Expanding Query Answers on Medical Knowledge Bases

Chuan Lei
IBM Research - Almaden
chuan.lei@ibm.com

Vasilis Efthymiou
IBM Research - Almaden
vasilis.efthymiou@ibm.com

Rebecca Geis*
IBM Germany
rebecca.geis@de.ibm.com

Fatma Özcan
IBM Research - Almaden
fozcan@us.ibm.com

ABSTRACT

Medical knowledge bases (KBs) are known to be vital for tasks like clinical decision support and medical question answering, since they provide well-structured relational information between entities. One of the main challenges for querying a medical KB is the mismatch between the terms in the KB and the colloquial and imprecise terminology used in user queries. To address this challenge, we propose a domain-specific query relaxation approach that leverages rich medical domain vocabularies and their semantic relationships from external knowledge sources, such as taxonomies, ontologies, and semantic networks, to expand the vocabulary of KBs. Our main goal is to expand both the set of queries that we can answer, as well as the set of answers to the queries, over the medical KB. We introduce a lightweight adaptation method to customize and incorporate external knowledge sources to work with the existing KB, and propose a novel similarity metric to leverage the information content in the KB, the structural information in the external knowledge source, and the contextual information from user queries. We implement our proposed techniques for a medical KB, and use SNOMED CT as the external knowledge source. We experimentally demonstrate the effectiveness of our proposed method and the improved quality of query results in terms of both precision and recall, compared to state-of-the-art approaches. Finally, we conduct user studies to evaluate how much a conversational interface can benefit from our proposed method in terms of its query capability on the medical KB.

1 INTRODUCTION

Medical knowledge bases (KBs) provide structured information about medical entities (such as drugs and diseases) and their relationships, which are invaluable in medical applications. Such KBs are often created from medical information sources, including medical literature, patient data, claims data, etc, and offer deep domain specialization with rich and detailed information, which is known to be vital for domain-specific tasks like clinical decision support and medical question answering. The medical KBs are different from cross-domain large-scale KBs such as DBpedia [5] and Freebase [9] which provide well-structured, encyclopedic knowledge but with less detail and precision.

When querying medical KBs, the users do not always formulate their queries precisely to match the terms in the KB, especially when they use natural language. For example, users are likely to use informal words, phrases, or abbreviations of certain terms in their natural language queries, which makes matching the mentioned entities to the medical KB a non-trivial task. Query relaxation [18] is one of the most prominent techniques used for query answering, allowing more domain-specific terms in user queries. Instead of returning no or incomplete answers, query relaxation can transform the query in a way that the user's intent is better represented, greatly improving the flexibility and usability of a medical KB.

The problem of query relaxation has been extensively studied in information retrieval and database systems with the goal of returning information beyond what is specified by a standard query [17, 26]. However, the techniques, traditionally designed for formally defined query languages such as SQL, cannot handle the complexity from natural language queries that involve complex semantic constraints and logic [37, 43]. Hence, they often fail to ensure query answering with high precision and recall. Recent work [3, 8, 14] demonstrated that deep learning models built at word or sentence level can be used for semantic similarity estimation. However, these methods demand high-quality training data, which is critical and expensive in reality.

In this paper, we focus on a medical KB (*MED*) which contains medication, disease and toxicology information to support informed diagnosis and treatment decisions for evidence-based clinical decisions and patient education. We observe that this and similar medical KBs can be further enriched by external knowledge sources, such as medical ontologies, taxonomies, and semantic networks (e.g., Unified Medical Language System [40], SNOMED Clinical Terms [38], and Gene Ontology [12]). These knowledge sources can be exploited by query relaxation to expand query answers.

We introduce a novel query relaxation method that leverages rich domain vocabularies and their semantic relationships from external medical knowledge sources, which largely consist of subsumption relationships (e.g., $A \sqsubseteq B$, where A and B are concepts in the external knowledge source). We first find the concept corresponding to a given query term in the external knowledge source, and then relax the term by exploring the concept's neighborhood to identify semantically related concepts.

The rich domain vocabulary and structural information of external knowledge sources empower query relaxation to generalize or specialize query terms beyond syntactic matching. However, external knowledge sources such as SNOMED CT are often not customized to the application's requirements. Using external knowledge sources without proper adaptation may introduce semantically unrelated information into the results, leading to low precision and recall. To provide high-quality results, a query relaxation method has to address the following challenges.

External knowledge source ingestion. External knowledge sources are often comprehensive, consisting of an excessive amount of information describing a domain. The given KB is often substantially smaller than the external knowledge sources.

*Rebecca Geis is currently a graduate student at Technische Universität Darmstadt.

Series ISSN: 2367-2005

10.5441/002/edbt.2020.67

This makes it challenging to identify semantically related results from the external knowledge source. For example, given a query *"what drugs treat pertussis"*, there might be no drug directly associated with *"pertussis"* in the given KB. Instead, a generalized clinical finding, *"bronchitis"*, in the KB has corresponding drug information. However, the distance (i.e., the number of hops) between *"pertussis"* and *"bronchitis"* in SNOMED CT is large, making it difficult for query relaxation to identify the semantic similarity between the two terms. Worse yet, many findings closer in distance but not semantically related might be returned as well, causing low-precision results.

Exploiting the query context. Contextual information has significant impact on the semantic correctness of the relaxed results. For example, a user may ask *"what drugs treat psychogenic fever"*, in which case the context is *"treatment"* and *"psychogenic fever"* is a query term in the given medical KB. This term appears in SNOMED CT as the name of a clinical finding, with both *"hyperpyrexia"* and *"hypothermia"* being similar findings. However, in the context of *"treatment"*, drugs for *"hypothermia"* should not be returned, as *"hypothermia"* is the opposite of *"hyperpyrexia"* and *"psychogenic fever"*.

To address these challenges, we propose a novel two-phase query relaxation method, consisting of external knowledge source ingestion phase and the online query relaxation phase. We implemented our techniques for the medical KB (*MED*), and used SNOMED CT as the external knowledge source. The main contributions of this paper are as follows.

• We present a lightweight, yet effective offline ingestion process that customizes the external knowledge source to the given KB.

• We propose a novel similarity metric to identify semantically related concepts, leveraging (*i*) the information content in the KB, (*ii*) the structural information in the external knowledge source, and (*iii*) the contextual information from the user query.

• We introduce a programmatic way to incorporate our method into two state-of-the-art systems, a conversational system [21] and a natural language query (NLQ) system [23, 35] for the medical KB, using SNOMED CT as the external knowledge source.

• Our experiments show that our query relaxation method for the medical KB outperforms state-of-the-art methods, including deep learning-based ones, in precision and recall. We also conduct a user study demonstrating how our query relaxation method improves the response quality of a conversational system.

Outline. The rest of the paper is organized as follows. In Section 2, we introduce the basic concepts used in this paper, and in Section 3, we provide an overview of our query relaxation approach. Section 4 describes context generation, extraction, and management. Section 5 introduces our query relaxation method in detail. We explain how to integrate our query relaxation technique into two natural language interface systems in Section 6, and provide experimental results in Section 7. We review related work in Section 8, and conclude in Section 9.

2 BACKGROUND

2.1 Knowledge Base

Following the standard notation of description logic [6], we assume that a KB is given in the form of TBox and ABox. In this paper, TBox is referred to as *domain ontology* and ABox is referred to as *instances* or data.

The domain ontology describes the concepts relevant to the domain, and the relationships (roles) among different concepts. The concepts associated with a relationship are provided by the domain (i.e., source) and range (i.e., destination) constraints of this relationship. The *context* of a query term used in a query can be represented by a relationship and its associated concepts from the domain ontology.

Figure 1 shows a fragment of a sample medical domain ontology. The concept *"Finding"* connects to both concepts *"Indication"* and *"Risk"* through the relationship *"hasFinding"*. This shows that *"Finding"* can be potentially used in two different contexts (i.e., *Risk-hasFinding-Finding* and *Indication-hasFinding-Finding*). Two example queries could be *"which drugs have the risk of* ***causing*** *diabetes"* and *"which drugs* ***treat*** *diabetes"*, where the query term is *"diabetes"*.

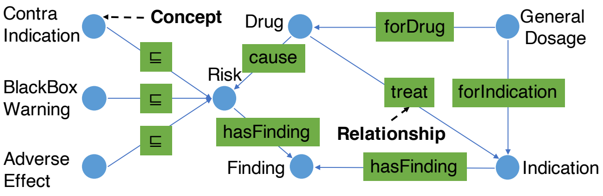


Figure 1: Snippet of a medical ontology.

