Narrative Query Graphs for Entity-Interaction-Aware Document Retrieval

Hermann Kroll[0000−0001−9887−9276], Jan Pirklbauer, Jan-Christoph Kalo, Morris Kunz, Johannes Ruthmann, and Wolf-Tilo Balke[0000−0002−5443−1215]

Institute for Information Systems, TU Braunschweig, Braunschweig, Germany
{kroll,kalo,balke}@ifis.cs.tu-bs.de and
{j.pirklbauer, morris.kunz, j.ruthmann}@tu-bs.de

Abstract. Finding relevant publications in the scientific domain can be quite tedious: Accessing large-scale document collections often means to formulate an initial keyword-based query followed by many refinements to retrieve a *sufficiently complete, yet manageable* set of documents to satisfy one's information need. Since keyword-based search limits researchers to formulating their information needs as a set of unconnected keywords, retrieval systems try to guess each user's intent. In contrast, distilling short narratives of the searchers' information needs into simple, yet precise entity-interaction graph patterns provides all information needed for a precise search. As an additional benefit, such graph patterns may also feature variable nodes to flexibly allow for different substitutions of entities taking a specified role. An evaluation over the PubMed document collection quantifies the gains in precision for our novel entity-interaction-aware search. Moreover, we perform expert interviews and a questionnaire to verify the usefulness of our system in practice.

Keywords: Narrative Queries, Graph-based Retrieval, Digital Libraries

1 Introduction

PubMed, the world's most extensive digital library for biomedical research, consists of about 32 million publications and is currently growing by more than one million publications each year. Accessing such an extensive collection by simple means such as keyword-based retrieval over publication texts is a challenge for researchers, since they simply cannot read through hundreds of possibly relevant documents, yet cannot afford to miss relevant information in retrieval tasks. Indeed, there is a dire need for retrieval tools tailored to specific information needs in order to solve the above conflict. For such tools, deeper knowledge about the particular task at hand and the specific semantics involved is essential. Taking a closer look at the nature of scientific information search, interactions between entities can be seen to represent a short narrative [8], a short story of interest: how or why entities interact, in what sequence or roles they occur, and what the result or purpose of their interaction is [3, 8].

Indeed, an extensive query log analysis on PubMed in [4] clearly shows that researchers in the biomedical domain are often interested in interactions between entities such as drugs, genes, and diseases. Among other results, the authors report that a) on average significantly more keywords are used in PubMed queries than in typical Web searches, b) result set sizes reach an average of (rather unmanageable) 14,050 documents, and c) keyword queries are on average 4.3 times refined and often include more specific information about the keywords' intended semantic relationships, e.g., *myocardial infarction AND aspirin* may be refined to *myocardial infarction prevention AND aspirin*. Given all these observations, native support for entity-interaction-aware retrieval tasks can be expected to be extremely useful for PubMed information searches and is quite promising to generalize to other kinds of scientific domains, too. However, searching scientific document collections curated by digital libraries for such narratives is tedious when being restricted to keyword-based search, since the same narrative can be paraphrased in countless ways [1, 4].

Therefore, we introduce the novel concept of *narrative query graphs for scientific document retrieval* enabling users to formulate their information need as entity-interaction queries explicitly. Complex interactions between entities can be precisely specified: Simple interactions between two entities are expressed by a basic query graph consisting of two nodes and a labeled edge between them. Of course, by adding more edges and entity nodes, these basic graph patterns can be combined to form arbitrarily complex graph patterns to address highly specialized information needs. Moreover, narrative query graphs support *variable nodes* supporting a far broader expressiveness than keyword-based queries. As an example, a researcher might search for treatments of *some disease* using *simvastatin*. While keyword-based searches would broaden the scope of the query far in excess of the user intent by just omitting any specific disease's name, narrative query graphs can focus the search by using a variable node to find documents that describe treatments of *simvastatin* facilitated by an entity of the type *disease*. The obtained result lists can then be clustered by possible node substitutions to get an entity-centric literature overview. Besides, we provide provenance information to explain why a document matches the query.

In summary, our contributions are:

1. We propose narrative query graphs for scientific document retrieval enabling fine-grained modeling of users' information needs. Moreover, we boost query expressiveness by introducing variable nodes for document retrieval.
2. We developed a prototype that processes arbitrary narrative query graphs over large document collections. As a showcase, the prototype performs searches on six million PubMed titles and abstracts in real-time.
3. We evaluated our system in two ways: On the one hand, we demonstrated our retrieval system's usefulness and superiority over keyword-based search on the PubMed digital library in a manual evaluation including practitioners from the pharmaceutical domain. On the other hand, we performed interviews and a questionnaire with eight biomedical experts who face the search for literature on a daily basis.

2 Related Work

Narrative query graphs are designed to offer complex querying capabilities over scientific document collections aiming at high precision results. Focusing on retrieving entity interactions, they are a subset of our conceptual overlay model for representing narrative information [8]. We discussed the first ideas to bind narratives against document collections in [9]. This paper describes the complete retrieval method and evaluation of narrative query graphs for document retrieval. In the last decade three major research areas were proposed to improve text-based information retrieval.

