.

This case is more difficult than the previous two. We are trying to discover the hidden disease entity, which is not in the query, that lies behind and connects strongly to the gene set query. As shown in Table 2, researchers can tell easily from the top 20 papers retrieved by SetSearch that those genes are highly relevant to Alzheimer's disease, while such signal is weak in the results of PubMed or Google Scholar. We also checked that SetSearch did satisfy the order criterion.

4 ABOUT THIS DEMO

The SetSearch system provides an easy-to-use web interface and easy-to-understand result visualization. We build a complete automatic workflow to search and analyze biomedical literature repositories including PubMed and PubMed Central. The current version of SetSearch system has included 26,450,721 papers in PubMed with title and abstract information, and over 2.27 millions full-text papers in PubMed Central. We implement the text mining pipeline, extract 230,231,831 in-corpus entity mentions, and type them using UMLS and MeSH as knowledge bases. We finally construct a unified index including document raw text, document metadata, and all entity information. In practice, the initial index building takes 6 hours for PubMed dataset and 4.5 hours for PubMed Central dataset on one single server. After that, we can incrementally update our structured index efficiently.

When a user inputs an entity set query, our system first types each entity mention in the query and then conducts the type-aware entity set search. Our primary interface is shown in Figure 5. At the center of the page, we will show the ranked list of papers along with their titles, authors, journal names, publication dates, abstract snippets, and matched query terms. Entity mentions in the title and abstract are highlighted and colored according to their corresponding entity types. If a user is interested in a specific paper, she can click the button "More" to expand the abstract snippet into the whole abstract, or just click the title, which directs the user to the paper page in PubMed. Beside the rank list of papers, SetSearch also presents some analysis results in the right column of result page, including detected entity mentions in the query and the most frequent entities of each type in the top returned results. In practice, it takes less than 3 seconds for returning top 10,000 relevant papers for a query containing five entities on average.

Finally, we state that SetSearch system can be extended and customized to set-oriented and entity-aware literature search in other scientific domains, since the methodology it adopts is very general. An example of the extended use of SetSearch in Physics can be querying ArXiv with "superconductivity" and a list of chemicals to retrieve literature support for candidates of superconducting materials.

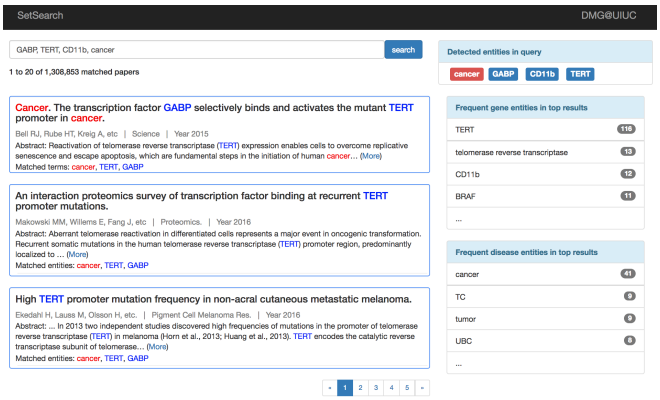

Figure 5: Snapshot of the result display for query "*GABP, TERT, CD11b, cancer*"

REFERENCES

[1] John Lafferty and Chengxiang Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 111–119.

[2] Jialu Liu, Jingbo Shang, Chi Wang, Xiang Ren, and Jiawei Han. 2015. Mining quality phrases from massive text corpora. In *SIGMOD*. ACM, 1729–1744.

[3] Zhiyong Lu. 2011. PubMed and beyond: a survey of web tools for searching biomedical literature. *Database* 2011 (2011).

[4] Joyce A Mitchell, Jane Fun, and Alexa T McCray. 2004. Design of Genetics Home Reference: a new NLM consumer health resource. *Journal of the American Medical Informatics Association* 11, 6 (2004), 439–447.

[5] Janet Piñero, Àlex Bravo, Núria Queralt-Rosinach, Alba Gutiérrez-Sacristán, Jordi Deu-Pons, Emilio Centeno, Javier García-García, Ferran Sanz, and Laura I Furlong. 2016. DisGeNET: a comprehensive platform integrating information on human disease-associated genes and variants. *Nucleic Acids Research* (2016).

[6] Hadas Raviv, Oren Kurland, and David Carmel. 2016. Document retrieval using entity-based language models. In *SIGIR*. ACM, 65–74.

[7] Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R Voss, and Jiawei Han. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *WWW*. ACM, 995–1004.

[8] Xiang Ren, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, and Jiawei Han. 2016. Label Noise Reduction in Entity Typing by Heterogeneous Partial-Label Embedding. In *KDD*. ACM.

[9] Mary Shultz. 2007. Comparing test searches in PubMed and Google Scholar. *Journal of the Medical Library Association: JMLA* 95, 4 (2007), 442.

5 PRACTICAL REQUIREMENTS

To present our demonstration at SIGIR 2017, we only need stable wireless network access, along with a table and poster mount backdrop. No additional resources or facilities are needed.