RDF Graph Summarization Based on Node Characteristic and Centrality

Jimao Guo and Yi Wang*

College of Computer and Information Science, Southwest University, Chongqing 400715, China
E-mail: guojimao105@foxmail.com; echowang@swu.edu.cn
**Corresponding Author*

Received 26 January 2022; Accepted 17 November 2022;
Publication 06 December 2022

Abstract

The explosive growth of RDF data makes it difficult to be efficiently queried, understood and used. RDF graph (RDFG) summarization aims to extract the most relevant and crucial data as summaries according to different criteria. Current summarization approaches mainly apply single strategies such as graph structure, pattern mining or relevance metrics to calculate RDFG summaries. Different to the existing approaches, this paper proposes a summarization approach to automatically generating RDFG summary, which can capture both structure and centrality information. Specifically, we present three algorithms, SumW (merging nodes based on node characteristics or similar types), SumS (merging nodes based on typed node characteristics) and SummaryFL (retrieving central nodes by combining node frequency and bridging coefficient). The three algorithms can be used by two summarization strategies: SumS or SumW only, and SumS+SummaryFL or SumW+SummaryFL. We conducted experiments over large and real-world

Journal of Web Engineering, Vol. 21_7, 2073–2094.
doi: 10.13052/jwe1540-9589.2174

RDF datasets to verify the effectiveness of our method with respect to time complexity, compression capability and coverage of the summary. The experiment results demonstrate that our approach outperformed the comparative algorithms.

Keywords: Knowledge graph summarization, node centrality, knowledge graph compression, node characteristic set, graph summarization.

1 Introduction

RDF Graph (RDFG) has been widely used in knowledge modeling and data reuse in various areas. Nowadays, the volumes of RDFGs continue growing explosively, which causes great difficulties in RDFG exploration, querying and error-detecting. For example, RDF datasets in geography, biology, vocabulary statistics, linguistics and sociology contain more than 20 billion triples and 3 billion nodes in the LinkedGeoData[1] dataset alone. Therefore, efficiently summarizing large-scale RDFGs becomes one of the challenging problems in the field of RDFG [1].

The problem of RDFG summarization has received a great deal of attention in recent years. The goal of RDFG summarization is to fast compress RDFGs in a meaningful way as much as possible. Since an RDFG is a directed labeled graph formed by RDF data, a number of approaches studied summarizing RDFGs from the aspect of graph structure by merging "similar" nodes and edges as supernodes and superedges [2–4]. For example, equivalence relations between nodes can be discovered and the quotient graphs formed by equivalence relations are used as RDFG summaries. How to quickly discover "similar" nodes in a large RDFG as candidates for merging is a challenging task. Because not only the structure formed by nodes and edges needs to be considered but also the semantics described by the types of nodes and edges is also crucial for a meaningful summary.

Some researchers studied using relevance and centrality metrics to rank entities or paths in RDFGs in order to select the most relevant or important nodes and paths as summaries [5–8]. Centrality or relevance metrics for RDFGs are defined and top nodes and related edges can be used as RDFG summaries. These methods are generally efficient because they are based on the statistics of centrality or relevance metrics. However, the drawback of these methods is that they only extract "important" nodes and edges as

[1] http://linkedgeodata.org/

summaries but neglect the structure and semantic information of the original RDFGs.

In this paper, we aim to efficiently generate RDFG summaries which can not only preserve the semantics and the structural information of the original datasets, but also include "important" nodes in terms of centralities into the summary. Different to the current methods which used single strategies, e.g., structure-based or statistic-based methods, to compute summaries and can only retain certain types of feature of the original RDFGs, our approach uses a hybrid summarization strategy. Specifically, our contributions include:

First, we present an RDFG summarization method based on the node Characteristic Set (CS) and the centrality metric. Our summarization method can compress the nodes and edges by their linking patterns and types. In addition, central nodes can be further extracted into the summary. Specifically, we present three algorithms, **SumW** (merging nodes based on node CS or similar type relations), **SumS** (merging nodes based on typed CS relation) and **SummaryFL** (retrieving central nodes by combining node frequency and bridging coefficient). The three algorithms can be used by two summarization strategies: SumS or SumW only, and SumS+SummaryFL or SumW+SummaryFL.

Second, we conducted experiments over four real-world large RDFG datasets: AGROVOC, DBpedia, LinkedGeoData, and Wikidata to verify the effectiveness of our method with respect to time complexity, compression capability and coverage of the summary. The results proved that our approach outperformed the current summarization methods.

We organize the paper as follows. Section 2 discusses the related work. Section 3 presents our summarization approach including the algorithms SumS, SumW and SummaryFL. Section 4 is the experiments and Section 5 concludes our work.

2 Related Work

As stated in Section 1, the goal of RDFG summarization is to fast compress RDFGs in a meaningful way as much as possible. We identify two mainstream methods for RDFG summarization: *merging-based summarization* and *extraction-based summarization*.

Merging-based summarization RDFGs are directed graphs in nature. The typical summarization methods for graphs by merging nodes/edges as supernodes/superedges [9–12] provide valuable experience for RDFG summarization. For example, the summarization method for undirected and

unlabeled graphs proposed by Ko et al. [11] merged neighbor nodes as supernodes by selecting candidate nodes with the probability related to their degrees. However, graph summarization methods cannot be directly applied to compress RDFGs because they are designed for compressing unlabeled graphs, where nodes and edges contain no meaning but only the connection relations. Differently, the nodes and edges in an RDFG have types, i.e., nodes and edges in an RDFG have different meanings. Moreover, RDFGs contain both the schema information (ontology) and the fact data. The same linking pattern in a common graph may represent different meaning and pattern in an RDFG [13, 16, 17].

