iSummary: Demonstrating Workload-based, Personalized Summaries for Knowledge Graphs

Giannis Vassiliou[1], Nikolaos Papadakis[1] and Haridimos Kondylakis[2,*]

[1] *School of Electrical & computer engineering - HMU, Greece*
[2] *Institute of Computer Science - FORTH, Greece*

Abstract
The rapid growth in both size and complexity of Knowledge Graphs (KGs), available on the web, has created a pressing demand for effective and efficient techniques to comprehend and explore them. Recently, semantic summaries have emerged as a promising solution to quickly grasp the contents of such KGs. However, these summaries often suffer from limitations, being static and lacking customization to accommodate user selections. Additionally, they struggle to handle the sheer scale of the Knowledge Graphs. In this demonstration, we present *iSummary*, a novel and scalable system able to construct personalized summaries. The main idea behind our approach is to exploit knowledge captured in existing user queries for identifying the most interesting resources and linking them, constructing as such high-quality, personalized summaries. Our algorithm provides theoretical guarantees on the summary's quality, with a computational complexity linearly dependent on the number of queries in the query log.

1. Introduction

Daily, a tremendous amount of new information becomes available online. RDF Knowledge graphs (KGs) rapidly grow to include millions or even billions of triples that are offered through the web. The complexity and the size of those data sources limit their exploitation potential and necessitate effective and efficient ways to explore and understand their content. In this direction, semantic summarization [1] has been proposed as a way to extract useful, minimized information out of large semantic graphs. Structural semantic summaries focus mostly on the structure of the graph, whereas non-quotient structural semantic summaries select the most important parts of the graph for generating the result summaries.

The problem. Most of the existing works in the area of structural, non-quotient semantic summarization, produce generic static summaries [2] and cannot be applied to big KGsḞurther, as different persons have different data exploration needs the generated summaries should be tailored specifically to the individual's interests. Although this has already been recognized by the research community, the approaches offering personalized summaries so far, rely on node weights selected by the users, then followed by algorithms making various vague assumptions about the relevant subsets out of the semantic graph that should complement the initial user choice [3, 4]. More recent approaches like [5, 6] exploit the individual user queries for mining

ISWC 2023 Posters and Demos: 22nd International Semantic Web Conference, November 6–10, 2023, Athens, Greece

*Corresponding author.

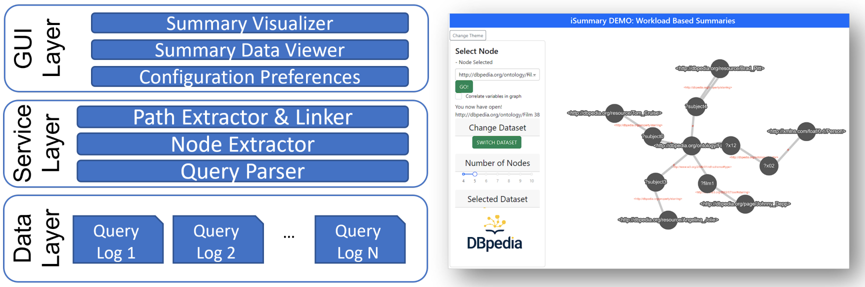
giannisvas@ics.forth.gr (G. Vassiliou); npapadak@cs.hmu.gr (N. Papadakis); kondylak@ics.forth.gr (H. Kondylakis)

user preferences but still rely on the KG to compute the summary which makes it computationally hard. Further, capturing a complete individual user query set is usually not feasible.

The iSummary solution. In this demonstration we present iSummary, which exploits generic logs already available through the SPARQL endpoints of the various KGs available online instead of relying on node weights or on individually provided sets of user queries. In order to generate a personalized summary iSummary only requires the selection of one or a few nodes that the user is mostly interested in. As previous users have already identified through their queries, the most common connections to the specific user-selected nodes, we exploit this information in order to formulate the generated summaries. In our demonstration, we explain how we can resolve the problem relying on existing query logs and we provide a solution to both the multiple weight assignment required and also to the computationally hard problem. The main paper of this work has already been presented at ESWC 2023 [7], whereas the system is available online[1]. To the best of our knowledge, it is the first approach to construct personalized, structural, non-quotient semantic summaries exploiting generic query workloads.

2. iSummary System Overview

We depict the high-level architecture of the iSummary system in Figure 1 (right). The system comprises three main components. The GUI enables setting the configuration parameters (the size of the summary, selection of the nodes, and the dataset), visualizes the results, and presents the triples in the result summary. The service layer includes the query parser that processes the query logs, the node extractor in order to select the nodes to be included in the summary, and the path extractor & linker which connects the selected nodes. Finally, the data layer includes the various query logs per dataset.


