Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2026.075613

ARTICLE

Lightweight Ontology Architecture for QoS Aware Service Discovery and Semantic CoAP Data Management in Heterogeneous IoT Environment

Suman Sukhavasi, Thinagaran Perumal*, Norwati Mustapha and Razali Yaakob

Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang, Malaysia

*Corresponding Author: Thinagaran Perumal. Email: thinagaran@upm.edu.my

Received: 04 November 2025; Accepted: 31 December 2025; Published: 12 March 2026

ABSTRACT: The Internet of Things (IoT) ecosystem is inherently heterogeneous, comprising diverse devices that must interoperate seamlessly to enable federated message and data exchange. However, as the number of service requests grows, existing approaches suffer from increased discovery time and degraded Quality of Service (QoS). Moreover, the massive data generated by heterogeneous IoT devices often contain redundancy and noise, posing challenges to efficient data management. To address these issues, this paper proposes a lightweight ontology-based architecture that enhances service discovery and QoS-aware semantic data management. The architecture employs Modified-Ordered Points to Identify the Clustering Structure (M-OPTICS) to cluster and eliminate redundant IoT data. The clustered data are then modelled into a lightweight ontology, enabling semantic relationship inference and rule generation through an embedded inference engine. User requests, transmitted via the Constrained Application Protocol (CoAP), are semantically enriched and matched to QoS parameters using Dynamic Shannon Entropy optimized with the Salp Swarm Algorithm. Semantic matching is further refined using a bidirectional recurrent neural network (Bi-RNN), while a State–Action–Reward–State–Action (SARSA) reinforcement learning model dynamically defines and updates semantic rules to retrieve the most recent and relevant data across heterogeneous devices. Experimental results demonstrate that the proposed architecture outperforms existing methods in terms of response time, service delay, execution time, precision, recall, and F-score under varying CoAP request loads and communication overheads. The results confirm the effectiveness of the proposed lightweight ontology architecture for service discovery and data management in heterogeneous IoT environments.

KEYWORDS: CoAP protocol; Internet of Things; interoperability; lightweight ontology; service discovery

1 Introduction

Internet of Things (IoT) is an environment that is spread over many applications like healthcare, agriculture, and so on, which collects different types of data and transfers it via multiple network devices to reach the end server. Hence, interoperability is a challenging task in this environment [1–3]. The IoT interoperability is to design a system that can transfer services with other devices that are different in its platform of operation. Due to the link of multiple platforms, it is an emerging challenge in the IoT ecosystem [4–6]. Therefore, the IoT generates a variety of data, hence semantics play a key role [7–9]. The heterogeneous data generated by IoT devices is typically managed through a centralized platform equipped with processing units to handle large data volumes. Consequently, the IoT system must be adaptable to support various processing tasks, including authentication, service discovery, data security, and other critical operations.

The discovery of services is growing in an IoT environment to achieve interoperability, where it aids in a variety of applications [10,11]. To facilitate service discovery, ontology is employed as a core tool that semantically represents and describes services. The Web Ontology Language (OWL) model is utilized to process and manage metadata, enabling the extraction of the most relevant services for user requests. Service discovery, in this context, involves identifying and retrieving the best-matching service instance from the storage or knowledge base. Ontology serves as a promising solution capable of inferring relationships based on the semantic meanings embedded in the data. In heterogeneous IoT environments, an inference engine plays a crucial role in the service discovery process [12–15]. This engine interprets and matches incoming user requests with the appropriate service descriptions within the ontology or other databases. Ontological identifiers, generated from semantic relationships, enable accurate and context-aware service matching. However, interoperability remains a significant challenge in IoT systems, which must integrate and process diverse data types originating from heterogeneous devices. This complexity often hampers the achievement of high Quality of Service (QoS) in service discovery. The primary challenges addressed in this paper are to facilitate efficient management of heterogeneous IoT data as well as ensure QoS-aware service discovery.