The instances (data) of the given KB are stored separately for query answering as shown in Figure 3. For example, *"fever"* and *"renal impairment"* are two instances of *"Finding"*. We assume that our input is in the form of a query term and its associated context. Following the previous example, the input to our query relaxation method would be [diabetes, *Risk-hasFinding-Finding*] or [diabetes, *Indication-hasFinding-Finding*]. Recent technologies [1, 21, 28] have been designed to extract the contextual information from natural language questions. In this paper, our method is integrated with Watson Assistant [21] to receive the contextual information, and finds semantically related instances for a given query term with high precision and recall. We provide more details on how we bootstrap the conversation space in Section 4.

2.2 External Knowledge Source

In this work, we utilize the rich medical domain vocabularies and their semantic relationships from external knowledge sources such as ontologies, semantic networks, and knowledge graphs. In particular, we are interested in the subsumption relationships in the form of $A \sqsubseteq B$, where A and B are concepts in the external knowledge source. In this case, we say that A *specializes* B, and that B *generalizes* A. We refer to the direct and implied (by transitivity) specializations of a concept A, excluding A, as the *descendants* of A. We assume that the external knowledge source is a directed acyclic graph (DAG), in which a top concept (owl:Thing in OWL) is the root and every concept is a descendant of the root. To avoid confusion with the concepts of the domain ontology, we refer to the concepts in the external knowledge source as *external concepts*.

2.3 Semantic Similarity Measures

Semantic similarity measures estimate the similarity between concepts, and are commonly used in various processing tasks (e.g., entity resolution [10, 15], link prediction [27], change detection [22]). The knowledge-based approach to semantic similarity exploits taxonomies like WordNet. Typically, path finding measures and information content (IC) measures are two common categories in knowledge-based approaches [19]. In addition to a knowledge source, the IC approach can leverage a frequency value $freq(A)$, accounting for the number of times a concept A is mentioned in a document corpus, to compute the similarity between concepts. Specifically, the IC of a concept A is defined as the inverse of the log of the concept's frequency [25, 34]:

$$IC(A) = -log(freq(A)), \tag{1}$$

where $freq(A)$ is recursively defined as:

$$freq(A) = |A| + \sum_{A_i \sqsubseteq A} freq(A_i), \tag{2}$$

with $|A|$ being the number of times concept A is directly mentioned in the document corpus, and A_i being the direct descendants of A in the taxonomy. The intuition is that the more general a concept is, the more likely it is that the concept or its descendants appear in the corpus. We describe how we compute this equation in the next section.

The IC-based similarity measure compares the IC of a pair of concepts to the IC of their Least Common Subsumer (LCS)[1]. The greater the IC of the LCS (i.e., the more specific the LCS), the more similar is the pair of concepts:

$$sim_{IC}(A, B) = \frac{2 \times IC(lcs(A, B))}{IC(A) + IC(B)}. \tag{3}$$

In general, the IC similarity measure is shown to outperform other approaches on various semantic similarity benchmarks [2, 19, 29]. Hence we adopt the above IC similarity measure and further integrate it with the structural information in the external knowledge source, as well as the contextual information from the natural language query.

3 APPROACH OVERVIEW

In this section, we provide an overview of our query relaxation method, as shown in Figure 2. We propose a two-phase approach: an offline phase, in which we construct context specifications, as well as incorporate an external knowledge source into the given KB, and an online phase, in which we take a query term associated with a context, and return the semantically related results as answers for the query.

Offline phase. In the offline phase, also called external knowledge source ingestion, we perform the following tasks: (i) we initialize a set of possible contexts based on the domain ontology, and optionally generate training examples for context classification (if needed by natural language interface (NLI) system), (ii) we compute the frequency of each external concept in the external knowledge source with respect to the associated contexts, and (iii) we generate mappings between instance data in the knowledge base and external concepts in the external knowledge source.

To initialize the set of possible contexts, we traverse the domain ontology and return all the relationships, along with their source and destination concepts. Those relationships constitute the set of possible contexts, which we provide to the NLI system. We can also provide labeled data for training a context classifier in the NLI system if required (described in Section 4).

To compute the frequency of each external concept, we leverage the document corpus from which the given knowledge base is curated. Additionally, we differentiate the frequency of the external concepts with respect to different contexts, as described in Section 5.1.

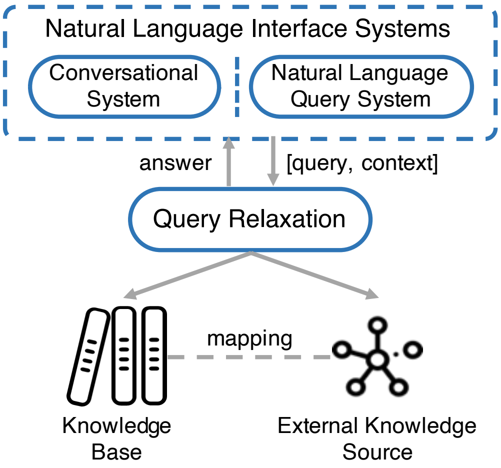

Natural Language Interface Systems
Conversational System
Natural Language Query System
answer
[query, context]
Query Relaxation
mapping
Knowledge Base
External Knowledge Source

Figure 2: Approach overview.

To map the instance data from the given KB to the external concepts in the external knowledge source, we provide three alternative methods, depending on the accuracy requirement from the application. Specifically, these methods include matching the instance data and external concepts with exactly the same names (exact match), very similar names in terms of edit distance, or similar names in terms of word embeddings, as described in Section 7.2.

Online phase. In the online phase, also called online query relaxation, we receives as input a [query term, context] pair and perform the following tasks: (i) we search for an external concept Q corresponding to the query term in the external knowledge source, and (ii) we retrieve the top-k similar external concepts having corresponding matching instances in the KB to Q as the answers.

To identify an external concept Q that corresponds to the input query term, we follow a similar process used in the offline external knowledge source ingestion: we identify Q as the external concept whose name either matches with the exact query term, or is very similar in terms of either edit distance or word embeddings. Additionally, several knowledge sources (e.g., SNOMED CT[2], DrugBank[3], DBpedia[4]) may offer a more sophisticated lookup service, which we can also utilize to find such mappings.

Finally, having identified the external concept Q that corresponds to the input query term, we retrieve its top-k similar external concepts that have corresponding instances in the given KB. For the similarity computation, we leverage the information content from the KB, the structural information in the external knowledge source, as well as the contextual information from the query, as described in Section 5.2.

[1] A LCS of two concepts always exists in the external knowledge source. When multiple LCSs exist, we choose the one with the shortest path to the pair of concepts. If multiple LCSs have equal distance to the pair of concepts, we use the average IC of these LCSs for the similarity measure.

[2] https://browser.ihtsdotools.org/

[3] https://www.drugbank.ca

[4] https://github.com/dbpedia/lookup

4 CONTEXT SPECIFICATION

As explained in Section 2.1, the context can be represented by a relationship and its associated concepts from the domain ontology. In this section, we provide a brief overview of: (*i*) how our method provides the necessary information that an NLI system requires for context recognition, and (*ii*) how the conversational context differs from simple question answering. We note that the process of context recognition is orthogonal to our method, and we refer the reader to [33] for more details.

Context generation and extraction. Context reflects the intent or goal expressed in the user query/input[5]. NLI systems typically use a learning-based model to identify the intent for a given user query within the current conversation. As a consequence, most of these systems require as input the specification of all possible contexts expected in a given workload with labeled query examples for training the intent classifier. These contexts are usually based on (*i*) the purpose of the application and the scope of questions that it intends to handle, (*ii*) the anticipated set of questions that the users might ask within the scope of the application.

To feed such NLI system with training data, we need to follow a two-step process. The first step is to generate all possible contexts, based on the domain ontology. For this step, we traverse the domain ontology and extract all relationships, along with their associated concepts, i.e., their source and destination concepts. Since a context can be represented by a relationship, we define the set of possible contexts (i.e., possible labels for training data) as the set of relationships.

