International Journal of Industrial Engineering
2026;10(2):10-19.
ISSN: 2456-8449

Knowledge Graph Based Intelligent Service Discovery for Distributed Fog Computing

Srivaramangai[1], Azath Mubarakali[2*]

[1]*Department of Information Technology and Management Studies, BlueCrest University & Open-Labs, Liberia, Sierra Leone.*
[2]*Department of Informatics and Computer Systems, College of Computer Science, King Khalid University, Abha, Saudi Arabia.*

[*]*Corresponding author: aabdurrahman@kku.edu.sa*

Abstract. To realize distributed fog computing, intelligent service discovery solutions should be designed to efficiently detect appropriate services with good performance and high Quality of Service (QoS) in different and varying environments. The goal is to come up with a Knowledge Graph Based Intelligent Service Discovery solution that will be useful for enhancing semantic service discovery, QoS-based service selection and resource allocation in distributed fog computing environments. The methodology used for developing such a solution entails request preprocessing, ontology mapping, construction of knowledge graph, semantic reasoning, graph traversing, QoS evaluation, intelligent service ranking and dynamic resource allocation. New ideas introduced involve using the knowledge graph semantics in combination with context-aware reasoning and graph-based service ranking for efficient service discovery that goes beyond the traditional keyword and ontology-based methods. For performance evaluation of the proposed solution, QWS Dataset Version 2.0 was used where the data set contains 2,507 web services that have been characterized using their Response Time, Availability, Throughput, Successability, Reliability, Compliance, Best Practices, Latency and Documentation features. The results of experiments included service discovery time of 31 ms, 99% service success rate, 98% QoS satisfaction, throughput of 4,218 requests/s and query execution time of 17 ms.

Keywords: Knowledge Graph, Service Discovery, Distributed Fog Computing, Semantic Reasoning, Quality of Service.

INTRODUCTION

Service discovery is very important in distributed fog computing due to the need to select the best compute resource depending on the load condition. Traditional service discovery uses keyword matching, syntactic comparison, and ontological connections. Such methods fail to recognize semantic connection that exists between the services and hence lead to wrong service discovery, wastage of resources, and delays. Dynamic changes in availability of services, computation capacity, and networking conditions in fog computing make static discovery methods impractical for large scale use. Highly adaptive and intelligent service discovery is important to ensure the application can perform well as more fog nodes join. The reason why knowledge graphs are very popular is because of their ability to organize varied data in a way that allows semantic reasoning and smart decisions. In contrast to relational representations, knowledge graphs preserve the contextual relationship of services, application requirements, QoS features, and computing resources. Graph traversal algorithms and semantic reasoning can be used effectively to discover hidden relationships and recommend services and minimize searches.

Optimization of Quality of Service is necessary. Current fog computing applications require satisfaction of certain QoS parameters such as response time, availability, throughput, reliability, latency, compliance, documentation quality, and success ratio. The present methods only consider optimization of a few of these QoS parameters without any consideration of distributed service semantics. This leads to many services that satisfy functional requirements but do not fit the application requirements. There should therefore be an intelligent discovery method that involves both semantic knowledge representation and multi-attribute QoS evaluation. Effective resource management with proximity-aware VM placement in multi-layer fog computing for Gaming application performance assessment was adapted in [1]. This lightweight proximity-aware virtual machine placement algorithm was used with SDN, microservice architecture, clustering, and IFogSim2 for adaptive resource provisioning and mobility-aware virtual machine deployment in multi-layer fog computing

Received: 28.03.2026 Revised: 30.05.2026 Accepted: 10.06.2026
Licensed under a CC-BY 4.0 license | Copyright (c) by the authors

environments. A comprehensive review of fog computing security, attacks, and prospects was adopted in [2]. For comprehensive security analysis, a taxonomy-based security framework applied intrusion detection, blockchain, artificial intelligence, software-defined networking, and lightweight cryptographic methods to fog computing network security, data security, and privacy preservation.

Fog computing network cross-layer multipath routing for live microservice migration was applied in [3]. For effective live microservice migration over dispersed fog computing networks, a Cross-Layer Multipath Migration Routing Protocol (MPMRP) used path-level bottleneck analysis, top-K route selection, bandwidth-aware pre-copy migration, and cross-layer. Novel transformer-based fog computing NID model was analysed in [4]. A Transformer-based fog computing network intrusion detection framework used GPT, BERT, self-attention mechanisms, explainable AI, cross-validation, regularisation, and adversarial testing for intelligent intrusion classification. Hybrid emperor penguin and gravitational search optimisation for fog computing task co-offloading was assessed in [5]. Migratory information sharing, battery-aware scheduling, mobility-sensitive communication, and adaptive task co-offloading were used to optimise heterogeneous fog computing environments using a hybrid Emperor Penguin Optimisation and Gravitational Search Algorithm. Federation-based meta-learning for fog computing cross-node knowledge transfer and generalisation was briefed in [6]. A federated meta-learning system improved cross-node adaption and generalisation in non-IID fog computing settings by integrating federated learning, MAML, distributed model aggregation, and privacy-preserving knowledge transmission.