In the process of service discovery in IoT, the main requirements that should be aware of before designing a service discovery in IoT are: (1) The services extracted for the submitted user request must be rich in semantic content that is given in the request packet. (2) Enable the perform of dynamic matching of services based on the description that needs to be flexible for heterogeneous service discovery. (3) Discovery of services for *n* number of users should be at less than in latency, which is one of the constraints in QoS. (4) The complexity in the system must be tolerable by the system, even though the data from IoT devices is in huge amounts. (5) The interoperability between low-power devices and back-end servers needs to be flexible without any lag in the connectivity, and hence it needs to retrieve services from the server. The process handled in service discovery on IoT is depicted in Fig. 1.

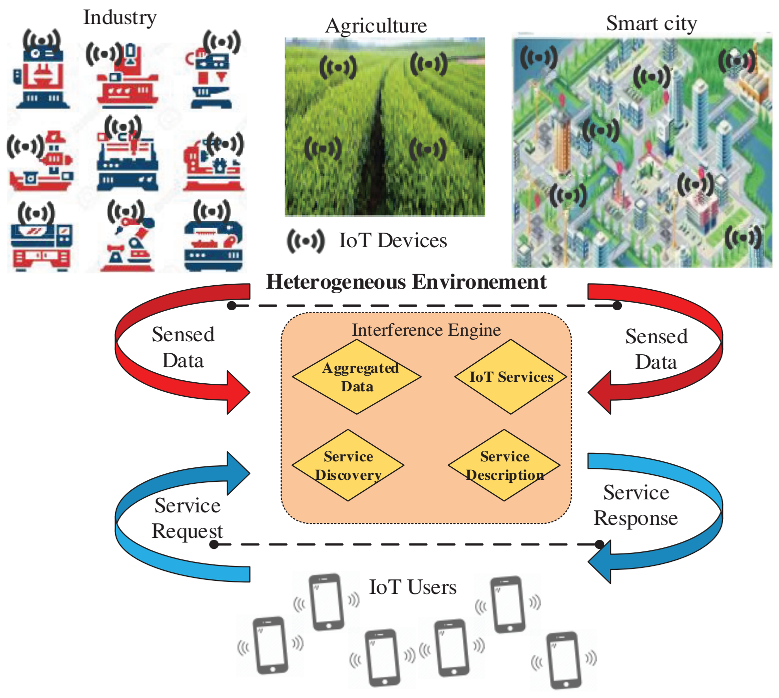


Figure 1: Service discovery for IoT applications

In service discovery, the IoT especially prefers to use lightweight protocols with some metadata to discover services from the database. The meta-data consists of some main information about the IoT devices, such as their identity, model number, and so on. Among all, the Constrained Application Protocol (CoAP) is popular to perform service discovery [16–18]. The CoAP can operate better in centralized and distributed. Service discovery can be handled based on the matching of identity, collected information, and others. Usually, the IoT devices register into the platform, and then they are allowed to submit data on the server. Then it is accessed by the IoT clients, i.e., users, by the server in the form of services. The CoAP protocol deals with the exchange of four messages as Confirmable (CON), Non-Confirmable (NON), Acknowledgement (ACK), and Reset (RST) [19,20]. Based on these messages, the CoAP protocol is also extended to work on rich semantics. Hereby, the IoT users are dynamic, and hence it is required to be flexible in discovering the services from the database. The main Research Questions (RQs) addressed in this paper are as follows:

- RQ1: How can heterogeneous IoT data be effectively managed within a unified database?
- RQ2: How can services be accurately matched with respect to Quality of Service (QoS) and semantic relationships?

Motivated by these questions, this paper proposes a heterogeneous, lightweight ontology–based architecture for IoT service discovery that ensures both QoS awareness and semantic richness. The architecture encompasses ontology management, rule generation, and the extraction of relevant services corresponding to user requests, with a key objective of minimizing latency in service discovery from heterogeneous data sources. The contribution of this proposed lightweight ontology in IoT is illustrated as follows: (1) The distributed management of data from heterogeneous IoT is dealt with by the construction of clusters using a Modified-Ordered Point to Identify the Clustering Structure (M-OPTICS). This clustering also reduces the effect of complexity in storage by reducing the redundancy in the data. The heterogeneous IoT environment will most often transfer sensed data that is similar in some cases. (2) The inference Engine is responsible for generating rules from a lightweight ontology, and also it is required to be updated concerning the arrival of new data. The semantic rules are defined using State-Action-Reward-State-Action (SARSA), i.e., a reinforcement learning algorithm. (3) Discovery of service not only supports QoS, but it is also extracted with relevance in semantic relations. To match QoS dynamic entropy, a Salp algorithm is used, and semantic rules matching a bidirectional recurrent neural network are used. (4) The development of semantic enrichment in CoAP ensures to provide perfect matching of the query with respect to the semantic information from the inference engine. This paper is organized as follows. Section 2 provides a detailed review of previous research on interoperability and service discovery in IoT systems. Section 3 presents an in-depth explanation of the proposed lightweight IoT framework, which manages heterogeneous data and enables semantic service discovery. Section 4 describes the experimental evaluation and performance analysis of the proposed system to justify its improvements over existing approaches. Finally, Section 5 concludes the paper and outlines potential directions for future research.