Machine Learning for Information Retrieval. Modern personalized systems try to guess each user's intent and automatically provide more relevant results by query expansion, see [1] for a good overview. Mohan et al. focus on information retrieval of biomedical texts in PubMed [13]. The authors derive a training and test set by analyzing PubMed query logs and train a deep neural network to improve literature search. Entity-based language models are used to distinguish between a term-based and entity-based search to increase the retrieval quality [16]. Yet, while a variety of approaches to improve result rankings by learning how a query is related to some document [13, 19, 20], have been proposed, gathering enough training data to effectively train a system for all different kinds of scientific domains seems impossible. Specialized information needs, which are not searched often, are hardly covered in such models.

Graph-based Information Retrieval. Using graph-based methods for textual information retrieval gained in popularity recently [3, 17, 18, 20]. For instance, Dietz et al. discuss the opportunities of entity linking and relation extraction to enhance query processing for keyword-based systems [3] and Zhao et al. demonstrate the usefulness of graph-based document representations for precise biomedical literature retrieval [20]. Kadry et al. also include entity and relationship information from the text as a learning-to-rank task to improve support passage retrieval [5]. Besides, Spitz et al. build a graph representation for Wikipedia to answer queries about events and entities more precisely [17]. But in contrast to our work, the above approaches focus on unlabeled graphs or include relationships only partially.

Knowledge Bases for Literature Search. GrapAl, for example, a graph database of academic literature, is designed to assist academic literature search by supporting a structured querying language, namely Cypher [2]. GrapAl mainly consists of traditional metadata like authors, citations, and publication information but also includes entities and relationship mentions. However, complex entity interactions are not supported, as only a few basic relationships per paper are annotated. As a more practical system that extracts facts from text to support question answering, QKBfly has been presented [14] . It constructs a knowledge base for ad-hoc question answering during query time that provides journalists with the latest information about emergent topics. However, they focus on retrieving relevant facts concerning a single entity. In contrast, our focus is on document retrieval for complex entity interactions.

3 Narrative Query Graphs

Entities represent things of interest in a specific domain: Drugs and diseases are prime examples in the biomedical domain. An entity $e = (id, type)$, where id is a unique identifier and $type$ the entity type. To give an example, we may represent the drug *simvastatin* by its identifier and entity type as follows: $e_{simvastatin} = (D019821, Drug)$. Typically, entities are defined by predefined ontologies, taxonomies, or controlled vocabularies, such as NLM's MeSH or EMBL's ChEBI. We denote the set of known entities as $\mathcal{E}$. Entities might also be classes as well, e.g., the entity *diabetes mellitus* (Disease) refers to a class of specialized diabetes diseases such as DM type 1 and DM type 2. Thus, these classes can be arranged in subclass relations, i.e., DM type 1 is the subclass of general diabetes mellitus. Since we aim to find entity interactions in texts, we need to know where entities are mentioned. In typical natural language processing, each sentence is represented as a sequence of tokens, i.e., single words. Therefore, an **entity alignment** maps a token or a sequence of tokens to an entity from $\mathcal{E}$ if the tokens refer to it.

We call an interaction between two entities a **statement** following the idea of knowledge representation in the Resource Description Framework (RDF) [12]. Hence, a **statement** is a triple (s, p, o) where $s, o \in \mathcal{E}$ and $p \in \Sigma$. Σ represents the set of all interactions we are interested in. We focus only on interactions between entities, unlike RDF, where objects might be literals too. For example, a *treatment* interaction between *simvastatin* and *hypercholesterolemia* is encoded as ($e_{simvastatin}$, *treats*, $e_{hypercholesterolemia}$). We call a set of extractions from a single document a so-called **document graph**.

Document graphs support narrative querying, i.e., the query is answered by matching the query against the document's graph. Suppose a user formulates a query like ($e_{simvastatin}$, *treats*, $e_{hypercholesterolemia}$). In that case, our system retrieves a set of documents containing the searched statement. Narrative query graphs may include typed variable nodes as well. A user might query ($e_{simvastatin}$, *treats*, *?X(Disease)*), asking for documents containing *some* disease treatment with *simvastatin*. Hence, all documents that include *simvastatin* treatments for diseases are proper matches. Formally, we denote the set of all variable nodes as $\mathcal{V}$. Variable nodes consist of a name and an entity type to support querying for entity types. We also support the entity type *All* to query for arbitrary entities. We write variable nodes by a leading question mark. Hence, a narrative query graph might include entities stemming from $\mathcal{E}$ and variable nodes from $\mathcal{V}$. Formally, a **fact pattern** is a triple $fp = (s, p, o)$ where $s, o \in (\mathcal{E} \cup \mathcal{V})$ and $p \in \Sigma$. A **narrative query graph** q is a set of fact patterns similar to SPARQL's basic graph patterns [15]. When executed, the query produces one or more matches μ by binding the variable symbols to actual entities, i.e., $\mu : \mathcal{V} \to \mathcal{E}$ is a partial function. If several fact patterns are queried, all patterns must be contained within a document forming a proper query answer. If queries include entities that are classes and have subclasses, then the query will be expanded to also query for these subclasses, i.e., direct and transitive subclasses.

