Aalborg Universitet

AALBORG UNIVERSITY
DENMARK

Graph-Query Suggestions for Knowledge Graph Exploration

Lissandrini, Matteo; Mottin, Davide; Palpanas, Themis; Velegrakis, Yannis

Published in:
The Web Conference 2020 - Proceedings of the World Wide Web Conference, WWW 2020

DOI (link to publication from Publisher):
10.1145/3366423.3380005

Creative Commons License
CC BY 4.0

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Link to publication from Aalborg University

Citation for published version (APA):
Lissandrini, M., Mottin, D., Palpanas, T., & Velegrakis, Y. (2020). Graph-Query Suggestions for Knowledge Graph Exploration. In *The Web Conference 2020 - Proceedings of the World Wide Web Conference, WWW 2020* (pp. 2549-2555). Association for Computing Machinery. https://doi.org/10.1145/3366423.3380005

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

? Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
? You may not further distribute the material or use it for any profit-making activity or commercial gain
? You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from vbn.aau.dk on: August 24, 2021

Graph-Query Suggestions for Knowledge Graph Exploration

Matteo Lissandrini
Aalborg University
matteo@cs.aau.dk

Davide Mottin
Aarhus University
davide@cs.au.dk

Themis Palpanas
University of Paris
themis@mi.parisdescartes.fr

Yannis Velegrakis
Utrecht University
i.velegrakis@uu.nl

ABSTRACT

We consider the task of exploratory search through graph queries on knowledge graphs. We propose to assist the user by expanding the query with *intuitive* suggestions to provide a more informative (full) query that can retrieve more detailed and relevant answers. To achieve this result, we propose a model that can bridge graph search paradigms with well-established techniques for information-retrieval. Our approach does not require any additional knowledge from the user and builds on principled language modelling approaches. We empirically show the effectiveness and efficiency of our approach on a large knowledge graph and how our suggestions are able to help build more complete and informative queries.

ACM Reference Format:
Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. 2020. Graph-Query Suggestions for Knowledge Graph Exploration. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3380005

1 INTRODUCTION

A knowledge graph models facts as a set of subject-predicate-object triples forming a graph [8, 38]. These resources gained much interest and hence the task of knowledge graph search via graph queries is of paramount importance. These graph-queries are structures that describe the characteristics of the elements of interest [6, 18, 27, 48]. Yet, in query answering, novice users, especially in *exploratory* use cases, are often unable to fully specify a structure that would have retrieved all the information of interest [22, 28, 44].

One useful exploratory query paradigm in knowledge graphs is *exemplar queries* [18, 24–27]. As opposed to traditional query answering, in which the query is a set of specifications for the elements of interest, an exemplar query is *an example* from the many elements of interest. As such, the answers are not the structures that comply exactly to the query-graph, but elements that have a *similar* structure to the one in the user input. For instance, in the query ⟨A. Kleiner, supervised, A. Einstein⟩, one answer can be ⟨D. Sciama, supervised, S. Hawking⟩.

Given a graph-query, there are many different ways it can be expanded. Yet, not all expansions may be of interest to the user, and a large number of expansions may overload the user. We are the first to propose an *interactive graph-query expansion* for graph exemplar queries on knowledge graphs. Interactive query expansion refers to the problem of retrieving a set of suggestions and the user selecting one of them [12]. Previous works on interactive graph-query expansion focus on the task of entity-list completion [36, 47], while existing systems like SPARKLIS [10], suggest entities that match particular user specification, disregarding structures with complementary information [36, 47]. As a consequence, they are not suitable for generic exploratory needs. In our case, the user provides a partial graph exemplar query and the system responds with the k most relevant expansions to complement such query.

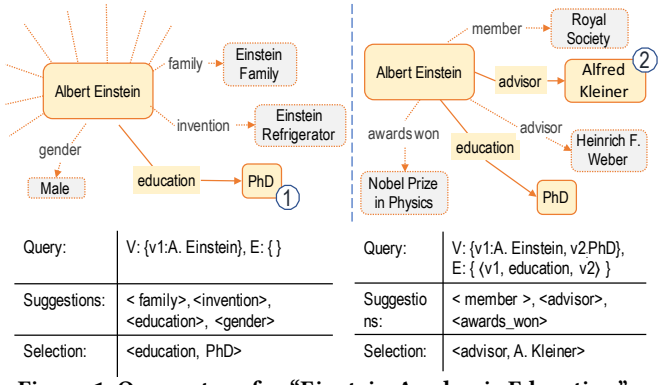

Figure 1: Query steps for "Einstein Academic Education".

Suggesting graph-query expansions requires a way to assess the likelihood a candidate expansion represents the actual user need. Graph-query suggestions have not been studied formally so far, and traditional IR approaches were not designed to work in such a domain. Traditional IR methods cannot readily adapt to the structure of knowledge bases that have no document structure. Recent work has proposed the contextualization of knowledge graph facts [43] as a *supervised learning* problem which requires complex training and manually labelled data and does not offer a model for query expansion compatible with the classical IR models.

In this work, we present **a novel approach to suggest query expansions for graph-queries**. Graph-query expansions are edges that can be added to the current query. Our model expands IR methods based on pseudo-relevance feedback and language-models [4, 31] by including *structural information* described by the query and answer graphs through neighbour edge-labels. Graph-query expansions are then ranked and proposed to the user. Our approach does not pose restrictions on the structure of the knowledge graph nor requires labeled data (e.g., query logs) as in previous methods [7, 16, 17, 43]. The simplicity of the model offers much higher flexibility and efficiency to run in massive graphs like Freebase.

Running example. Consider a student querying for facts about *the education of famous scientists*, who knows only A. Einstein as an example. To obtain information about the scientists' education using the exemplar query formalism, it is necessary to specify a graph query describing the relationships or attributes of interest by including in the exemplar query A. Kleiner as Einstein's advisor. An exemplar query engine [18, 27] will retrieve all the other similar

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '20, April 20–24, 2020, Taipei, Taiwan
© 2020 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380005

structures that include those aspects. A person oblivious to these details would only know Einstein's name and would not be able to specify such example. Our goal is to help these users.

