Keyword Search on RDF Datasets

Dennis Dosso[1][0000−0001−7307−4607]

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

Abstract. In the last years, the Resource Description Framework (RDF) has gained popularity as the *de-facto* representation format for heterogeneous structured data on the Web. RDF datasets are interrogated via the SPARQL language, which is often not intuitive for a user since it requires the knowledge of the syntax, the underlying structure of the dataset and the IRIs. On the other hand, today users are accustomed to Web-based search facilities that propose simple keyword-based interfaces to interrogate data. Hence, in order to ease the access to the data to users, we aim to develop of an effective and efficient system for keyword search over RDF graphs. Furthermore, we propose a methodology to properly evaluate these systems. Finally, we aim to address the problem of the explainability of the information contained in the answers to non-expert users.

Keywords: RDF Graphs · Keyword Search · Explainability

1 Motivation

Recently, the continuous growth of knowledge-sharing communities like Wikipedia and the advances in automated information-extraction from Web pages have made it possible to build large-scale knowledge bases [6]. In the meantime the Web of Data emerged as one of the principal means to expose structured data [12]. These datasets are usually built using RDF, a family of W3C specification, where data are represented in the form of a directed graph. RDF is the *de-facto* standard for publishing, accessing and sharing data, because it allows for flexible manipulation, enrichment, discovery, and reuse of data across applications, enterprises, and community boundaries. RDF is used by applications in industry, biology and human science such as Eagle-i [13], Europeana [11], Dbpedia [1], Disgenet [10] and many others, proving its importance and validity [8].

The standard way to interrogate RDF is the SPARQL query language, but it can become very difficult to write complex queries in this language, even for expert users, since it requires the knowledge of the language and of the underlying structure of the dataset.

Our research task is to enable the non-expert user to interrogate real-world RDF databases in a more intuitive and easy way. The keyword search paradigm can become a tool to enable the general user to gain access to these datasets.

Amongst the difficulties regarding keyword search systems, as described in [4] and [5], there are the long execution times (more than one hour on average)

and the memory required by most of the systems in the literature. These systems often cannot complete their execution even on small databases (1M triples) and thus cannot scale to real-world sizes. Our aim is to develop a keyword search system to enable non-expert users to perform keyword query on real-world RDF datasets. The output of such a system is a list of answer graphs ranked in decreasing order of relevance with respect to the information need.

The system needs to be effective and efficient: (i) to be effective it needs to create answer graphs that correctly cover the information need of the user, ranked from the most relevant to the least relevant; and (ii) to be efficient, it needs to operate in a time in the order of seconds or minutes even on real-world datasets of tens of millions of triples.

Keyword Search has been thoroughly studied in the contest of structured databases such as relational DB and Knowledge Bases. There are good reviews about this topic, such as [2], [15] with a particular focus on RDB, and [14] with a focus on graph data.

The rest of the paper is organized as follows: Section 2 describes the proposed research and our methodology; Section 3 provides an outlook on future challenges.

2 Description of the Proposed Research and Methodology

The description of our research stems from four principal research topics related to keyword search.

Efficiency The answer time of a system is crucial since users are accustomed to the rapid response of web search engines. A keyword search system needs to be efficient in order to be perceived as useful by the general user. This means that it needs to complete its execution in a reasonable time and with an efficient use of memory. [5] in particular showed that many approaches in the literature cannot complete their task due to time or memory issues.

In our system in particular we execute computations off-line (before the user submits her query) in order to create data structures that will help fasten the execution of the on-line phase. In particular, we create off-line a collection of subgraphs from the dataset. This collection is called "representative collection" since our aim is to build the subgraphs in such a way that they cover one single topic each. Such a collection can be efficiently indexed, searched and used to build the final answers to the query. We were able to achieve high on-line efficiency (in the order of minutes) on a dataset of more than 100M of triples. Our datasets are two orders of magnitude bigger than the ones that can be found in works like [5], [7] and [9].