GUI Layer
Summary Visualizer
Summary Data Viewer
Configuration Preferences
Service Layer
Path Extractor & Linker
Node Extractor
Query Parser
Data Layer
Query Log 1
Query Log 2
...
Query Log N
iSummary DEMO: Workload Based Summaries
Change Theme
Select Node
- Node Selected
http://dbpedia.org/ontology/Fil...
GO!
Correlate variables in graph
Change Dataset
SWITCH DATASET
Number of Nodes
4
5
6
7
8
9
10
Selected Dataset
DBpedia
<http://dbpedia.org/resource/Brad_Pitt>
?subject4
<http://dbpedia.org/resource/Tom_Cruise>
?subject0
<http://dbpedia.org/ontology/Fi
?x12
<http://xmlns.com/foaf/0.1/Person>
?x02
?subject3
?film1
<http://dbpedia.org/page/Johnny_Depp>
<http://dbpedia.org/resource/Angelina_Jolie>

You now have open!
http://dbpedia.org/ontology/Film 38

Figure 1: High-level architecture of the iSummary system (left) and a screenshot of the web GUI (right)

Next, we present the problem of computing λ/κ-Personalized Summaries and we show that, although they are useful, their computation is both impractical and computationally hard.

Definition 2.1 **(λ/κ-Personalized Summary).** *Given (1) a knowledge graph $G = (V, E)$, (2) a non-negative weight assignment to all nodes, capturing user preferences in G, (3) λ seed nodes, (4)*

[1] http://gianos.pythonanywhere.com/

and a number κ ($\lambda \leq \kappa$), find the smallest maximum-weight tree $G' = (V', E') \in G$ including the κ most preferred nodes.

Note that we don't actually require a weight to be assigned to all nodes, as the weight of all nodes can be by default zero, and the user only adds weights to a subset of them. Also, a solution to the λ/κ-Personalized Summary problem is not unique, as there might be many maximum-weight trees with the smallest size that are equally useful for the user. A nice property of the λ/κ-Personalized Summaries is that their quality is *monotonically increasing* as the κ increases. This means that as the summary size increases more relevant information is added to the summary for the same seed nodes selected by the user.

Lemma 2.1. *Let S_κ be a λ/κ-Personalized Summary and $S_{\kappa+1}$ be a $\lambda/(\kappa+1)$-Personalized Summary for G. Then $W(S_{\kappa+1}) \geq W(S_\kappa)$, where $W(S)$ the sum of all node weights in S.*

Computing the λ/κ-Personalized Summary is both *impractical*, as different weights should be assigned to the graph nodes for each distinct user query, and computationally *expensive*, as it is equivalent to the Steiner-Tree problem which is NP-complete. Next, we present how iSummary provides an elegant approximate solution based on query workloads assuming that λ=1, i.e. to the κ-Personalized Summary problem, which can be easily generalizable for λ>1 as well.

Resolving the problem of multiple weight assignments. Assume now that for the KG G we have available a query log $Q = \{q_1, \cdots, q_n\}$ available. This assumption is reasonable, as all big KGs offer a SPARQL endpoint that logs user queries for various purposes. Having such a query log available, we can *use it to mine user preferences* for the specific seed node that the user is interested in. The idea here is that if a user is interested in a κ-Personalized Summary for a node s then we can use Q to identify *relevant queries* to s, i.e., queries that include s. In those queries, other nodes relevant to the user input will be available. In fact, as those queries have been issued by thousands of users, we assume that *the most useful related nodes will be the ones that appear more frequently* there.

Based on this assumption we can have multiple weight assignments, one per user input, as they occur from thousands of user queries that involve the provided user input and that are based on past users' preferences, as expressed in their queries. Note here that we don't need weights for the whole graph, as by default we can set the weight of the nodes that do not appear in the filtered user queries to zero.

Resolving the computational problem. Now that we have a way to assign personalized weights to the nodes, we next provide a computationally efficient procedure in order to link the selected nodes over a big graph. We start with a solution including a single node, the s selected by the user, adding one node each time of the ones with the maximum weight till all remaining $k - 1$ nodes are included in the summary. However, for doing so we will not use the original data graph but again *relevant user queries.* The main idea here is the following: link s with the $k - 1$ maximum weight nodes using *the most frequent shortest paths* from the user queries. The following theorem can be proved.

Theorem 2.2. *The iSummary algorithm finds an approximate solution to the κ-Personalized Summary problem with a worst-case bound of 2, i.e., $W/W_{opt} \leq 2 \times (1 - l/k)$, where W and W_{opt}*

denote the total weight of a feasible solution and an optimal solution respectively, and l *a constant, and scales linearly to the number of queries in the workload.*

Example. Consider as an example, the KG shown in Figure 2, on the university domain.

