Q-KGSWS: Querying the Hybrid Framework of Knowledge Graph and Semantic Web Services for Service Discovery

Pooja Thapar, Lalit Sen Sharma
Department of Computer Science and IT, University of Jammu, J&K, India

***Abstract*—In the era of big data, Knowledge Graphs (KGs) have become essential tools for managing interconnected datasets across various domains. This paper introduces a novel RDF (Resource Description Framework) based Knowledge Graph of Semantic Web Services (KGSWS), designed to enhance service discovery. Leveraging the versatile SPARQL query language, the framework facilitates precise querying operations on KGSWS, enabling customized service matching for user queries. Through comprehensive experimentation and analysis, notable improvements in accuracy (69.75% and 90.01%) and rapid response times (0.61s and 1.57s) across two semantic search levels are demonstrated, validating the efficacy of the approach. Furthermore, research questions regarding the interlinking of ontologies, methods for formulating automatic queries, and efficient retrieval of services are addressed, offering insights into future avenues for research. This work represents a significant advancement in the domain of semantic web services, with potential applications across various industries reliant on efficient service identification and integration. Future phases of research will focus on logical inference and the integration of machine learning-based graph embedding models, promising even greater strides in knowledge discovery within the KGSWS framework, thus reshaping the domain of semantic web services.**

Keywords*—Ontologies; knowledge graph; semantic web services; SPARQL query language; OWLS; data integration; service discovery

I. INTRODUCTION

In today's rapidly changing computing environment, the fundamental paradigm of Service Oriented Architecture (SOA) is supported by core engineering principles of Reusability, Discoverability, and Interoperability. These principles provide a robust framework for orchestrating communication in distributed computing environments characterized by services encapsulated in discrete units. SOA leverages standardized interfaces and protocols, enabling seamless integration and communication between heterogeneous systems, thereby fostering a modular and extensible architecture [1]. Web Services (WS), the embodiment of SOA, play a pivotal role in contemporary enterprise solutions by enabling the integration of systems across organizational boundaries. WS, being self-described and disseminated by organizations, facilitate interoperable, machine-to-machine interactions and promote code reusability across networks. The orchestration of this integration relies on the convergence of components such as WSDL (Web Service Description Language), UDDI (Universal Description, Discovery and Integration), and SOAP (Simple Object Access Protocol), including the Service Consumer, Service Registry, and Service Provider. These components empower manual examination of WS within the UDDI repository, enabling users to identify services based on their specific functionalities [2] . However, the use of XML as the standard language for describing WS capabilities, while crucial for service description, introduces ambiguity in keyword-based matchmaking due to the absence of machine-understandable semantic information. This challenge in the software landscape necessitates the introduction of formal knowledge representation—a shared, universally harnessed resource within intricate software engineering systems [3] [4].

Semantic Web Services (SWS) build upon the foundation of WS by incorporating semantic extensions through metadata vocabularies from the Semantic Web. These extensions are either realized through semantic annotations known as SAWSDL (Semantic Annotations for WSDL) or by employing domain ontologies rooted in description logic, as seen in OWL-S. Ontologies, acting as metadata vocabularies, provide a formal, machine-understandable representation of concepts and their relationships within a domain, enhancing knowledge comprehension. Among the conceptual models defining SWS, OWL-S distinguishes itself through the integration of domain ontologies into WS descriptions. This integration enables logical reasoning, facilitating more precise and automated service discovery. Rules and constraints, defined on the concepts and properties of domain ontologies, serve as reference points for logical inference, fortifying specification, consistency, and conceptualization during service discovery. It, therefore empower domain experts to navigate knowledge contexts across domains with unambiguous precision. However, the intricacies of description logic formalism within domain ontologies, and the proliferation of SWS services have presented efficiency challenges for industries and organizations in terms of service discovery [5-7].

This paper confronts these challenges by leveraging the burgeoning paradigm of Knowledge Graphs (KGs). The unique capability of KGs to incorporate semantic metadata descriptions of entities makes them a prime candidate for complex querying through semantic web technologies, particularly SPARQL querying [8] [9]. This work introduces a pioneering approach: the creation of an RDF-based Knowledge Graph of Semantic Web Services (KGSWS) for service discovery, empowered by SPARQL-based question

answering. At its core, this approach centers on generating interlinked data from domain ontologies to significantly augment service discovery accuracy. By employing advanced KG techniques and querying methodologies, SPARQL queries utilize the knowledge encapsulated in OWL-S service descriptions and their linked ontologies from KGSWS to address queries based on diverse user input/output requests. In essence, KGSWS uncover the hidden correlations between service descriptions and domain ontologies; and provide insightful recommendations to refine the user experience effectively and thereby serving as a bridge between the realm of SWS and KGs.

The paper's structure unfolds as follows: Section II delves into the Background elements and techniques underpinning this research. Section III explores related work, elucidating the context that motivated our research endeavor. Section IV frames the research questions to be in the proposed work. Section V meticulously details the proposed methodology, laying out the innovative approach that unifies SWS ontologies and KGs. Following this, Section VI unveils our experimental results, providing a comprehensive comparative analysis with related work. Finally, in Section VII, we conclude and outline future avenues for this research, illuminating the potential for further advancements in the domain of semantic web services.

II. BACKGROUND

This section lays the groundwork for understanding the key components and concepts that underpin our research:

A. Semantic Web

Semantic Web is an advancement to the existing Web 2.0, designed to enable machines to process information intelligently, akin to human reasoning. This capability is achieved through the strategic use of semantic tags, imbuing data with what we call "Semantic Metadata." To exemplify, consider the concept of a "Service" in our context. Semantic Metadata goes beyond the term itself, distinguishing between a "Web Service" as a functional software component and a "Semantic Web Service" as a service enriched with semantic information, by providing specific details about its functionalities, inputs, and outputs, thereby facilitating precise and automated discovery. Structured Linked Open Data (LOD) is represented as a graph, interconnecting data across servers via Universal Resource Identifiers (URIs). RDF (Resource Description Framework) leverages URIs to denote relationships within this graph. In this structured environment, edges serve as the conduits for relationships between two resources, culminating in triplets within a directed graph. Ontologies, integral to the Semantic Web stack, enhance data within specific domains with semantic metadata. These ontologies furnish explicit, machine-understandable descriptions of concepts and their relationships, thus deepening data comprehension. OWL (Web Ontology Language), a logic-based language, forms the foundation of these ontology models, empowering RDF triple stores with rigorous constraints. The presence of OWL reasons ensures not only logical consistency but also real-time computation of inferred knowledge, thereby propelling data automation and interoperability. Incorporating Knowledge Graphs and the SPARQL Query language into the Semantic Web framework enhances our methodology for precise service discovery [10-13].

B. Knowledge Graph (KG)

Knowledge Graphs (KGs) are intricate representations of real-world entities as interconnected nodes, serving as central hubs for information retrieval and complex web searches. Their significance in encapsulating machine-understandable contextual information within heterogeneous environments has spurred intensive research in semantic matching. In 2012, Google pioneered the concept with its Google Knowledge Graph, featuring an impressive array of 570 million entities [14]. Subsequently, KGs such as Geonames, FactForge [15], Yago [16], Wikidata [17], and more have been carefully built, containing a lot of Linked Open Data (LOD). Across various domains, including retail, entertainment, healthcare, finance, and more, KGs have revolutionized question answering and knowledge discovery, underlining their versatility. KGs typically adopt either the subject-property-object (s,p,o) notation or the entity-relation-entity representation. Formally, a KG is defined as a subset of $(\mathcal{E} \times R \times \mathcal{E}) \cup (\mathcal{E} \times L_r \times L)$, where:

- $\mathcal{E}$ refers to the set of entities.
- R signifies the set of relations between entities.
- L_r signifies the set of relations linking entities with object of literals.
- L refers to the set of literals.