Effectiveness The output of a keyword search system is a ranked list of answer subgraphs. Effectiveness can be measured on two aspects: the ability to create *accurate* answer graphs, and the ability to rank them accordingly to their *relevance* to the user query. With "accurate" answer graph we mean a graph that contains many relevant triples, i.e. triples that contain useful information

for the user. Moreover, it should not contain noisy triples, i.e. triples that are not interesting for the user. Since we are returning a ranking, once the user has seen a relevant triple in one answer graph, the same triple presented in graphs down below in the ranking is considered redundant, and therefore no more useful. This means that the system should also be able to produce answer graphs *different enough* one from the other in order to completely cover the information need without too much redundancy.

In our system, we rely on heuristics to build and then prune graphs. We use the representative collection to look for subgraphs containing all the query words. Then, we create answer subgraphs that contain all the query words and we prune them to remove the noisy triples. We use an adapted version for RDF of the ranking function described in [9] to order the answer graphs using both their topological structure and text content.

In our experiments, we took the IMDB database[1] and parsed it in order to produce an RDF dataset of circa 116M triples. We also took a subset of IMDB of 1M triples, called rIMDB, in order to confront our algorithms to the others that cannot scale. As benchmarks, we implemented the keyword search system SLM described in [7] and adapted the keyword system MRF-KS for RDB in [9] in order for it to work on RDF. Our approach is called TSA+VDP (Topologic Syntactic Aggregator + Virtual Document Pruning).

Evaluation The Cranfield paradigm is a well-established method to evaluate IR systems [4]. However, in the context of keyword search over structured data (in particular relationa DBs), its implementation reported some limits as highlighted in [3]. Our aim is to develop a general evaluation framework for keyword search systems on RDF that follows the Cranfield paradigm and that can face the lacks shown so far in the literature.

In this regard, we developed an evaluation framework based on couples made of a keyword query and its counter-part SPARQL version. The SPARQL version produces the "exact" answer graph, which is used as ground truth. The answer graphs produced by the system are compared to this GT graph in order to understand if they are relevant to the user. Then, their position in the ranking is considered in order to understand the quality of the ranking itself. We designed two different functions: a Signal-To-Noise Ratio function (SNR) to decide if an answer graph is relevant to the user query; and a triple-based Discounted Cumulative Gain (tb-DCG) function to evaluate the quality of a ranking. The higher the tb-DCG, the better.

Table 1 reports the average performances obtained by the two benchmarks SLM and MRF-KS, compared to our TSA+VDP in terms of effectiveness via the tb-DCG and in terms of efficiency via time and memory usage. We used 50 different topics created by hand. This is not the only algorithm we developed, but it is the most promising one in terms of trade-off between efficiency and effectiveness. As can be seen, in the rIMDB dataset our algorithms obtain the best performances in terms of effectiveness and efficiency.

[1] https://datasets.imdbws.com/

Table 1. Performances in rIMDB, 1M triples. † indicates the systems in the top performing group with $p < 0.01$. The best system is in bold.

Systems	tb-DCG	time (sec)	memory (MB)
SLM	0.0030±0.00	**34.5000**± 1.59†	3.2733±0.06
MRF-KS	0.4217±0.06	440.6800±189.77	**0.9858**±0.44
TSA+VDP	**0.5449**±0.12†	35.8710± 9.53†	19.8860±3.66

Only our algorithm could be deployed on the whole IMDB database, obtaining a tb-DCG of 0.4490 with an average time of 200 seconds. TSA+VDP is the only system to complete its execution on a dataset of such a dimension. None of the algorithms of the state of the art could be deployed in such instances.

Explainability Another crucial aspect is the ability of the system to *explain* the answers provided to the user. In other words, since we are assuming that the user is not an expert, it is reasonable that she does not know SPARQL and, possibly, does not know RDF. An output composed of a list of RDF triples can be useless to her, since she will be unable to read it. More useful would be a description of the answers in a more user-friendly format. For example, a text snippet in natural language that describes the content of the answer graphs.

3 Research Issues

In this section, we list some open issues and research directions.