The second step is to associate a query workload to the generated contexts. There are different options for this step, which go beyond the scope of this work. One simple approach is to retrieve an existing query workload, and ask domain experts to label each query in the workload with the most relevant context. Once we have such an annotated query workload, we can either stop the process here, or we can further enrich the query workload. For enriching the query workload, we can replace identified instances with other instances of the same concept. For example, we can generate more queries in our workload from a given query *"what drugs treat fever"*, labeled with the context *Indication-hasFinding-Finding*), by replacing *"fever"* with other instances of *"Finding"*, such as *"headache"*, *"sore throat"*, and *"pain in throat"*.

The result of this two-step process is a set of queries, each labeled with a context, which we can provide to a NLI system as training data for context recognition.

Context management. In Figure 2, we show two alternative NLI systems that can benefit from our query relaxation method. The main difference between a conversational system and a natural language query system is that the latter can be stateless. Namely, a conversational system needs to keep track of the conversational flow, i.e., the state of the dialogue and its history. This way, the current context can be inferred from the previous state, even if not explicitly mentioned in the current query. For example, if the current query is *"what about fever?"*, there is no clear context, if this query is processed individually. However, if it is processed as part of a conversation, in which the previous query was *"which drugs treat diabetes"*, then a conversational system can infer that the previous context *Indication-hasFinding-Finding*) remains unchanged. More details on that subject can be found in [23]. On the other hand, a natural language query system typically handles contexts in one-shot queries without considering previously asked queries.

5 QUERY RELAXATION METHOD

Our query relaxation method has two phases, the offline external knowledge source ingestion and the online query relaxation. In this section, we first focus on describing how to customize and incorporate an external knowledge source into the given KB, and then we show how to use the adapted knowledge sources in online query relaxation.

5.1 External Knowledge Source Ingestion

The external knowledge source ingestion addresses the following two issues. First, we need to count the frequency $freq(A)$ of an external concept A (Equation 2) based on the corpus from where the given medical KB is curated. Its frequency should reflect the context in which the concept is used. Second, the external knowledge source typically contains an excessive amount of information in terms of domain vocabulary and relationships. It is imperative to customize and incorporate the external knowledge source in accordance with the given KB for query relaxation.

Concept frequency. First, we need to map the instances from the given KB to their corresponding external concepts in the external knowledge source, as illustrated in Figure 3. A variety of techniques can be leveraged to produce such mappings, ranging from exact string matching, approximate string matching using edit distance, to word embeddings. In this paper, we use these techniques in a pluggable fashion, and we compare the effectiveness of these algorithms in the experimental evaluation. If an instance is mapped to an external concept, then this concept is marked with a flag (the concepts in yellow in Figure 3). The online query relaxation only returns flagged concepts as semantically related results to a given query term, since the given KB only contains information about those concepts.

Next, as mentioned earlier, a concept could be used in different contexts depending on the natural language query, and the semantic meaning can be completely different in different contexts. For example, a condition treated by a drug is different from an adverse effect (i.e., condition) caused by the same drug. In this case, having a single frequency associated with a concept (Equation 2) would not be sufficient to capture the semantic differences over all possible contexts.

To resolve this issue, we identify all the contexts where an external concept A can be used. Specifically, we use the relationships associated to a concept in the domain ontology as the contexts of A, if an instance of this concept is mapped to A. Then, we compute the concept frequency with respect to each context. The online query relaxation phase chooses the appropriate concept frequency according to the query context as described later in this section.

To compute the frequency of external concepts, we assume that the KB is curated based on a document corpus, and we count the number of times that each external concept name is mentioned within this corpus. To account for the sparsity of certain concept names in the corpus, the concept frequency is further adjusted based on the number of documents in which the concept name appears. For example, *"asthma"* is mentioned in 54 drug descriptions in DrugBank [16], whereas *"lung cancer"* has only a handful of associated specialty drugs. Hence, we utilize the commonly used tf-idf weighting to alleviate this bias.

[5] We use the terms context and intent interchangeably in this section.

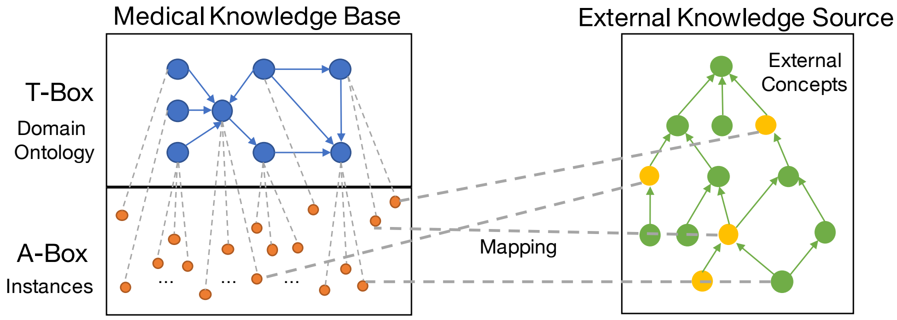


Figure 3: External knowledge source ingestion.

Example 1. In the medical ontology depicted in Figure 1, the concept *"Finding"* is connected to both *"Indication"* and *"Risk"*. In this case, the external concepts associated with instances of *"Finding"*, have two concept frequencies corresponding to the contexts *"Indication-hasFinding-Finding"* and *"Risk-hasFinding-Finding"*. Depending on the query context, one of the concept frequencies is used in query relaxation. In Figure 4, we show a snippet of SNOMED CT with the external concept frequency populated. The external concepts in brackets are all mapped from different instances of *"Finding"* in the domain KB, so they can be used in two different contexts (*"Indication-hasFinding-Finding"* and *"Risk-hasFinding-Finding"*). Hence, they are associated with two concept frequencies. For example, *"headache"* is the only direct descendant of *"craniofacial pain"*, and the frequency of *"craniofacial pain"* is the frequency of itself, together with the one of *"headache"*. Accordingly, the frequency of *"pain of head and neck region"* is a summation of the frequencies of *"craniofacial pain"*, *"pain in throat"*, and itself, which is 19164 (i.e., 18878 + 283 + 3) in the context of *"Indication-hasFinding-Finding"* and 1656 in the context of *"Risk-hasFinding-Finding"*.

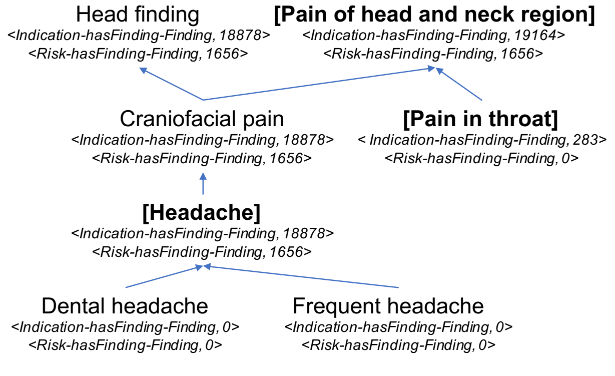

Head finding
<Indication-hasFinding-Finding, 18878>
<Risk-hasFinding-Finding, 1656>
[Pain of head and neck region]
<Indication-hasFinding-Finding, 19164>
<Risk-hasFinding-Finding, 1656>
Craniofacial pain
<Indication-hasFinding-Finding, 18878>
<Risk-hasFinding-Finding, 1656>
[Pain in throat]
< Indication-hasFinding-Finding, 283>
<Risk-hasFinding-Finding, 0>
[Headache]
<Indication-hasFinding-Finding, 18878>
<Risk-hasFinding-Finding, 1656>
Dental headache
<Indication-hasFinding-Finding, 0>
<Risk-hasFinding-Finding, 0>
Frequent headache
<Indication-hasFinding-Finding, 0>
<Risk-hasFinding-Finding, 0>

Figure 4: Snippet of SNOMED CT with frequencies.

Finally, all of these frequencies are normalized between [0, 1], which corresponds to the probability of a concept appearing in the corpus. The root concept has the highest normalized frequency of 1, because all concepts in the external knowledge source are its descendants.

Sparsity of external knowledge source. The commonly used external knowledge sources, such as SNOMED CT, are comprehensive, consisting of an excessive amount of information describing the domain. They often consist of rich domain vocabularies associated with deep hierarchies. On the contrary, the given KB is often relatively smaller compared to the external knowledge source. Therefore, only a small subset of the external concepts may have corresponding instances in the KB. Any pair of concepts could be connected through multiple intermediate ones, which makes finding semantically related concepts prohibitively time-consuming for an online system. One straightforward approach would be computing the pairwise similarity between all concepts offline. However, this leads to unnecessary computations and space consumption, since most of these precomputed similarities may not even be used during the online query relaxation.