In our work, the user starts with the query "A. Einstein". The system then suggests additional information, e.g., the facts that he invented the "Einstein Refrigerator", or that he had a PhD. The user will then select the most suitable suggestion among the options, e.g., ⟨education, PhD⟩ (Figure 1, left). This latter selection is interpreted as a new query and the system will retrieve a list of scientists accompanied by their education. The system can also respond with a new set of suggestions. For example, given the expansion ⟨education, PhD⟩, the system proposes more related options (i.e., advisors, and awards, as in Figure 1, right). The process continues until the user is satisfied with the answers. Without such a system, the user would be forced to a laborious search among all available edges, often hundreds or thousands.

Therefore, in this work: (i) We formally define the problem of Suggesting Graph-Query Expansions (Section 3) and provide a model based on intuitions from language-modelling and relevance feedback (Section 4). (ii) We propose two unsupervised methods to estimate the relevance of expansions based on edge frequency and compare them with two strong baselines: *surprise* [33] and Markov models [3]. (iii) We show that our framework outperforms other methods even when the query contains just one edge, obtaining an NDCG score between 0.5 and 0.6 on the top-10 suggestions. (iV) We show with a user-study that our framework provides useful suggestions quickly by exploiting only edge frequency. Yet, the framework can integrate more complex structures if needed.

2 RELATED WORK

Note that oftentimes in *exploratory search*, the user is not able to express their information needs in clear terms [44]. Hence we cannot rely on the user to provide a precise description of the graph structures they are interested in. To our knowledge, no previous work tackles the problem of *example driven* graph-query expansion in knowledge graphs. Related works for query expansions and knowledge graph exploration are outlined in Table 1, yet most of them miss important features to address this problem.

Query Expansion and Suggestion. Query expansion improves document search by including additional terms in the user's query either automatically (implicitly), or interactively. *Automatic Query Expansion (AQE)* [5] is a one-shot approach that augments the query with additional terms and uses the expanded query to retrieve a different set of documents. Such automatic methods do not incorporate user feedback and potentially deliver irrelevant results [32]. Rather than changing the query directly, *Interactive Query Expansion (IQE)* shows alternative (expanded) queries to the user, generated either via user *explicit* feedback on the results [12, 39], or *implicitly* via some prior information [2, 11]. Yet, the scarcity of user feedback can hinder the applicability of such methods.

Other works harness external sources [14, 19] or pseudo-relevant feedback [4, 20] to help generating expansions. With *external sources*, expansion terms are retrieved from thesauri [14] or knowledge graphs [19]. Knowledge graphs and thesauri are accessed as dictionaries, thus such methods do not deal with the expansion of graph queries. On another vein, *Pseudo-Relevant Feedback (PRF)* techniques [4] build expansions in a *data-driven* manner by considering the words contained in the user query's results. Pseudo-relevant feedback embodies language models [4] based on word co-occurrences and provides an effective solution for keyword query expansion in documents. *However, until now there has been no proper adaptation of such models on graphs.* Graph Relevance Feedback (GRF) [37] proposes the use of a ground-truth query set similar to pseudo-relevant feedback, but only to re-rank answers for keyword queries on graphs, without proposing any graph-query expansion.

Exploratory search in knowledge graphs. Exploratory methods overcome the rigidity of declarative languages in graphs, such as SPARQL [35]. *Entity search* allows for automatic completion of a set of seed entities (persons, organizations, places). Such an expanded list of entities can complement an ambiguous user query [19] or provide explanations of the original seed entities [47]. However, these methods do not interact with the users towards their intended answers, nor do they provide any *query construction* mechanism. Exemplar queries [27] allows users to specify a representative of the results of an unknown query; the algorithm then discovers and returns the other results. Similarly, Graph Query by Example (GQBE) [18] extends the idea to multiple exemplar entity tuples. *Example-based* queries, albeit expressive, do not interact with the user and require the input of a full example (a subgraph or a tuple).

Graph navigation systems. User interfaces for query formulation [9, 10, 15, 17, 34] either (i) assume the user can navigate through the entire graph structure (often ranking suggestions in decreasing frequency order [10]), but do not restrict the potentially large number of expansions [9, 15, 30], or (ii) rely on heuristic approaches based on query logs [17]. Similarly, *faceted search* allows smart filtering of large result sets along different attributes [13]. Faceted search proposes no preferred suggestion, leaving the user unassisted. On the other hand, our approach provides concrete and relevant suggestions on how to expand graph-queries.

Knowledge graph fact contextualization [43] augments a given knowledge graph fact (edge) with additional facts that help the user understand its context. A *supervised* fact contextualization method (NFCM) trains a neural network on hand-crafted features and human-annotated data enriched with meta-data extracted from Wikipedia. This supervised machine-learning process is aimed at producing short natural language descriptions of a fact. Instead, we assist the graph-query construction in an unsupervised manner.

3 PROBLEM FORMULATION

A knowledge graph is a set of entities and relationships among them. Knowledge graphs are represented as directed labelled multi-graphs, in which *nodes* model entities, *edges* relationships among them, and *labels* the names of entities and relationships. Given a finite set $\mathcal{L}$ of entity and relationship labels, a *knowledge graph* $\mathcal{K}$ is described by a triple $\langle V, E, \mathcal{L}\rangle$, where V is the set of entities and $E \subseteq V \times V \times \mathcal{L}$ is a set of relationships (or facts) among entities represented as labeled edges. In the following, to ease the presentation, we also indicate with $\ell : V \cup E \to \mathcal{L}$ a labelling function on entities and relationships.

