A Scalable Virtual Document-Based Keyword Search System for RDF Datasets

Dennis Dosso
Department of Information Engineering, University of Padua
Padua, Italy
dosso@dei.unipd.it

Gianmaria Silvello
Department of Information Engineering, University of Padua
Padua, Italy
silvello@dei.unipd.it

ABSTRACT
RDF datasets are becoming increasingly useful with the development of knowledge-based web applications. SPARQL is the official structured query language to search and access RDF datasets. Despite its effectiveness, the language is often difficult to use for non-experts because of its syntax and the necessity to know the underlying data structure of the database queries. In this regard, keyword search enables non-expert users to access the data contained in RDF datasets intuitively.

This work describes the TSA+VDP keyword search system for effective and efficient keyword search over large RDF datasets. The system is compared with other state-of-the-art methods on different datasets, both real-world and synthetic, using a new evaluation framework that is easily reproducible and sharable.

CCS CONCEPTS
• Information systems → Retrieval models and ranking: Information retrieval; Document representation; Evaluation of retrieval results.

KEYWORDS
RDF graphs, keyword search, evaluation

ACM Reference Format:

1 INTRODUCTION
In the last decade, the Web of Data has become one of the principal means to expose structured data on the Web [15]. The actual paradigm of the Web of Data is the Linked Data realized via Resource Description Framework (RDF) datasets (or RDF graphs). An RDF dataset is a set of triples (subject-predicate-object) each connected to the other to form a directed graph. RDF allows for flexible manipulation, enrichment, data discovery, and reuse of data across applications, enterprises, and community boundaries. Today, RDF datasets typically contain thousands of millions of edges [16].

The SPARQL structured query language is the standard means to retrieve and access data from RDF graphs. Its complexity — i.e., syntax and the need to know the underlying data schema — is, however, an obstacle to non-experts. In this regard, keyword search will enable users to easily access these data, overcoming SPARQL complexity.

Over the past two decades, the research community and data technology vendors have developed new approaches for keyword search over structured databases [2, 12, 19], but no prototype has led to a transition from proof-of-concept implementations into fully deployed systems [5]. The main limitations of state-of-the-art systems regard both effectiveness and efficiency. To target effectiveness, we need to develop methods for retrieving and ranking candidate answers, which are RDF subgraphs in our case. The structure and the semantic relevance to the query of the latter is the object of evaluation. To target efficiency, in relation to query execution time and memory consumption, systems that scale to real-world dataset sizes have to be designed. Scalability is "the most pressing challenge" that search systems over structured data have to face and this aspect is particularly striking for graph data [16] such as RDF.

The research community has dedicated great effort in searching relational databases, conducting extensive evaluation, and identifying benchmarks to evaluate system efficiency and effectiveness [5]. Regarding RDF, a common evaluation framework and shared benchmark to evaluate keyword search systems are still missing.

Markov Random Fields-Keyword Search (MRF-KS)[13] is one of the top systems that has been evaluated using the relational benchmark [4]. Although MRF-KS has been tested on relational databases, in principle it can be used for any kind of data graph. Based on the "virtual documents" approach [7, 17], the text documents are built starting from the content of nodes and edges of the graph and then indexed and retrieved by means of IR techniques. One of the few native RDF search systems, also based on the virtual documents approach, is the Structured Language Model (SLM) [6].

The virtual documents approach is promising from the scalability viewpoint, since it moves part of the computations off-line (e.g., documents creation) and exploits IR ranking techniques and data structures (e.g., inverted indexes) to speed up the search process. Even though IR techniques scale well to large datasets, they need to be adapted in dealing with the complexity of structured data.

Contributions of this paper include: (i) TSA+VDP, a virtual documents-based keyword search system that is both efficient and effective over different databases, which overcomes the limitations of other state-of-the-art systems; (ii) a new evaluation framework...
2 RELATED WORK

Keyword Search has been extensively studied in the context of structured databases such as Relational DB and Knowledge Bases. Good reviews about these topics are [2, 18, 19].

There are three main approaches to keyword search.

The first one is the schema-based (e.g. DISCOVER [1]), which uses the schema of the database to formulate SQL queries starting from the keyword query.