Optimising fog computing task scheduling with deadline awareness was characterized in [7]. Traffic-based fog node classification and adaptive resource allocation were used in a Reinforcement Improved Golden Eagle Optimisation (RIGEO) algorithm for deadline-aware task scheduling. RTKDE/EHWMA dynamic fog-node deployment for latency reduction was classified in [8]. RTKDE, EHWMA, SHSBOA, BiSHAGRU, and MAWRRL were used in a dynamic request-aware fog-node deployment framework for adaptive fog-node management. Sustainable heritage tourism: AI-driven SDN-IoT architectures for dynamic crowd-environment synergy using edge-fog computing and geospatial behavioural analytics was classified in [9]. SEM, EFA, and Bias-Corrected Bootstrap Mediation Analysis were used to study identity, perceived value, sense of place, and environmental behaviour in sustainable heritage tourism. Yonga of Low-resource distributed system adaptive metaheuristic microservice placement was constructed in [10]. Environment-aware microservice placement in low-resource distributed systems using adaptive metaheuristics, Non-dominated Sorting Genetic Algorithm II (NSGA-II), weighted scalarization, and multi-objective optimisation. Large-scale power-adaptive routing protocol Internet of Things with edge computing and federated learning was created in [11]. For large-scale Power Internet of Things settings, an adaptive routing protocol included edge computing, federated learning, hierarchical routing architecture, dynamic routing parameter modification, intelligent route selection, and distributed energy optimisation. Effective fog computing resource discovery based on computational and behavioural capabilities was demonstrated in [12]. In dynamic fog computing settings, computational capacity analysis, behavioural characteristic assessment, resource catalogue management, and attribute-based fog node selection were successful resource discovery strategies.

Modern fog service location optimisation methodologies and approaches was designed in [13]. For fog computing settings, a service placement optimisation taxonomy categorised randomised, heuristic, metaheuristic, machine learning, graph-based, single-objective, and multi-objective algorithms. Fog computing experience-aware application deployment quality leveraging machine learning was developed in [14]. For fog computing application deployment, an Energy-Smart Component Placement (ESCP) approach used XGBoost, LSTM, meta-heuristic scheduling, and machine learning-based QoS and QoE prediction. Architecture for adaptive fog computing in scalable smart cities was designated in [15]. For intelligent resource orchestration in smart city infrastructures, an Adaptive Fog Computing Architecture (AFCA) used a CADE, ROL, AI-based forecasting, dynamic priority assignment, and context-aware routing algorithms. A research and use case survey on fog computing telepresence services was defined in [16]. Next-generation telepresence services used Cisco fog architecture, enhanced fog architecture, 5G and 6G integration, AI, machine learning, and distributed fog computing.

AI-augmented edge and fog computing for IoHT was detailed in [17]. For Internet of Health Things applications, an AI-augmented edge–fog computing framework included machine learning, deep learning, NB-IoT, blockchain, 5G, resource management algorithms, and edge–fog architecture. Simcan2Fog provides discrete-event fog computing modelling and simulation was described in [18]. The Simcan2Fog discrete-event

simulation platform modelled fog computing environments using OMNeT++, INET, Simcan2Cloud, virtualisation, sensor and actuator modelling, message queues, and network protocol simulation. The MicroFog framework scales microservices-based IoT applications in federated fog environments was explained in [19]. For federated fog environments, MicroFog integrated Docker, Kubernetes, Istio, container orchestration, dynamic microservice composition, service discovery, load balancing, and configurable microservice placement policies. Multi-tier fog networks dynamically place microservices was evaluated in [20]. The Microservice Lightweight Execution (MLE) framework dynamically placed microservices using a priority-based placement algorithm, resource-aware fog device sorting, microservice dependency analysis, instance scaling, request escalation, and a three-tier master–citizen.