In semantic network modeling, triplets (s,p,o) denote metadata statements resulting from the mapping of subject (s) and object (o) to nodes or entities and their associated properties (p) to links or relations. In this structured framework, $\mathcal{E}$ must be a URI or a blank node representing the subject, while R and Lr must be URIs, and the object entity can be a URI, blank node, or literal. In the context of SWS discovery, KGs herald a transformative shift from big data discovery to intelligent data discovery. Knowledge integration from domain ontologies plays a pivotal role in this transformation [18].

C. SPARQL Query

SPARQL Protocol and RDF Query Language (SPARQL) stands as the preeminent query language for Semantic Web data, providing a means to interrogate extensive RDF-based KGs. Unlike SQL, tailored for relational databases, SPARQL caters to NoSQL graph databases like KGs, enabling the integration of knowledge from diverse sources to derive new insights. Built on the HTTP transport layer protocol, SPARQL facilitates querying information from multiple KGs via federated queries. It employs graph pattern matching, represented as subject, predicate, and object (s,p,o) triple patterns, to uncover solutions. SPARQL supports four query types: ASK, SELECT, CONSTRUCT, and DESCRIBE, each designed for specific querying purposes [19]. By harnessing the capabilities of SPARQL, our methodology transcends conventional querying techniques, providing a sophisticated means to navigate and utilize the rich semantic information encapsulated within the KG.

III. Related Work and Motivation

Over the past decade, substantial research efforts have been dedicated to enhancing the efficiency of Semantic Web Services (SWS) discovery algorithms. These algorithms aim to facilitate the discovery, selection, composition, classification, and ranking of SWS based on various functional and non-functional parameters. Numerous matchmaking algorithms have been devised, leveraging both logical and non-logical functional parameters of SWS. These algorithms enhance the selection and recommendation process for SWS. Initial variant of OWLS-M0 [20] used only logic based semantic similarity measure on I/O for service discovery. However, the other variants in [20] and notable work by [21, 22] employed hybrid matchmakers that consider logical and non-logical parameters from OWL-S and WSMO services. Semantic similarity metrics such as cosine similarity, loss-of-information, and Jensen-Shannon divergence were employed to assess the compatibility of services. One challenge encountered in this context is the time-consuming creation of matchmaker ontologies, particularly when handling a substantial number of services. To address this, [23, 24] introduced a caching-based mechanism called Service Discovery Caching (SDC). SDC involves the construction of a graph to cache frequently used services, thereby mitigating the time overhead. Another line of research [25] employed First Order Logic (FOL) to formally describe service capabilities. The study utilized SPARQL 1.0 for querying services based on their descriptions, particularly in cases where services provided Preconditions and Effects (PE) descriptions. Although promising, this method was constrained by the limited availability of services with PE descriptions in the dataset, limiting its generality. Efforts to reduce the problem space of service searching led to prefiltering mechanisms [26-28]. These mechanisms were designed to enhance existing matchmaker engines using SPARQL queries, significantly improving response times. Nevertheless, the challenge of semantic matching persisted, particularly for complex concepts and relations within domain ontologies. Working on the same OWL-S dataset, the work in [29] have used unsupervised learning methods like DBSCAN clustering to cluster the services semantically closed and thereby finding the semantically closed cluster to user requirement using Latent Semantic Analysis (LSA). An ensemble model based approach in [30] use decision tree and logistic regression for service classification and recommendation of top-10 services. However, none of these works implemented the concept of KGs in the domain of SWS. The work cited in study [31] [32] employed machine learning methodologies; specifically K-Means clustering was used in [31] to generate a Service Ontology from SWS corpus. Results demonstrated that the generated ontology can be used for the discovery mechanism. Later work in [32] utilized first K-Means clustering and then K-Nearest Neighbors (KNN) for classification. This analysis yielded a noteworthy accuracy rate of 89.28%. However, it's essential to emphasize that this algorithm was rigorously tested within a restricted range of domains. Another work in [33] used part of OWL-S dataset i.e. 30000 triplets to add domain knowledge in KG and also for training and testing purpose of user's intent. However, with advanced machine learning methods, the overall accuracy of SWS discovery has improved. However, most of these works [29-32] focused on a subset of domains within the OWLS collection, and none of the works have implemented an automatic querying method for user queries. Instead of using the relevance file provided for OWLS-TC as per user queries, different methods have been employed to compute evaluation metrics.

In parallel with advancements in SWS discovery, Knowledge Graphs (KGs) have emerged as a powerful tool for representing and retrieving knowledge from large interlinked datasets. These KGs integrate domain ontologies at a universal level, creating extensive graphical representations of interlinked data that support complex queries and unified knowledge discovery. Recent work on KGs has generated KGs in different domains whereas some works perform querying over the existing KGs to improve the tasks like recommendation systems, link prediction, node classification, and knowledge discovery [34-36]. Another line of works in [37-40] generate SPARQL queries from natural language for existing KGs like DBpedia, Wikidata for complex querying. Furthermore, Graph Neural Network-based learning models have been employed to enhance KGs' performance in tasks such as link prediction and multihop querying [41-43]. Our motivation for this research stems from the intersection of SWS discovery and KGs. Both domains utilize ontologies as a foundation, making them amenable to integration. While previous works have primarily focused on matchmaking within SWS or querying KGs independently, there is a notable gap in seamlessly integrating SWS discovery with Knowledge Graphs (KGs) specific to our domain of SWS for service discovery. This is the key motivation behind our research. We aim to bridge these domains by creating a Knowledge Graph of Semantic Web Services (KGSWS). This integrated approach enables enhanced SWS discovery using KG-based querying techniques.

IV. Research Questions

The motivation for this proposed work arose from an extensive review of related research papers and the challenges discussed in the field. Our objective is to establish a centralized Knowledge Graph (KG) for Semantic Web Services (SWS) and address the following research questions:

1) How can ontologies from different domains be interlinked to form an extensive Knowledge Graph enriched with semantic metadata thereby enhancing the discovery of SWS?

2) What methods can be used to formulate the automatic queries on the KG, aligned with varying numbers of inputs-outputs for effective querying purposes?

3) How can we efficiently retrieve SWS from big KG that precisely matched the user requirements for service discovery? Additionally, how can we identify the closely related services when an exact match is not available, maintaining the integrity of the user's query?

V. PROPOSED APPROACH

In this section, the detailed experimentation done to design a framework for discovery of relevant services across KGSWS has been discussed. The framework includes the steps required to create, visualize, and query the KGSWS as shown in Fig. 1.

The detailed insights to these steps have been discussed in Phase 1 and Phase 2 of the framework. Phase 3 discussed the validation process of the results.