A graph $G' : \langle V', E', \mathcal{L}'\rangle$ is a *subgraph* of $G : \langle V, E, \mathcal{L}\rangle$, denoted as $G' \sqsubseteq G$, if $V' \subseteq V$, $E' \subseteq E$, and $\mathcal{L}' \subseteq \mathcal{L}$. We denote as $\mathcal{P}_{\sqsubseteq}(G)$ the set of all subgraphs of G. A graph-query is also a graph $Q : \langle V_Q, E_Q, \mathcal{L}_Q\rangle$. We adopt the exemplar queries semantics for the user's query. In

Method	External source	Data model	Structured query	Query expansion	Interactive	Data-driven	Example-based
AQE [5]	*query-logs*	*documents*	✘	✔	✘	✔	✘
Explicit feedback [12, 39]	*user feedback*	*documents*	✘	✔	✔	✘	✘
Implicit feedback [2, 11]	*query-logs*	*documents*	✘	✔	✔	✔	✘
External-sources [14, 19]	*knowledge graphs*	*documents*	✘	✔	✔	✘	✘
Pseudo-relevant [4, 20]	*none*	*documents*	✘	✔	✔	✔	✘
GRF [37]	*Wikipedia*	*graphs*	✘	✘	✘	✔	✘
Entity search [19, 47]	*none*	*graphs*	✘	✘	✘	✔	✔
Contextual Ent. search [1, 41]	*keywords*	*graphs*	✘	✘	✘	✔	✔
Exemplar queries [25, 27]	*none*	*graphs*	✔	✘	✘	✔	✔
GQBE [18]	*none*	*graphs*	✔	✘	✘	✔	✔
Sparklis [10]	*none*	*graphs*	✔	✘*	✔	✔	✘
NFCM [43]	*Wikipedia*	*graphs*	✘	✔	✘	✔	✔
Graph Query Suggestion	*none*	*graphs*	✔	✔	✔	✔	✔

Table 1: Outline of related work in terms of fulfilled (✔) and missing (✘) properties of query expansions for exploratory search.

exemplar queries [6, 18, 27], the user issues queries on a knowledge graph $\mathcal{K}$ by means of a subgraph $Q \sqsubseteq \mathcal{K}$. Thereby, a graph-query is interpreted as an example or representative of the intended results.

Example 3.1. Given a user looking for advisors of famous scientists: one example is Alfred Kleiner that can be provided as the exemplar query identified by the edge ⟨A. Einstein, advisor, A. Kleiner⟩.

It follows that, when performing the search task, an answer to a graph-query is a subgraph *similar* to the one the user provides.

Definition 3.2 (Exemplar Queries [18, 27]). The *answers* to a query Q is a set of subgraphs of the knowledge graph that is similar to Q for some similarity relation $\sim$, i.e., $\mathcal{A} = \{A \mid A \sqsubseteq \mathcal{K} \wedge A \sim Q\}$.

The above definition depends on the specific choice of the similarity $\sim$. This problem has been extensively studied in the context of graph search [17, 27]. Here, we use edge-label preserving subgraph isomorphism as our default similarity [17, 27]. For instance, for the user query in Example 3.1, expected answers are other subgraphs like ⟨Hendrik Lorentz, advisor, Pieter Rijke⟩, and ⟨Niels Bohr, advisor, Christian Christiansen⟩. Hence, we want to suggest expansions to graph-queries represented by graph-structures as defined above.

Graph-Query Expansions. It is unrealistic to assume that a user is able to operate on complex graphs and provide a complete subgraph representing an example of the intended results. We instead consider a user that provides an initial query Q, which is treated as a partial specification of the example. Then, we enable the system to recommend additional information (in the form of edges) to be added to such example to better specify the user intent. The suggested edges are called *query expansions*. In this work, we accept as a starting query Q any connected subgraph of $\mathcal{K}$; i.e., both *arbitrary structures* as well as *queries with just a single entity* (i.e., a node in the graph). An expanded graph-query is then a super-graph of the original query.

Definition 3.3 (Expanded Query). An *expanded query* $Q' : \langle V_{Q'}, E_{Q'}, \mathcal{L}_{Q'} \rangle$ of a query $Q : \langle V_Q, E_Q, \mathcal{L}_Q \rangle$ is a connected graph $Q \sqsubseteq Q'$, such that $|E_{Q'}| > |E_Q|$. $E_{Q'} \setminus E_Q$ is the set of query expansions.

Example 3.4. Assume the user query in Example 3.1. Possible query expansions are edges like ⟨A. Einstein, invention, Einstein Refrigerator⟩, or ⟨A. Kleiner, employer, University of Zurich⟩.

Suggesting Query Expansions. A graph-query suggestion system needs to select only a subset of the possible expansions to be presented to the user. Indeed, the straightforward approach, which generates all possible expansions through all the neighbouring relationships around the initial query, overloads the user with too many options. For instance, the entity Albert Einstein in Freebase has 500+ outgoing relationships with other entities, not counting attributes like age or height. As such, we require an intelligent way to select only those relationships that are more likely to describe the type of information the user had in mind. We formulate the problem in a ranking-retrieval fashion by defining a *relevance function* ρ on the possible relationships that can be added to the current query Q to obtain the expansion Q' on the knowledge graph $\mathcal{K} : \langle V, E, \mathcal{L} \rangle$.

We denote the set of candidate expansion edges as E_δ. Hence, once presented with the set of expansions E_δ, the user either selects one expansion edge $e' \in E_\delta$ to form the expanded query Q', or interrupts the process if no expansion is required. Note that for interactive exploration, when an expansion is selected, the expanded query Q' can be provided as input for further expansion, hence we need only to model the single step. Hence, we tackle the following:

Problem 1 (Graph Query Suggestion). *Given a knowledge graph $\mathcal{K} : \langle V, E, \mathcal{L} \rangle$, a number $k > 0$, and an initial query Q, retrieve the top-k edges set $E_\delta \subset E$, $|E_\delta| \leq k$, ranked according to relevance function ρ. The elements from E_δ are returned as suggested query expansions.*

4 GRAPH-QUERY SUGGESTION MODEL

Next, we provide an effective model for the unknown user relevance function ρ. With such function we obtain a list of k expansion edges $E_\delta \subseteq E$ that is shown to the user, among which they can choose the desired expansion. We devise models to discover the implicit user preference given the scarce feedback and relying solely on the data and the query. Our model instantiates ρ as an estimate of the likelihood of the user selecting a specific expansion edge given the query they provided. To this effect, we first provide an appropriate model for the graph-query, and then expand such a model for computing likelihood of a candidate expansion.