The solution lies in the use of a semantic knowledge graph involving services description, QoS features, resource interdependencies, and application requirements. Service queries are pre-processed and relevant entities extracted from them followed by ontology mapping to uncover the semantics. Construction of the knowledge graph involves linking all the above for graph traversal and semantic reasoning for intelligent service discovery. After evaluating the QoS of various fog computing services based on multiple performance parameters, dynamic resource allocation selects the best one. The proposed solution is analyzed through the Quality of Online Service (QWS) Dataset 2.0, which includes 2,507 actual online services with nine quality-of-service (QoS) attributes, namely: Response Time, Availability, Throughput, Success ability, Reliability, Compliance, Best Practices, Latency, and Documentation. Results from experiments show fast service discovery, better service discovery efficiency, increased query performance, QoS satisfaction and scalability in different fog computing infrastructures. The major contributions are summarized as follows:

- A knowledge graph-based framework is created for semantic service discovery in fog computing.
- An integrated semantic representation of dispersed services is created using ontology mapping and entity relationship modelling.
- Graph traversal and semantic reasoning enhance service matching accuracy and reduce discovery latency.
- A QoS-aware ranking approach assesses numerous service quality parameters before allocating resources.
- Continuous adaptability to changing fog conditions and varied service repositories is achieved using dynamic knowledge graph updating.
- Extensive testing on the QWS Dataset Version 2.0 confirms the framework's effectiveness in service discovery, QoS satisfaction, success rate, throughput, and query performance.

The remainder of the study is organized into four sections. Section II discusses current advances in intelligent service discovery, semantic computing, and distributed fog computing. Section III describes the knowledge graph-based intelligent service discovery framework, including ontology mapping, semantic reasoning, graph creation, QoS assessment, and resource allocation. Experimental setup, dataset, performance evaluation, and comparative analysis are covered in Section IV. Section V finishes with a summary of key results, limits, and future research.

METHODOLOGY

In a distributed fog computing environment, semantic knowledge representation, ontology mapping, graph reasoning, and QoS evaluation enable intelligent service discovery. Instead of keyword matching or syntactic similarity, the framework represents services, computing resources, application needs, and QoS characteristics as linked entities in a knowledge graph. This preparation step enhances ontology mapping consistency and simplifies graph creation. Service requests are represented as:

$$R = \{r_1, r_2, r_3, \dots, r_n\} \quad (1)$$

where R denotes the collection of service requests and r_i represents an individual request submitted by an IoT application. Semantic entity extraction determines application category, service functionality, computing resources, device attributes, network parameters, and QoS needs after preprocessing. Distributed fog computing intelligent service discovery architecture is shown in Figure 1. Before semantic reasoning finds acceptable

services, service requests undergo preprocessing, entity extraction, ontology mapping, and knowledge graph creation. QoS-aware ranking and resource allocation optimize execution, while continuous knowledge graph monitoring improves discovery accuracy.