Thus, merging-based RDFG summarization needs to consider the types of nodes and edges during the summarization. Stefanoni et al. [4] proposed a typed summary, called SumRDF for RDFG. A resource d was described by 4-tuple: ($C(d)$, $O(d)$, $I(d)$, $P(d)$), where $C(d)$ was the type of d, $O(d)$ and $I(d)$ were the vectors representing the types of outgoing and incoming entities, and $P(d)$ was the partition for the resource d. The basic idea was to put same-typed resources into a partition. Some methods defined equivalence relations called, bisimulation, between graph nodes [3, 14, 15]. For example, Čebirić et al. [15] identified equivalence relations of nodes in RDFGs, and the quotient graphs consisting of sets of equivalent nodes (merged as supernodes) were defined as the summaries.

Merging-based summarization has the advantage of capturing the structure of graphs and thus helps users better understand the original RDFGs. However, due to the massive volume of current RDFGs which usually contain hundreds of millions of triples, the overhead for calculating summaries is expensive. A major problem with current merging-based summarization methods is that the scalability of algorithms needs to be improved for very large RDFGs.

Extraction-based summarization Extraction-based summarization methods usually extract important and relevant nodes/edges as summaries by defining metrics to rank nodes/edges in RDFGs [18, 20]. Some of the methods focused on extracting both important nodes and edges as summaries [5–7, 9, 19], while others focused on extracting the most important nodes (also called entity summarization) [21–24].

Pires et al. [5] studied the summarization in the context of a Peer Data Management System (PDMS), where each peer was an autonomous source defined by an ontology. The authors proposed using the concepts of centrality and frequency to select the most relevant resources as the summaries for ontologies. Similarly, Troullinou et al. [6] also proposed a summary method,

called RDF Digest, to extract the most relevant paths in ontologies. In [7], user preference was considered when creating ontology summaries. Safavi et al. [8] also studied personalized summaries of RDFGs. The personalized summarization was defined as a set of triples that maximized a user's "utility" over a given RDFG. Presutti et al. [19] extracted the key knowledge pattern paths based on the metrics of type betweenness and property betweenness to represent the knowledge of datasets.

As the objective of entity summarization is to retrieve the most relevant nodes in RDFGs, metrics related to nodes were proposed to rank nodes. Gunaratna et al [21] presented an approach called faceted entity summaries for selecting a small subset of the original triples associated with an entity as a summary for quick access of entity-related information. Thalhammer et al. [22] proposed a relevance-oriented summarization of entities. It was a combination of PageRank algorithm with the Backlink method. In [23], the authors selected the set of most representative entities in an RDFG. The metric for ranking entities was based on structural and textual features and the summarization process was modeled as an optimization problem. Yang et al. [24] focused on discovering outstanding facts from knowledge graphs for target entities under the context specified by context entities.

Extraction-based summarization methods are usually efficient because they only rely on the statistics of centrality or relevance metrics. However, the drawback of these methods is that they only extract "important" nodes/edges as summaries but neglect the structural and semantic information of the original RDFGs.

To improve the performance and quality of summaries for large-scale RDFGs, we propose a hybrid and efficient summarization method that consists of two-phase summarization: merging nodes by the node CS and extracting nodes by the centrality. The resulting summaries can capture both the structural and central information of the original RDFGs.

3 Proposed RDFG Summarization Approach

3.1 Overall Approach

Figure 1 gives an overview of our approach for RDFG summarization. Firstly, an RDFG is summarized based on the node *Characteristic Set* (CS) by the SumW or SumS algorithms. This step generates the summary containing the structural information of the original RDFG. Then, the resulting summary generated by SumW or SumS is further processed by the SummaryFL algorithm, which extracts central nodes from the RDFG and adds them to the

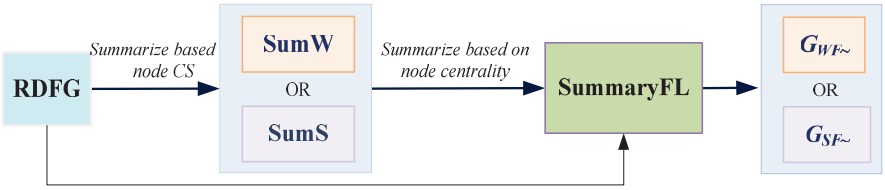

Figure 1 Overall approach.

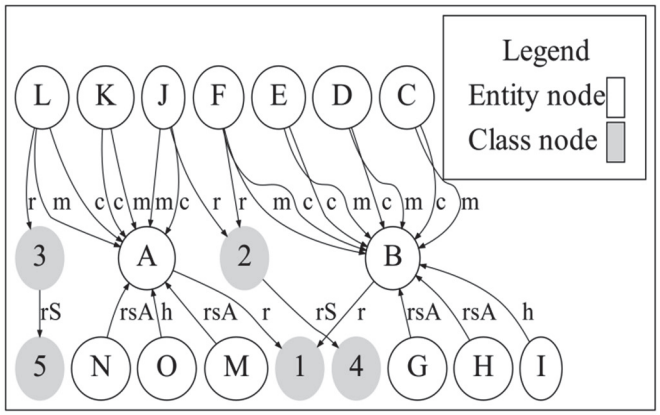

Figure 2 DBLP RDFG.

final summary. Thus, the final summary retains both the structural and also the central information of graph nodes.

The following example provides more details of our approach.

Example Figure 2 shows part of the datasets of DBLP.[2] It can be found that although the nodes A and B link to different numbers of edges, they have the same *set of properties* (called CS). Moreover, they both belong to the same type. Our intuition is that nodes with the same CS and *similar types* are candidates for merging during the creation of summary. We say similar types because it is common that two instance nodes have multiple types and these types may be not completely same but they are similar to some extent.