4.1 Bag-of-Labels Model for Graph-Query

One of our core contributions is a simple, yet powerful, modelling that *bridges the gap between knowledge graphs and the well-known retrieval models applied to documents*. We first establish a parallel between graph-query-suggestion and query expansion for keyword queries [12] with reference to language models [31]. The language model in keyword queries relates the relevance of a document D to the intent expressed by a keyword query $\mathbf{q}$. The intent behind a query is defined by a distribution over the words from which the user samples the content of the query [31]. In language models, each document D in a collection C is described by a multinomial

distribution $\hat{p}(w|D)$ over each keyword w [31]. Under such model, the likelihood $\hat{p}(D|\mathbf{q})$ for a query $\mathbf{q}=\langle w_1, w_2 \ldots w_n\rangle$ is proportional to $\hat{p}(\mathbf{q}|D)\hat{p}(D)$, where $\hat{p}(D)$ is a prior on D. Intuitively, the user's query describes a summary of a document through a sample of words drawn from a distribution over the document collection. Therefore, identifying which new keyword w' could be added to $\mathbf{q}$ turns into estimating the probability $\hat{p}(w'|\mathbf{q})$. Hence, we estimate the probabilistic model $M_{\mathbf{q}}$ (a multinomial distribution over keywords) that generated $\mathbf{q}$ as well as the one generating D, i.e., M_D.

To bridge the gap between traditional search models and graph-search, we propose an adapted representation for a graph. In particular, we note that a graph query describes a set of relationships, which are characterized by the respective edge labels. As such, a graph can be modelled as follows:

Definition 4.1 (Bag of Labels Model of a Graph). Given a graph $G:\langle V_G, E_G, \mathcal{L}_G\rangle, G \sqsubseteq \mathcal{K}$, its adapted representation as a multiset (or bag) is $Bag(G):\left\{l^{m(l)} | \langle v_1, v_2, l\rangle \in E_{\mathcal{K}} \wedge (v_1 \in V_G \vee v_2 \in V_G)\right\}$.

$m(l)$ represents the cardinality of label l in the $Bag(G)$ (duplicate edges are preserved) as $|\left\{\langle v_1, v_2, l\rangle \in E_{\mathcal{K}} \wedge (v_1 \in V_G \vee v_2 \in V_G)\right\}|$.

Under this definition, two graphs are similar if they contain similar bags of edge labels. Note that an edge label can appear multiple times around nodes in a graph, so the frequency of edge labels is as important as the frequency of keywords in the corresponding bag-of-words model. Furthermore, we do not include only edges that appear in the current graph, but *also edges on the fringe that connects the graph-nodes with the surrounding portion of the knowledge graph.* This choice has two positive effects: (1) it enriches the description of $Bag(Q)$ and $Bag(G)$, and (2) allows the model to be applicable even when Q contains just a single node. Our experiments show that this modelling choice effectively captures enough information and still enables fast computations, despite its simplicity. Given our *bag of labels model*, we estimate M_G for any graph $G \in \mathcal{P}_{\sqsubseteq}(\mathcal{K})$ using label frequencies in $Bag(G)$. That is, M_G is a multinomial distribution over edge-labels, which is estimated according to the frequency of edge labels in the corresponding $Bag(G)$.

4.2 Baseline Scoring-Functions

We present two baseline techniques for computing $\rho(Q,e)$ as the likelihood of choosing the edge e with label $\ell(e)=l$, given a graph query Q with some modeled edge-label distribution M, i.e., $\hat{p}(l|M_Q)$. The first method is based on maximum likelihood estimation, where the score of a label l is proportional to its relative frequency around the graph-query and in the entire repository, i.e., frequent labels are considered more likely to be part of a query. The model is estimated with Dirichlet smoothing (according to previous work [45]) as:

$$\hat{p}(l|M_Q)_{MLE} = \frac{|E^l_Q| + \varepsilon \hat{p}(l|\mathcal{K})}{|E_Q| + \varepsilon}, \tag{1}$$

where ϵ is the Dirichlet prior (a system-wide constant usually between 1000 and 2000 [45]), $|E^l_Q|$ is the number of edges in Q with label l, $|E_Q|$ is the total number of edges in Q, and $\hat{p}(l|\mathcal{K})$ represents the probability of l in the collection of target graphs, but in our case is approximated by $|E^l|/|E|$ (E^l are edges with label l in $\mathcal{K}$).

The second technique favours distinctive labels, i.e., labels that are frequent around the query, but infrequent in the dataset. This score is computed with the KL-divergence [20], as follows:

$$\hat{p}(l|M_Q)_{KL} \propto exp\left(\frac{1}{(1-\lambda)}\log(\hat{p}(l|M_Q)) - \frac{\lambda}{(1-\lambda)}\log(\hat{p}(l|\mathcal{K}))\right) \tag{2}$$

The parameter $\lambda \in [0,1)$ is a weighting parameter that for document search depends on the frequency of keywords in the collection, while in our case, depends on the frequency of labels in the graph.

4.3 Scoring with Pseudo-Relevance Feedback

Next, we describe a model-based approach inspired by the pseudo-relevance feedback framework [4, 21, 46]. Under this model, we estimate the likelihood of a candidate expansion-edge based on its relative frequency within a *pseudo-relevance set* of graphs retrieved by the original query. Based on that, a new model is estimated (M_{rel}) to evaluate the likelihood of a candidate query expansion.