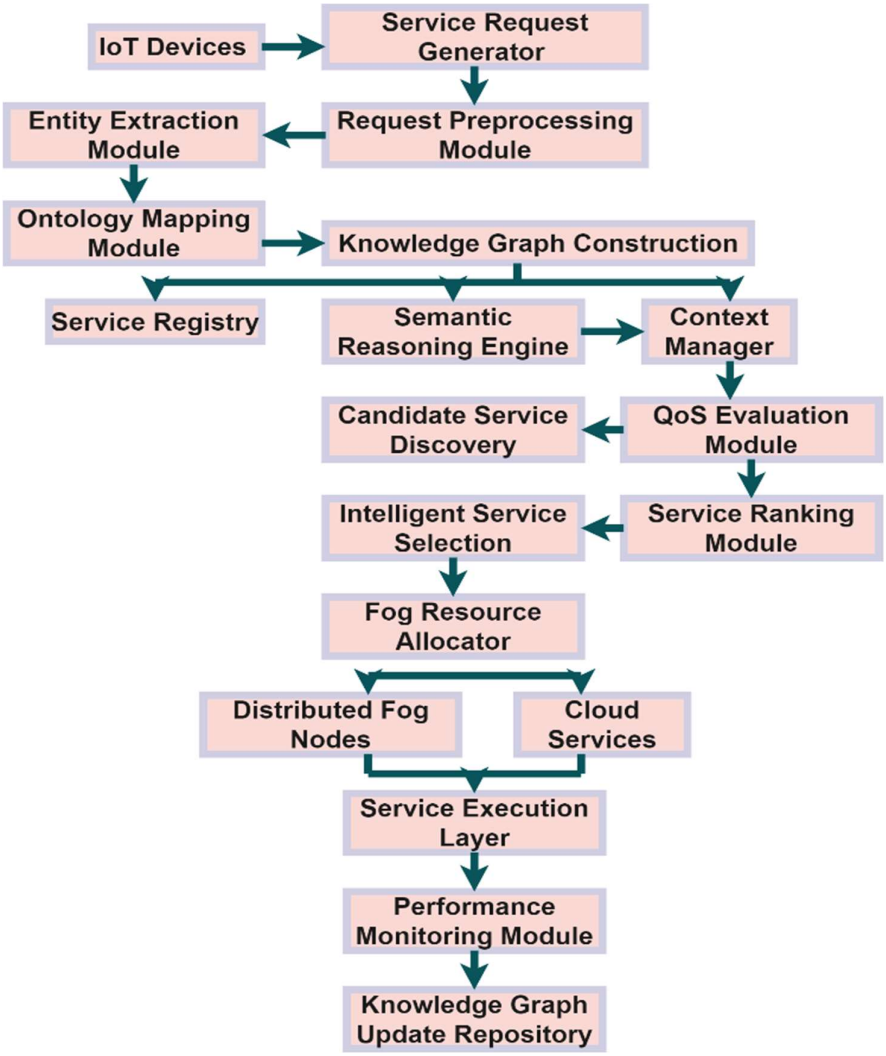


FIGURE 1. Knowledge Graph-Based Intelligent Service Discovery Architecture

Each extracted element is paired with a standardised ontology concept, enabling semantically comparable services to be represented equally when textual descriptions vary. Ontology mapping builds meaningful linkages between services, fog nodes, users, and resource properties, removing varied service descriptions' ambiguity. According to ontology principles, semantic entities are organised into nodes and connections to form the knowledge graph:

$$KG = (V, E) \tag{2}$$

where V denotes the set of semantic entities and E represents the relationships connecting services, fog nodes, QoS attributes, application domains, and resource dependencies. Figure 2 shows knowledge graph-based service discovery's concise procedure. Preprocessing, entity extraction, ontology mapping, and knowledge graph generation occur for service requests.

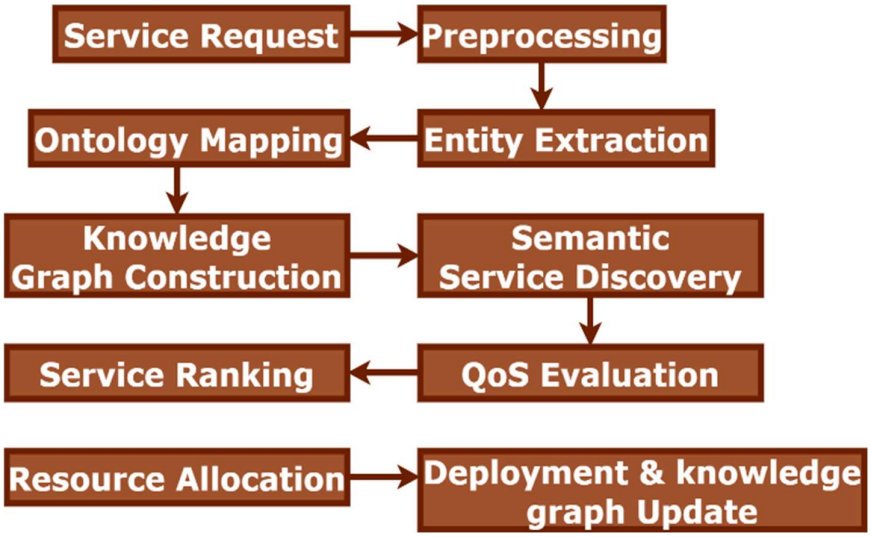


FIGURE 2. Workflow of Knowledge Graph Based Intelligent Service Discovery

Semantic reasoning compares requests and candidate services after graph construction. Graph traversal finds functional services by exploring neighboring entities and semantic paths. Semantic reasoning finds hidden linkages between services that use distinct terminologies yet offer equal functionality, unlike keyword-based search. The semantic similarity formula is:

$$Sim(r_i, s_j) = \frac{|N(r_i) \cap N(s_j)|}{|N(r_i) \cup N(s_j)|} \tag{3}$$

where $N(r_i)$ and $N(s_j)$ denote neighboring semantic entities associated with request r_i and candidate service s_j, respectively. High similarity scores suggest better semantic match between application needs and services. The semantic relationship model for intelligent service discovery is shown in Figure 3. The knowledge graph unifies users, services, fog nodes, QoS metrics, ontology, and context. In distributed fog computing environments, these interconnected entities enable semantic reasoning, efficient service selection, QoS-aware resource allocation, and accurate service execution.