Figure 3(a) demonstrates the summarization process for the proposed algorithms **SumS**+**SummaryFL**, which merges nodes with the same CS and similar types first and then extracts central nodes as the summary. Firstly, nodes A and B are merged as a supernode N1 and N2 is created to represent

[2]https://dblp.org/xml/

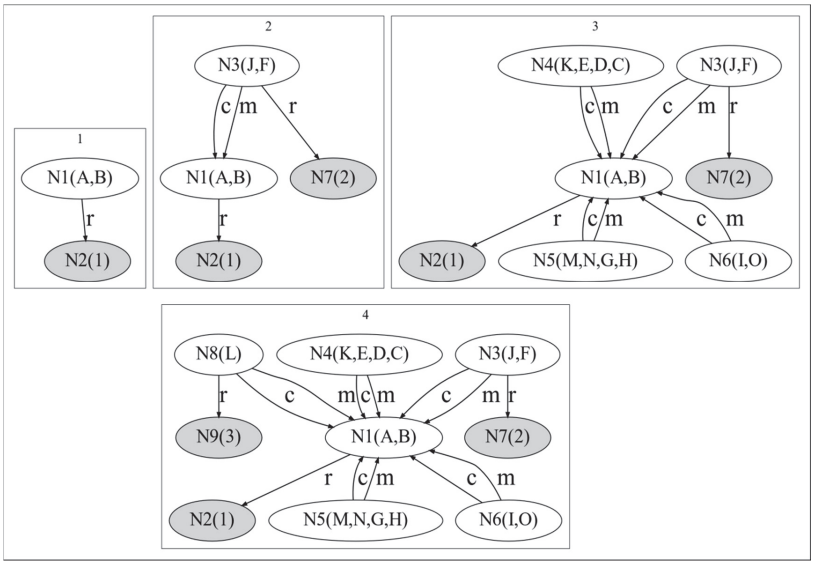

Figure 3(a) Summarization process based on typed CS and centrality (SumS+SummaryFL).

its type. The edges between nodes are also created between supernodes, as shown in step 1. Then, **SumS** traverses its neighbors from nodes A and B, and finds that J and F belong to the same type and also have the same CS. Thus N3 is created as well as N7 (the type of J and F) as shown in step 2. Similarly, supernodes N4, N5 and N6 are created at step 3. The last step applies **SummaryFL** to find central nodes in the original RDFG by the combination of the bridging coefficient and frequency. The central nodes N8 and N9 are then added to the summary at step 4.

Figure 3(b) demonstrates the summarization process of the proposed algorithm **SumW+ SummaryFL**, which merges nodes with same CS or similar types and then extracts central nodes as the summary.

3.2 RDFG Summary Based on Node Characteristics

In this section, we present the RDFG summary method based the CS of nodes and the SumS and the SumW algorithms for creating RDFG summaries.

3.2.1 The summary methods based on CS

An RDFG can be defined as $G = (V, E, P, L)$, where V is the set of nodes including classes, instances, blank nodes and literals, E is the set of relations

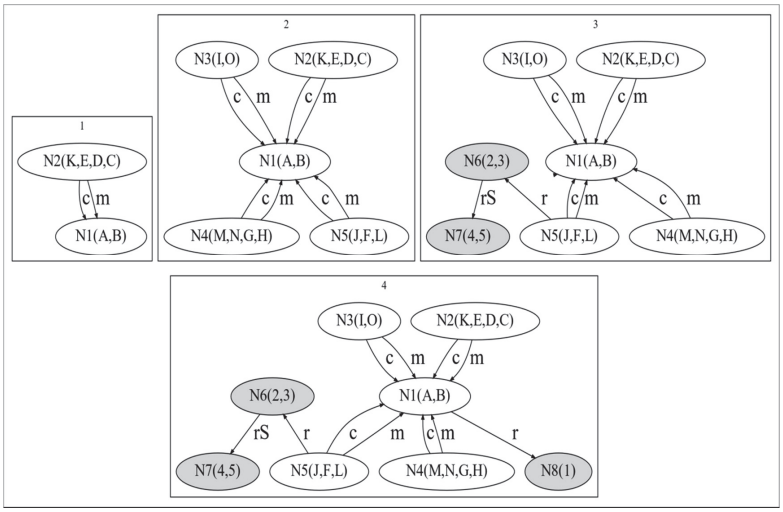

Figure 3(b) Summarization process based on CS and centrality (SumW+ SummaryFL). ***Notation***: Step 1: **SumW** creates supernodes N1 and N2 for (A, B) and (K, E, D, C) which both have the same CSs. Step 2: N3, N4, N5 for (I, O), (M, N, G, H) and (J, F, L) are created. Step 3: N6 and N7 are created according to the property edges linked by the class nodes (2 and 3) and (4 and 5). Step 4: N8 is created.

between nodes, P is the set of properties, and L: $E \rightarrow P$ is a labeling function that maps each relation to its type.

Definition 1 (*Characteristic set*, *CS*) Given an instance node v of an RDFG G, the CS of v, denoting as $P(v)$ is defined as the set of all properties linked with v except rdf:type. Furthermore, $P(v) = OP(v) \cup IP(v)$, where $OP(v)$ is the out-CS including all the outgoing properties of v and $IP(v)$ is the in-CS including all the incoming properties of v.

Definition 2 (*Same-CS relation*) Given two different nodes u and v: (1) u and v have the same-out-CS relation denoting as $(u, v) \in R_{OP}$ if $OP(u) = OP(v)$; (2) u and v have the same-in-CS relation denoting as $(u, v) \in R_{IP}$ if $IP(u) = IP(v)$; and (3) u and v have the same-CS relation denoting as $(u, v) \in R_P$ if $P(u) = P(v)$.

Definition 2 defines the same-CS relation between nodes. For example, in Figure 2, since the nodes C and D both have the outgoing properties creator and maker, they have the same-out-CS relation.

An instance may belong to multiple classes or types. Given two instances, we need to judge how similar two instances are in terms of types. Zheng

et al. [25] defined the semantic distance between two types in an ontology. The intuition is that two types are similar to each other if they have more common super-types. Similarly, we define the type similarity between two instance nodes in Definition 3.