4 Narrative Document Retrieval

In the following section we describe our system for narrative query graph processing. First, we perform a pre-processing that involves entity linking, information extraction, cleaning, and loading. It extracts document graphs from text and stores them in a structured repository. Then, a query processing that matches a user's query against the document graphs takes place. In this way, we can return a structured visualization of matching documents. An overview of the whole system is depicted in Figure 1.

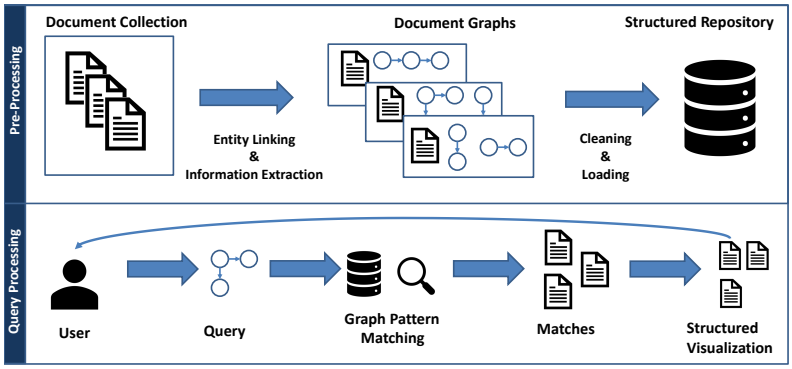

Fig. 1. System Overview: Document graphs are extracted from texts, cleaned, indexed, and loaded into a structured repository. Then, narrative query graphs can be matched against the repository to retrieve the respective documents.

4.1 Document Graph Extraction

The pre-processing step, including entity linking and information extraction, utilizes our toolbox for the nearly-unsupervised construction of knowledge graphs [10]. The toolbox requires the design of two different vocabularies: 1. An entity vocabulary that contains all entities of interest. An entry consists of a unique entity id, an entity name, and a list of synonyms. 2. A relation vocabulary that contains all relations of interest. An entry consists of a relation and a set of synonyms.

For this paper, we built an entity vocabulary that comprises *drugs, diseases, dosage forms, excipients, genes, plant families, and species*. Next, we wanted to extract interactions between these entities from texts since interactions between entities are essential to support retrieval with narrative query graphs. Although the quality of existing open information extraction like OpenIE 6 sounded promising [6], we found that open information extraction methods highly lack recall when processing biomedical texts, see the evaluation in [10]. That is why we developed a recall-oriented extraction technique **PathIE** in [10] that flexibly extracts interactions between entities via a path-based method. This method was evaluated and shared in our toolbox as well.

PathIE yields many synonymous predicates (treats, aids, prevents, etc.) that represent the relation *treats*. The relation vocabulary must have clear semantics and was built with the help of two domain experts. We designed a relation vocabulary comprising 60 entries (10 relations plus 50 synonyms) for the cleaning step. This vocabulary enables the user to formulate her query based on a well-curated vocabulary of entity interactions in the domain of interest. We applied our semi-supervised predicate unification algorithm to clean the extractions. To increase the quality of extractions, we introduced type constraints by providing fixed domain and range types for each interaction. Extracted interactions that did not meet the interaction's type constraints were removed. For example, the interaction *treats* is typed, i.e., the subject must be a drug, and the object must be a disease or species. Some interactions in our vocabulary like *induces* or *associated* are more general and thus were not annotated with type constraints.

4.2 Document Retrieval

Finally, the extracted document graphs had to be stored in a structured repository for querying purposes. For this paper, we built upon a relational database, namely PostgresV10. Relational databases support efficient querying and allowed us to provide additional provenance information and metadata for our purposes. For example, our prototype returned document titles, sentences, entity annotations, and extraction information to explain matches to the user. Due to our focus on pharmaceutical and medical users, we selected a PubMed subset that includes drug and excipient annotations. Therefore, we annotated the whole PubMed collection with our entity linking component, yielding 302 million annotations. Around six million documents included a drug or excipient annotation. Performing extraction and cleaning on around six million documents yielded nearly 270 million different extractions. Hence, the current prototype's version comprises about six million documents. We incrementally have increased the available data, but we entirely covered the relevant pharmaceutical part (drug and excipient).

As a reminder, a narrative query graph consists of fact patterns following simple RDF-style basic graph patterns. Our system automatically translates these narrative query graphs into a structured query language: They are translated into SQL statements for querying the underlying relational database. A single fact pattern requires a selection of the extraction table with suitable conditions to check the entities and the interaction. Multiple fact patterns require self-joining of the extraction table, and adding document conditions in the where clause, i.e., the facts matched against the query must be extracted from the same document. We developed an in-memory and hash-based matching algorithm that quickly combines the results. Another point to think about were ontological subclass relations between entities. For example, querying for treatments of *Diabetes Mellitus* would require to also search for the subclasses *Diabetes Mellitus Type 1* and *Diabetes Mellitus Type 2*. Query rewriting is necessary to compute complete results for queries that involve entities with subclasses [11]. We rewrite queries that include entities with subclasses to also query for these subclasses. Due to

the long-standing development of databases, such a query processing can be performed very quickly when using suitable indexes. We computed an inverted index, i.e., each extraction triple was mapped to a set of document ids. Besides, we implemented some optimization strategies to accelerate the query processing, e.g., match fact patterns with concrete entities first and fact patterns with variable nodes afterward. We remark on our system's query performance in our evaluation.