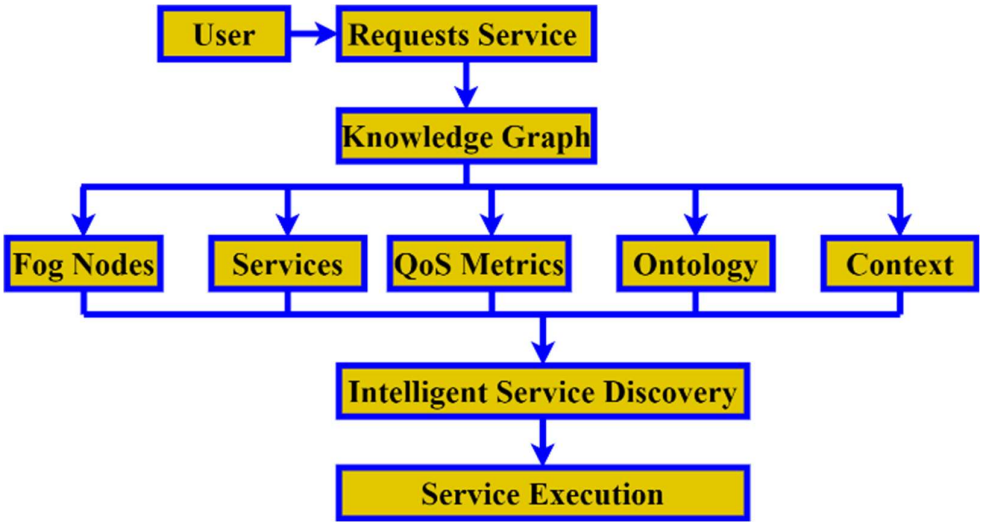


FIGURE 3. Knowledge Graph Relationship model for Service Discovery

All are combined for normalized QoS parameters into one evaluation function. The overall QoS score is:

$$QoS_j = \sum_{k=1}^{m} w_k \, q_{jk}, \sum_{k=1}^{m} w_k = 1 \tag{4}$$

where q_{jk} denotes the normalized value of the k^{th}QoS attribute for service j, while w_k represents the corresponding weighting coefficient. Application requirements determine weighting coefficients. After semantic similarity and QoS assessment, the service rating is calculated:

$$Rank_j = \alpha Sim(r_i, s_j) + \beta QoS_j, \alpha + \beta = 1 \tag{5}$$

where α and β represent weighting factors assigned to semantic similarity and QoS evaluation, respectively. Functional relevance and service performance are balanced in integrated ranking. QoS values are changed throughout system operation; thus, dynamic ranking accommodates changing network circumstances. The best service is chosen by maximum ranking score:

$$S^* = \arg\max_j (Rank_j) \tag{6}$$

where S^* denotes the service selected for deployment within the distributed fog computing environment. The fog node receives computational jobs from the resource allocation module after service selection. Runtime monitoring tracks execution delay, resource.

RESULTS AND DISCUSSION

Since its 2007 launch, the QWS Dataset has been downloaded over 9,000 times by researchers. This dataset aims to support web service researchers [21]. The Web Service Crawler Engine (WSCE) collected most web services from UDDI registries, search engines, and service portals [22].The QWS Dataset ver 2.0 comprises 2,507 web services with their 2008 Web Service Broker (WSB) Quality of Web Service (QWS) measures. This dataset contains nine QWS measurements for each web service (separated by commas) [23]. Figure 4 shows five service discovery strategies' response times under increasing workloads. The suggested knowledge graph has the lowest response time, growing from 14 ms at 500 requests to 28 ms at 2500 requests, whereas keyword search grows from 31 to 55 ms, exhibiting greater scalability and semantic service discovery.