Definition 3 (*Type similarity*) Given two nodes *u* and *v* in *G*, *T*(*u*) and *T*(*v*) are the sets of types related to *u* and *v*. Let $T(u) \cup T(v) = \{c_1, c_2, \ldots, c_m\}$, $\boldsymbol{T}(u)$ and $\boldsymbol{T}(v)$ are the vectors in P^m representing *T*(*u*) and *T*(*v*) respectively, where the *i*th element is 0 (denoting the nonexistence of c_i) or 1 (denoting the existence of c_i) ($i = 1, \ldots, m$). The type similarity for *u* and *v* *SimTp*(*u*, *v*) is defined as:

$$SimTp(u, v) = \frac{\boldsymbol{T}(u) \cdot \boldsymbol{T}(v)}{\|\boldsymbol{T}(u)\|^2 + \|\boldsymbol{T}(v)\|^2 - \boldsymbol{T}(u) \cdot \boldsymbol{T}(v)} \tag{1}$$

Definition 3 defines the metric *SimTp*(*u*, *v*) based on the Tanimoto coefficient [26]. For example, assume $T(u) = \{c_1, c_2, c_3\}$ and $T(v) = \{c_1, c_2, c_5\}$, then $\boldsymbol{T}(u) = <1, 1, 1, 0, 0>$ and $\boldsymbol{T}(v) = <1, 1, 0, 0, 1>$. By Equation (1), the type similarity between *u* and *v* is: $SimTp(u, v) = 2/(3 + 3 - 2) = 0.5$.

Now we can define the same-type relation between instance nodes.

Definition 4 (*Same-Type relation*, $\tilde{C}$) Assume *u* and *v* are two instance nodes in an RDFG, then *u* and *v* have the same-type relation denoted as: $(u, v) \in \tilde{C}$ if: (i) $T(u) \subseteq T(v)$ or $T(v) \subseteq T(u)$ and (ii) $SimTp(u, v) \geq \gamma_s$, where γ_s is the similarity threshold.

Suppose $\gamma_s = 0.5$. If $T(u) = \{c_1, c_2, c_3\}$ and $T(v) = \{c_1, c_2, c_5\}$, we know that $SimTp(u, v) = 0.5$. However, they do not have the same-type relation because *T*(*u*) and *T*(*v*) do not subsume each other. If $T(u') = \{c_1, c_2\}$ and $T(v') = \{c_1, c_2, c_5\}$, then $T(u') \subseteq T(v')$ and $SimTp(u', v') \approx 0.67 > \gamma_s$. We regard that *u*′ and *v*′ have the same-type relation, i.e., $(u', v') \in \tilde{C}$.

Definition 5 (*Typed Same-CS relation*, ***S***) Given two nodes *u* and *v* in an RDFG, *u* and *v* have the typed same-CS relation denoting as $(u, v) \in \boldsymbol{S}$ if $(u, v) \in \tilde{C} \cap R_P$.

Definition 5 defines a **strong** relation between instance nodes, i.e., having same types and same CSs. For example, it can be found that A and B in Figure 2 have both the same-CS and same-type relations, and thus they have the **typed same-CS relation**, i.e., ***S*** relation. Similarly, we use ***W*** to denote *u* and *v* having the same-type or same-CS relation, which is a weaker relation compared to ***S***. For example, F and L link with the same properties, but belong to different types, and thus they have ***W*** relation. Obviously if two nodes have

the ***S*** relation, they must have the ***W*** relation, but the vice versa does not hold. Definition 6 defines the summaries of RDFGs based on ***S*** and ***W*** relations.

Definition 6 (*RDFG Summary based on* ***S*** (***W***) *relation*) Given $G = (V, E, P, L)$, the summary of the RDFG based on the ***S*** (***W***) relation is $G_{S\sim} = (N_S, E_S)$ ($G_{W\sim} = (N_W, E_W)$), where N_S (N_W) is the set of supernodes and E_S (E_W) is the set of superedges. $\forall\ N_i \in N_S$ (N_W), all the nodes in N_i have the ***S***(***W***) relation. $\forall\ E_k = (N_1, N_2) \in E_S$ (E_W), E_k includes all the property edges between the nodes in N_1 and N_2 of the original graph G.

3.2.2 The summary algorithms based on CS

The SumS Algorithm. The **Algorithm 1 SumS** calculates the summary of an RDFG based on the ***S*** relation, i.e., typed same-CS relation. The algorithm accepts G and γ_s (the type similarity threshold) as the input. In steps 2-3, the algorithm randomly selects an instance node u and creates the supernode N_u containing u. Then the procedure FindST retrieves nodes V_u with similar types as u satistfying γ_s. The procedure ScreateSN further determines if the nodes in V_u having the ***S*** relation with u and adds the confirmed nodes into N_u. Step 6 creates the superedges according to the property edges linked by the neighbors of nodes in N_u. Step 7 update N_S with N_u and step 8 removes the nodes in N_u from the candidate nodes. The total time complexity for SumS is $O(\Delta \cdot |V| + |E|)$, where FindST and ScreateSN takes $O(\Delta \cdot |V|)$ and ScreateSE takes $O(E)$ (Δ is the max degree of G).

Algorithm 1 SumS

Input: $G = (V, E, P, L)$, γ_s
Output: $G_{S\sim} = (N_S, E_S)$
1: *V_Candidate* $\leftarrow V$, $N_S \leftarrow \varnothing$, $E_S \leftarrow \varnothing$
2: **for** u in *V_Candidate* **do**
3: $N_u \leftarrow \{u\}$ // N_u *contains* all the nodes having ***S*** relation with u
4: $V_u \leftarrow$ FindST(u, $T(u)$, γ_s) //find instance nodes with similar types as u
5: $N_u \leftarrow$ ScreateSN(u, V_u) // create supernode for nodes by ***S*** relation with u
6: *Es* $\leftarrow$ ScreateSE (N_S, N_u) //create superedges for N_u
7: $N_S \leftarrow N_S \cup N_u$
8: *V_Candidate* $\leftarrow$ *V_Candidate*$\backslash N_u$ //remove nodes in N_u from *V_Candidate*
9: **end for**
return N_S, E_S

The SumW Algorithm. The **Algorithm** 2 **SumW** calculates the summary of an RDFG based on the ***W*** relation, i.e., same-CS or same-type relation.