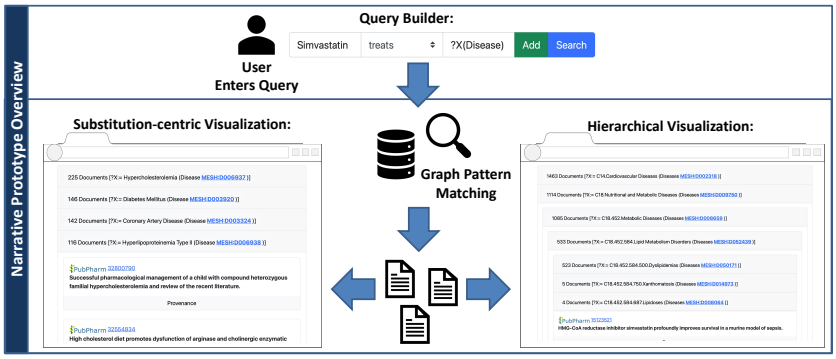

Fig. 2. A schematic overview of our prototype implementation. A query builder helps the users to formulate their information need. If the narrative query involves variable nodes, the results can be visualized in a substitution-centric visualization (left side) or in a hierarchical visualization (right side).

4.3 Prototype Design

We present a prototype resulting from joint efforts by the university library, the institute for information systems, and two pharmaceutical domain experts who gave us helpful feedback and recommendations. The prototype[1] offers precise biomedical document retrieval with narrative query graphs. A general overview of our prototype is shown in Fig. 2. We implemented a REST service handling queries and performing the query processing on the backend side. Furthermore, we supported the user with a query builder and suitable result visualization on the frontend side. In an early prototype phase, we tested different user interfaces to formulate narrative query graphs, namely, 1. a simple text field, 2. a structured query builder, and 3. a graph designer tool. We found that our users preferred the structured query builder which allows them to formulate a query by building a list of fact patterns. For each fact pattern, the users must enter the query's subject and object. Then, they can select an interaction between both in a predefined selection. The prototype assists the user by suggesting around three million terms (entity names plus synonyms). Variable nodes can be formulated, e.g., by writing *?X(Drug)* or just entering the entity type like *Drug* in

[1] http://www.pubpharm.de/services/prototypes/narratives/

the subject or object field. When users start their search, the prototype sends the query to the backend and visualizes the returned results. The returned results are sorted by their corresponding publication date in descending order. The prototype represents documents by a document id (PubMedID), a title, a link to the digital library entry (PubMed), and provenance information. Provenance includes the sentence in which the matching fact was extracted. We highlight the linked entities (subject and object) and their interaction (text term plus mapping to the interaction vocabulary). Provenance may be helpful for users to understand why a document is a match. If a query contains multiple fact patterns, we attach a list of matched sentences in the visualization. Visualizing document lists is comparable to traditional search engines, but handling queries with variable nodes requires novel interfaces. We will discuss such visualizations for queries, including variable nodes, subsequently.

4.4 Retrieval with Variable Nodes

Variable nodes in a narrative query graph may be restricted to specific entity types like *Disease*. We also allow a general type *All* to support querying for arbitrary entities. For example, a user might formulate the query (*Simvastatin*, *treats*, *?X(Disease)*). Several document graphs might match the query with different variable substitutions for $?X$. A document d_1 with the substitution $\mu_1(?X) =$ *hypercholestorelemia* as well as a document d_2 with $\mu_2(?X) =$ *hyperlipidemia* might be proper matches to the query. How should we handle and present these substitutions to the users? Discussions with domain experts led to the conclusion that aggregating documents by their substitution seems most promising. Further, we present two strategies to visualize these document result groups in an user interface: *substitution-centric* and *hierarchical visualization*.

Substitution-centric Visualization. Given a query with a variable node, the first strategy is to aggregate by similar variable substitutions. We retrieve a list of documents with corresponding variable substitutions from the respective document graph. Different substitutions represent different groups of documents, e.g., one group of documents might talk about the treatment of *hypercholestorelemia* while the other group might talk about *hypertriglyceridemia*. These groups are sorted in descending order by the number of documents in each group. Hence, variable substitutions shared by many documents appear at the top of the list. Our query prototype visualizes a document group as a collapsible list item. A user's click can uncollapse the list item to show all contained documents. Provenance information is used to explain why a document matches her query, i.e., the prototype displays the sentences in which a query's pattern was matched. Provenance may be especially helpful when working with variable nodes.