In the offline ingestion phase, we alleviate the above issue by introducing additional application-specific edges to the external knowledge source. Specifically, an additional directed edge is introduced from an external concept A to an external concept B, if all the following conditions are satisfied: (1) A and B are not directly connected (i.e., one-hop neighbors), (2) A is a descendant of B, and (3) at least one of the two concepts has a corresponding instance in the given KB. Consequently, they become one-hop neighbors with respect to the application. The distance between two external concepts is attached to the new edge so that the original semantic information between two concepts is preserved. This way, external concepts come closer, avoiding unnecessary delays in the online query relaxation phase.

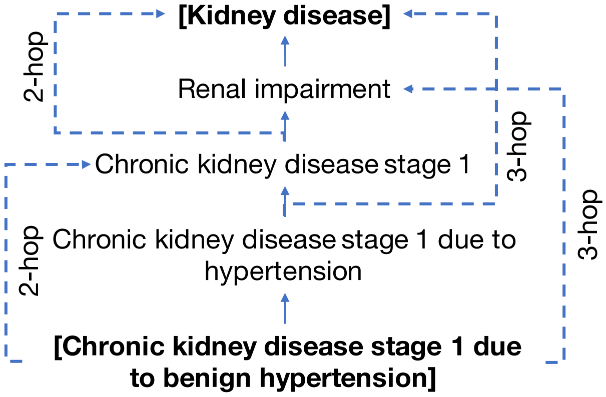

[Kidney disease]
Renal impairment
Chronic kidney disease stage 1
Chronic kidney disease stage 1 due to hypertension
[Chronic kidney disease stage 1 due to benign hypertension]
2-hop
3-hop
2-hop
3-hop

Figure 5: External knowledge source customization.

Example 2. In Figure 5, *"chronic kidney disease stage 1 due to hypertension"* is 3 hops away from *"kidney disease"*, which has a corresponding instance in the KB. By introducing an additional edge (the dashed line) between these two external concepts, they are only 1 hop away. Therefore, more semantically related concepts are within a close distance, and the semantic similarity between two external concepts remains unchanged since the original path information (3-hop) between them is attached to the new edge.

Overall, the offline external knowledge source ingestion process is summarized in Algorithm 1. The algorithm receives as input the medical KB (as domain ontology O and instances I) and the external knowledge source EKS, and it returns the set of possible contexts C, the frequencies F of the external concepts for each context, the mappings M from instances to external concepts, and the set of external concepts that are marked with a flag FEC. The algorithm consists of 3 almost independent procedures: context generation, mappings, and concept frequency. Additionally, for efficiency, sparsity of external knowledge source is also handled in the same loop as concept frequency, even though one does not depend on the other.

In Lines 1-4, we create the set of possible contexts, based on the domain ontology's relationship, along with their domains (source concepts) and ranges (destination concepts). In Lines 5-11, we find an external concept A as a mapping for every instance $i \in I$, if such exists, based on a chosen mapping function, and return the set of mappings M. At the end of this loop, the set FEC contains all the external concepts that have been marked with a flag, i.e., all the external concepts that have a corresponding instance in I. In Line 12, we sort the external concepts in topological order, such that the descendants precede their ancestors (note that EKS is a DAG). This way, we can easily compute the frequency of each external concept for a given context (Lines 14-18), using the recursive function of Equation 2. Since the external concepts are already topologically sorted, we add application-specific edges to alleviate the sparsity of EKS in the same loop (Lines 19-23). Specifically, in Line 21, we add an edge from external concept A to external concept B, when all three conditions (previously described) are satisfied, while attaching their original distance (as $|shortestPath(A, B)|$) to the new edge.

Time complexity analysis. The time complexity of external knowledge source ingestion can be broken down into the following parts. First, creating all possible contexts requires iterating through all relationships ($|R|$) in the domain ontology. Hence the time complexity is $\Theta(|R|)$. Second, the time complexity of finding an external concept A for every instance depends on the chosen method. For example, using word embeddings to find mappings requires $\Theta(|I| \cdot Cost(lookup))$, where $|I|$ denotes the total number of instances and $Cost(lookup)$ denotes the constant cost of embedding lookup for each instance. If an approximate string matching algorithm is used, then the time complexity becomes $O(|I| \cdot mn)$, where $O(mn)$ denotes the time complexity of typical approximate string matching algorithm (m and n are the lengths of two strings). Third, topological sorting of all external concepts requires $O(|V| + |E|)$ time complexity, assuming that V is the set of concepts and E is the set of relationships in the external knowledge source. Fourth, the time complexity of computing frequency of each external concept for all possible context is $O(|V| \cdot AVG(contexts))$, where $AVG(contexts)$ denotes the average number of contexts per concept in the domain ontology. Regarding adding application-specific edges to EKS, we ignore its time complexity since the number of concepts satisfying is much less than $|V|$. In summary, the total time complexity is $\Theta(|R|)$ + $\Theta(|I| \cdot Cost(lookup))$ + $O(|V| + |E|)$ + $O(|V| \cdot AVG(contexts))$. Note that the external knowledge source ingestion is an offline process that is executed only once.

Algorithm 1 Knowledge Source Ingestion Algorithm.

```
Input: Domain ontology $O$, Instances $I$, External Knowledge
    Source $EKS$
Output: (Set of contexts $C$), External concept frequencies $F$,
    Mappings $M$, Flagged external concepts $FEC$
    ▷ Context generation
 1: $C \leftarrow \emptyset$
 2: for each $r \in Relationships(O)$ do
 3:     $C \leftarrow C \cup \{(domain(r), r, range(r))\}$
 4: end for
    ▷ Mappings
 5: $M \leftarrow \emptyset$
 6: $FEC \leftarrow \emptyset$                         // Flagged external concepts
 7: for each $i \in I$ do
 8:     $A \leftarrow mapping(i, EKS)$  // map $i$ to an external concept $A$
 9:     $M \leftarrow M \cup \{(i, A)\}$
10:     $FEC \leftarrow FEC \cup \{A\}$
11: end for
    ▷ Concept frequency
12: $Q \leftarrow topol.Sort(Concepts(EKS))$  // children before parents
13: $F \leftarrow \emptyset$                            // Frequencies wrt. context
14: while $Q$ is not empty do
15:     $A \leftarrow Q.next()$
16:     for each $c \in C$ do
17:         $F \leftarrow F \cup \{(A, c, freq(A))\}$            // see Equation 2
18:     end for
        // External knowledge source customization
19:     for each $B \in ancestors(A, EKS) \setminus parents(A, EKS)$ do
20:         if $A \in FEC$ or $B \in FEC$ then
21:             $EKS.addEdge(A, |shortestPath(A, B)|, B)$
22:         end if
23:     end for
24:     $Q.remove(A)$
25: end while
26: return $C, F, M, FEC$
```

5.2 Online Query Relaxation

Given a query term, the goal of online query relaxation is to identify the semantically related instances contained in the given KB by leveraging the external knowledge source. In this subsection, we present a novel similarity measure that leverages the information content from the medical KB, the structural information in the external knowledge source, as well as the contextual information from the query.

Contextual information. As described earlier, the possible contexts of a query term mapped to a concept in the domain ontology, are the relationships of the concept to its adjacent concepts. With the contextual information, the online query relaxation phase can choose the appropriate concept frequency to use in Equation 2.

Example 3. For the query *"what are the risks of using aspirin"*, the context is *"Drug-cause-Risk"*. As shown in Figure 1, *"Risk"* has three descendants *"Black Box Warning"*, *"Adverse Effect"*, and *"Contra Indication"*. Assuming that the query term *"aspirin"* cannot be found in the given KB, then the online query relaxation would consider related conditions in the context of *"Drug-cause-Risk"*. Hence, the concept frequency used for the similarity measure should be the total frequency of all three descendants of *"Risk"*.

In case the contextual information is not available for online query relaxation, our method can aggregate the frequencies (i.e., all possible contexts) associated with an external concept. As verified in the experimental evaluation section, the contextual information greatly improves the quality of the results.

Structural (Path) information. As described above, external knowledge sources contain generalization and specialization relationships between concepts. Generalizing the query term in a user query may cause information loss [24]. For example, as shown in Figure 4, *"headache"* can be generalized to *"craniofacial pain"*, which can in turn be generalized as *"pain of head and neck region"* including *"pain in throat"*. Apparently, *"pain in throat"* no longer describes pain in head, even if it is as close to *"craniofacial pain"* as *"frequent headache"*.