The second one is the graph-based (e.g. BANKS [3]), which models relational databases as graphs, where the tuples are nodes and the foreign-primary key relationship between them define the edges. While this approach is one of the most commonly used in literature (cfr. [4]), it specifically relies on the exploration of the graph structure. This prevents the system from scaling to very large databases.

The third approach is the document-based, that concatenates the text contained in the nodes and edges of the graph into textual documents and leverages on IR methods for ranking. This approach seems promising from the scalability perspective since online graph traversals, which require a lot of time and memory, can be avoided.

Two main systems that follows this approach are considered: SLM and MRF-KS. The first was designed to work for RDF graphs [6]. It is based on the construction of potential answer subgraphs containing query keywords and on a ranking function that uses an adapted language model. The second system, MRF-KS [13], was originally designed for relational databases. In order to work over RDF graphs, it was re-implemented. This system is based on answer trees and their ranking with an MRF function [14]. The system requires performing off-line pre-computations, and include exploring the graph via the Dijkstra algorithm.

3 TSA+VDP SEARCH SYSTEM

The TSA+VDP search system takes as input an RDF graph \( G \) and a user keyword query \( Q_k \). It creates a set of potential answer graphs \( G_j \) and the corresponding textual document \( D_{G_j} \), by combining the strings extracted from IRIs and literals contained in the graphs. A retrieval function on the graphs and documents is applied for ranking according to their relevance to \( Q_k \). The ranking is then returned to the user.

Figure 1 shows the main components of the system along with a simple example. There are two main phases: the off-line phase, performed before the user query is issued, and the on-line phase.

The off-line phase is performed by the Topological Syntactical Aggregator (TSA) algorithm. As shown in Figure 1, the first step of the algorithm is to build a set of subgraphs – the Representative Collection (RC) – from the RDF dataset. The idea is to create subgraphs via a series of BFS-like explorations of the dataset. TSA considers an IRI node and all the literals connected to it as a big meta-node. For every iteration, the exploration starts from a fresh meta-node (i.e. a source node) with an out-degree greater than a given threshold. Only edges labeled with frequent predicates are traversed. An edge is traversed only if it leads to a fresh source node. The algorithm stops when it encounters a visited node or when it reaches a given graph radius. Each subgraph is then associated with a Virtual Document created by connecting the words in the given...
subgraph nodes and edges. These documents are then indexed with standard IR methods.

The second step of the algorithm begins when the user submits the query. Firstly, a ranking is obtained (top 1,000 results) over all the virtual documents with the BM25 model (step 2 in Figure 1). In step 3 (note that the sub-graph triples containing query keywords are highlighted) the sub-graphs with overlapping triples are merged using a decision algorithm based on a learned threshold.

Afterwards, the Virtual Document Pruning (VDP) algorithm (step 4) is applied. First of all, VDP considers the collection of merged graphs and keeps those containing all the query keywords. These graphs are then merged in a not always connected and usually quite small graph, called query graph. VDP explores it via BFS and produces new sub-graphs. Finally, the sub-graphs that do not contain all of the query keywords are discarded and the remaining ones go through a pruning phase which eliminates the triples that do not contain at least a keyword. The final result is a collection of answer graphs that are further ordered by a Markov Random Field-based [14] ranking function.

4 EVALUATION FRAMEWORK

The efficacy evaluation is quite straightforward since it is enough to measure the (online and offline) time and memory used by the search systems (i.e. TSA+VDP, SLM and MRF-KS) to run the off-line and the on-line search phases.

To measure the effectiveness of the search system a critical difference between standard text retrieval and virtual database-based keyword search systems needs to be taken into account. In the former case, the collection of documents is fixed, while in the latter the answer graphs and the virtual documents of the collection are generated dynamically (sometimes on-the-fly) with different strategies and heuristics. Hence, the relevance of a document to a query cannot be verified in advance, as can be done for standard text retrieval.

While [5] defines a shared test collection to evaluate the effectiveness of search systems over relational data, there is no such shared test collection for RDF.