As such, we process the current graph-query and obtain the pseudo-relevance set $\mathcal{G}_{rel}=\{G_1,\ldots,G_k\}$ of answer graphs that match the original user query. We retrieve those answers by applying entity similarity [36] for single entities, or exemplar queries [18, 27] for generic graph-queries. The relevance score for the candidate expansion label l can be obtained through maximum likelihood estimation from the set $\mathcal{G}_{rel}$ of (pseudo-)relevant graphs and the corresponding set of bags of labels $Bag(\mathcal{G}_{rel})$

$$\hat{p}(l|M_{rel})_{MLE} \approx \sum_{G \in \mathcal{G}_{rel}} \hat{p}(l|M_G)\hat{p}(Q|M_G), \tag{3}$$

where $\hat{p}(Q|M_G) \propto \prod_{l \in Bag(Q)} \hat{p}(l|M_G)$, and each $\hat{p}(l|M_G)$ is computed according to Equation 1.

For instance, assume the query to be the edge $\langle$ A.Einstein, education, PhD$\rangle$ as in Figure 1 (middle). We will first retrieve as a pseudo-relevance set all other edges (or a top-k subset) with the same label (those are, by construction isomorphic graphs). Then we rank the edges around the original query that were not contained in the query, e.g., awards won or advisor, and we will do so by exploiting the frequency of such labels in the pseudo-relevance set. In particular, if we rank them based on their score in Equation 3 we will favour expansions that appear frequently in the pseudo-relevant set.

As we observed earlier (Equation 2), one could use alternatively the intuition behind the KL-divergence scoring model [46]. In practice, the scoring assigns a high probability to expansions that are common in $\mathcal{G}_{rel}$, but not so common within the rest of the graph. Within such a model, a candidate expansion label l has a score proportional to the following:

$$\hat{p}(l|M_{rel})_{KL} \propto exp\left(\frac{1}{(1-\lambda)}\sum_{G}^{\mathcal{G}_{rel}} \frac{\log(\hat{p}(l|M_G))}{|\mathcal{G}_{rel}|} - \frac{\lambda}{(1-\lambda)}\log(\hat{p}(l|\mathcal{K}))\right) \tag{4}$$

Surprise-based Heuristic We complete our study including a scoring technique for expansions based on the concept of *surprise* [33]. This heuristic is still implemented within the pseudo-relevance feedback framework.

This method adopts the same intuition seen earlier for the case of the KL-divergence scoring model, i.e., the expansion terms that obtain a higher score are those with a relative frequency higher in the result-set than in the rest of the dataset. Therefore, for documents, given a set of terms $T=\{t_1,t_2,\ldots,t_n\}$, $\hat{p}(t_i)$ is the probability of term t_i to appear in one document, and $\hat{p}(t_1,t_2,\ldots,t_n)$ is the probability of

all terms to occur together in one document. The surprise is then measured by the ratio $\hat{p}(t_1,t_2,...,t_n)/\hat{p}(t_1)\cdot\hat{p}(t_2)\cdots\hat{p}(t_n)$

Therefore, given the query $Q=\{t_1,t_2,...,t_n\}$, we can score an expansion term t' by considering the increment in the surprise score obtained by $Surprise(T)-Surprise(T\cup\{t'\})$. Note that, opposed to the earlier models, this score counts the number of documents in which a term appears, but is not affected if in some documents the term is more or less frequent. The final score is:

$$Surprise(l)\propto\hat{p}(l|M_{rel})/\hat{p}(l|\mathcal{K}). \quad (5)$$

5 EXPERIMENTAL EVALUATION

Here we show that suggesting labels based only on their absolute frequency in the graph does not provide any useful information. The same is true when we favour expansions based only on node popularity or proximity: they result more frequently in unhelpful suggestions. In contrast, *our method based on KL-divergence with PRF inclusion provides relevant suggestions even when the query contains one edge.* Therefore, the KL-divergence with PRF is the most appropriate score for graph-query suggestion and it is able to effectively support users unfamiliar with the graph schema.

5.1 Experimental settings

Dataset: We validate our approach on Freebase, one of the largest available knowledge graphs. We use two snapshots. First the full dataset (after removing unnecessary metadata, e.g., Freebase users) with 76M nodes, 314M edges, and ~4.5K distinct edge labels. The other is a smaller snapshot (used in KG contextualization [43]), which focuses on People, Films, Music, Awards, Government, Business, Organizations, and Education (~500 edge labels).

Implementation: We implement our four methods [23]: **MLE**: The maximum likelihood estimation (Eq. 1); **KL**: The KL-divergence method (Eq. 2); **MLE-rel**: The maximum likelihood estimation with PRF (Eq. 3); **KL-rel**: The KL-divergence method with PRF (Eq. 4). Additionally, *we implement three baseline methods*: **PPR**: entity recommendation based on random walks [40]; **Srp**: Surprise heuristic described in Equation 5; **Rnd**: uniformly random suggestion. With Personalized Page Rank [3], in our case, edge suggestions are those directed towards nodes with the highest PPR score.

Queries: We tested two different sets of queries. For the first set (QALD-7), we obtained graph queries by translating questions and answers from the QALD-7 dataset [42]. We manually selected 65 queries covering diverse topics and with a clear exploratory intent, having potentially multiple answers and an explicit mapping into Freebase nodes and edges[1]. The choice of Freebase is motivated by previous studies [18, 27, 43] and allow us to evaluate our methods on one of the largest available knowledge graphs. For instance, the first query in the QALD-7 dataset is "doctoral supervisor, Albert Einstein", this can be translated into the single exemplar entity A. Einstein, or in the exemplar edge ⟨A. Einstein, advisor, A. Kleiner⟩ (see Figure 1). We then assigned to each graph-query a descriptive, yet unbiased, sentence to describe the exploratory information need. For instance, in the previous example, the assigned exploratory need was "*Academic information about Albert Einstein*".

The second set (KG contextualization) is a larger set of 325 facts designed to test fact contextualization methods [43]. While this set has more queries, it is designed for a different task, where the query is limited to a single edge and answers are expected to help the user understand the context of such fact. Moreover, all the queries in this set are people-centric (i.e., facts that involve a person).

Running time: We remark that the goal of this work is to study ranking models for graph query suggestion. There are only two time-consuming tasks in our algorithm: retrieving neighbour edges, to compute candidate expansions, and exemplar-query answering for PRF ranking. Both enjoy very efficient implementations [18, 27] that run in *less than 1 sec.* on average, while the time required to compute the score given the computed results is of few milliseconds.