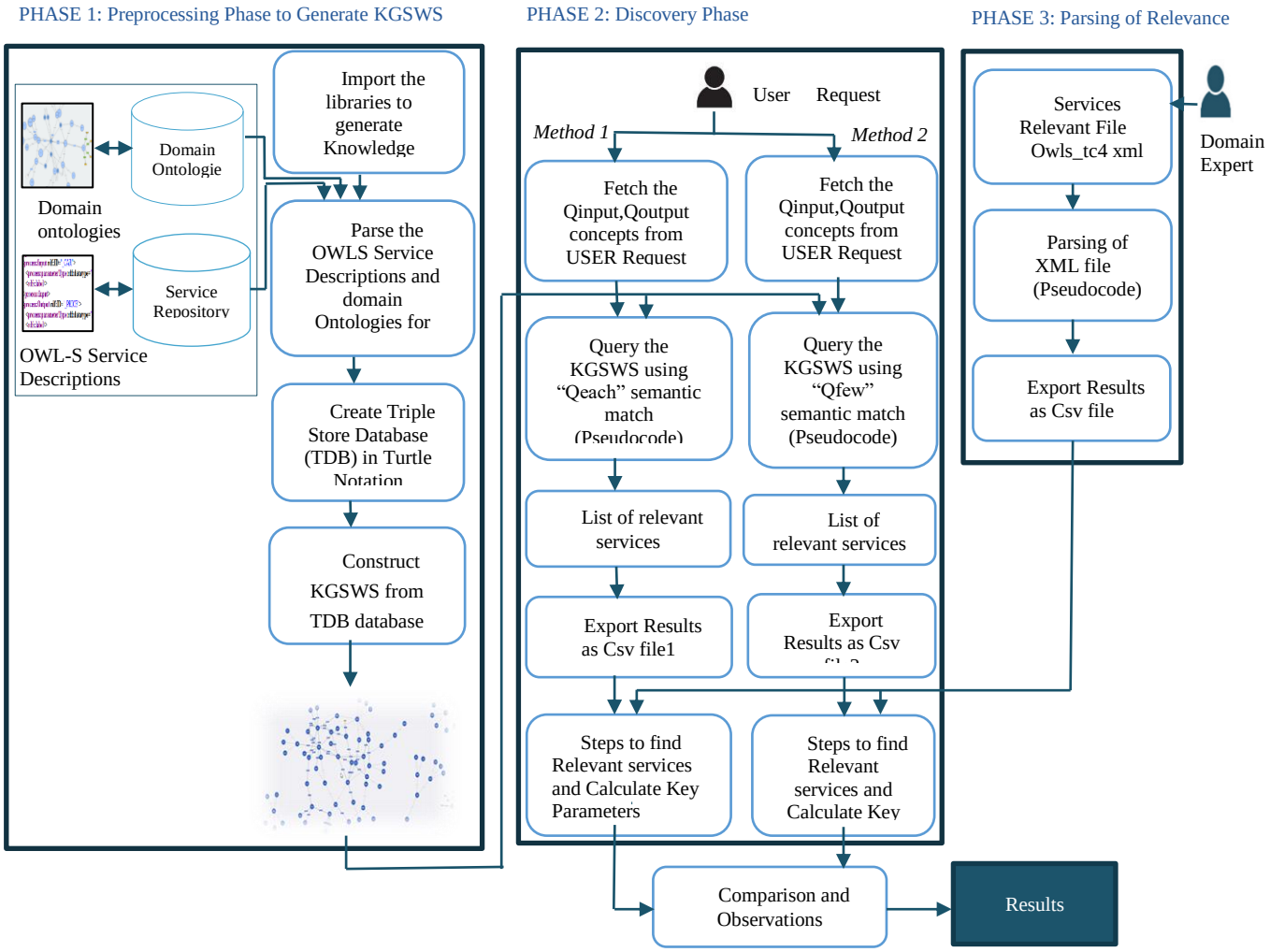


Fig. 1. Proposed work methodology.

A. Phase 1: Preprocessing Phase to Generate Knowledge Graph

The core step of our approach is the creation of a Knowledge Graph designed explicitly for Semantic Web Services (KGSWS). This phase describes the steps involved in constructing the KGSWS from the OWL-S service descriptions and their associated domain ontologies with an example. Through this, semantic metadata was integrated with KGSWS that forms the backbone of the proposed approach.

1) OWL-S service descriptions and KGSWS Construction: OWL-S services are based on frameworks that add semantic extensions to non-semantic web services using domain ontologies. In OWLS approach, Service Profile describes the capabilities of the service and refer as the upper ontology, Service Modelling demonstrates data flow and control flow within the services and Service Grounding specify the procedure of interaction with these services using protocols and message formats. OWL-S service description is mainly provided by the upper ontology Service Profile where different terms describe the inputs, outputs, preconditions, and their effects (IOPE); postconditions, assumptions based on the transition rules, if required. Once the service based on the brief description of IOPE concepts in the Profile is selected; Process Model related with Service Profile is used for further interaction with these services by linking them to their domain ontologies. For instance, consider the Profile and Process Model in the Service Description of finding author of the book as shown in Fig. 2(a). In the Profile of the service <profile:serviceName>, <profile:textDescription>, <profile:hasInput> and <profile:hasOutput> provides the information of the book whereas the <process:

parameterType> in the process model linked the I/O concepts "Book" and "Author" to the ontology "books.owl" for the semantic matching of the concepts. In order to construct KGSWS from of these services, these I/O terms are referred as entities and their associated properties and restrictions form the relations are then mapped to literal or another entity. Since these I/O concepts refer to some domain ontologies description in the Process Model, these ontologies are also linked in KGSWS by extracting an interoperable definition of each concept from their properties and restrictions from domain ontology. This was done by first serializing the service descriptions concepts and their associated ontologies to Triple Notation (Turtle Notation. ttl). Apache Jena Fuseki was used for integration, thus all the OWL mappings, alignment axioms that indicate equivalence or relationships between terms were preserved. Afterwards, KGSWS graph was created using network and rdf based libraries like rdflib, networkx in Python which consists of 1,21,542 Triplets. Fig. 2(b) shows the snippet of turtle representation for example shown in Fig. 2(a).

```
<service:Service rdf:ID="TITLE_BOOK_SERVICE">
<service:presents rdf:resource="#TITLE_BOOK_PROFILE"/>
<service:describedBy rdf:resource="#TITLE_BOOK_PROCESS"/>
<service:supports rdf:resource="#TITLE_BOOK_GROUNDING"/>
</service:Service>

<profile:Profile rdf:ID="TITLE_BOOK_PROFILE">
<service:isPresentedBy rdf:resource="#TITLE_BOOK_SERVICE"/>
<profile:serviceName xml:lang="en">
BookSearch
</profile:serviceName>
<profile:textDescription xml:lang="en">
A book search engine service, which provides information of books whose title be
</profile:textDescription>
<profile:hasInput  rdf:resource="#_TITLE"/>
<profile:hasOutput rdf:resource="#_BOOK"/>

<profile:has_process rdf:resource="TITLE_BOOK_PROCESS" /></profile:Profile>
```