SumW accepts G and γ_s as the input and generate the summary $G_{W\sim}$ as output. Steps 2–4 find the CSs for each node by the hash function. This task takes $O(E)$ time complexity. In steps 5–6, the algorithm randomly selects an instance node u and creates the supernode N_u containing u. Steps 8–12 merge nodes having same CS with u; otherwise, merge nodes having same types with u. Finally supernode N_u is added to N_W and superedges are created. The total time complexity for SumW is also $O(\Delta \cdot |V| + |E|)$.

Algorithm 2 SumW

Input: $G = (V, E, P, L), \gamma_s$
Output: $G_{W\sim} = (N_W, E_W)$
1: $V\_Candidate \leftarrow V, N_W \leftarrow \varnothing, E_W \leftarrow \varnothing, mapP \leftarrow \varnothing$
2: **for** v in V **do**
3: $h_F \leftarrow h(P(v))$ // $P(v)$ is the set of node features for v and h is a hash function
4: $mapP[h_F] \leftarrow mapP[h_F] \cup \{v\}$ // $mapP$ stores the characteristics of nodes
5: **end for**
6: **for** u in $V\_Candidate$ **do**
7: $N_u \leftarrow \{u\}$ // N_u *contains* all the nodes having $\boldsymbol{W}$ relation with u
8: $V_p = mapP(h(P(u))$
9: **if** $V_p \neq \varnothing$ **then**
10: $N_u \leftarrow$ ScreateSN(u, V_p) //create supernode having same-CS with u
11: **else**
12: $V_T \leftarrow$ FindST(u, $T(u)$, γ_s) // find instance nodes with similar types as u
13: $N_u \leftarrow$ ScreateSN(u, V_T) // create supernode having similar types with u
14: **end if**
15: $E_W \leftarrow$ ScreateSE (N_W, N_u) // create superedges for N_u
16: $N_W \leftarrow N_W \cup N_u$
17: $V\_Candidate \leftarrow V\_Candidate \backslash N_u$ //remove nodes in N_u from $V\_Candidate$
18: **end for**
return N_W, E_W

3.3 RDFG Summary Based on Node Centrality

In this section, we extend our summary approach by the notion of node centrality in order to process the nodes which are "important" in the RDFG but are not merged into supernodes by the $\boldsymbol{S}$ and $\boldsymbol{W}$ relations.

3.3.1 Node centrality in RDFG

We calculate the centrality of a node v, for both instance nodes and also class nodes, by combining the frequency of properties related to v and its bridging characteristics.

Definition 7 (*Node frequency*) Given an RDFG *G*, the frequency of a node *v* is defined as:

$$Freq(v) = \frac{\deg(v)}{|\{e|e \in E \wedge L(\mathrm{e}) \in PT(v)\}|} \tag{2}$$

where *PT*(*v*) is all the associated properties for the node *v*.

For an instance node, $PT(v) = P(v) \cup \{\text{rdf:type}\}$, and for class node, *PT*(*v*) includes properties such as rdfs:subClassOf or rdf:type. The frequency *Freq*(*v*) indicates the importance of *v* either instance node, or class node in terms of its properties. For example, in the Figure 2, the class node 1 (Agent) incidents to two edges with properties rdfs:subclassOf and rdf:type. Thus, the frequency of Agent is $Freq(Agent) \approx 0.29$.

The central node of RDF graph is not only related to its frequency of properties, but also related to its bridging characteristics.

Definition 8 (*Bridging coefficient*) Given an RDFG *G*, the bridging coefficient of node *v*, denoted as *Ln*(*v*), is defined as:

$$Ln(v) = \frac{\deg(v)^{-1}}{\sum_{u \in Neighbor(v)} \deg(u)^{-1}} \tag{3}$$

The bridging coefficient defined by (3) reflects how well the node is located between high degree nodes. A high value of the metric *Ln*(*v*) indicates that the node *v* connects to densely connected nodes. We define the central node using a linear combination of the frequency and bridging coefficient.

Definition 9 (*Node centrality*) The node centrality *FL* is a defined as a linear combination of the frequency and the bridging coefficient as shown in formula (4), where α is the weight between 0 and 1.

$$FL(v) = \alpha Freq(v) + (1 - \alpha) Ln(v) \tag{4}$$

A high value of *FL*(*v*) indicates both high property importance and connection centrality.

3.3.2 The SummaryFL algorithm

With concept of node centrality, we extend our summary approach by processing central nodes that are not summarized by the SumS or SumW algorithms. The SummaryFL accepts the summary based on ***S*** or ***W*** relation, i.e., $G_{S\sim}$ or $G_{W\sim}$ as input, and the output is the improved summary $G_{SF\sim}$ or $G_{WF\sim}$ by adding central nodes of *G* to the summary. For instance node

that is not summarized by SumS (SumW), we calculate its *FL* value and add it to the summary if it meet the centrality threshold (i.e., 0.5). In addition, its type nodes are also added to the summary and corresponding edges are also created. Since the algorithm SummaryFL needs to access the nodes in the RDFG, the time complexity is $O(|V|)$.