5.2 User assessment

For the QALD-7 query set, we computed the top-20 edge expansion suggestions for each query and each method, i.e., for each graph query we obtained a list of 20 distinct edges that could be added to it. We obtained 7 expansion lists each one containing 20 edges one for each suggestion method. Each expansion has been translated into natural language[1] in order to allow the raters to judge its relevance to the task. Through crowd-sourcing, for each query and candidate expansion, we collected human judgments assessing their relevance on a four-point scale: irrelevant (0), uninteresting (1), fairly interesting (2), really interesting (3). In total, we obtained more than 25 thousands judgments, with *at least* three judgments for each query-suggestion pair. In particular, the average stddev among ratings for each question-answer pair is 0.413, median stddev of 0.471, and 90th percentile of 0.942. The query set for KG contextualization provides judgments with a similar format also obtained through crowdsourcing. In their case, facts are scored based not only on the edge-types but also on the target entity involved. We aggregated their relevance scores to obtain relevant edge-labels coherent with our application.

When analyzing the proportion of ratings we obtained from the human assessors, we found on average only five really interesting, or fairly interesting suggestions per query, among all methods. This demonstrates that *usually the user intent can be described by a very small number of edges.* This finding is consistent with the other query set, where less than 10% of the answers are judged relevant [43]. That is, finding the right edge to expand the user query, resembles the proverbial needle in the haystack. For both query datasets we can report a moderate agreement among raters.

5.3 Ranking quality

We compare first the ranked suggestions for the QUALD-7 dataset in terms of Normalized Discounted Cumulative Gain (NDCG) from top-1 to top-20, for the case of queries composed by one single entity (Figure 2), one edge (Figure 3), and two or more edges (Figure 4). We report the average NDCG score for each query and each method using the averages of the human judgments.

When the query is a single entity (Figure 2), no approach performs better than guessing. Except for PPR, all other methods perform similarly and the difference among methods does not significantly deviate (with p-value <0.05) from random (Rnd). The low performance of PPR is statistically significant (p-value <0.05) compared to the other methods. As expected, knowing just the entity of interest is too little information to predict the user interest. In such cases, an effective strategy could maximize suggestion-list

[1] http://people.cs.aau.dk/~matteo/files/exp-queries.zip

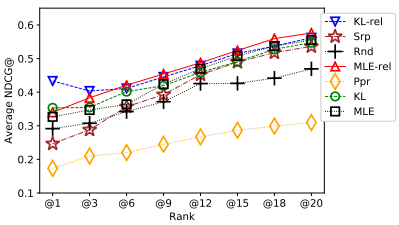

Figure 2: NDCG score at top-1 to top-20 when the starting query contains 1 Entity

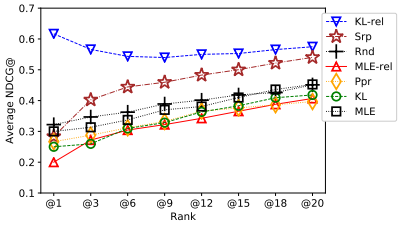

Figure 3: NDCG score at top-1 to top-20 when the starting query contains 1 Edge (2 Entities)

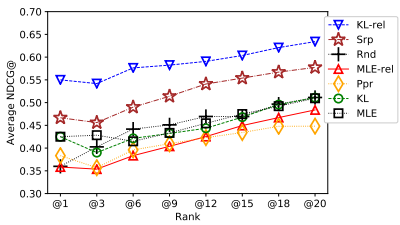

Figure 4: NDCG score at top-1 to top-20 when the starting query contains 2 or more Edges (3+ Entities)

diversity, in order to cover many different aspects and help the user disambiguate their intention.

The outcome is different for queries composed by one edge (Figure 3). We see that the KL-rel function outperforms the other methods with an average NDCG score around 5.5 (with p-value <0.05, w.r.t Rnd, PPR, KL, and MLE). The Surprise (Srp) heuristic provides good results as more suggestions are included in the top-k, and the difference with KL-rel is not statistically significant after the top-12. *PRF paired with a strategy that favours unexpectedly frequent relationships, pays-off in terms of quality, pushing relevant suggestions first.* On the other hand, we notice that favouring really frequent edge labels is detrimental to the quality of the suggestions. This is confirmed by the queries with 2 or more edges (Figure 4), where the KL-rel outperforms the competitors (p-value <0.05).

The results in Table 2 on the KG contextualization query set [43] confirm the above findings. The NDCG scores indicate (with p-value <0.05) that the KL-rel scoring function provides consistently better suggestions than the other methods. We note that our technique presents structure-query suggestions, where relevance is scored against relationship types, whereas the focus of the KG contextualization task is on specific fact instances. Moreover, NFCM [43] is a supervised method, while our approach is fully unsupervised. For this reason we abstain from a direct comparison of the two methods. Instead, in a different experiment, we investigate whether our scores – combined – can form simple and effective features on a supervised learning-to-rank method for the NFCM task. In this experiment, we use all the scores (Srp, MLE, KL, MLE-rel, KL-rel, and PPR) as features. Each query-expansion pair is encoded as a query-document pair to train a state-of-the-art learning-to-rank model [29] using the labels in NFCM [43]. We report that in this different settings our approach obtains comparable results to NFCM (e.g., NDCG@5 of 0.4513 vs. 0.5110 declared; NDCG@10 of 0.4954 vs. 0.5289 declared) and generally superior to all the baselines against which NFCM was tested [43]. This result indicates the applicability of our measures on related tasks, such as contextualization.

NDCG	**@3**	**@5**	**@10**	**@12**	**@15**	**@18**	**@20**
EM-rel	0.2961	0.2625	0.4185	0.4185	0.4185	0.4185	0.4457
Srp	0.2346	0.2617	0.3470	0.3470	0.3764	0.4333	0.4333
KL-rel	**0.3743**	**0.4928**	**0.5819**	**0.5819**	**0.6112**	**0.6394**	**0.6394**

Table 2: Avg. NDCG for the KG Contextualization data [43].