2 Related Work

This section reviews prior research in the field of service discovery within IoT environments. The authors highlight that semantic interoperability is still a major challenge in heterogeneous IoT environments [21]. Their study notes that variations in data models, device semantics, and platform-specific representations often hinder seamless integration across IoT systems. They emphasize the need for consistent semantic descriptions and standardized vocabularies to enable reliable data exchange and automated reasoning. This work underscores the importance of lightweight and adaptable semantic frameworks for scalable IoT deployments. To enhance service discovery, authors in [22] introduced a Progressive Search Algorithm (ProSA) that filters services according to user requirements. A primitive search strategy was adopted to estimate semantic similarity, classifying query results into perfect match, partial match, and irrelevant match categories based on ontology concepts. However, Quality of Service (QoS) parameters were treated as optional, and thus not validated across all user queries, reducing reliability for QoS-sensitive applications. In [23], a dynamic topology for smart discovery was presented to enable autonomous interactions among devices. This approach utilized a Semantic Web of Things Environment (SWTE), incorporating context information into service discovery. Ontology-based semantic subgraphs were dynamically constructed over time, and actions were executed according to SPARQL query results. Nevertheless, this framework required frequent ontology updates, which propagated changes to all dependent subgraphs, increasing system complexity and maintenance overhead. Similarly, ref. [24] employed semantic service descriptions stored in a repository to support the discovery process. While this method improved semantic expressiveness, it remained limited by the lack of adaptive mechanisms for managing dynamic and large-scale heterogeneous IoT environments.

A primary limitation of conventional discovery mechanisms is their inability to ensure QoS compliance when selecting services with similar functionalities. In [25], the authors proposed a hybrid meta-heuristic framework combining Whale Optimization and Genetic Algorithms to achieve QoS-aware service selection. The model optimizes latency, cost, and energy utilization during the discovery process, resulting in better response times and reduced resource consumption. However, the computational complexity of the hybrid algorithm makes it unsuitable for constrained IoT devices, and its validation is limited to simulation environments. Centralized service registries are another major bottleneck in IoT ecosystems, leading to scalability, trust, and privacy concerns. To overcome these challenges, the authors of [26] developed a decentralized service registry (DSR) based on Distributed Hash Tables (DHTs) with cryptographically verifiable records. The proposed peer-to-peer framework eliminates single points of failure and supports cross-network service discovery. Despite these advantages, large-scale scalability remains untested, and the dependency on stable P2P connectivity limits deployment in constrained IoT scenarios.

In article [27], a DNS-based discovery approach using LOC resource records to embed geographical data within service registration. This enables IoT devices to locate nearby services dynamically, reducing network latency and improving responsiveness. The main limitation lies in the modification of existing DNS infrastructure and potential privacy concerns related to location disclosure. The authors have proposed an autonomous discovery of ontology, especially on landslip-based data [28]. It was used to verify and predict landslides. The three mains' topics in this work are storage, searching, and processing. In this work, the authors have not focused on the use of any algorithm, and so the discovery of service consumes a larger execution time. The authors have investigated a semantic approach based on the constructed ontology in [29]. The designed framework was designed with four processes such as data mapping & pre-processing, model generation, storage, and exploitation. This work was mainly operated on the water management system, so a water-based ontology was created with real-time data. This work is suitable only for a particular type of data, and hence it cannot support heterogeneous data. Then it also fails to process effectively, since the arrival of bunches of requests needs fast operating algorithms to process. In [30], the heterogeneous IoT constructs automatic semantic annotations. The Naïve Bayes algorithm was used for classifying the function from the collected data from IoT devices. Then the annotation files were created automatically. To minimize the tedious computations in searching for a lightweight semantic model was proposed [31]. In this proposed IoT-lite, a set of ten rules was generated from the semantic model based on the semantic relationship. The semantics are dynamic in this work, which uses MathML annotations in the constructed lightweight ontology. By this, the proposed work aims to minimize the processing time. A lightweight semantic model from ontology to search data was given in the paper [32].