We define a new evaluation framework on the basis of the use of SPARQL queries and their counterpart keyword queries that is easily reproducible and shareable. In our context, the SPARQL query is associated to a given topic \( t_k \) in \( T \), returns the “perfect answer graph” – i.e. the ground truth graph \( G_{t_k} \). Thus, all the triples in the answer graph \( G_{t_k} \) are deemed as “relevant” to the topic \( t_k \) while all the triples outside \( G_{t_k} \) are “not-relevant”.

Definition 4.1. Let \( Q_k \) be a keyword query derived from the topic \( t_k \) in \( T \) and submitted to a keyword search system and let \( G_{t_k} \) be the ground-truth graph of \( t_k \). We define the ranking returned by the system as \( R_k = \{ap_1, ap_2, \ldots, ap_n\} \), where \( ap_i = (G_i, sim_i) \in R_k \), \( i \in [1, n] \) is called answer pair s.t. \( G_i \) is the RDF answer graph at rank \( i \) and \( sim_i \in \mathbb{R} \) is a degree indicating the similarity of \( G_i \) to \( G_{t_k} \). For every couple of answer pairs \( \{ap_i, ap_j\} \in R_k \) s.t. \( i < j \), \( sim_i \geq sim_j \) holds true.

A good keyword search system should return a ranking that is: (i) top-heavy: the answer graphs ranked at the top contain more relevant triples than noisy triple; and, (ii) essential: redundant triples in different graphs ins the ranking are irrelevant to the user.

Definition 4.2. Given a topic \( t_k \in T \), a ranking \( R_k \) and the ground-truth graph \( G_{t_k} \), the Signal-to-Noise Ratio (SNR) of \( G_i \in R_k \) and the Graph Relevance Weight (GRW) of \( G_i \in R_k \) as defined are as follows:

\[
SNR(G_i) = \frac{|(G_i \cap G_{t_k}) \setminus S|}{|G_i|} \quad \text{GRW}(G_i) = \frac{|(G_i \cap G_{t_k}) \setminus S|}{|G_{t_k}|}
\]

where \( S \) is the union of all the relevant triples in \( G_j \in R_k \), \( j \in [1, i] \).

The SNR recognizes precise and essential graphs, while the GRW is a weighting function which rewards the graphs that contain new relevant triples seen for the first time by the user.

Definition 4.3. Let \( t_k \in T \) be a topic, \( R_k \) a ranking with length \( n \in \mathbb{N}^+ \), \( \lambda \in \{0, 0.1, 0.2, \ldots, 1\} \) be a threshold value indicating the level of redundancy in the ranking \( 1 \) and \( b \in \mathbb{N}^+ \) a log base value indicating how often a keyword may appear in the ranking. We define the Relevance Gain (RG) of \( G_i \in R_k \) as

\[
RG_b(G_i) = \begin{cases} 
\frac{GRW(G_i)}{\log_b i} & \text{if } i < b \land SNR(G_i) > \lambda, \\
0 & \text{if } i \geq b \land SNR(G_i) > \lambda, \\
\lambda & \text{otherwise}.
\end{cases}
\]

The triple-based Discounted Cumulative Gain (tb-DCG) of the ranking \( R_k \) is defined as

\[
tb-DCG_b(R_k) = \sum_{i=1}^{n} RG_b(G_i)
\]

The tb-DCG is a variation of the DCG measure [11], widely-used in text retrieval.

5 EXPERIMENTAL RESULTS

Three RDF datasets have been considered: LinkedMDB [10] (7M triples), IMDB \( ^2 \) (116M triples) and the synthetic dataset LUBM (two versions of 1M and 10M triples) \([8]\). LinkedMDB and IMDB are too big for SLM (> 1000 sec online execution time for every query) and MRF-KS (> 48 hours offline execution time). Hence, to allow effectiveness comparison to TSA+VDP, two reduced versions – rLinkedMDB and rIMDB – of 1M triples each are created by randomly sampling connected components.

We created 50 queries for both LinkedMDB and IMDB; whereas, for LUBM, we used the 14 official queries \([3]\). The source code of the search systems, the scripts to obtain the test datasets and the code to run the experiments along with the topics and the keyword queries are made available in our git repository \( ^4 \).