Comparison with different query-sizes: We study the effect of increasing the query size on the ability to predict relevant expansions (we mark as relevant suggestions with an average score larger than 2 on the scale [0–3]). We studied the mean average precision (MAP) at top-1,3,5,10 for all the methods, comparing different query sizes (not reported for space constraints). We recorded both better precision provided by the KL-methods (when presented with at least one edge in the query), but also the fact that, as the user provides additional information, the score is able to better capture the user intention and provide more relevant information. Note that as the size of the query increases, the number of possible expansions also increase: this makes the task harder, yet precision is not hurt, proving that the additional information is effectively exploited.

5.4 Measuring user effort

We estimate the user effort spared on obtaining the desired graph query. We assume each query's target graph is obtained by selecting the edges with the highest user rating. The initial query is the fact mentioned by keyword queries in the QALD-7 dataset (e.g., for "*Academic information on A. Einstein*" we consider ⟨A. Einstein, advisor, A. Kleiner⟩). The overall effort compares the number of suggestions required to retrieve all the edges from the initial query to the total number of possible edge types that a user would be required to inspect without our system (i.e., the number of edge-types for each entity). For instance, for the query above, the final query graph would contain facts about the advisor, the PhD degree, and the university of affiliation. Considering that the entity A. Einstein has 41 edge-types around it and presenting to the user just one instance of each type, the user would be required to inspect all of them to identify the desired edges. With our best scoring (KL-rel), the first relevant fact (⟨A. Einstein, education, PhD⟩) was within the top-5 suggestions, while the information about the department and the university were among the top-6 in the subsequent set of suggestions. This allows the user to limit the inspection to just 11 edge types, instead of a total of 50. We repeated the process for all queries. We estimate that, with our system, the user can inspect 60% fewer edge-types, significantly reducing their effort in formulating their queries. Therefore, our system makes the difference between a task that users will most probably never complete (with traditional tools), and a task they can easily carry out (using our suggestions).

6 CONCLUSIONS

We consider the novel task of graph query expansion in the context of exemplar search. We introduce query expansion methods for knowledge graphs to help users navigate queries and answers at the same time. Our experimental evaluation on real users demonstrates the expressiveness and the effectiveness of our methods in assisting the users in exploratory search. Our score based on KL-divergence outperforms strong baselines based on frequent edge labels and Personalized PageRank, providing useful graph-query suggestions.

REFERENCES

[1] Sumit Bhatia and Harit Vishwakarma. 2018. Know Thy Neighbors, and More!: Studying the Role of Context in Entity Recommendation *(HT '18)*. 87–95.

[2] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. 2011. Query reformulation mining: models, patterns, and applications. *IR* 14, 3 (2011), 257–289.

[3] Ilaria Bordino, Yelena Mejova, and Mounia Lalmas. 2013. Penguins in Sweaters, or Serendipitous Entity Search on User-generated Content *(CIKM '13)*. 109–118.

[4] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. 2008. Selecting Good Expansion Terms for Pseudo-relevance Feedback. In *SIGIR*. 243–250. https://doi.org/10.1145/1390334.1390377

[5] Claudio Carpineto and Giovanni Romano. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *ACM Comput. Surv.* 44, 1, Article 1 (Jan. 2012), 50 pages.

[6] Jiefeng Cheng, Xianggang Zeng, and Jeffrey Xu Yu. 2013. Top-k graph pattern matching over large graphs. In *ICDE*. IEEE, 1033–1044.

[7] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity Query Feature Expansion Using Knowledge Base Links *(SIGIR '14)*. ACM, 365–374. https://doi.org/10.1145/2600428.2609628

[8] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A Web-scale Approach to Probabilistic Knowledge Fusion *(KDD '14)*. 601–610. https://doi.org/10.1145/2623330.2623623

[9] Ahmed El-Roby, Khaled Ammar, Ashraf Aboulnaga, and Jimmy Lin. 2016. Sapphire: Querying RDF data made simple. *PVLDB* 9, 13 (2016), 1481–1484.

[10] Sébastien Ferré. 2017. Sparklis: an expressive query builder for SPARQL endpoints with guidance in natural language. *Semantic Web* 8, 3 (2017), 405–418.

[11] Jianfeng Gao, Gu Xu, and Jinxi Xu. 2013. Query Expansion Using Path-constrained Random Walks. In *SIGIR*. 563–572.

[12] D. Harman. 1988. Towards Interactive Query Expansion. In *SIGIR*. 321–331.

[13] Philipp Heim, Jürgen Ziegler, and Steffen Lohmann. 2008. gFacet: A Browser for the Web of Data. In *Proceedings of the International Workshop on Interacting with Multimedia Content in the Social Semantic Web (IMC-SSW'08)*, Vol. 417. Citeseer, 49–58.

[14] Guangjun Huang, Shuili Wang, and Xiaoguo Zhang. 2011. Query Expansion based on Associated Semantic Space. *Journal of Computers* 6, 2 (2011).

[15] Kai Huang, Sourav S Bhowmick, Shuigeng Zhou, and Byron Choi. 2017. PICASSO: exploratory search of connected subgraph substructures in graph databases. *PVLDB* 10, 12 (2017), 1861–1864.

[16] Zhipeng Huang, Bogdan Cautis, Reynold Cheng, and Yudian Zheng. 2016. Kb-enabled query recommendation for long-tail queries *(CIKM '16)*. ACM, 2107–2112.

[17] Nandish Jayaram, Sidharth Goyal, and Chengkai Li. 2015. VIIQ: auto-suggestion enabled visual interface for interactive graph query formulation *(VLDB)*, Vol. 8. 1940–1943.

[18] Nandish Jayaram, Arijit Khan, Chengkai Li, Xifeng Yan, and Ramez Elmasri. 2015. Querying knowledge graphs by example entity tuples. *TKDE* 27, 10 (2015), 2797–2811.