Hierarchical Visualization. Entities are arranged in taxonomies in many domains. Here, diseases are linked to MeSH (Medical Subject Heading) descriptors arranged in the MeSH taxonomy. The hierarchical visualization aims at showing document results in a hierarchical structure. For example, *hypercholestorelemia* and *hypertriglyceridemia* share the same superclass in MeSH, namely *hyperlipidemias*. All documents describing a treatment of *hypercholestorelemia* as well as

hypertriglyceridemia are also matches to *hyperlipidemias*. Our prototype visualizes this hierarchical structure by several nested collapsible lists, e.g., *hyperlipidemias* forms a collapsible list. If a user's click uncollapses this list, then the subclasses of *hyperlipidemias* are shown as collapsible lists as well. We remove all nodes that do not have any documents attached in their node or all successor nodes to bypass the need to show the whole MeSH taxonomy.

5 System Evaluation & User Study

Subsequently, we analyze our retrieval prototype concerning two research questions: *Do narrative query graphs offer a precise search for literature? And, do variable nodes provide useful entity-centric overviews of literature?* We performed three evaluations to answer the previous questions:

1. Two pharmaceutical experts created test sets to quantify the retrieval quality (100 abstracts and 50 full-text papers). Both experts are highly experienced in pharmaceutical literature search.
2. We performed interviews with eight pharmaceutical experts who search for literature in their daily research. Each expert was interviewed twice: Before testing our prototype to understand their information need and introducing our prototype. After testing our prototype, to collect feedback on a qualitative level, i.e., how they estimate our prototype's usefulness.
3. Finally, all eight experts were asked to fill out a questionnaire. The central findings are reported in this paper.

5.1 Retrieval Evaluation

After having consulted the pharmaceutical experts, we decided to focus on the following typical information needs in the biomedical domain: I1: Drug-Disease treatments (*treats*) play a central role in the mediation of diseases. I2: Drugs might decrease the effect of other drugs and diseases (*decrease*). I3: Drug treatments might increase the expression of some substance or disease (*induces*). I4: Drug-Gene inhibitions (*inhibits*), i.e., drugs disturb the proper enzyme production of a gene. I5: Gene-Drug metabolisms (*metabolizes*), i.e., gene-produced enzymes metabolize the drug's level by decreasing the drug's concentration in an organism. Narrative query graphs specify the exact interactions a user is looking for. For each information need (I1-5), we built narrative query graphs with well-known entities from the pharmaceutical domain: Q1: *Metformin treats Diabetes Mellitus (I1)*, Q2: *Simvastatin decreases Cholesterol (I2)*, Q3: *Simvastatin induces Rhabdomyolysis (I3)*, Q4: *Metformin inhibits mtor (I4)*, Q5: *CYP3A4 metabolizes Simvastatin AND Erythromycin inhibits CYP3A4 (I4/5)*, and Q6: *CYP3A4 metabolizes Simvastatin AND Amiodarone inhibits CYP3A4 (I4/5)*.

Further, we used the entities for each query to search for document candidates on PubMed, e.g., for Q1 we used *metformin diabetes mellitus* as the PubMed

Table 1. Expert evaluation of retrieval quality for narrative query graphs in comparison to PubMed and a MeSH-based search on PubMed. Two experts have annotated PubMed samples to estimate whether the information need was answered. Then, precision, recall and F1-measure are computed for all systems.

Query	**#Hits**	**#Sample**	**#TP**	**PubMed**	**MeSH Search**			**Narrative QG**		
				Prec.	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
Q1	12.7K	25	19	0.76	0.82	0.47	0.60	**1.00**	0.42	0.59
Q2	5K	25	16	0.64	**0.73**	0.50	0.59	0.66	0.25	0.36
Q3	427	25	17	0.68	0.77	0.59	0.67	**1.00**	0.35	0.52
Q4	726	25	16	0.64	**0.78**	0.44	0.56	0.71	0.31	0.43
Q5	397	25	6	0.24	-	-	-	**1.0**	0.17	0.25
Q6	372	25	5	0.20	-	-	-	**1.0**	0.20	0.33

query. We kept only documents that were processed in our pipeline. Then, we took a random sample of 25 documents for each query. The experts manually read and annotated these sample documents' abstracts concerning their information need (true hits / false hits). Besides, we retrieved 50 full texts documents of PubMed Central (PMC) for a combined and very specialized information need (Q5 and Q6). The experts made their decision for PubMed documents by considering titles and abstracts, and for PMC documents, the full texts. Subsequently, we considered these documents as ground truth to estimate the retrieval quality. We compared our retrieval to two baselines, 1) queries on PubMed and 2) queries on PubMed with suitable MeSH headings and subheadings.

PubMed MeSH Baseline. PubMed provides so-called MeSH terms for documents to assists users in their search process. MeSH (Medical Subject Headings) is an expert-designed vocabulary comprising various biomedical concepts (around 26K different headings). These MeSH terms are assigned to PubMed documents by human annotators who carefully read a document and select suitable headings. Prime examples for these headings are annotated entities such as drugs, diseases, etc., and concepts such as study types, therapy types, and many more. In addition to headings, MeSH supports about 76 subheadings to precisely annotate how a MeSH descriptor is used within the document's context. An example document might contain the subheading *drug therapy* attached to *simvastatin*. Hence, a human annotator decided that *simvastatin* is used in drug therapy within the document's context. The National Library of Medicine (NLM) recommends subheadings for entity interactions such as treatments and adverse effects. In cooperation with our experts who read the NLM recommendations, we selected suitable headings and subheadings to precisely query PubMed concerning the respective entity interaction for our queries.