(a)

```
<http://127.0.0.1/services/1.1/BookSearchService.owls#TITLE_BOOK_SERVICE>
        a       <http://www.daml.org/services/owl-s/1.1/Service.owl#Service> ;
        <http://www.daml.org/services/owl-s/1.1/Service.owl#describedBy>
                <http://127.0.0.1/services/1.1/BookSearchService.owls#TITLE_BOOK_PROCESS> ;
        <http://www.daml.org/services/owl-s/1.1/Service.owl#presents>
                <http://127.0.0.1/services/1.1/BookSearchService.owls#TITLE_BOOK_PROFILE> ;
        <http://www.daml.org/services/owl-s/1.1/Service.owl#supports>
                <http://127.0.0.1/services/1.1/BookSearchService.owls#TITLE_BOOK_GROUNDING> .

<http://127.0.0.1/services/1.1/BookSearchService.owls#TITLE_BOOK_PROFILE>
        a       <http://www.daml.org/services/owl-s/1.1/Profile.owl#Profile> ;
        <http://www.daml.org/services/owl-s/1.1/Profile.owl#hasInput>
                <http://127.0.0.1/services/1.1/BookSearchService.owls#_TITLE> ;
        <http://www.daml.org/services/owl-s/1.1/Profile.owl#hasOutput>
                <http://127.0.0.1/services/1.1/BookSearchService.owls#_BOOK> ;
        <http://www.daml.org/services/owl-s/1.1/Profile.owl#has_process>
                <http://127.0.0.1/services/1.1/TITLE_BOOK_PROCESS> ;
        <http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceName>
                "\nBookSearch\n"@en ;
<
```

(b)

Fig. 2. (a) Example of service description structure [42]. (b) Shows the snippet of turtle notation for above service description example used to construct KGSWS.

B. Phase 2: Discovery Phase

The main objective of this phase was to perform queries over KGSWS created in phase 1 of the proposed framework.

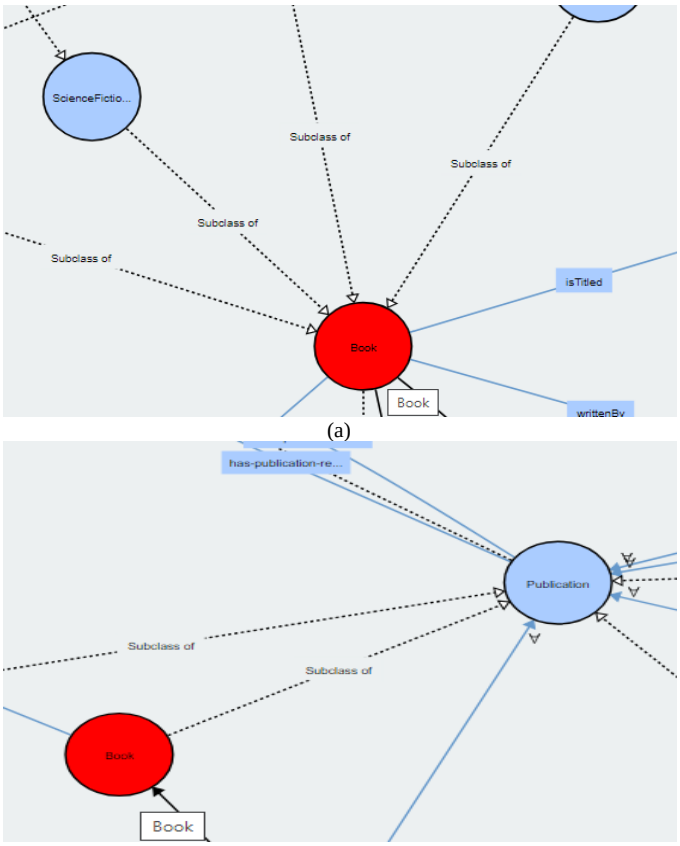
1) Querying and Searching from KGSWS: KGSWS, generated in Phase 1, comprises millions of entities and their attributes represented as nodes and edges. To query such specialized KGs, specialized knowledge of the querying language and a deep understanding of the underlying structure are required. Addressing the challenge of searching for relevant services based on the I/O concepts mentioned in the user request, we automated the generation of SPARQL queries based on the OWL-S service description model. These queries are then executed across the KGSWS to search and retrieve relevant services. The proposal used SPARQL 1.1 query due to its efficient results in pre filtering of domain wise services [28] and recommendation by W3C for querying over graphical databases. However, the work lacked its own matchmaker and has not integrated different domain ontologies to create big KG due to which services based on equivalent concepts were not retrieved. For instance, to find the name of books giving title or other information as input retrieve the service no.1 and 2 with output concept "Book" from "books.owl" domain ontology as valid services as shown in Table I. In the given query, the service no. 3 having interface description "Publication-number" as input and "Book" as output concept from "univ.owl" is also valid due to equivalence relation of "books:Book ≡ univ:Book" where books and univ are namespaces for "books.owl" and "univ.owl" ontologies respectively.

As show in Fig. 3(a) of "books.owl" ontology, the "Book" concept was used to describe information of different types of books and in Fig. 3(b) of "univ.owl", the same highlighted concept was added as a subclass of "Publication" concept. To query the KGSWS, two methods were implemented with a view to include different degree of semantic matching.

2) Semantic matching: To evaluate the performance of different degree of semantic match of concepts during discovery of services from their knowledge graph, two methods namely Method 1 and Method 2 were used to generate query wise relevant list. These two matchmaking algorithms focus solely on the input and output parameters of the service and employ two levels of semantic concept matching, namely 'Q_{each}' and 'Q_{few}' as outlined in Table II. This is followed by the pseudocodes for both discovery methods. The Phase 1 of constructing KGSWS is followed by the process of parsing I/O concepts based on user requests. Subsequently, a SPARQL query, guided by steps 9-12, was executed to retrieve a list of relevant services using Select, Where, Filter, Bind, Regular expressions options in the query. For the determination of services using "Q_{each}," each concept in the user request was considered, and services aligned with equivalent concepts in other ontologies within the same domain were retrieved through the 'owl:equivalentClass' restriction. Additionally, for "Q_{few}," beyond the steps involved in "Q_{each}," a 'UNION' operation in the SPARQL query was introduced to filter out entirely irrelevant services.

TABLE I. SOME RELEVANT SERVICES TO FIND THE BOOK NAME FROM ITS GIVEN INFORMATION

S.No.	WSName	Textual Description of the Functionality	Interface Description	
			Input	Output
1.	BookFinder.owls	The services retrieves the book information having title as input	books:Title	books:Book
2.	BookSearchService.owls	Search engine for book	books:Title	books:Book
3.	Publication_book_service.owls	The service retrieve the book name with given publication number	niv:Publication	univ:Book



(a)

(b)

Fig. 3. (a) Books.owl. (b) Univ.owl.

TABLE II. DEMONSTRATED THE METHODS FORMULATION IN DESCRIPTION LOGIC

Method No.	Method Name	Description Logic Representation	Description
Method 1	Q_{each}	If $I_{concept}(S) \equiv I_{concept}(R_q) \wedge O_{concept}(S) \equiv O_{concept}(R_q)$	Service S concepts are semantically equivalent to R_q Concepts
Method 2	Q_{few}	If $I_{concept}(R_q) \subseteq I_{concept}(S) \wedge O_{concept}(R_q) \subseteq O_{concept}(S)$	R_q concepts matched to some concepts of service S
where $I_{concept}(S)$, $I_{concept}(R_q)$, $O_{concept}(S)$, and $O_{concept}(R_q)$ represents the inputs/outputs concepts of advertised and request services respectively			