[19] Vitaly Klyuev and Yannis Haralambous. 2011. Query expansion: Term selection using the ewc semantic relatedness measure. In *FedCSIS*. 195–199.

[20] John Lafferty and Chengxiang Zhai. 2017. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. *SIGIR Forum* 51, 2 (Aug. 2017), 251–259.

[21] Victor Lavrenko and W. Bruce Croft. 2001. Relevance Based Language Models. In *SIGIR*. 120–127.

[22] Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. 2018. *Data Exploration Using Example-Based Methods*. Synthesis Lectures on Data Management, Vol. 10. Morgan & Claypool Publishers. 1–164 pages.

[23] Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. 2018. X2Q: Your Personal Example-based Graph Explorer *(VLDB '18)*. 901–904.

[24] Matteo Lissandrini, Davide Mottin, Dimitra Papadimitriou, Themis Palpanas, and Yannis Velegrakis. 2014. Unleashing the Power of Information Graphs. *SIGMOD Rec.* (2014), 21–26.

[25] Matteo Lissandrini, Davide Mottin, Yannis Velegrakis, and Themis Palpanas. 2018. Multi-Example Search in Rich Information Graphs. In *ICDE*.

[26] Steffen Metzger, Ralf Schenkel, and Marcin Sydow. 2013. QBEES: query by entity examples. In *CIKM*. 1829–1832. https://doi.org/10.1145/2505515.2507873

[27] Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. 2016. Exemplar queries: a new way of searching. *VLDB J.* (2016), 1–25. https://doi.org/10.1007/s00778-016-0429-2

[28] Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. 2019. Exploring the Data Wilderness Through Examples. In *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*. ACM, New York, NY, USA, 2031–2035. https://doi.org/10.1145/3299869.3314031

[29] Rama Kumar Pasumarthi, Sebastian Bruch, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork, Jan Pfeifer, Nadav Golbandi, Rohan Anil, and Stephan Wolf. 2019. TF-Ranking: Scalable TensorFlow Library for Learning-to-Rank. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[30] Robert Pienta, Fred Hohman, Alex Endert, Acar Tamersoy, Kevin Roundy, Chris Gates, Shamkant Navathe, and Duen Horng Chau. 2018. VIGOR: interactive visual exploration of graph query results. *IEEE transactions on visualization and computer graphics* 24, 1 (2018), 215–225.

[31] Jay M. Ponte and W. Bruce Croft. 1998. A Language Modeling Approach to Information Retrieval. In *SIGIR*. ACM, 275–281.

[32] Ian Ruthven. 2003. Re-examining the Potential Effectiveness of Interactive Query Expansion. In *SIGIR*. ACM, 213–220.

[33] Nikos Sarkas, Nilesh Bansal, Gautam Das, and Nick Koudas. 2009. Measure-driven keyword-query expansion. *PVLDB* 2, 1 (2009), 121–132.

[34] Bahareh Sarrafzadeh and Edward Lank. 2017. Improving Exploratory Search Experience Through Hierarchical Knowledge Graphs *(SIGIR '17)*. 145–154.

[35] Bahareh Sarrafzadeh, Olga Vechtomova, and Vlado Jokic. 2014. Exploring Knowledge Graphs for Exploratory Search. In *Proceedings of the 5th Information Interaction in Context Symposium (IIiX '14)*. ACM, 135–144. https://doi.org/10.1145/2637002.2637019

[36] Grzegorz Sobczak, Mateusz Chochół, Ralf Schenkel, and Marcin Sydow. 2015. iQbees: Towards Interactive Semantic Entity Search Based on Maximal Aspects. In *Foundations of Intelligent Systems*. Springer, 259–264.

[37] Yu Su, Shengqi Yang, Huan Sun, Mudhakar Srivatsa, Sue Kase, Michelle Vanni, and Xifeng Yan. [n.d.]. Exploiting Relevance Feedback in Knowledge Graph Search *(KDD '15)*. 1135–1144.

[38] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge *(WWW '07)*. 697–706.

[39] Bin Tan, Atulya Velivelli, Hui Fang, and ChengXiang Zhai. 2007. Term Feedback for Information Retrieval with Language Models. In *SIGIR*. ACM, 263–270.

[40] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast Random Walk with Restart and Its Applications. In *ICDM*. https://doi.org/10.1109/ICDM.2006.70

[41] Nam Khanh Tran, Tuan Tran, and Claudia Niederée. 2017. In *The Semantic Web*. 353–368.

[42] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, Anastasia Krithara, Michael Röder, and Giulio Napolitano. 2017. *7th Open Challenge on Question Answering over Linked Data (QALD-7)*. 59–69.

[43] Nikos Voskarides, Edgar Meij, Ridho Reinanda, Abhinav Khaitan, Miles Osborne, Giorgio Stefanoni, Prabhanjan Kambadur, and Maarten de Rijke. 2018. Weakly-supervised Contextualization of Knowledge Graph Facts. In *SIGIR*. 765–774. https://doi.org/10.1145/3209978.3210031

[44] Ryen W. White and Resa A. Roth. 2009. *Exploratory Search: Beyond the Query-Response Paradigm*. Morgan and Claypool Publishers. http://dx.doi.org/10.2200/S00174ED1V01Y200901ICR003

[45] Chengxiang Zhai and John Lafferty. [n.d.]. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval *(SIGIR '01)*. 334–342.

[46] Chengxiang Zhai and John Lafferty. 2001. Model-based Feedback in the Language Modeling Approach to Information Retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM '01)*. ACM, New York, NY, USA, 403–410. https://doi.org/10.1145/502585.502654

[47] Xiangling Zhang, Yueguo Chen, Jun Chen, Xiaoyong Du, Ke Wang, and Ji-Rong Wen. 2017. Entity Set Expansion via Knowledge Graphs *(SIGIR '17)*. 1101–1104.

[48] P. Zhao and J. Han. 2010. On graph query optimization in large networks. *VLDB J.* 3, 1-2 (2010), 340–351.