Results. The corresponding interaction and the retrieval quality (precision, recall, and F1-score) for each query are depicted in Table 1. The sample size and the number of positive hits in the sample (TP) are reported for each query. The PubMed search contains only the entities as a simple baseline, and hence, achieved a recall of 1.0 in all cases. PubMed search yielded a precision of around

0.64 up to 0.76 for abstracts and 0.2 up to 0.24 for full texts. The PubMed MeSH search achieved a moderate precision of about 0.73 to 0.82 and recall of about 0.5 for PubMed titles and abstracts (Q1-Q4). Unfortunately, the important MeSH annotations were missing for all true positive hits for Q5 and Q6 in PMC full texts. Hence, the PubMed MeSH search did not find any hits in PMC for Q5 and Q6. Narrative query graphs (Narrative QG) answered the information need with good precision: Q1 (*treats*) and Q3 (*induces*) were answered with a precision of 1.0 and a corresponding recall of 0.42 (Q1) and 0.47 (Q3). The minimum achieved precision was 0.66, and the recall differed between 0.17 and 0.42. Our prototype could answer Q5 and Q6 on PMC full texts: One correct match was returned for Q5 as well as for Q6, leading to a precision of 1.0.

5.2 User Interviews

The previous evaluation demonstrated that our system could achieve good precision when searching for specialized information needs. However, the next questions are: How does our prototype work for daily use cases? And, what are the prototype's benefits and limitations in practice? Therefore, we performed two interviews with each of the eight pharmaceutical experts who search for literature in their daily work. All experts had a research background and worked either at a university or university hospital.

First Interview. In the first interview, we asked the participants to describe their literature search. They shared two different scientific workflows that we have analyzed further: 1. searching for literature in a familiar research area, and 2. searching for a new hypothesis which they might have heard in a talk or read in some paper. We performed think-aload experiments to understand both scenarios. They shared their screen, showed us at least two different literature searches, and how they found relevant documents answering their information need. For scenario 1), most of them knew suitable keywords, works or journals already. Hence, they quickly found relevant hits using precise keywords and sorting the results by their publication date. They already had a good overview of the literature and could hence answer their information need quickly. For scenario 2), they guessed keywords for the given hypothesis. They had to refine their search several times by varying keywords, adding more, or removing keywords. Then, they scanned titles and abstracts of documents looking for the given hypothesis. We believe that scenario 1) was recall-oriented: They did not want to miss important works. Scenario 2) seemed to be precision-oriented, i.e., they quickly wanted to check whether the hypothesis may be supported by literature. Subsequently, we gave them a short introduction to our prototype. We highlighted two features: The precision-oriented search and the usage of variable nodes to get entity-centric literature overviews. We closed the first interview and gave them three weeks to use the prototype for their literature searches.

Second Interview. We asked them to share their thoughts about the prototype: What works well? What does not work well? What could be improved? First, they considered querying with narrative query graphs, especially with variable nodes, different and more complicated than keyword-based searches.

Querying with variable nodes by writing *?X(Drug)* as a subject or an object was deemed too cryptic. They suggested that using *Drug*, *Disease*, etc. would be easier. Another point was that they were restricted to a fixed set of subjects and objects (all known entities in our prototype). For example, querying with pharmaceutical methods like *photomicrography* was not supported. Next, the interaction vocabulary was not intuitive for them. Sometimes they did not know which interaction would answer their information need. One expert suggested to introduce a hierarchical structure for the interactions, i.e., some general interactions like *interacts* that can be specified into *metabolizes* and *inhibits* if required. On the other side, they appreciated the prototype's precise search capability. They all agreed that they could find precise results more quickly using our prototype than other search engines. Besides, they appreciated the provenance information to estimate if a document match answers their information need. They agreed that variable nodes in narrative query graphs offered completely new search capabilities, e.g., *In which dosage forms was Metformin used when treating diabetes?* Such a query could be translated into two fact patterns: (*Metformin*, *administered*, *?X(DosageForm)* and (*Metformin*, *treats*, *Diabetes Mellitus*). The most common administrations are done *orally* or via an *injection*. They agreed that such information might not be available in a specialized database like DrugBank. DrugBank covers different dosage forms for Metformin but not in combination with diabetes treatments. As queries get more complicated and detailed, such information can hardly be gathered in a single database. They argued that the *substitution-centric visualization* helps them to estimate which substitutions are relevant based on the number of supporting documents. Besides, they found the *hierarchical visualisation* helpful when querying for diseases, e.g., searching for (*Metformin*, *treats*, *?X(Disease)*). Here, substitutions are shown in an hierarchical representation, e.g., *Metabolism Disorders*, *Glucose Disorders*, *Diabetes Mellitus*, *Diabetes Mellitus Type 1*, etc. They liked this visualization to get a drug's overview of treated disease classes. All of them agreed that searches with variable nodes were helpful to get an entity-structured overview of the literature. Four experts stated that such an overview could help new researchers get better literature overviews in their fields.