In this case, solely relying on the IC similarity measure (Equation 3) with contextual information can be insufficient as it cannot differentiate the semantic difference between specialization (i.e., the opposite direction of subsumption edges in the external knowledge source) and generalization (i.e., following the direction of subsumption edges). We tackle this challenge by assigning different weights to generalization and specialization in the external knowledge source. The weight of a path connecting two external concepts A and B is thus computed as:

$$p_{A,B} = \prod_{i}^{|D|} w_i^{D-i}, \tag{4}$$

where the distance between external concepts A and B is $|D|$, and w_i indicates the weight of the i-th edge from A to B. The intuition is that we penalize a generalization and the penalty is more if it appears early on in a path from A to B. In fact, such distinction helps us to better capture the semantic similarity between a pair of concepts based on their relative locations in the external knowledge source.

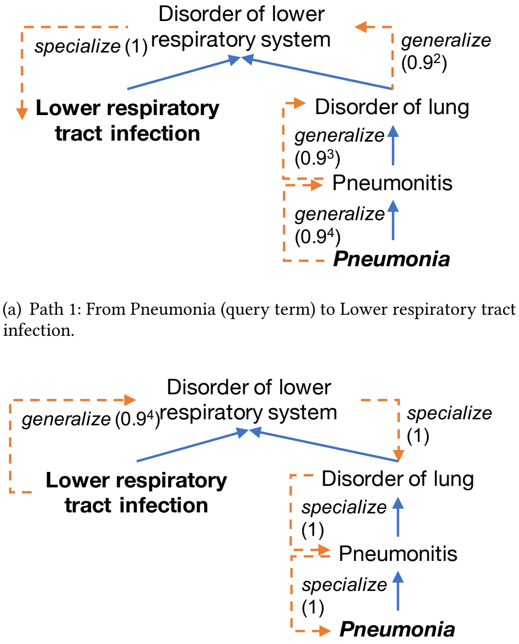
To learn the weights of both generalization and specialization, simple statistical regression analysis [7] such as logistic regression can be used. In our empirical study, the weights of generalization and specialization are set to 0.9 and 1, respectively.

Example 4. In Figure 6, the penalty associated with the path (dashed orange lines) connecting two external concepts can be different, depending on which concept corresponds to the query term. In this example, there are 4 hops between *"pneumonia"* and *"lower respiratory tract infection"*. If the query term is *"pneumonia"*, it would be penalized more as the first 3 hops in the path starting from *"pneumonia"* to *"lower respiratory tract infection"* are all generalizations (Figure 6(a)). On the other hand, if the query term is *"lower respiratory tract infection"*, it only suffers from one generalization at the beginning (Figure 6(b)).

Putting it all together. Overall, the online query relaxation uses a novel similarity measure, which takes as input the information content, the contextual information, and the structural information to find semantically related concepts.

$$sim(A, B) = p_{A,B} \times sim_{IC}(A, B). \tag{5}$$

For a given query term, the query relaxation method first finds the corresponding external concept A in the external knowledge source. Then, it searches for the concepts within a distance r from A. Last, our method retrieves the pre-computed similarity between A and each external concept in its neighborhood. Top-k relaxed results are returned based on their similarity scores, where k is application-specific and defined by users. The radius r can be set in different ways. Namely, it can be set as a fixed value by empirical studies, or dynamically decided if a fixed r cannot provide k results.



(a) Path 1: From Pneumonia (query term) to Lower respiratory tract infection.

(b) Path 2: From Lower respiratory tract infection (query term) to Pneumonia.

Figure 6: Example of paths between two external concepts.

Overall, the online query relaxation process is summarized in Algorithm 2. The algorithm receives as input a query term q, along with its associated context c, the instances I from the given KB, the external knowledge source EKS, the set of external concepts that are marked with a flag FEC, the mappings M from I to EKS concepts, the radius r and an integer k, and it returns the top-k results Res from I.

Algorithm 2 Online Query Relaxation Algorithm.

```
Input: Query term $q$, Context $c$, Instances $I$, External Knowledge
    Source $EKS$, Flagged external concepts $FEC$, Mappings $M$,
    radius $r$, integer $k$
Output: Top-$k$ results $Res \subseteq I$
1: $A \leftarrow mapping(q, EKS)$ // concept $A$ corresponds to $q$ in $EKS$
   ▷ candidates are flagged concepts within radius $r$ from $A$
2: $Candidates \leftarrow neighbors(A, EKS, r) \cap FEC$
3: $sort(Candidates, sim(A, B))$          // Equation 5 for context $c$
4: $Res \leftarrow \emptyset$
5: while $|Res| \leq k$ and $|Candidates| > 0$ do
6:     $B \leftarrow Candidates.pop()$ // get next element and remove it
7:     $Res \leftarrow Res \cup \{i|(i, B) \in M\}$
8: end while
9: return $Res$
```

In Line 1, we retrieve the external concept A that corresponds to the query term q, using the same mapping function as in Algorithm 1. Then, we get the set of candidate external concepts within radius r from A, which are marked with a flag, i.e., members of FEC (Line 2), and we sort them in descending order of similarity to A (Line 3). Finally, we iterate through the sorted candidates and add to the results Res the instances i that map to those candidates, until k results have been retrieved, or there are no more candidates (Lines 5-8).

Time complexity analysis. For online query relaxation, we again assume that V is the set of concepts in the external knowledge source and the total number of flagged external concepts FEC is N. Then finding a corresponding external concept A corresponding to the query term q requires $O(|V|)$ time in the worst case. Returning all flagged external concepts within radius r from A requires $O(N)$ time in the worst case (i.e., r is large enough to include all flagged external concepts). Sorting these candidates and returning the top-k results take $\Theta(NlogN)$ and $\Theta(k)$ time, respectively. Hence, the total time complexity of online query relaxation is $\Theta(NlogN)$.

6 APPLICATIONS OF QUERY RELAXATION

Our proposed method is applicable and beneficial to various natural language interface systems, including conversational systems, question and answer systems, as well as natural language query systems to KBs. In this section, we describe how to integrate the query relaxation method with two state-of-the-art systems for the medical data set *MED*, that is used to support evidence-based clinical decisions and patient education, and SNOMED CT as the external knowledge source, since it is one of the most comprehensive and widely used medical knowledge sources.

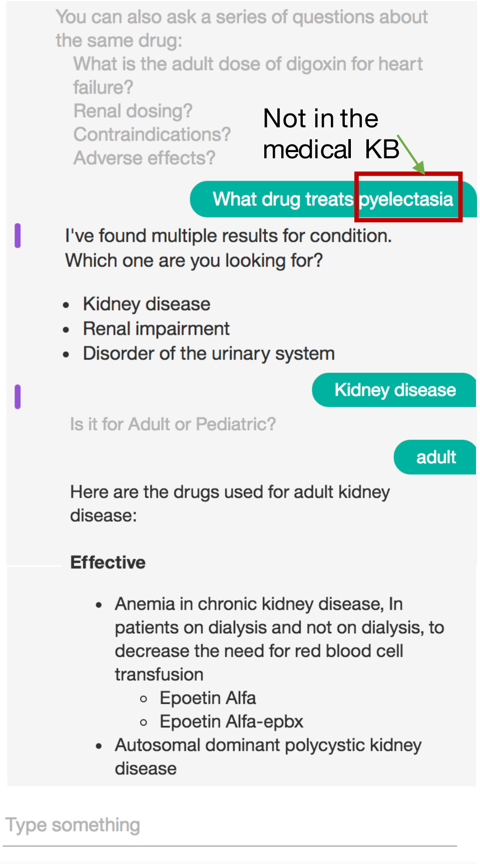
6.1 Integration with a Conversational System

In the following, we describe how we extended a conversational system [33] that is built on top of IBM Watson Assistant [21], to include query relaxation. The query relaxation method is implemented in Java and is deployed on IBM Cloud™ to interact with Watson Assistant. The medical data set is stored in IBM Db2® Database and the external knowledge source (SNOMED CT) is stored in a graph database (i.e., JanusGraph[6]).

As described in Section 4, the possible intents (i.e., contexts) are bootstrapped based on the domain ontology, and our query relaxation method provides training examples to Watson Assistant for intent classification. At query time, Watson Assistant provides the input to our query relaxation method in the form of a [query term, context] pair.