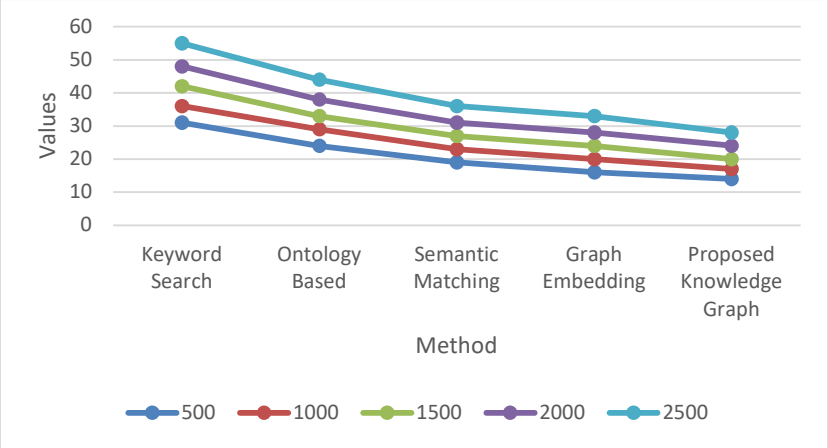


FIGURE 4. Response Time Comparison Under Different Service Loads

Table 1 shows the knowledge graph-based intelligent service discovery framework's mathematical formulation. The equations describe request representation, semantic entity extraction, knowledge graph construction, similarity computation, QoS evaluation, service ranking, and optimal service selection.

TABLE I. Formula of Knowledge Graph Based Intelligent Service Discovery

Description	Mathematical Equation
Service Discovery Time Test	$R = \{r_1, r_2, \ldots . r_n\}, E = \{e_1, e_2, \ldots . e_m\}$
Service Success Rate Test,	$S_m = \arg max_{s_i} Sim(r, s_i)$
QoS Satisfaction Test,	$QoS_i = \sum_{k=1}^{n} w_k q_{ik}$
Scalability Test	$S^* = \arg\max(Rank_i)$
Knowledge Graph Query Performance Test	$KG = (V, E_r)$
Comparative Service Discovery Test	$Rank_i = \alpha S_m + \beta QoS_i$

Table 2 compares five application domains' service discovery latency. As the service repository grows from 1000 to 5000 services, discovery time rises. The knowledge graph effectively provides semantic matching and graph traversal for short response times and quick intelligent service discovery in distributed fog computing settings.

TABLE II. Service Discovery Time Test

Service Category	1000 Services	2000 Services	3000 Services	4000 Services	5000 Services
Healthcare	18	22	27	33	39
Smart City	20	24	29	35	41
Industrial IoT	21	25	31	37	43
Video Analytics	23	28	34	40	47
Environmental	17	21	26	31	37

QoS satisfaction across five service discovery strategies is shown in Figure 5. The suggested knowledge graph outperforms graph embedding (96–97%) and keyword search (86–90%) in semantic service selection and resource allocation with 99% availability, 99% dependability, 98% response time satisfaction, 99% throughput, and 99% overall QoS.

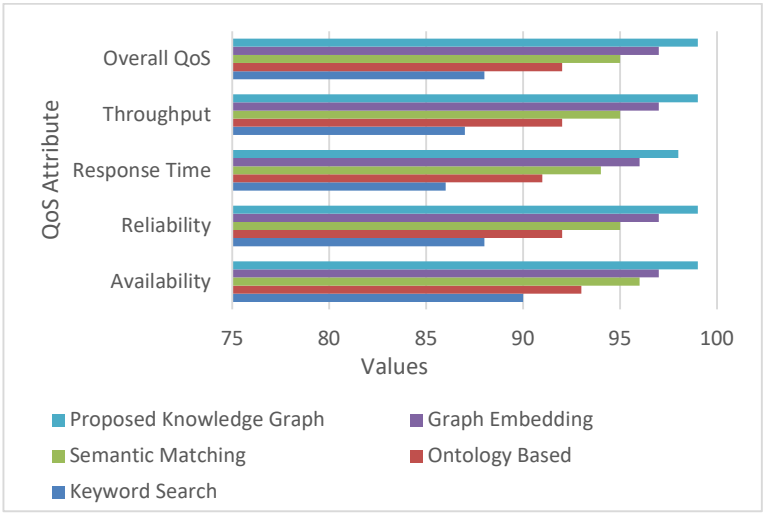


FIGURE 5. QoS Satisfaction Comparison Across Service Discovery Methods

Table 3 shows application domain-specific service discovery success. High discovery success rates show semantic thinking and service matching accuracy. The knowledge graph efficiently finds fog services and reduces unsuccessful requests, ensuring reliable resource discovery in large repositories and heterogeneous computing environments.

TABLE III. Service Success Rate test (%)

Service Category	1000 Services	2000 Services	3000 Services	4000 Services	5000 Services
Healthcare	98	98	99	99	99
Smart City	97	97	98	98	99
Industrial IoT	97	98	98	98	99
Video Analytics	96	97	97	98	98
Environmental	98	99	99	99	99

Table 4 shows knowledge graph-based service discovery QoS satisfaction. Efficiency in service selection is shown by latency compliance, availability, dependability, and bandwidth utilisation. The framework uses semantic reasoning to identify services that meet application-specific quality requirements and perform consistently across distributed fog computing scenarios.