1. We aim to develop a new automatic technique to build explanatory text document to be attached to the answer graphs in order to better explain their meaning to the non-expert user. We will investigate how data provenance and data citation can be deployed and integrated in order to solve this problem.
2. We want to investigate how the creation of fielded documents can affect the ranking function and the possibility to explain them to the user. We will use the text content of the IRIs and the structure of the dataset to better leverage on the presence of specific words and their frequency.
3. Often an RDF dataset does not present human-readable IRIs. These IRIs are made of numbers and acronyms. Thus, many keywords used in the queries are not matched in triples that are relevant to the information need. A possibility in this regard is to implement a *query expansion* technique, with the support of a dictionary, created ad-hoc for every dataset, that will enhance the ability of the keyword query to match with words in the dataset. This will help the algorithms to produce answer graphs and rank them in the best possible way.
4. There are many other databases of big dimensions and wide informative scope (like DBpedia) to be tested. Our aim is to test our system in these other databases and adapt it to the different peculiarities.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference. pp. 722–735. ISWC'07/ASWC'07, Springer-Verlag (2007)
2. Bast, H., Buchhold, B., Haussmann, H.: Semantic search on text and knowledge bases. Foundations and Trends in Information Retrieval **10**(2-3), 119–271 (2016)
3. Bergamaschi, S., Ferro, N., Guerra, F., Silvello, G.: Keyword-Based Search Over Databases: A Roadmap for a Reference Architecture Paired with an Evaluation Framework. Trans. Computational Collective Intelligence **21**, 1–20 (2016)
4. Coffman, J., Weaver, A.C.: A framework for evaluating database keyword search strategies. In: Proc. of the 19th ACM International Conference on Information and knowledge management. pp. 729–738. ACM Press (2010)
5. Coffman, J., Weaver, A.C.: An Empirical Performance Evaluation of Relational Keyword Search Systems. IEEE Transactions on Knowledge and Data Engineering **26**(1) (2014)
6. Doan, A., Ramakrishnan, R., Vaithyanathan, S.: Managing information extraction: state of the art and research directions. In: Proceedings of the 2006 ACM SIGMOD international conference on Management of data. pp. 799–800. ACM, ACM (2006)
7. Elbassuoni, S., Blanco, R.: Keyword Search over RDF Graphs. In: Proc. of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011,. pp. 237–242. ACM Press, New York, USA (2011)
8. Feigenbaum, L., Herman, I., Hongsermeier, T., Neumann, E., Stephens, S.: The semantic web in action. Scientific American **297**(6), 90–97 (2007)
9. Mass, Y., Sagiv, Y.: Virtual Documents and Answer Priors in Keyword Search over Data Graphs. In: Proc. of the Workshops of the EDBT/ICDT 2016 Joint Conference. CEUR Workshop Proceedings, vol. 1558. CEUR-WS.org (2016)
10. Paschke, A., Burger, A., Romano, P., Marshall, M.S., Splendiani., A. (eds.): Proceedings of the 6th International Workshop on Semantic Web Applications and Tools for Life Sciences, Edinburgh, UK, December 10, 2013, CEUR Workshop Proceedings, vol. 1114. CEUR-WS.org (2014)
11. Petras, V., Hill, T., Stiller, J., Gäde, M.: Europeana - a search engine for digitised cultural heritage material. Datenbank-Spektrum **17**(1), 41–46 (2017)
12. Pound, J., Mika, P., Zaragoza, H.: Ad-hoc object retrieval in the web of data. In: Proc. of the 19th International Conference on World Wide Web, WWW 2010. pp. 771–780. ACM Press, New York, USA (2010)
13. Torniai, C., Bourges-Waldegg, D., Hoffmann, S.: eagle-i: Biomedical research resource datasets. Semantic Web **6**(2), 139–146 (2015)
14. Wang, H., Aggarwal, C.C.: A survey of algorithms for keyword search on graph data. In: Managing and Mining Graph Data, pp. 249–273. Springer (2010)
15. Yu, J.X., Qin, L., Chang, L.: Keyword Search in Relational Databases: A Survey. IEEE Data Eng. Bull. **33**(1), 67–78 (2010)