In this case, the context comes from the intent classifier of the conversational system. Regarding the query term, Watson Assistant extracts entity mentions from an input natural language query[7] and passes the unknown entity mentions as query terms to our relaxation method. Next, we showcase two scenarios in which our query relaxation are used (Figures 7 and 8).

The first scenario is to expand the set of queries by using query relaxation when there is no answer in the KB for the user query. For example, when the query term (*"pyelectasia"*) is unknown (i.e., no matching concept in the given KB), Watson Assistant triggers the query relaxation method to find a list of semantically related concepts that are contained in the given KB by utilizing the external knowledge source (i.e., SNOMED CT). As shown in Figure 7, these additional concepts are then used as a means to "repair" the conversation and smoothly redirect the user to the information contained in the KB. Consequently, the conversation can continue with follow-up questions around the expanded result, *"kidney disease"*. Without our query relaxation, Watson Assistant would not be able to return any useful information except replying messages such as *"I don't understand"*. Worse yet, it may return irrelevant and incorrect information to the user, as illustrated in Section 7.2.



You can also ask a series of questions about the same drug:

What is the adult dose of digoxin for heart failure?

Renal dosing?

Contraindications?

Adverse effects?

What drug treats pyelectasia

I've found multiple results for condition. Which one are you looking for?

- Kidney disease
- Renal impairment
- Disorder of the urinary system

Kidney disease

Is it for Adult or Pediatric?

adult

Here are the drugs used for adult kidney disease:

Effective

- Anemia in chronic kidney disease, In patients on dialysis and not on dialysis, to decrease the need for red blood cell transfusion
    - Epoetin Alfa
    - Epoetin Alfa-epbx
- Autosomal dominant polycystic kidney disease

Type something

Figure 7: Integration with Watson Assistant (Scenario 1).

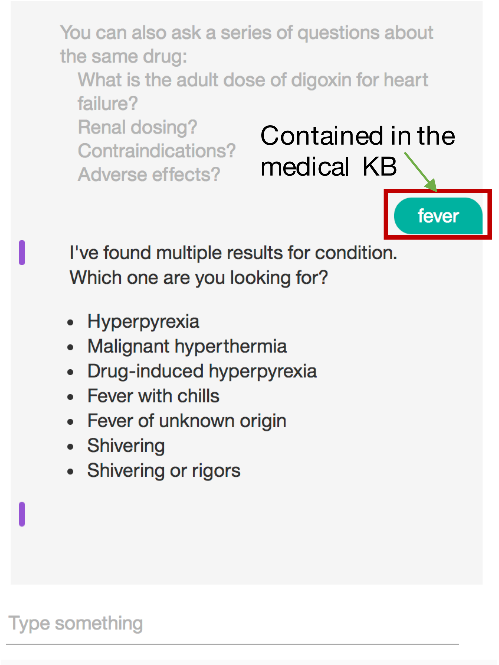
In the second scenario, we use query relaxation to expand query answers beyond what matches to the query term in the KB directly. As seen in Figure 8, the query term (*"fever"*) is identified by Watson Assistant as an instance of the concept (*Finding*) in the medical KB. Without query relaxation, identifying *"fever"* as *Finding* triggers one predefined intent *"Indication-hasFinding-Finding"* in Watson Assistant. Hence, the information such as syndromes and treatments for *"fever"* would be returned to the user. With our query relaxation method for concept expansion, 7 additional concepts related to *"fever"* are returned before providing any information specific to *"fever"*. Hence, it offers more opportunities for the user to explore the information contained in the given knowledge base.

6.2 Integration with a Natural Language Query System

Currently, we are also working on incorporating our method with a natural language query (NLQ) system [23, 35]. The NLQ system is different from the previously described conversational system as it targets one-shot complex queries. In this case, the NLQ system receives a natural language query as input and interprets it over the domain ontology to produce a structured query such as SQL. The proposed query relaxation method is utilized to increase the flexibility and robustness of query interpretation.

[6] https://janusgraph.org/

[7] For a demo, refer to https://natural-language-understanding-demo.ng.bluemix.net.



You can also ask a series of questions about the same drug:

What is the adult dose of digoxin for heart failure?

Renal dosing?

Contraindications?

Adverse effects?

Contained in the medical KB

fever

I've found multiple results for condition. Which one are you looking for?

- Hyperpyrexia
- Malignant hyperthermia
- Drug-induced hyperpyrexia
- Fever with chills
- Fever of unknown origin
- Shivering
- Shivering or rigors

Type something

Figure 8: Integration with Watson Assistant (Scenario 2).

We outline the solution with a running example from the medical domain as captured in Figure 1 for the query: *"What are the risks caused by using Aspirin with pyelectasia"*.

Evidence generation. In the very first step, the NLQ system tries to identify all the different mentions of ontology elements in the input natural language query [35]. The NLQ system iterates through all the word tokens in the query and collects evidences of one or more elements which have been referenced in the input query. These elements can be concepts or relationships in the domain ontology, as well as the instances of those concepts in the knowledge base.

In general, a token can match multiple elements in the ontology. For example, the token *"risks"* is mapped to a concept *"Risk"*, and the phrase *"caused by"* is mapped to *"cause"* relationship in the ontology. *"Aspirin"* is mapped to the concept of *"Drug"*.

There are two types of evidence: (*i*) a *metadata* evidence is generated by matching the token to the ontology elements (e.g., *"Risk"*), and (*ii*) a *data-value* evidence is generated by looking up a token in the knowledge base (e.g., *"Aspirin"*). The evidence for a token can either be *metadata* or *data-value*, but not both [35].

Due to the colloquial and imprecise terminology used in natural language queries, our query relaxation method is particularly useful to increase the capability of query understanding. The NLQ system relies on the semantically related results from our query relaxation method to match a token to either a metadata or a data-value evidence. The NLQ system associates these semantically relevant results with ontology elements on the fly, as shown in Figure 9. Again, if *"pyelectasia"* is not contained in either domain ontology or knowledge base, our query relaxation returns semantically relevant results (e.g., *"kidney disease"*, *"nephropathy"*, etc.), which can be then mapped to the concept of *"Finding"* in the domain ontology.

Interpretation generation. Note that only one element from the evidence set of each token corresponds to the correct query. In this phase, the NLQ system tries all such combination of elements from each evidence set. Each such combination, called *selection set*, is used to generate an interpretation, which is represented as a sub-graph in the semantic graph connecting one evidence for each token for each ontology element [35]. This semantically grounds the words in the input query to specific contexts by referring to elements in the semantic graph. Connecting these referred elements produces a unique interpretation for the given natural language query based on the ontology semantics.

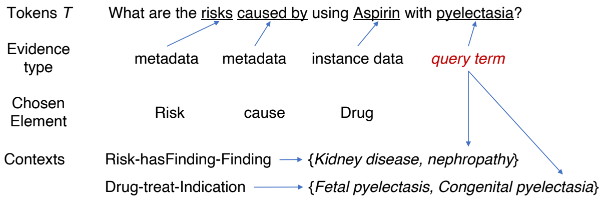


Figure 9: Evidence set.

For each selection set, a sub-tree, called *Interpretation Tree*, is computed, which uniquely identifies the relationship paths among the evidences in the selected set [35]. It is computed by connecting all the elements in the selected set in the semantic graph and satisfying the following constraint. The NLQ system uses a Steiner-tree-based algorithm [35] and ranks the interpretations according to their compactness to generate an interpretation of minimal size for a selected set. Note that query relaxation returns a similarity score associated with each result value. We are currently extending the ranking algorithm to take this ranking score into account, in addition to compactness.

For example, the top ranked interpretation as found from selected set is *ITree* = { (*Drug* → *cause* → *Risk* → *hasFinding* → *Finding*), (*Drug* → *treat* → *Indication* → *hasFinding* → *Finding*) }. Two interpretations have the same compactness. In this case, if we take into consideration the similarity scores associated with the relaxed results, the former interpretation would be more preferable since *"Kidney disease"* is the most semantically similar concept to the search term *"pyelectasia"*.

7 EXPERIMENTAL EVALUATION

In this section, we describe experiments using a medical data set (*MED*) to show the efficacy of our proposed query relaxation method in terms of precision, recall, and F1-score. We also conduct user studies, in which we use Watson Assistant as the conversational interface.