Pseudocode for Method 1: Q_{each}

1. **Input:**
2. Given user query refer to required Q_{input} and Q_{ouput} where:
   Q_{input}: List of required input concepts in the query
   Q_{ouput}: List of required output concepts in the query
3. **Output:**
4. RSL: Relevant Service List
5. **Local Resources:**
6. KGWS = Knowledge Graph of Services and their Ontologies
7. For each query, Parse Q_{input} and Q_{output}
8. Parse the TDB knowledge graph of services
9. Check if the predicate of the triplet is service:Service
10. For each service, retrieve the process Input and Output tags from the triplet
11. Retrieve <process:parameterType> tags of process Input and Output to retrieve domain ontology
12. Apply filters for semantic matching of "each" concept of Q_{input} and Q_{ouput}.
13. Generate RSL(Q_{input}, Q_{ouput}, RSL- Relevant Service List)
14. Repeat

Pseudocode for Method 2: Q_{few}

1. **Input:**
2. Given user query refer to required Q_{input} and Q_{ouput} where:
   Q_{input}: List of required input concepts in the query
   Q_{ouput}: List of required output concepts in the query
3. **Output:**
4. RSL: Relevant Service List
5. **Local Resources:**
6. KGWS = Knowledge Graph of Services and their Ontologies
7. For each query, Parse Q_{input} and Q_{output}
8. Parse the TDB knowledge graph of services
9. Check if the predicate of the triplet is service:Service
10. For each service, retrieve the process Input and Output tags from the triplet
11. Retrieve <process:parameterType> tags of process Input and Output to retrieve domain ontology
12. Apply filters for semantic matching of "few" concepts of Q_{input} and Q_{ouput}.
13. Generate RSL(Q_{input}, Q_{ouput}, RSL- Relevant Service List)
14. Repeat

C. Phase 3: Parsing of Relevance File

In this phase, the list of services generated in Phase 2 was validated against the services in the relevance file provided by domain experts for "OWL-TC4."

The file features a <binaryrelevanceset> root tag containing <request>, <name>, and <uri> tags for each service request, as shown in the above structure of XML relevance

file. These tags were employed to assign a unique ID to each request, with service request names based on the requested service's functionality. The <uri> tag contains a unique address to identify the request on the web. Furthermore, each <request> has a <ratings> tag containing multiple <offer> tags specifying the <name>, <uri>, and binary relevance in the <relevant> tag of the service. These requests have been parsed and the relevant information was stored in a CSV relevance file using Python libraries for hierarchical structures as shown in pseudocode. The service list from Phase 2 was compared with this relevance file to determine the key parameters of the model, as discussed in the next section.

Pseudocode: To Parse XML DOM	Structure of the XML Relevance File
1. Import the XML libraries and other hierarchical libraries 2. Parse the relevance file 'owls-tc4.xml' 3. Get the root of the structure *//To match the name and uri of each query request given by the user* 4. For each user request as child in xml file 5. Retrieve the name and uri of the request *//Find the name, uri, and relevance value of services for particular request* 6. For each item offer giving the details of the service 7. Retrieve the service name, uri and its relevance value. 8. Append the results in data frame //As hierarchical structure 9. Export the results as "outputxml.csv"	<binaryrelevanceset> *//for the binary relevance sets* <request ...> <name ... /> <uri .../> <ratings> <offer ...> *//each request contains multiple offers* <name ... /> <uri .../> <relevant>value</relevant> *//here the value can be either 0 or 1* </offer> ... </ratings> </request> ... </binaryrelevanceset>

VI. Results and Discussion

In this section, a comprehensive analysis of the performed experiments has been discussed to assess the performance and efficiency of our proposed framework. The section commenced by describing the experimental scenario and the dataset used for testing. Following this, an in-depth analysis of the results has been done to emphasize significant observations and comparisons between our approach and baseline methods.

A. Experimental Scenario

In order to test the performance of our proposal, ontology based SWS test collection OWLS-TC v4 [44] was used to create KGSWS. As shown in Table III, the collection contains 1083 services from nine domains namely Education, Medical Care, Food, Travel, Communication, Economy, Weapon, Geography, and Simulation. These services were based on 48 domain ontologies. KGSWS consists of 1,21,542 triplets generated from the integration of these 1083 service descriptions and their corresponding 48 domain ontologies. Further, 38 test queries with varying number of input output concepts based on the domains were executed against KGSWS using two different SPARQL query methods.

TABLE III. Domain Names and Number of Services in each Domain

S.No.	*Domain*	*No. of Services*
1	*Education*	279
2	*Medical Care*	73
3	*Food*	34
4	*Travel*	197
5	*Communication*	59
6	*Economy*	325
7	*Weapon*	40
8	*Geography*	60
9	*Simulation*	16
	Total	*1083*

B. Analysis of Results

In this subsection, we provide a comparative analysis of our proposed methods i.e. Method 1 (Q_{each}) and Method 2 (Q_{few}), against baseline approaches. The key parameters that were used to compare the performance of these methods are given below in Table IV [45]. The Macro-Averaged Precision Recall metrics was used to give equal relevance to each test query and its value lies in the range [0, 1]. The relevance file provided for OWLS-TC (as discussed in Phase 3) was utilized to compute the key parameters of the methods in the proposed framework.

1) Analysis of method 1 results: As discussed, Method 1 employed each concept matching approach, aligning service requests with the concepts within the KGSWS. This precision-oriented method demonstrated noteworthy results as shown in Fig. 4(a) and Fig. 4(b). It retrieved fewer or zero irrelevant services, leading to significantly higher precision in most cases. Also, Method 1 exhibited an average query response time of a mere 0.61 seconds, showcasing its efficiency. However, this method faced challenges in test cases 8, 17 and 22, where it failed to locate any relevant service matching each concept, leading to a precision value of zero and due to equal importance of each query it dropped the overall precision value even if in most of the cases the values reaches 1 as shown in Fig. 4(a). Furthermore, the proposed model did not include logical inferred "subclass" concepts during the matchmaking of each concept. Due to this, all the relevant services having input-output concepts as direct subclass concepts of requested concepts in the query were not retrieved dropping the macro-averaged recall value to 49.14%. But, the overall accuracy of the model is 69.75% better than one of the existing work [20].

TABLE IV. KEY PARAMETERS USED FOR THE PERFORMANCE EVALUATION OF THE MODEL

Precision	Precision in case of SWS is defined as the number of relevant services out of the total number of services retrieved by the framework. Mathematical equation to calculate the Precision value is given as follows: $\frac{S_{relevant}}{S_{relevant} + S_{irrelevant}}$ where, $S_{relevant}$ gives the number of relevant services retrieved, and $S_{irrelevant}$ gives the number of irrelevant services retrieved
Recall	Recall is computed by taking the fraction of relevant services retrieved out of the total relevant services given in the relevance file by domain expert. Mathematical equation to calculate the Recall value is given as follows: $\frac{S_{relevant}}{S_{relevant} + S_{relevantNR}}$ where, $S_{relevant}$ gives the number of relevant services retrieved, and $S_{relevantNR}$ gives the number of relevant services not retrieved by the framework
Macro-Average Precision	Macro Average Precision is used to compute the arithmetic mean of the precision of each test query in case of multiclass classification. $\sum_{q=1}^{N} \frac{Precision}{N}$ where, q represents the number of test queries and $Precision_q$ gives the precision value of q^{th} query
Macro-Average Recall	Macro-Average Recall is used for multiclass classification and is computed by taking arithmetic mean of each test query recall value. $\sum_{q=1}^{N} \frac{Recall_q}{N}$ where, q represents the test query and takes values up to N, and $Recall_q$ gives the recall value of q^{th} query
Accuracy	Accuracy provides the overall performance of the model by computing the ratio of correct predictions out of total predictions done by the model.
Average Query Response Time	Average Query Response Time (Avg. Q_{rt}) is computed by taking the average of total response time taken to execute all the test queries.