5.3 Questionnaire

We asked each domain expert to answer a questionnaire after completing the second interview. The essential findings and results are reported subsequently. First, we asked to choose between precision and recall when searching for literature. Q1: *To which statement would you rather agree when you search for related work?* The answer options were (rephrased): A1a: *I would rather prefer a complete result list (recall). I do not want to miss anything.* A2a: *I would rather prefer precise results (precision) and accept missing documents.* Six of eight experts preferred recall, and the remaining two preferred precision. We asked a similar question for the second scenario (hypothesis). Again, we had let them select between precision and recall (A1a and A1b). Seven of eight preferred precision, and one preferred recall when searching for a hypothesis. Then,

Table 2. Questionnaire Results: Eight participants were asked to rate the following statements about our prototype on a Likert scale ranging from 1 (disagreement) to 5 (agreement). The mean ratings are reported.

Statement about the Prototype	**Mean**
The prototype allows me to formulate precise questions by specifically expressing the interactions between search terms.	4.0
The formulation of questions in the prototype is understandable for me.	4.0
The displayed text passage from the document (Provenance) is helpful for me to understand why a document matches my search query.	5.0
The prototype provides precise results for my questions (I quickly find a relevant match).	3.5
Basically, grouping results is helpful for me when searching for variable nodes.	4.5
When searching for related work, I would prefer the prototype to a search using classic search tools (cf. PubPharm, PubMed, etc.).	2.8
When searching for or verifying a hypothesis, I would prefer the prototype to a search using classic search tools (cf. PubPharm, PubMed, etc.).	3.4
I could imagine using the prototype in my literature research.	3.9

we asked Q3: *To which statement would you rather agree for the vast majority of your searches?* Again, seven of eight domain experts preferred precise hits over complete result lists. The remaining one preferred recall. The next block of questions was about individual searching experiences with our prototype: different statements were rated on a Likert scale ranging from 1 (disagreement) to 5 (full agreement). The results are reported in Table 2. They agreed that the prototype allows to formulate precise questions (4.0 mean rating), and the formulation of questions was understandable (4.0). Besides, provenance information was beneficial for our users (5.0). They could well imagine using our prototype in their literature research (3.9) and searching for a hypothesis (3.4). Still, users were reluctant to actually switch to our prototype for related work searches (2.8). Finally, the result visualization of narrative query graphs with variables was considered helpful (4.5).

5.4 Performance Analysis

The query system and the database ran on a server, having two Intel Xeon E5-2687W (3,1GHz, eight cores, 16 threads), 377GB of DDR3 main memory, and SDDs as primary storage. The preprocessing took around one week for our six million documents (titles and abstracts). We randomly generated 10k queries asking for one, two, and three interactions. We measured the time of query execution on a single thread. Queries that are not expanded via an ontology took in average 21.9ms (1-fact) / 52ms (2-facts) / 51.7ms (3-facts). Queries that are expanded via an ontology took in average 54.9ms (1-fact) / 158.9ms (2-facts) / 158.2ms (3-facts). However, the query time heavily depends on the interaction (selectivity) and how many subclasses are involved. In sum, our system can retrieve documents with a quick response time for the vast majority of searches.

6 Discussion and Conclusion

In close cooperation with domain experts using the PubMed corpus, our evaluation shows that overall document retrieval can indeed decisively profit from graph-based querying. The expert evaluation demonstrates that our system achieves a moderate up to good precision for highly specialized information needs in the pharmaceutical domain. Although the precision is high, our system has only a moderate recall. Moreover, we compared our system to manually curated annotations (MeSH and MeSH subheadings), which are a unique feature of PubMed. Most digital libraries may support keywords and tags for documents but rarely support how these keywords, and primarily, how entities are used within the document's context. Therefore, we developed a document retrieval system with a precision comparable to manual metadata curation but without the need for manual curation of documents.

The user study and questionnaire reveal a strong agreement for our prototype's usefulness in practice. In summary, the user interface must be intuitive to support querying with narrative query graphs. Further enhancements are necessary to explain the interaction vocabulary to the user. We appreciate the idea of hierarchical interactions, i.e., showing a few basic interactions that can be specified for more specialized needs. Especially the search with variable nodes in detailed narrative query graphs offers a new access path to the literature. The questionnaire reveals that seven of eight experts agreed that the vast majority of their searches are precision-oriented. Next, they agreed that they prefer our prototype over established search engines for precision-oriented searches. The verification of hypotheses seems to be a possible application because precise hits are preferred here. We believe that our prototype should not replace classical search engines because there are many recall-oriented tasks like related work searches. The recall will always be a problem by design when building upon error-prone natural language processing techniques and restricting extractions to sentence levels. Although the results seem promising, there are still problems to be solved in the future, e.g., improve the extraction and the user interface.