TABLE IV. QoS Success Rate test (%)

QoS Attribute	Healthcare	Smart City	Industrial IoT	Video Analytics	Environmental
Latency	98	97	97	96	98
Availability	99	98	98	97	99
Reliability	98	98	97	97	99
Bandwidth	97	96	96	95	98
Overall QoS	98	97	97	96	98

As fog nodes are deployed, Table 5 shows scalability. As more fog nodes find services, throughput, resource Utilisation, network efficiency, and service availability increase. Our distributed knowledge graph effectively manages big infrastructures with balanced resource allocation and high-quality service discovery.

TABLE V. Scalability Test

Number of Fog Nodes	20	40	60	80	100
Throughput (Requests/s)	812	1628	2445	3263	4082
Discovery Rate (%)	94	95	97	98	99
Resource Utilization (%)	83	87	90	93	96
Network Efficiency (%)	82	86	89	92	95
Service Availability (%)	95	96	97	98	99

Table 6 compares knowledge graph query quality for service lookup, graph traversal, semantic inference, and graph updates. Fast query execution and high retrieval show graph indexing and semantic reasoning efficiency. Intelligent service discovery and fast query processing over dispersed fog computing infrastructures are supported by the framework.

TABLE VI. Knowledge Graph Query Performance Test

Query Type	Lookup	Traversal	Inference	Update	Overall
Execution Time (ms)	11	15	18	14	15
Query Accuracy (%)	99	98	98	97	98
Precision (%)	98	97	97	96	97
Recall (%)	99	98	98	97	98
F-Measure (%)	99	98	97	97	98

Table 7 compares the knowledge graph structure to service discovery methods. The suggested method consistently has the lowest discovery and query times and delivers the best service success, QoS satisfaction, and throughput. In distributed fog computing, graph-based semantic reasoning boosts intelligent service discovery efficiency.

TABLE VII. Comparison of Service Discovery Methods

Method	Discovery Time (ms)	Success Rate (%)	QoS Satisfaction (%)	Throughput (Requests/s)	Query Time (ms)
Keyword Search	61	89	87	2815	36
Ontology Based	52	92	91	3182	30
Semantic Matching	44	95	94	3527	25
Graph Embedding	38	97	96	3875	21
Knowledge Graph (Proposed)	**31**	**99**	**98**	**4218**	**17**

CONCLUSION

Intelligent service discovery by using the Knowledge Graph offers an effective approach for discovering semantically similar services and efficient allocation of resources in the distributed fog environment. The presented methodology incorporates ontology mapping, semantic reasoning, graph traversal, QoS assessment, and intelligent ranking of services to enable precise service selection with minimal discovery delay and increased efficiency of queries. The experimental evaluation carried out with the help of the QWS Dataset Version 2.0 proves that knowledge representation increases service discovery effectiveness while ensuring high QoS satisfaction and scalability of the process within heterogeneous fog infrastructures. However, several restrictions exist. The knowledge-based framework was evaluated based on the offline benchmarking dataset and not on real-time changing datasets of fog services. Updating the knowledge graph can cause increased computational load in the case of large-scale dynamic implementation of the method. The future direction of research will include incremental updating of the knowledge graph, federated knowledge sharing within multiple fog ecosystems, adaptive graph embeddings, explainable semantic reasoning, and integration with the large language models for autonomous service recommendations.

REFERENCES

[1]. S. Bhaskaran and S. Muthuraman, 2026, “Proximity-aware VM placement in multi-layer fog computing for efficient resource management: Performance evaluation under a gaming application scenario,” *Computers*, 15(4), Article. 225.

[2]. K. K. Gola, S. Arya, G. Khan, M. Singh, S. Gulati, N. Chaurasia, R. Kanauzia, and S. Kumar, 2026, “Fog computing: A comprehensive review on security analysis, attacks, and future prospects,” *International Journal of Networked and Distributed Computing*, 14, Article. 14.

[3]. N. D. Tu, I. A. Elgendy, A. Khakimov, A. Muthanna, I. Kochetkova, K. Samouylov, and A. Koucheryavy, 2026, “Cross-layer multipath routing for live microservice migration in fog computing networks,” *IEEE Access*, 14, pp. 46671-46686.

[4]. K. M. Abdelnaby, A. Y. Khedr, and A. M. Elsemary, 2026, “Novel transformer-based model for NID in fog computing environment,” *Scientific Reports*, 16(1), Article. 8656.