The effectiveness and the efficiency of the considered keyword search systems were tested in terms of tb-DCG (\( \lambda = 0.1 \)), offline time to build the virtual documents, average online time and memory used by the queries for every dataset. The results are shown in Table 1. An ANOVA test among the results was performed to assess the statistical reliability of the systems’ performances.

In terms of offline time required to build and index the virtual documents, we can see that SLM has a negligible offline phase, whereas MRF-KS requires up to 2 days to process the rIMDB database. TSA+VDP reports the best trade-off between offline and online time and constantly keeps the offline time under 1h with the solely exception of LUBM10M. The offline execution time is the main bottleneck for MRF-KS. Considering the databases of 1M triples, it is

1. \( \lambda = 0 \) redundant triples allowed, \( \lambda = 1 \) no redundant triples allowed in the ranking.
Table 1: From left to right: dataset name; system name; offline time to build and index the virtual documents; average tb-DCG; average online time; average central memory. *indicates the systems in the top performing group with $p < 0.01$. The best system is in bold.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>System</th>
<th>offline (sec)</th>
<th>tb-DCG</th>
<th>online (sec)</th>
<th>memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>rLinkedMDB</td>
<td>MRF-KS</td>
<td>1.76</td>
<td>0.546 ± 0.157</td>
<td>111.22 ± 6.67</td>
<td>22.45 ± 2.68</td>
</tr>
<tr>
<td>rLinkedMDB</td>
<td>TSA-VDP</td>
<td>0.21</td>
<td>0.546 ± 0.131</td>
<td>30.69 ± 0.43</td>
<td>21.96 ± 0.48</td>
</tr>
<tr>
<td>rMBDB</td>
<td>MRF-KS</td>
<td>2.85</td>
<td>0.427 ± 0.06</td>
<td>440.68 ± 15.07</td>
<td>0.985 ± 0.44</td>
</tr>
<tr>
<td>rMBDB</td>
<td>TSA-VDP</td>
<td>1.13</td>
<td>0.541 ± 0.127</td>
<td>42.87 ± 5.91</td>
<td>19.88 ± 0.46</td>
</tr>
<tr>
<td>AMK</td>
<td>MRF-KS</td>
<td>0.97</td>
<td>0.287 ± 0.05</td>
<td>248.18 ± 15.37</td>
<td>33.51 ± 0.51</td>
</tr>
<tr>
<td>LUBM1M</td>
<td>SLM</td>
<td>1.0</td>
<td>0.640 ± 0.02</td>
<td>19.68 ± 1.93</td>
<td>5.468 ± 1.14</td>
</tr>
<tr>
<td>LUBM1M</td>
<td>MRF-KS</td>
<td>1.44</td>
<td>0.965 ± 0.157</td>
<td>286.80 ± 0.10</td>
<td>1.846 ± 0.184</td>
</tr>
<tr>
<td>LUBM</td>
<td>TSA-VDP</td>
<td>0.06</td>
<td>0.245 ± 0.346</td>
<td>23.46 ± 0.257</td>
<td>23.14 ± 0.69</td>
</tr>
<tr>
<td>LUBM1DM</td>
<td>TSA-VDP</td>
<td>0.13</td>
<td>0.208 ± 0.170</td>
<td>17.21 ± 1.166</td>
<td>105.18 ± 21.28</td>
</tr>
</tbody>
</table>

6 CONCLUSIONS

The TSA+VDP search system is presented herein for keyword search over graph data. The system exploits state-of-the-art IR methods and scales to real-world RDF dataset sizes.

A new evaluation framework for keyword search systems based on structured SPARQL query and corresponding keyword query pairs has been illustrated. This framework can be easily reproduced. It is also based on tb-DCG, a new user-oriented measure, which awards top-heavy and non-redundant rankings.

In the future, TSA+VDP can be tested on other datasets like DBpedia Infoblox to further assess the ability of TSA+VDP to scale. Different methods in the construction of the virtual documents and their impact on effectiveness will also be tested.

Acknowledgments This work is partially supported by the Computational Data Citation (CDC-STARs) project of the University of Padua and by the ExaMode project, as part of the European Union Horizon 2020 program under Grant Agreement no. 825292.

REFERENCES


2019, July 21–25, 2019, Paris, France D. Dosso and G. Silvello