Conclusion. Entity-based information access catering even for complex information needs is a central necessity in today's scientific knowledge discovery. But while structured information sources such as knowledge graphs offer *high query expressiveness* by graph-based query languages, scientific document retrieval is severely lagging behind. The reason is that graph-based query languages allow to describe the desired characteristics of and interactions between entities in sufficient detail. In contrast, document retrieval is usually limited to simple keyword queries. Yet unlike knowledge graphs, scientific document collections offer *contextualized knowledge*, where entities, their specific characteristics, and their interactions are connected as part of a coherent argumentation and thus offer a clear advantage [7,8]. The research in this paper offers a novel workflow to bridge the worlds of structured and unstructured scientific information by performing graph-based querying against scientific document collections. But as our current workflow is clearly precision-oriented, we plan to improve the recall without having to broaden the scope of queries in future work.

References

1. Azad, H.K., Deepak, A.: Query expansion techniques for information retrieval: A survey. Information Processing & Management **56**(5), 1698–1735 (2019)
2. Betts, C., Power, J., Ammar, W.: GrapAL: Connecting the dots in scientific literature. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations. pp. 147–152. Association for Computational Linguistics, Florence, Italy (Jul 2019). https://doi.org/10.18653/v1/P19-3025
3. Dietz, L., Kotov, A., Meij, E.: Utilizing knowledge graphs for text-centric information retrieval. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. p. 1387–1390. SIGIR '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3209978.3210187
4. Herskovic, J.R., Tanaka, L.Y., Hersh, W., Bernstam, E.V.: A Day in the Life of PubMed: Analysis of a Typical Day's Query Log. Journal of the American Medical Informatics Association **14**(2), 212–220 (03 2007)
5. Kadry, A., Dietz, L.: Open relation extraction for support passage retrieval: Merit and open issues. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 1149–1152. SIGIR '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3077136.3080744
6. Kolluru, K., Adlakha, V., Aggarwal, S., Mausam, Chakrabarti, S.: OpenIE6: Iterative Grid Labeling and Coordination Analysis for Open Information Extraction. In: Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP). pp. 3748–3761. ACL (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.306
7. Kroll, H., Kalo, J.C., Nagel, D., Mennicke, S., Balke, W.T.: Context-compatible information fusion for scientific knowledge graphs. In: Digital Libraries for Open Knowledge. pp. 33–47. Springer (2020). https://doi.org/10.1007/978-3-030-54956-5_3
8. Kroll, H., Nagel, D., Balke, W.T.: Modeling narrative structures in logical overlays on top of knowledge repositories. In: Conceptual Modeling. pp. 250–260. Springer (2020). https://doi.org/10.1007/978-3-030-62522-1_18
9. Kroll, H., Nagel, D., Kunz, M., Balke, W.T.: Demonstrating narrative bindings: Linking discourses to knowledge repositories. In: Fourth Workshop on Narrative Extraction From Texts, Text2Story@ECIR2021. CEUR Workshop Proceedings, vol. 2860, pp. 57–63. CEUR-WS.org (2021), http://ceur-ws.org/Vol-2860/paper7.pdf
10. Kroll, H., Pirklbauer, J., Balke, W.T.: A toolbox for the nearly-unsupervised construction of digital library knowledge graphs. In: Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2021. JCDL '21, Association for Computing Machinery, New York, NY, USA (2021)
11. Krötzsch, M., Rudolph, S.: Is your database system a semantic web reasoner? KI-Künstliche Intelligenz **30**(2), 169–176 (2016). https://doi.org/10.1007/s13218-015-0412-x
12. Manola, F., Miller, E., McBride, B., et al.: RDF primer. W3C recommendation **10**(1-107), 6 (2004)
13. Mohan, S., Fiorini, N., Kim, S., Lu, Z.: A fast deep learning model for textual relevance in biomedical information retrieval. In: Proceedings of the 2018

World Wide Web Conference. p. 77–86. WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2018). https://doi.org/10.1145/3178876.3186049
14. Nguyen, D.B., Abujabal, A., Tran, N.K., Theobald, M., Weikum, G.: Query-driven on-the-fly knowledge base construction. Proc. VLDB Endow. **11**(1), 66–79 (Sep 2017). https://doi.org/10.14778/3151113.3151119
15. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of sparql. ACM Transactions on Database Systems **34**(3) (Sep 2009). https://doi.org/10.1145/1567274.1567278
16. Raviv, H., Kurland, O., Carmel, D.: Document retrieval using entity-based language models. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 65–74. SIGIR '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2911451.2911508
17. Spitz, A., Gertz, M.: Terms over load: Leveraging named entities for cross-document extraction and summarization of events. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 503–512. SIGIR '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2911451.2911529
18. Vazirgiannis, M., Malliaros, F.D., Nikolentzos, G.: Graphrep: Boosting text mining, nlp and information retrieval with graphs. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. p. 2295–2296. CIKM '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3269206.3274273
19. Xiong, C., Power, R., Callan, J.: Explicit semantic ranking for academic search via knowledge graph embedding. In: Proceedings of the 26th International Conference on World Wide Web. p. 1271–1279. WWW '17, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2017). https://doi.org/10.1145/3038912.3052558
20. Zhao, S., Su, C., Sboner, A., Wang, F.: Graphene: A precise biomedical literature retrieval engine with graph augmented deep learning and external knowledge empowerment. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. p. 149–158. CIKM '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3357384.3358038