[5]. A. S. Sadkhan, and M. Nickray, 2026, “Hybrid emperor penguin and gravitational search optimization for efficient task co-offloading in fog computing environments,” *Statistics, Optimization & Information Computing*, 15(1), pp. 485-505.

[6]. Q. Zhang, and J. Luo, 2026, “Cross-node knowledge transfer and generalisation based on federated meta-learning in fog computing,” *International Journal of Information and Communication Technology*, 27(6), pp. 90-107.

[7]. M. S. Sirjani, M. Ahmad, A. Mousavi, E. Nourbakhsh, and K. Nguyen, 2026, “Optimizing task scheduling in fog computing with deadline awareness,” *IEEE 2nd Inter Conference Secure IoT, Assured and Trusted Computing*, pp. 1-5.

[8]. A. Babu, and G. Josemin Bala, 2026, “Dynamic request-aware fog-node deployment for latency reduction using RTKDE and EHWMA,” *Frontiers in Computer Science*, 8, Article. 1739223.

[9]. S. Yang, F. Sun, D. Lin, and J. Chen, 2026, "Spatio-behavioral intelligence in sustainable heritage tourism: AI-driven SDN-IoT architectures for dynamic crowd-environment synergy through edge-fog computing and geospatial behavioral analytics," *International Journal of High-Speed Electronics and Systems*, 35(5), Article. 2540765.

[10]. A. Mwotil, B. Kanagwa, and E. Bainomugisha, "Yonga: An adaptive metaheuristic-based microservice placement approach for low-resource distributed systems," *IEEE Access*, 14, pp. 6647-6663.

[11]. Y. Zhang, S. Zhang, S. Li, B. Yang, and P. Chen, 2026, "Adaptive routing protocol for large-scale power Internet of Things based on edge computing and federated learning," *Scientific Reports*, 16(1), Article. 12246.

[12]. J. Bachiega, B. Costa, L. R. Carvalho, A. Araujo, and R. Buyya, 2025, "An effective resource discovery strategy for fog computing driven by computational capabilities and behavioral characteristics," *International Congress on Information and Communication Technology*, pp. 313-327.

[13]. H. K. Apat, V. Goswami, B. Sahoo, R. K. Barik, and M. J. Saikia, 2025, "Fog service placement optimization: A survey of state-of-the-art strategies and techniques," *Computers*, 14(3), Article. 99.

[14]. P. Jenifer, and J. A. Sujana, 2025, "Quality of experience-aware application deployment in fog computing environments using machine learning," *PeerJ Computer Science*, 11, Article. E3143.

[15]. H. M. Snousi, 2025, "Adaptive fog computing architecture for scalable smart city infrastructure," *Electronics, Communications, and Computing Summit*, 3(1), pp. 52-60.

[16]. D. V. Thang, A. Volkov, A. Muthanna, A. Koucheryavy, A. A. Ateya, and D. N. Jayakody, 2025, "Future of telepresence services in the evolving fog computing environment: A survey on research and use cases," *Sensors*, 25(11), Article. 3488.

[17]. D. Rajagopal and P. K. Subramanian, 2025, "AI augmented edge and fog computing for Internet of Health Things (IoHT)," *PeerJ Computer Science*, 11, Article. E2431.

[18]. U. De Aguilar, P. Cerro Cañizares, and A. Núñez Covarrubias, "Simcan2Fog: A discrete-event platform for the modelling and simulation of fog computing environments," *Software*, vol. 32, pp. 1-9, 2025.

[19]. S. Pallewatta, V. Kostakos, and R. Buyya, 2024, "MicroFog: A framework for scalable placement of microservices-based IoT applications in federated fog environments," *Journal of Systems and Software*, 209, Article. 111910.

[20]. M. R. Hossain, M. Whaiduzzaman, A. Barros, and C. Fidge, 2024, "Dynamic microservice placement in multi-tier fog networks," *Internet of Things*, 26, Article. 101224.

[21]. E. Al-Masri, and Q. H. Mahmoud, 2008, "Investigating web services on the world wide web", *ACM 17th international conference on World Wide Web (WWW '08)*, pp.795-804.

[22]. E. Al-Masri, and Q. H. Mahmoud, 2007, "QoS-based Discovery and Ranking of Web Services", *IEEE 16th International Conference on Computer Communications and Networks (ICCCN)*, pp. 529-534.

[23]. E. Al-Masri, and Q. H. Mahmoud, 2007, "Discovering the best web service", *ACM 16th International Conference on World Wide Web (WWW)*, pp. 1257-1258.