7.1 Experimental Setup

Data set. We use a medical data set (*MED*) that is used to support evidence-based clinical decisions and patient education. The total size of this data set is around 1.2 GB. The ontology corresponding to *MED* consists of 43 concepts and 58 relationships. We chose SNOMED CT as the external knowledge source since it consists of comprehensive information regarding terms, synonyms, and definitions used in clinical documentation and reporting.

Users. 20 Subject Matter Experts (SMEs) participated in our experiments. They all have deep knowledge and understanding of the medical domain, and are able to distinguish between a correct and a wrong answer.

Methodologies. We conducted two sets of experiments to evaluate the efficacy of our proposed query relaxation method with respect to precision and recall. First, we show the superiority of our method compared to alternative methods. Second, a user study demonstrates the benefit of applying our query relaxation method in a conversational system.

7.2 Evaluation Results

Precision and recall. We chose 100 commonly used concepts of medical conditions, and used the methods described below to identify the semantically related concepts. The participants were asked to evaluate whether these relaxed concepts are indeed related to the given ones.

We first study the effectiveness of the methods used for mapping instances from the given KB to the external knowledge source, including exact string matching (*EXACT*), approximate string matching with an edit-distance threshold $\tau = 2$ (*EDIT*), and a variant of word embedding to support longer pieces of text [3] (*EMBEDDING*). Table 1 reports *Precision*, *Recall* as well as *F1-score* of these three mapping methods.

We observe that word embedding achieves the best overall result quality. This provides our query relaxation method a solid foundation to identify semantically related concepts in the neighborhood of these concepts. On the contrary, exact and approximate string matching suffer from lower recall compared to the word embedding method. Hence we used word embeddings in the rest of the experiments as the matching method.

Table 1: Accuracy of mapping methods.

Methods	Precision	Recall	F1
EXACT	**100**	83.33	90.01
EDIT	96.36	88.33	92.17
EMBEDDING	96.49	**91.67**	**94.02**

Next, we compare our proposed query relaxation method (*QR*) against its variants as well as alternative approaches, including our proposed method without the frequency information from the corpus (*QR-no-corpus*), our proposed method without the contextual information (*QR-no-context*), a baseline IC-based semantic measure (*IC*) [2], a baseline method [3, 8] using both pre-trained embeddings [32] (*Embedding-pre-trained*) and embeddings we trained on a given medical document corpus (*Embedding-trained*).

Table 2: Overall effectiveness.

Methods	P@10	R@10	F1
QR	**90.51**	**82.64**	**86.40**
QR-no-context	85.45	77.27	81.15
QR-no-corpus	78.23	70.91	74.39
IC	75.55	68.18	71.68
Embedding-pre-trained	66.14	60.13	62.99
Embedding-trained	79.37	71.81	75.40

Table 2 reports *Precision@10*, *Recall@10* as well as *F1-score* against the *MED* data set. *Precision@10* corresponds to the number of relevant results among the top 10 returned concepts, *Recall@10* is the proportion of relevant results found in the top 10 returned concepts to the total number of relevant results, and *F1-score* is the harmonic mean of *Precision@10* and *Recall@10*. Our proposed methods including (*QR-no-corpus* and *QR-no-context*) are more accurate than the baseline *IC*. Specifically, we observe that *QR-no-context* still returns higher quality results than the baseline *IC*. This shows that differentiating specialization and generalization relationships helps capturing the semantic similarity between a pair of concepts. It is not surprising that *QR-no-context* further improves the result quality when the frequency information is available from the corpus. Regarding *Embedding-pre-trained*, we observe that it achieves the lowest precision and recall among all methods. This is expected as the model was trained on a different medical corpus and many of the words contained in SNOMED CT are out of its vocabulary. For the embedding of multi-word query terms, we used the average its words' embeddings. The result quality of *Embedding-trained* is much improved as we trained the embedding model on our medical document corpus, and the embedding of a (multi-word) query term is further computed based on [3]. However, without the contextual information from the query, many concepts in the given KB are cluttered with the query term in the low-dimensional vector space. Hence the quality of *Embedding-trained* is still not as good as *QR*. In summary, our method *QR*, which incorporates both the frequency information from the corpus, as well as the context from the query, achieves the highest precision and recall.

User study. In this user study, participants were asked to complete two tasks to evaluate the query understanding capability of a conversational system with and without our query relaxation method. The participants were allowed to get familiar with the conversational system over the given KB. In task 1 (*T1*), we asked participants to come up with 20 questions around 20 given concepts (i.e., condition names). For example, the participant may ask "what drugs are used to treat [*condition*]" or "what drugs cause [*condition*]". In task 2 (*T2*), the participants were allowed to come up with 10 questions of their own choice about anything in *MED*.

The participants were then asked to grade the quality of the relaxed results in a scale of 1-5. If the system returns a correct response in the first attempt, it receives 5 points. If the system fails to return a correct response, the participants can rephrase their questions for at most 4 more times. Each time a wrong result is returned, the participant subtracts a point. For example, if the correct answer is returned after 3 attempts (i.e., 2 failed attempts) the participant gives in total 5-2 = 3 points. In addition to the score, we also asked the participants to provide detailed feedback.

Table 3: Watson Assistant with and without QR.

	QR		***no QR***	
Score	***T1***	***T2***	***T1***	***T2***
1 (Very dissatisfied)	2.1%	10.55%	13.06%	11.11%
2 (Dissatisfied)	10.35%	11.07%	16.87%	38.26%
3 (Okay)	25.59%	29.33%	36.29%	30.85%
4 (Satisfied)	35.21%	33.37%	18.25%	12.47%
5 (Very satisfied)	26.85%	15.68%	15.53%	7.31%
AVG	3.73	3.31	3.06	2.67

Table 3 shows the aggregated grades received by our query relaxation method for the two tasks described above. The numbers in the table show the percentage of each particular grade. Clearly, the conversation system with query relaxation achieves a substantially higher score than the one without query relaxation in both tasks. The average grades of the system with query relaxation in both tasks are 20% higher than the ones without relaxation. Specifically, our query relaxation method performed slightly better in *T1* than *T2*. The reasons are that the questions used in *T2* were completely from the participants and a non-trivial number of questions (9 out of 200) do not have an answer in the given KB, as opposed to *T1*, where 20 concepts were provided to the users. Moreover, the feedback from the participants indicates that the lower grades in Table 3 are due to other reasons orthogonal to the quality of our query relaxation method.

Specifically, there are 7 incidences in which the expected answers are not contained in the given KB (*MED*). There are 11 incidences where the users complained about the conversational flow irrespective of the query relaxation results. For example, some users prefer smaller number of interactions with the conversational system in order to complete their tasks. Some users failed to follow the instructions, and hence ran into unexpected follow-up questions. Moreover, there are 10 incidences that the users did not provide any negative feedback but gave a low grade such as 1 or 3. Last but not the least, the SMEs reported 6 instances that the amount of information returned is overwhelming even though the relaxed results are semantically correct. All these cases resulted in low grades, including 1, 2, and 3. Note that all the above cases are counted in Table 3. One straightforward solution to address these issues would be to incorporate the user's relevance feedback [39] in the query relaxation method, and to progressively improve the relaxed results.

8 RELATED WORK

Natural language querying over KBs. Several approaches have been recently investigated to build natural language interfaces to KBs. In [13], query templates are learned from KBs and question answering corpora. Wang et al. [41] leverage terms and their relationships from a web corpus and map them to related concepts using a KB. Then, a random walk-based algorithm is proposed for understanding the terms in a given query. The semantic parsing methods proposed in [4] use a domain-independent representation derived from combinatory categorical grammar parsers. Most recently, a supervised learning framework [20] is introduced to exploit sentence embedding for the medical question answering task. However, they usually require a large labeled corpus or pairs of questions and answers. Our approach is complementary to these methods since they only focus on answering queries with 'strict' execution, which often results in no answers. Our query relaxation expands the domain vocabulary used in queries and provides more semantically related results.

Query relaxation for databases. The database community has developed query relaxation methods that return information beyond a standard query. Query relaxation expands the query selection criteria to include additional relevant information, often by consulting a semantic model of the data domain. Gaasterland [18] introduces query relaxation techniques in deductive databases, using logic rules to specify legal relaxation constraints. Query relaxation [11] is introduced to relational databases using type-abstraction hierarchies to find semantically similar query results. A taxonomy-based relational algebra is proposed to extend the capability of selection and join by relating values occurring in the tuples with values in the query using the taxonomy [26]. Poulovassilis et al. [31] applied query approximation and query relaxation techniques based on RDFS inference rules to the evaluation of conjunctive regular path queries (CRPQs) over graph data. Our approach is different from the above work as we leverage external knowledge sources to expand the domain vocabulary. Further, our approach uses the domain ontology and context information to differentiate the semantic subtleties among instances.