Algorithm 3 SummaryFL

Input: $G = (V, E, P, L)$, α, δ, $G_{S\sim}$ ($G_{W\sim}$) // δ is the centrality threshold
Output: $G_{SF\sim}(G_{WF\sim})$
1: $N_{SF} \leftarrow N_S$, $E_{SF} \leftarrow E_S$
2: **for** u in V **do**
3: **if** u not in N_S **then**
4: $FL(u) \leftarrow$ ComputeFL(u, α) // ComputeFL obtains the centrality value of u
5: **if** $FL(u) > \delta$ **then**
6: $N_u \leftarrow$ ScreateSN(u, V_u) // create supernode for u
7: $E_u \leftarrow$ ScreateSE (N_u, $PT(u)$) //create superedges for N_u
$N_{SF} \leftarrow N_{SF} \cup N_u$, $E_{SF} \leftarrow E_{SF} \cup E_u$
8: **end if**
9: **end if**
10: **end for**
return N_{SF}, E_{SF}

4 Experiments

In this section, we set up the experiments to evaluate our approach from the three aspects: **efficiency** (Section 4.1), **compression capability** (Section 4.2) and **coverage** (Section 4.3) of the summarization algorithms. The efficiency is measured by the execution time spent for summarizing RDFGs. The compression capability is measured by the size of the RDFG summary described by the set of supernodes and edges. The coverage refers to the ratio of nodes and edges summarized by the algorithms over the nodes and edges in the original RDFG. This metric denotes the ability of a summary to represent the original RDFG.

We selected two state-of-the-art RDFG summarization methods SumRDF [4] and PDMS [5] for the comparisons. SumRDF is a typical merging-based summarization method which generates summaries by the type information related to entities. PDMS is a typical extraction-based summarization which extracts central nodes from RDFGs. Since our approach uses the hybrid summarization strategies consisting of summarizing by node CS and centrality, comparing our approach to the typical single-strategy methods can demonstrate the effectiveness of our approach.

Table 1 RDFGs

RDF Datasets	Entities	Relations	Triples
AGROVOC	2002171	2830	5012375
DBpedia 1	20555	9120	31050
DBpedia 2	1790653	18730	3822106
Linkgeodata 1	2591324	30516	4914217
Linkgeodata 2	639274	16504	1225338
Wikidata	65349	10470	93867

Table 1 lists the datasets for the experiments: AGROVOC,[3] DBpedia,[4] LinkedGeoData, and Wikidata.[5] We used the AGROVOC Core containing 711M data for experiments, which includes 39,000 concepts and more than 800,000 terminologies. We used two subsets of DBpedia: DBpedia ontology (referred to as DBpedia 1) and the Linkeddata (referred to as DBpedia 2) containing 565 M geographic data. We used two subsets of LinkedGeoData: sport datasets (referred to as Linkgeodata 1) containing 706 M data and historic datasets (referred to as Linkgeodata 2) containing 164 M data. We used a subset of Wikidata including person and geodata.

Our approach was implemented using Java. The experiments were conducted on the machine equipped with Intel Core (TM) i7-7700 CPU @ 3.6 GHz 512SSD and 16GB Memory.

4.1 Performance of Summarization Algorithms

We compared the execution time of our proposed summarization algorithms, i.e., SumW, SumS, SumW+SummaryFL($G_{WF\sim}$), and SumS+SummaryFL ($G_{SF\sim}$), with PDMS and SumRDF. Figure 4 shows the time performance of the six algorithms. Overall, SumW, SumS, and PDMS performed better than other algorithms and SumRDF took the longest time to generate summaries. However, PDMS ignored the instance nodes of RDFGs and only summarized the schema nodes and therefore it provided limited information for the entire RDFGs. Different to PDMS, our algorithms summarized both instance and schema nodes, i.e., the entire RDFGs. Among our algorithms, SumW and SumS performed better than SumW+SummaryFL($G_{WF\sim}$), and SumS+SummaryFL($G_{SF\sim}$), because the former two algorithms only

[3]http://www.fao.org/agrovoc/

[4]https://wiki.dbpedia.org/

[5]https://www.wikidata.org/wiki/

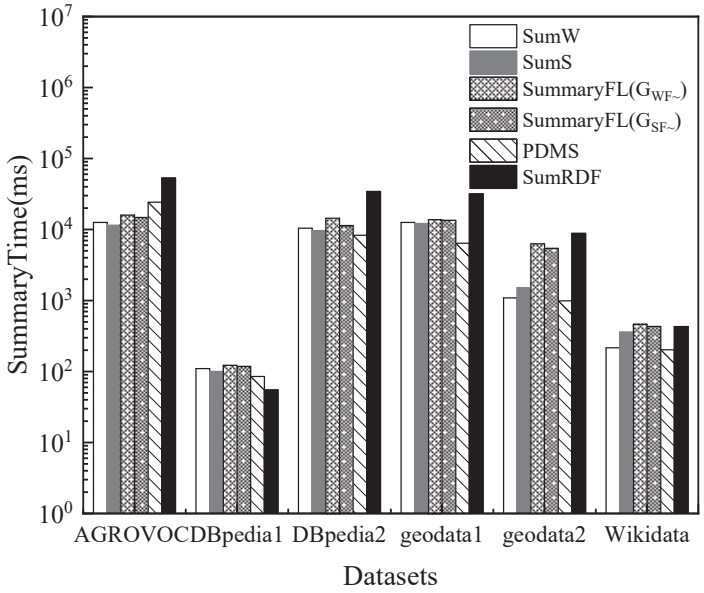

Figure 4 Summarization time.

Table 2 The size of summary generated by SummaryFL and SumRDF

			SummaryFL($G_{WF\sim}$)		SummaryFL($G_{SF\sim}$)		SumRDF	
Ddatasets	Nodes	Edges	Summary Nodes	Summary Edges	Summary Nodes	Summary Edges	Summary Nodes	Summary Edges
AGROVOC	2002171	3010204	183	1420	191	1922	7346	123999
DBpedia 1	20555	10495	3767	18056	892	19230	182	1414
DBpedia 2	1790653	2031453	946	10620	938	10610	18253	267884
Linkgeodata 1	2591324	2322893	3364	38741	3340	42081	26033	376508
Linkgeodata 2	639274	586064	1412	14660	1433	16072	11389	161575
Wikidata person	38872	54995	233	545	477	1566	1313	3704

compute the summaries based on the node CS, while the latter have an extra step of extracting central nodes.