- Logical Inferencing over KGSWS

The incorporation and interconnection of ontologies within KGSWS not only enhance the outcomes, as discussed in the previous section, but also pave the way for easier reasoning, composition, and classification in the future. Reasoning over KGSWS involves incorporating logically inferred concepts through the "subclass" symmetric relation. For example, when searching for services that provide the price of a given book, services that yield "MaxPrice" or "RecommendedPrice" as output, with "book" as input, are also deemed relevant. This relevance stems from their subclass relationship with the "Price" concept. However, the proposal did not account for the addition of logically inferred subclass concepts from the given concepts, leading to a decrease in the overall accuracy of Method 1.

2) Analysis of Method 2 results: Method 2, characterized by looser restrictions on the semantic match of I/O concepts, yielded contrasting results as shown in Fig 4(c) and Fig 4(d). This enhancement allowed the automatic retrieval of services sharing subclass relationships with some concepts of user concepts thereby capturing more closely related services. This flexibility leads to improved recall values, as Method 2 can identify a broader range of relevant services. This method proved beneficial in the worst cases of Method 1, where it retrieved some relevant services and achieved non-zero precision. However, this came at the trade-off of retrieving some irrelevant services, resulting in a decrease in average precision. This highlights the potential of Method 2 for more comprehensive service discovery, albeit at the expense of precision. Table V demonstrates the macro-averaged results, accuracy and average Q_{rt} of the two methods.

3) Comparison with existing frameworks: Comparing our experimental results with some published results of other existing works on the same dataset (see Table VI), it was observed that our proposed method demonstrated superior performance in terms of average response time and accuracy. Notably, the two-step approach in [28] involving prefiltration and subsequent matchmaking incurred a higher average query response time compared to our approach. This is a significant achievement, considering that our search was conducted over the integrated KGSWS, underlining the efficiency and swiftness of our approach. Moreover, the matchmakers of [20] exhibited longer response times, primarily attributable to the addition of concepts to a new matchmaker ontology for each request. In contrast, our approach, which seamlessly integrates the advantages of in depth querying over Knowledge Graph, overcame this bottleneck, leading to a more streamlined and efficient service discovery process. An important observation in our results was that while some subsequent studies [29-32] explored machine learning-based classification techniques, none had harnessed the potential of the Knowledge Graph within this domain.

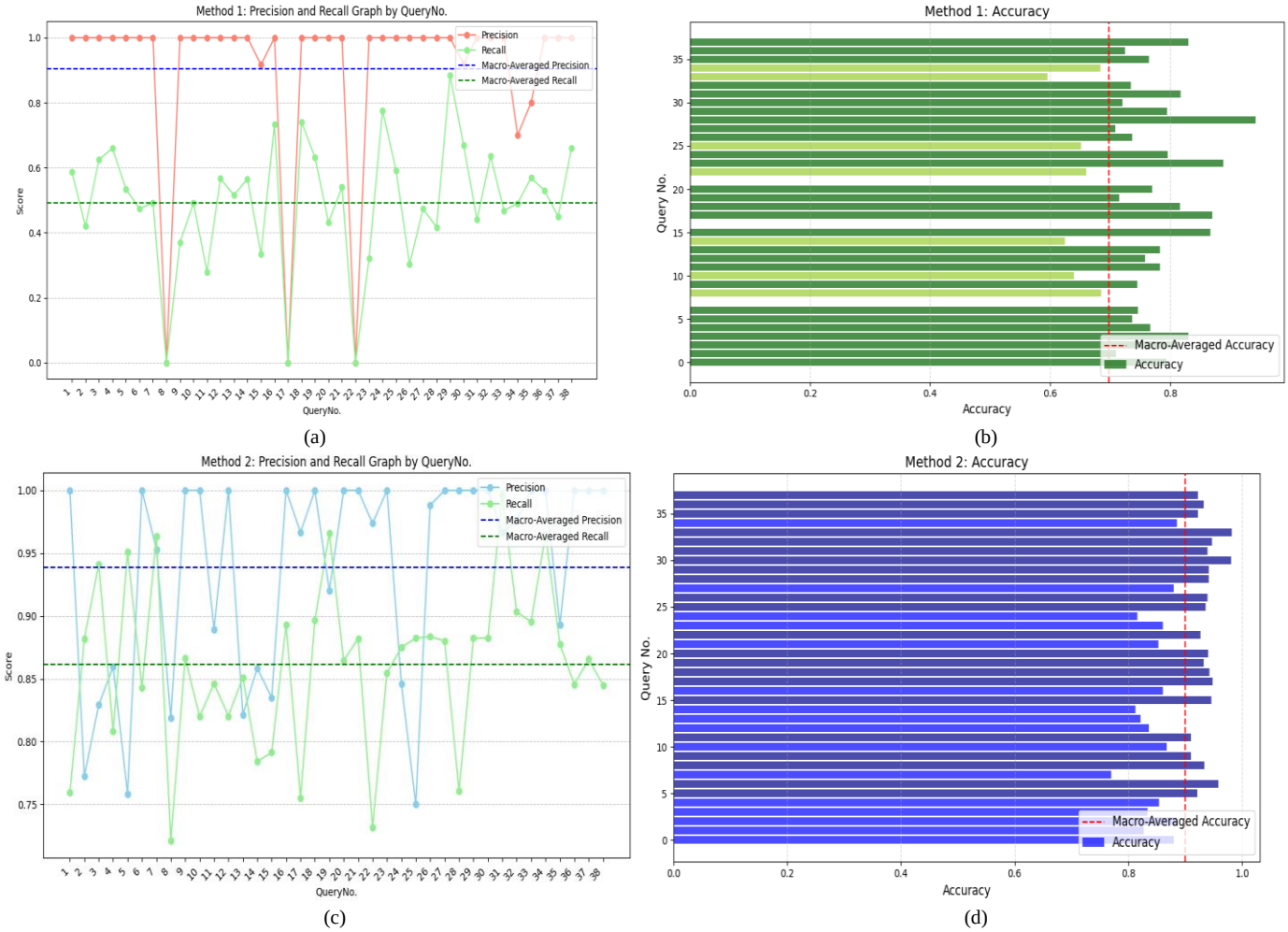


Fig. 4. (a) Macro-averaged precision recall of method 1. (b) Query wise accuracy of method 1 and the dotted redline shows the macro-averaged accuracy of the model. (c) Macro-averaged precision recall of method 2. (d) Query wise accuracy of method 2 and the dotted redline shows the macro-averaged accuracy of the model.