Semantic similarity measures. Among various semantic measures, path finding measure [42] is based on the shortest path separating concepts, which traverses the LCS of two concepts. IC-based measures can be estimated solely from the structure of a taxonomy [36], or from the distribution of concepts in a text corpus and a taxonomy [34]. Our semantic similarity measure is designed upon these measures and overcomes their limitations by utilizing the domain ontology to differentiate the semantic subtleties. Hence our method can achieve significant gain of recall without sacrificing the precision.

Recent works demonstrated that deep learning models built at word [8, 30] or sentence [3, 14] level can be used for semantic similarity estimation. However, these methods demand high quality training data sets, which is critical and expensive in reality. Moreover, directly applying word or sentence embeddings to our problem is not sufficient since the structural and contextual information are not considered when the model is trained. Hence, we use word and sentence embeddings for linking the given KB to the external knowledge source and build our similarity measure on top of it.

9 CONCLUSION

In this paper, we present a novel two-phase query relaxation method that leverages external knowledge sources to expand answers for querying medical KBs. We introduce a novel similarity metric to empower our query relaxation method to identify semantically related concepts. Our method is successfully integrated with two exemplary systems, a conversational system and a natural language query system, respectively. Our experiments show that our query relaxation method for the medical KB outperforms state-of-the-art methods, including deep learning-based ones, in precision and recall. We also conduct a user study demonstrating how our query relaxation method expands the query results and improves their quality for medical KBs.

REFERENCES

[1] Amazon. 2019. Amazon Comprehend Medical. https://aws.amazon.com/comprehend/medical/.

[2] Mohamed Ben Aouicha and Mohamed Ali Hadj Taieb. 2016. Computing semantic similarity between biomedical concepts using new information content approach. *Journal of Biomedical Informatics* 59 (2016), 258–275.

[3] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *ICLR*.

[4] Yoav Artzi, Nicholas FitzGerald, and Luke Zettlemoyer. 2013. Semantic Parsing with Combinatory Categorial Grammars. In *ACL (Tutorials)*.

[5] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC*. 722–735.

[6] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (Eds.). 2003. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press.

[7] Christopher M. Bishop. 2007. *Pattern recognition and machine learning, 5th Edition.* Springer.

[8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[9] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*. 1247–1250.

[10] Vassilis Christophides, Vasilis Efthymiou, and Kostas Stefanidis. 2015. *Entity Resolution in the Web of Data.* Morgan & Claypool Publishers.

[11] Wesley W. Chu, Hua Yang, Kuorong Chiang, Michael Minock, Gladys Chow, and Chris Larson. 1996. CoBase: A Scalable and Extensible Cooperative Information System. *J. Intell. Inf. Syst.* 6, 2-3 (1996), 223–259.

[12] The Gene Ontology Consortium. 2019. The Gene Ontology knowledgebase. http://geneontology.org/.

[13] Wanyun Cui, Yanghua Xiao, and Wei Wang. 2016. KBQA: An Online Template Based Question Answering System over Freebase. In *IJCAI*. 4240–4241.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[15] Xin Luna Dong and Divesh Srivastava. 2015. *Big Data Integration.* Morgan & Claypool Publishers.

[16] DrugBank. 2019. DrugBank. https://www.drugbank.ca/.

[17] Marcus Fontoura, Vanja Josifovski, Ravi Kumar, Christopher Olston, Andrew Tomkins, and Sergei Vassilvitskii. 2008. Relaxation in text search using taxonomies. *PVLDB* 1, 1 (2008), 672–683.

[18] Terry Gaasterland. 1997. Cooperative Answering through Controlled Query Relaxation. *IEEE Expert* 12, 5 (1997), 48–59.

[19] Vijay Garla and Cynthia Brandt. 2012. Semantic similarity in the biomedical domain: An evaluation across knowledge sources. *BMC bioinformatics* 13 (2012), 261.

[20] Yu Hao, Xien Liu, Ji Wu, and Ping Lv. 2019. Exploiting Sentence Embedding for Medical Question Answering. In *AAAI*. 938–945.

[21] IBM. 2019. Watson Assistant. https://www.ibm.com/cloud/watson-assistant/.

[22] Danai Koutra, Neil Shah, Joshua T. Vogelstein, Brian Gallagher, and Christos Faloutsos. 2016. DeltaCon: Principled Massive-Graph Similarity Function with Attribution. *TKDD* 10, 3 (2016), 28:1–28:43.

[23] Chuan Lei, Fatma Özcan, Abdul Quamar, Ashish R. Mittal, Jaydeep Sen, Diptikalyan Saha, and Karthik Sankaranarayanan. 2018. Ontology-Based Natural Language Query Interfaces for Data Exploration. *IEEE Data Eng. Bull.* 41, 3 (2018), 52–63.

[24] Jiaqing Liang, Yi Zhang, Yanghua Xiao, Haixun Wang, Wei Wang, and Pinpin Zhu. 2017. On the Transitivity of Hypernym-Hyponym Relations in Data-Driven Lexical Taxonomies. In *AAAI*. 1185–1191.

[25] Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. In *ICML*. 296–304.

[26] Davide Martinenghi and Riccardo Torlone. 2014. Taxonomy-based relaxation of query answering in relational databases. *VLDB J.* 23, 5 (2014), 747–769.

[27] Víctor Martínez, Fernando Berzal, and Juan Carlos Cubero Talavera. 2017. A Survey of Link Prediction in Complex Networks. *ACM Comput. Surv.* 49, 4 (2017), 69:1–69:33.

[28] Microsoft. 2019. Language Understanding (LUIS). https://www.luis.ai/home.

[29] Ted Pedersen. 2010. Information Content Measures of Semantic Similarity Perform Better Without Sense-tagged Text. In *NAACL*. 329–332.

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.

[31] Alexandra Poulovassilis, Petra Selmer, and Peter T. Wood. 2016. Approximation and relaxation of semantic web path queries. *Journal of Web Semantics* 40 (2016), 1 – 21.

[32] Sampo Pyysalo, Filip Ginter, Hans Moen, et al. 2013. Distributional semantics resources for biomedical text processing. In *Proceedings of Languages in Biology and Medicine.*

[33] Abdul Quamar, Chuan Lei, Dorian Miller, Fatma Özcan, Jeffrey Kreulen, Robert J Moore, and Vasilis Efthymiou. 2020. An Ontology-Based Conversation System for Knowledge Bases. *Under review* (2020).

[34] Philip Resnik. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *IJCAI*. 448–453.

[35] Diptikalyan Saha, Avrilia Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R. Mittal, and Fatma Özcan. 2016. ATHENA: An Ontology-Driven System for Natural Language Querying over Relational Data Stores. *PVLDB* 9, 12 (2016), 1209–1220.

[36] Nuno Seco, Tony Veale, and Jer Hayes. 2004. An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *ECAI*. 1089–1090.

[37] Jaydeep Sen, Fatma Özcan, Abdul Quamar, Greg Stager, Ashish R. Mittal, Manasa Jammi, Chuan Lei, Diptikalyan Saha, and Karthik Sankaranarayanan. 2019. Natural Language Querying of Complex Business Intelligence Queries. In *SIGMOD*. 1997–2000.

[38] SNOMED. 2019. SNOMED Clinical Terms. https://www.snomed.org/snomed-ct/what-is-snomed-ct.

[39] Yu Su, Shengqi Yang, Huan Sun, Mudhakar Srivatsa, Sue Kase, Michelle Vanni, and Xifeng Yan. 2015. Exploiting Relevance Feedback in Knowledge Graph Search. In *KDD*. 1135–1144.

[40] UMLS. 2019. Unified Medical Language System. https://www.nlm.nih.gov/research/umls/.

[41] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query Understanding Through Knowledge-based Conceptualization. In *IJCAI*. 3264–3270.

[42] Zhibiao Wu and Martha Palmer. 1994. Verbs Semantics and Lexical Selection. In *ACL*. 133–138.

[43] Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. 2016. Neural Enquirer: Learning to Query Tables in Natural Language. In *IJCAI*. 2308–2314.