4.2 Compression Capability of Summary

To evaluate the compression capability of summaries, we analyzed the sizes of the summaries generated by our algorithms and SumRDF. The sizes of the summaries, i.e., nodes and edges generated in the summaries, are shown in Table 2. SummaryFL generated more compact summaries for the datasets except DBpedia 1. For the AGROVOC Core dataset, the number of summary nodes generated by SumRDF was 40 times larger than that of the summary

Table 3 Details of the summary by generated by SumW, SumS, and SummaryFL

		Summary Nodes			
Datasets	Nodes	SumW	SummaryFL ($G_{WF\sim}$)	SumS	SummaryFL ($G_{SF\sim}$)
AGROVOC	2002171	165	183	150	191
DBpedia 1	20555	280	3767	285	892
DBpedia 2	1790653	520	946	723	938
Linkgeodata 1	2591324	1575	3364	1552	3340
Linkgeodata 2	639274	795	1412	774	1433
Wikidata person	38872	227	233	169	477

nodes generated by SummaryFL($G_{WF\sim}$). In addition, the number of summary edges by SumRDF was 87 times larger than that of the summary edges by SummaryFL($G_{WF\sim}$). The results were similar for other datasets except DBpedia 1. The reason for a larger summary size for DBpedia 1, the DBpedia ontology dataset, is that it covers a wide range of areas and contains richer properties than other datasets.

Table 3 provides more details about the summary results for our algorithms. The first-phase summarization process: SumW or SumS generated the majority of the supernodes which demonstrate that the CS-based summarization indeed capture the linking patterns of the RDFGs. In addition, the second-phase summarization: SummaryFL extracted central nodes as complementary to the summaries. For example, SumW generated 165 nodes, and SummaryFL ($G_{WF\sim}$) contained 183 nodes, which means that 18 central nodes were extracted and added into the summary by SumW.

4.3 Coverage of Summarization Algorithms

In this section, we analyzed the coverage of the RDFG summaries by the five algorithms: SumRDF, PDMS, SummaryFL($G_{WF\sim}$), SumS, and SumW. The criterion is to judge how many nodes and edges were covered by the RDFG summaries. *The coverage of a summary is defined as the ratio of nodes and property edges summarized by the summary over the nodes and property edges in the original RDFG*.

The evaluation results of coverage for the summaries over the six datasets are shown in Figure 5. It can be observed that the coverage rates of SumS and SumW were higher than that of PDMS and SumRDF algorithms. The coverage of PDMS was lower than that of SumS and SumW, especially for LinkedGeoData. This is because that PDMS only summarized schema graph. The coverage of SumRDF is lower than that of SumW, SumS, and

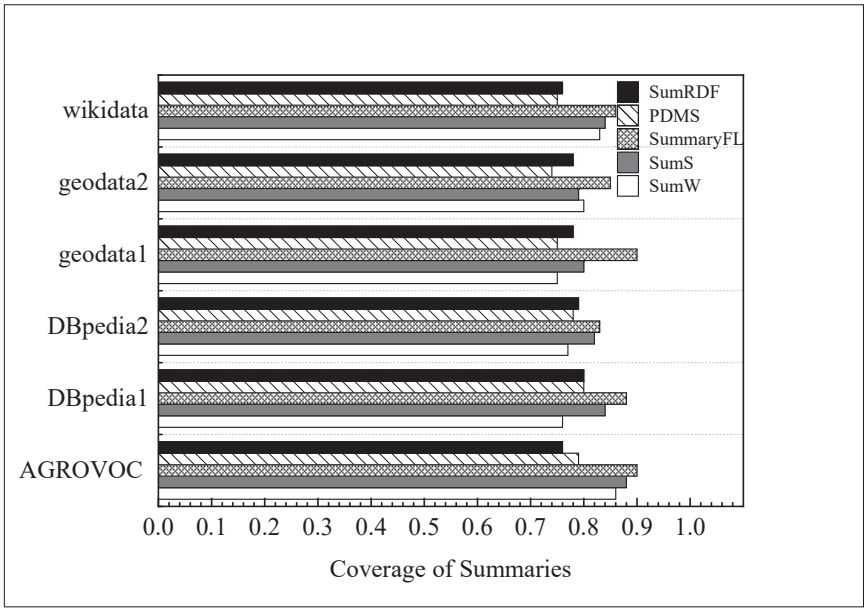

Figure 5 Coverage of the RDFG summaries.

SummaryFL, especially for the AGROVOC Core dataset. In general, the coverage of SumS is lower than that of SumW, which was consistent to the expectation. Furthermore, SummaryFL improved the coverage of the summaries, which had an average coverage as 0.87. The results showed that the proposed summarization method based on CS and centrality could summarize the major and key information of the original RDFGs.

5 Conclusions

In this paper, we present the summarization approach which considers both the node characteristic and centrality. The proposed RDFG summary can capture not only the graph structure but also the central nodes. Our approach consists of three algorithms: SumS, SumW and SummaryFL. The three algorithms can be used by two summarization strategies: SumS ($G_{S\sim}$) or SumW ($G_{W\sim}$) only, and SumS+SummaryFL ($G_{SF\sim}$) or SumW+SummaryFL ($G_{WF\sim}$). The summaries $G_{S\sim}$ and $G_{W\sim}$ capture certain types of the graph structure, whereas the summaries $G_{SF\sim}$ and $G_{WF\sim}$ keep not only the graph structure but central nodes of the original graph. We carried out experiments to compare our algorithms with competitive methods in terms of time

complexity, compression capability, and coverage rate. The results showed that our approach outperformed the compared methods for the three aspects. Our future work shall focus on personalized summarization that considers specific requirements of users and stream graph summarization.

References

[1] Čebirić, S., Goasdoué, F., Kondylakis, H., Kotzinos, D., Manolescu, I., Troullinou, G., Zneika, M.: Summarizing Semantic Graphs: A Survey. VLDB J. 28, 295–327 (2019).