TABLE V. KEY PARAMETERS RESULTS GIVEN BY TWO METHODS "Q_{EACH}" AND "Q_{FEW}"

Key Parameters	Method 1	Method 2
Macro-Averaged Precision	0.9036	0.9388
Macro-Averaged Recall	0.4914	0.8614
Accuracy	0.6975	0.9001
Average Query Response Time (in s)	0.61 s	1.57s

C. Discussion

The proposed work, used the advanced potential of KGs for semantic enrichment and querying, overcomes the limitations of previous approaches by introducing an integrated composite schema known as KGSWS for querying. This schema allows Method 1 and Method 2 to incorporate more relevant services within the same domain using different levels of filtering and regular expressions during the discovery process. The integration of domain ontologies and service descriptions in KGSWS enables the alignment of heterogeneous concepts within the same domain with user-requested concepts, thereby increasing the accuracy of both Method 1 and Method 2. However, as the work does not include logical reasoning over KGSWS, the performance metrics of Method 1 experienced a decline to 69.75% due to the matching of each concept of the user-requested query. Additionally, while Method 2 benefits from the inclusion of more equivalent concepts within the same domain, allowing for the automatic inclusion of more relevant services with loose concept matching using filters and regular expressions, this also entails the inadvertent inclusion of some irrelevant services. Compared to previous approaches, our proposed framework offers a comprehensive querying solution rather than relying on pre-filtering through querying on exiting matchmakers [28], which can lead to increased response times. Furthermore, unlike existing methods [20] that create matchmaking ontologies for discovery, our approach does not require such intermediary steps. For validation purposes, our work utilized the relevance file provided for the OWLS dataset in Phase 3 instead of employing alternative methods [29-33] to find relevant services, thereby enhancing the reliability of the framework. Additionally, we considered the complete OWLS-TC dataset rather than using its subset to generate KGSWS and further for service discovery. The

framework also allows for the automatic generation of queries based on user-requested concepts, offering a generic and streamlined approach to querying.

TABLE VI. COMPARISON OF ACCURACY AND QUERY RESPONSE TIME OF EXISTING FRAMEWORKS WITH THE PROPOSED FRAMEWORK

	Accuracy (%)	Average Q_{rt} (in sec.)
Method 1	69.75	0.61 s
Method 2	**90.01**	**1.57s**
OWLS-M0 [20]	49.55	57.33s
OWLS-MX3 (M3) [20]	82.96	58.46s
SPARQLent [28]	72.02	55.00s
HELSWSR [30]	85.60	-
-[29]	83.09	-
- [32]	89.28	-

VII. CONCLUSION AND FUTURE SCOPE

In this paper, we have presented a framework for the automatic discovery of SWS through the use of SPARQL querying over KG. By introducing the KGSWS framework, a paradigm shift has been made by offering more precise and machine understandable context during automatic matchmaking of services. The integration of domain ontologies in KGSWS introduces a new level of semantic richness that effectively resolved the ambiguity associated with keyword-based matchmaking of WS. In conclusion, our work contributes to the field of SWS discovery by efficiently retrieving the relevant services aligned with the concepts in user request. Our approach also addresses several research questions discussed at the outset of this work.

A. Addressing Research Questions

1) How can ontologies from different domains be interlinked to form an extensive Knowledge Graph enriched with semantic metadata thereby enhancing the discovery of SWS?

Our framework successfully accomplished this by constructing the KGSWS from OWL-S service descriptions and their associated domain ontologies that forms a centralized repository of semantic metadata to enhance the discovery process of services.

2) What methods can be used to formulate the automatic queries on the KG, aligned with varying numbers of inputs-outputs for effective querying purposes?

Our experiments demonstrated the effectiveness of our approach in generating and executing general-purpose queries over KGSWS. The capabilities of two semantic matching methods namely "Q_{each}" and "Q_{few}," have been evaluated in answering user queries based on varying input/output parameters.

3) How can we efficiently retrieve SWS from big KG that precisely matched the user requirements for service discovery? Additionally, how can we identify the closely related services when an exact match is not available, maintaining the integrity of the user's query?

The semantic matching methods provide a practical solution to this challenge. The "Q_{each}" method excels in retrieving precisely matched services whereas the "Q_{few}" method retrieved closely related services when exactly matched services are not available.

B. Future Scope

While our proposed work has achieved promising results, several avenues for improvements and future research exist:

1) Logical inference: In future work, the inclusion of logical inferencing techniques in Q_{each} semantic matching can enhance the macro-averaged recall and thereby accuracy of our framework.

2) Scalability and machine learning based models: As the number of SWS and their associated ontologies continues to grow, scalability remains a critical concern. The incorporation of machine learning based graph embedding models can efficiently handle large KGs and also enable more accurate service recommendations.

VIII. DECLARATIONS

Author contribution P.T and L.S conceived the idea. P.T executed the experiments and wrote the article. L.S did edition and corrections.

Data Availability Not applicable

Code Availability Not applicable

Conflict of Interest the authors declare no competing interests

REFERENCES

[1] N. B. Kurniawan, Y. Bandung, and P. Yustianto, 'Services computing systems engineering framework: a proposition and evaluation through soa principles and analysis model', IEEE Syst. J., vol. 14, no. 3, pp. 3105–3116, 2019.

[2] J. B. Merin and W. A. Banu, 'Social based Web Service Discovery for Multiple Domains and Recommendation', Webology, vol. 19, no. 1, pp. 6396–6407, 2022.

[3] X. Zhang, J. Liu, M. Shi, and B. Cao, 'Word embedding-based Web service representations for classification and clustering', in 2021 IEEE International Conference on Services Computing (SCC), IEEE, 2021, pp. 34–43.

[4] G. Lampropoulos, E. Keramopoulos, and K. Diamantaras, 'Enhancing the functionality of augmented reality using deep learning, semantic web and knowledge graphs: A review', Vis. Informatics, vol. 4, no. 1, pp. 32–42, Mar. 2020, doi: 10.1016/J.VISINF.2020.01.001.

[5] A. Bennaceur and B. Nuseibeh, 'The Many Facets of Mediation: A Requirements-Driven Approach for Trading Off Mediation Solutions', Manag. Trade-offs Adapt. Softw. Archit., pp. 299–322, Jan. 2017, doi: 10.1016/B978-0-12-802855-1.00012-5.

[6] H. Guermah, T. Fissaa, H. Hafiddi, and M. Nassar, 'Exploiting Semantic Web Services in the Development of Context-Aware Systems', Procedia Comput. Sci., vol. 127, pp. 398–407, Jan. 2018, doi: 10.1016/J.PROCS.2018.01.137.

[7] M. Hu and Y. Liu, 'E - maintenance platform design for public infrastructure maintenance based on IFC ontology and Semantic Web services', Concurr. Comput. Pract. Exp., vol. 32, no. 6, p. e5204, 2020.

[8] R. Hammami, H. Bellaaj, and A. H. Kacem, 'Semantic web services discovery: A survey and research challenges', Int. J. Semant. Web Inf. Syst., vol. 14, no. 4, pp. 57–72, 2018, doi: 10.4018/IJSWIS.2018100103.

[9] C. Peng, F. Xia, M. Naseriparsa, and F. Osborne, Knowledge Graphs: Opportunities and Challenges, no. March. Springer Netherlands, 2023. doi: 10.1007/s10462-023-10465-9.

[10] T. Berners-Lee, J. Hendler, and O. Lassila, 'The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities', in Linking the World's Information: Essays on Tim Berners-Lee's Invention of the World Wide Web, 2023, pp. 91–103.

[11] A. Patel and S. Jain, 'Present and future of semantic web technologies: a research statement', Int. J. Comput. Appl., vol. 43, no. 5, pp. 413–422, 2021, doi: 10.1080/1206212X.2019.1570666.

[12] A. Kuzzaman, Metadata format and Standards, Nov. 6, 2018. Accessed on: May 20, 2022. [Online]. Available: http://www.lisbdnet.com/introduction-to-metadata/.

[13] D. Brickley and R. V. Guha, RDF Schema 1.1, W3C, Feb. 25, 2014. Accessed on: July 24, 2022. [Online]. Available: http://www.w3.org/TR/rdf-schema/.