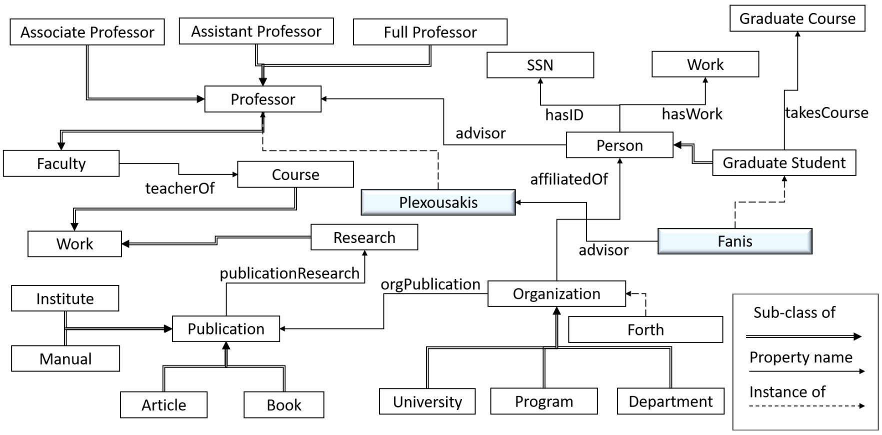


Figure 2: Example RDF KG.

Assume that for our example KG, we have available a query log consisting of the following SPARQL queries:

```
Q1. SELECT ?x WHERE {x? a Person. y? a Professor. ?x advisor ?y.}
Q2. SELECT ?x WHERE {x? a Person. y? a Organization. ?y affiliatedOf ?x.}
Q3. SELECT ?x WHERE {x? a Person. y? a Organization. ?y affilatedOf ?x.
    ?y orgName "FORTH".}
```

Now assume that a user is interested in a 2-Personalized Summary for the node *Person*. Based on the query log we can identify that relevant queries to user input are $Q1$, $Q2$, and $Q3$. Examining those queries we can identify that the useful nodes are the *Professor* and *Organization*. In fact, as *Organization* is used in two queries it should be most useful according to the available query log. As we are looking for a 2-Personalized Summary it will be included in the result. Now instead of searching the graph shown in Figure 2 for linking *Person* with *Organization* we will additionally filter the queries including *Person* and *Organization*, i.e. Q2 and Q3. For each one of those queries, we calculate the shortest path for linking *Person* and *Organization* and we eventually select the most frequent shortest path to include in the summary. As such the 2-Personalized Summary for the node *Person* includes a single triple (*Organization*, *addiliatedOf*, *Person*).

3. Demo Overview

To demonstrate the functionalities of iSummary, we will use DBpedia and Wikidata along with their corresponding query logs (58K queries for DBpedia and 192K queries for WikiData). The demonstration will proceed in six phases:

1. Introduction. In the first step we will explain the problem of computing a λ/κ-Personalized Summary motivating its usefulness. Further, we will explain why it is computationally hard and also inconvenient by requiring users to provide multiple weights. Then we will explain the main ideas behind our approximate solution, exploiting query logs for effectively tackling both hardness of the problem and also multiple weights assignment.

2. Configuration. Then we will start the summarization process by selecting the KG to be summarized along with the corresponding query logs. In the configuration menu, the user can select the node for which she requires a summary and also the number of nodes to be included in the summary (i.e. the κ).

3. Summary Graph. In this phase the result summary will be presented visually. Here we will explain how our algorithm mines the queries in order to identify the nodes that most frequently co-occur with the user input and then how again the queries are exploited to link the selected nodes by picking the smallest most frequent path that appears in the query log. The triples included in the summary will also be shown.

4. Summary for different sizes. We will explain the properties of a good summary explaining that it is better when it is able to maximize the fragments of queries that include user-selected nodes (i.e. to maximize the query coverage). Then we will demonstrate how coverage monotonically increases as the size of the summary increases as well.

5. Mini-Game. In this phase we will let conference participants ”play” with the system. We will play a mini-game asking them to return a summary for a few nodes and contrasting them with the result from iSummary.

Acknowledgments

This research project was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the “2nd Call for H.F.R.I. Research Projects to support Post-Doctoral Researchers” (iQARuS Project No 1147) and by the SafePolymed (GA 101057639) EU project.

References

[1] S. Cebiric, F. Goasdoué, H. Kondylakis, D. Kotzinos, I. Manolescu, G. Troullinou, M. Zneika, Summarizing semantic graphs: a survey, VLDB J. 28 (2019) 295–327.

[2] G. E. Trouli, A. Pappas, G. Troullinou, L. Koumakis, N. Papadakis, H. Kondylakis, Summer: Structural summarization for RDF/S kgs, Algorithms 16 (2023) 18.

[3] A. Alzogbi, G. Lausen, Similar structures inside rdf-graphs, in: LDOW, 2013.

[4] G. Wu, J. Li, L. Feng, K. Wang, Identifying potentially important concepts and relations in an ontology, in: International Semantic Web Conference, Springer, 2008, pp. 33–49.

[5] T. Safavi, C. Belth, L. Faber, D. Mottin, E. Müller, D. Koutra, Personalized knowledge graph summarization: From the cloud to your pocket, in: ICDM, 2019, pp. 528–537.

[6] G. Vassiliou, G. Troullinou, N. Papadakis, H. Kondylakis, Wbsum: Workload-based summaries for RDF/S kbs, in: SSDBM, 2021, pp. 248–252.

[7] G. Vassiliou, F. Alevizakis, N. Papadakis, H. Kondylakis, isummary: Workload-based, personalized summaries for knowledge graphs, in: ESWC, 2023, pp. 192–208.