[2] Kaushik, R., Shenoy, P., Bohannon, P., Gudes, E.: Exploiting local similarity for indexing paths in graph-structured data. Proc. Int. Conf. Data Eng. 129–140 (2002).

[3] Schätzle, A., Neu, A., Lausen, G., Przyjaciel-Zablocki, M.: Large-scale bisimulation of RDF graphs. In: Proceedings of the Fifth Workshop on Semantic Web Information Management (SWIM 2013) (2013).

[4] Stefanoni, G., Motik, B., Kostylev, E. V.: Estimating the cardinality of conjunctive queries over RDF data using graph summarisation. In: Proceedings of the World Wide Web Conference. pp. 1043–1052 (2018).

[5] Pires, C.E., Sousa, P., Kedad, Z., Salgado, A.C.: Summarizing ontology-based schemas in PDMS. In: Proceedings of International Conference on Data Engineering. pp. 239–244 (2010).

[6] Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: Ontology Understanding without Tears: The Summarization Approach. Semant. Web Jouranl. 8, 797–815 (2017).

[7] Queiroz-Sousa, P.O., Salgado, A.C., Pires, C.E.: A Method for Building Personalized Ontology Summaries. J. Inf. Data Manag. 4, 236–250 (2013).

[8] Safavi, T., Belth, C., Faber, L., Mottin, D., Muller, E., Koutra, D.: Personalized knowledge graph summarization: From the cloud to your pocket. In: Proceedings of IEEE International Conference on Data Mining, ICDM. pp. 528–537 (2019).

[9] Liu, Y., Safavi, T., Dighe, A., Koutra, D.: Graph summarization methods and applications: A survey. ACM Comput. Surv. 51, 1–34 (2018).

[10] LeFevre, K., Terzi, E.: GraSS: Graph Structure Summarization. In: Proceedings of the 10th SIAM International Conference on Data Mining, SDM 2010. pp. 454–465 (2010).

[11] Ko, J., Kook, Y., Shin, K.: Incremental Lossless Graph Summarization. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. 317–327 (2020).
[12] Lee, K., Jo, H., Ko, J., Lim, S., Shin, K.: SSumM: Sparse Summarization of Massive Graphs. Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. 144–154 (2020).
[13] Faralli, S., Finocchi, I., Ponzetto, S.P., Velardi, P.: Efficient pruning of large knowledge graphs. In: Proceedings of International Joint Conference on Artificial Intelligence. pp. 4055–4063 (2018).
[14] Luo, Y., Fletcher, G.H.L., Hidders, J., Wu, Y., De Bra, P.: External memory k-bisimulation reduction of big graphs. In: Proceedings of International Conference on Information and Knowledge Management. pp. 919–928 (2013).
[15] Čebirić, Š., Goasdoué, F., Manolescu, I., Šejlač, Š., Šejlaèebirić, Š.: Query-Oriented Summarization of RDF Graphs. In: In: Maneth S. (eds) Data Science. BICOD 2015. Lecture Notes in Computer Science, vol. 9147. Springer, Cham. (2015).
[16] Song, Q., Wu, Y., Dong, X.L.: Mining summaries for knowledge graph search. In: Proceedings of IEEE International Conference on Data Mining, ICDM. pp. 1215–1220 (2017).
[17] Zneika, M., Lucchese, C., Vodislav, D., Kotzinos, D.: RDF Graph Summarization Based on Approximate Patterns. Commun. Comput. Inf. Sci. 622, 69–87 (2016).
[18] Pappas, A., Troullinou, G., Roussakis, G., Kondylakis, H., Plexousakis, D.: Exploring importance measures for summarizing RDF/S KBs. Lect. Notes Comput. Sci. 10249 LNCS, 387–403 (2017).
[19] Presutti, V., Aroyo, L., Adamou, A., Schopman, B., Gangemi, A., Schreiber, G.: Extracting core knowledge from Linked Data. In: CEUR Workshop (2011).
[20] Liu, Q., Cheng, G., Gunaratna, K., Qu, Y.: Entity summarization: State of the art and future challenges. J. Web Semant. 69, 100647 (2021).
[21] Gunaratna, K., Thirunarayan, K., Sheth, A.: FACES: Diversity-aware entity summarization using incremental hierarchical conceptual clustering. In: Proceedings of the National Conference on Artificial Intelligence. pp. 116–122 (2015).
[22] Thalhammer, A., Lasierra, N., Rettinger, A.: LinkSUM: Using link analysis to summarize entity data. In: Proceedings of the International Conference of Web Engineering (ICWE),Lecture Notes in Computer Science. pp. 244–261 (2016).

[23] Liu, Q., Cheng, G., Qu, Y.: Entity summarization with high readability and low redundancy. Sci. Sin. Informationis. 50, 845–861 (2020).
[24] Yang, Y., Li, Y., Karras, P., Tung, A.K.H.: Context-aware Outstanding Fact Mining from Knowledge Graphs. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD). pp. 2006–2016 (2021).
[25] Zheng, W., Zou, L., Peng, W., Yan, X., Song, S., Zhao, D.: Semantic SPARQL similarity search over RDF knowledge graphs. In: Proceedings of the VLDB Endowment. pp. 840–851 (2016).
[26] Sankara Rao, A., Durga Bhavani, S., Sobha Rani, T., Bapi, R.S., Narahari Sastry, G.: Study of Diversity and Similarity of Large Chemical Databases Using Tanimoto Measure. Commun. Comput. Inf. Sci. 157 CCIS, 40–50 (2011).

Biographies

Jimao Guo received her M.S. degree in Computer Science from the Southwest University (China) in July 2022. She is now working at the Neijiang Education and Examination Institute, SiChuan, China. Her research interests include Semantic technology and knowledge modeling.

Yi Wang received her M.S. degree in Computer Science from the Southwest University (China) in 2004 and the Ph.D. in Computer Science from the Macquarie University (Australia) in 2012. She is an associate professor at the Southwest University (China) since 2014. Her research interests include knowledge representation and knowledge graph refinement.