[14] L. Ehrlinger and W. Wöß, 'Towards a definition of knowledge graphs.', Semant. (Posters, Demos, SuCCESS), vol. 48, no. 1–4, p. 2, 2016.

[15] C. Gutierrez and J. F. Sequeda, 'Knowledge graphs', Commun. ACM, vol. 64, no. 3, pp. 96–104, 2021, doi: 10.1145/3418294.

[16] T. Pellissier Tanon, G. Weikum, and F. Suchanek, 'Yago 4: A reason-able knowledge base', in The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17, Springer, 2020, pp. 583–596.

[17] D. Vrandečić, L. Pintscher, and M. Krötzsch, 'Wikidata: The Making Of', in Companion Proceedings of the ACM Web Conference 2023, 2023, pp. 615–624.

[18] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, 'A survey on knowledge graphs: Representation, acquisition, and applications', IEEE Trans. neural networks Learn. Syst., vol. 33, no. 2, pp. 494–514, 2021.

[19] C. B. Aranda, O. Corby, S. Das, L. Feigenbaum, P. Gearon, B. Glimm et al., SPARQL 1.1 Overview, W3C, Mar. 21, 2013. Accessed on: Jan. 20, 2021. [Online]. Available: https://www.w3.org/TR/sparql11-overview/.

[20] M. Klusch, B. Fries and K. Sycara, 'OWLS-MX: a hybrid Semantic Web service match-maker for OWL-S services', Web Semantics, vol. 7, no. 2, pp. 121–133, 2009.

[21] R. Amorim, D. B. Claro, D. Lopes, P. Albers and A. Andrade, 'Improving web service dis-covery by a functional and structural approach', in Proceedings of the IEEE 9th International Conference on Web Services (ICWS'11), pp. 411–418, IEEE, Washington, DC, USA, July 2011.

[22] M. Klusch and F. Kaufer, 'WSMO-MX: a hybrid SemanticWeb service matchmaker', Web Intelligence and Agent Systems, vol. 7, no. 1, pp. 23–42, 2009.

[23] M. Stollberg, M. Hepp and J. Hoffman, 'A caching mechanism for semantic web service discovery', in The Semantic Web, pp. 480-493, Springer, Berlin, Heidelberg, 2007.

[24] M. Stollberg, J. Hoffmann and D. Fensel, 'A caching technique for optimizing automated service discovery', International Journal of Semantic Computing (World Scientific), vol. 5, no. 1, pp. 1–31, 2011.

[25] M. L. Sbodio, D. Martin and C. Moulin, 'Discovering Semantic Web services using SPARQL and intelligent agents', Journal of Web Semantics, vol. 8, no. 4, pp. 310-328, 2010.

[26] T. Khdour, 'Towards semantically filtering web services repository', in International Conference on Digital Information and Communication Technology and Its Applications, vol. 167, pp. 322-336. Springer, Berlin, Heidelberg, 2011.

[27] K. Mohebbi, S. Ibrahim and M. Zamani, 'A pre-matching filter to improve the query response time of semantic web service discovery', Journal of Next Generation Information Technology, vol. 4, no. 6, pp. 9-18, 2013.

[28] J. M. Garc´ıa, D. Ruiz and A. Ruiz-Cort´es, 'Improving semantic web services discovery using SPARQL-based repository filtering', Web Semantics: Science, Services and Agents on the World Wide Web, vol. 17, pp. 12–24, 2012.

[29] N. El Allali, M. Fariss, H. Asaidi, and M. Bellouki, 'Towards Semantic Web Services Density Clustering Technique', in International Conference on Digital Technologies and Applications, Springer, 2021, pp. 543–553.

[30] S. Sagayaraj and M. Santhoshkumar, 'Heterogeneous ensemble learning method for personalized semantic web service recommendation', Int. J. Inf. Technol., vol. 12, no. 3, pp. 983–994, 2020, doi: 10.1007/s41870-020-00479-9.

[31] M. Kaouan, D. Bouchiha, S. M. Benslimane, and S. Boukli-Hacene, 'Towards Service Ontology for Web Services Storage and Discovery', in 2020 4th International Symposium on Informatics and its Applications (ISIA), IEEE, 2020, pp. 1–6.

[32] B. S. Balaji, S. Balakrishnan, K. Venkatachalam, and V. Jeyakrishnan, 'Automated query classification based web service similarity technique using machine learning', J. Ambient Intell. Humaniz. Comput., vol. 12, no. 6, pp. 6169–6180, 2021, doi: 10.1007/s12652-020-02186-6.

[33] L. Guodong, Q. Zhang, Y. Ding, and W. Zhe, 'Research on service discovery methods based on knowledge graph', IEEE Access, vol. 8, pp. 138934–138943, 2020.

[34] T. Yu et al., 'Knowledge graph for TCM health preservation: Design, construction, and applications', Artif. Intell. Med., vol. 77, pp. 48–52, 2017, doi: 10.1016/j.artmed.2017.04.001.

[35] A. B. Kamran, B. Abro, and A. Basharat, 'SemanticHadith: An ontology-driven knowledge graph for the hadith corpus', J. Web Semant., vol. 78, p. 100797, Oct. 2023, doi: 10.1016/J.WEBSEM.2023.100797.

[36] A. Rivas, D. Collarana, M. Torrente, and M.-E. Vidal, 'A neuro-symbolic system over knowledge graphs for link prediction', Semant. Web, no. Preprint, pp. 1–25, 2022.

[37] S. Ravishankar et al., 'A Two-Stage Approach towards Generalization in Knowledge Base Question Answering', Find. Assoc. Comput. Linguist. EMNLP 2022, pp. 5600–5609, 2022.

[38] G. Rossiello et al., 'Generative Relation Linking for Question Answering over Knowledge Bases', Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 12922 LNCS, pp. 321–337, 2021, doi: 10.1007/978-3-030-88361-4_19.

[39] G. Maheshwari, P. Trivedi, D. Lukovnikov, N. Chakraborty, A. Fischer, and J. Lehmann, 'Learning to Rank Query Graphs for Complex Question Answering over Knowledge Graphs', Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 11778 LNCS, pp. 487–504, 2019, doi: 10.1007/978-3-030-30793-6_28.

[40] S. Purkayastha, S. Dana, D. Garg, D. Khandelwal, and G. P. S. Bhargav, 'Knowledge Graph Question Answering via SPARQL Silhouette Generation', 2021, [Online]. Available: http://arxiv.org/abs/2109.09475.

[41] Abu-Salih, Bilal, 'Domain-specific knowledge graphs: A survey', Journal of Network and Computer Applications, vol. 185, 103076, 2021.

[42] Gutierrez, Claudio, and Juan F. Sequeda, 'Knowledge graphs', Communications of the ACM 64, no. 3, pp. 96-104, 2021.

[43] Chen, Xiaojun, Shengbin Jia, and Yang Xiang, 'A review: Knowledge reasoning over knowledge graph', Expert Systems with Applications 141, 112948, 2020.

[44] OWLS-TC version 4.0, Semantic Web Central, Sep. 21 2010, Accessed on: Aug 19, 2020. [Online]. Available: http://projects.semwebcentral.org/projects/owls-tc/.

[45] M. Sokolova and G. Lapalme, 'A systematic analysis of performance measures for classification tasks', Inf. Process. Manag., vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/J.IPM.2009.03.002.