This work considers both raw data and processed data for the annotation process. Initially, the raw data is collected using the data mining algorithm Symbolic Aggregate Approximation (SAX) for every 10 min, and then it is annotated. To reduce the problem of annotation of individual data, this aggregation is done. Based on this annotation, the query search is done in this work. Here, not all the sensor data involves annotation; that is, it only considers the raw data every 10 min. In the case of IoT, the sensing information may or may not change frequently, but each data needs to be noted. Also, it lacks some event information for this reason. Here, Symbolic Aggregate Approximation (SAX) is used to collect raw data (10 min), which has the conventional problem of missing significant information since it defines a mean average feature from the aggregated data.

The survey by [33] provides a comprehensive overview of IoT multi-protocol gateway architectures, supported protocols, and cybersecurity measures, identifying key challenges like scalability, standardization gaps, and compatibility issues across technologies. However, its limitations include a primarily descriptive approach that lacks quantitative performance comparisons or empirical evaluations of gateway implementations, potentially limiting practical guidance for deployment. Additionally, the paper does not deeply explore emerging threats like quantum-resistant cryptography or address real-world testing gaps common in IoT surveys, such as interoperability, scalability, and heterogeneity. Table based database model was constructed that defines user topics, identities, services, and others. In this database, the request was mapped, and then the results were given to the user. The exchange of messages depends on the MQTT, HTTP, and CoAP protocols. A Semantically Enhanced CoAP gateway as SECoG to improve accuracy in matching was proposed [34]. The proposed gateway receives the request and uses a triple store, and then it responds to the request. In this gateway, knowledge-based groups are constructed based on the similarity in semantics. Later group management includes the elements of location, resource type, and URI. While the request arrived, the semantic relationship was predicted from the location and the resource type using the utility function. The matching of services is only based on the location and resource type, which cannot match exactly in a heterogeneous data environment. SECoG results in higher execution time in case of an increase in the density of traffic, and this minimizes the reliability of the system. The authors in [35] have proposed a proactive resource directories discovery mechanism. The registered clients submit CoAP requests, and it uses the trickle algorithm that enables reducing traffic. In the quest, the client attaches to all the essential constraints. According to these parameters, the CoAP maps with the directories and gives a response to the client. On the other hand, CoAP can support service discovery, semantic annotation process, and full integration of web application [36–38].

The work by Ferranti et al. systematically reviews metaheuristic-based ontology meta-matching methods, but its scope introduces some notable limitations [39]. It mainly aggregates and classifies existing techniques without providing a unified experimental benchmark, making it difficult to directly compare the effectiveness and computational cost of the surveyed approaches on common datasets. The paper also pays limited attention to highly dynamic or streaming environments, where ontologies and data evolve over time, and it does not deeply analyze practical deployment issues such as parameter tuning, scalability to very large ontologies, or integration with real-world tools and workflows. The static threshold for the elimination of non-optimal concepts will not be suitable for different types of concepts that are in a heterogeneous environment. Hence, a fixed threshold for the entire data results in low accuracy. Hereby, the obtained semantic discovery of service is not verified, which is the key to showing the efficiency in searching for optimization. Further, the management of heterogeneous data in IoT was presented with clustering algorithms as iterative agglomerative fuzzy k-means, density peak based clustering, and adaptive Self-Organizing Map (SOM) clustering [40–43]. There are some critical issues in clustering, while the conventional-issue of k-means is the absolute estimation of the center in the cluster. Hereby, it combines with fuzzy, which requires validating rules each time new data arrives for storage. Service provisioning and discovery are closely linked, particularly in delay-sensitive IoT applications. In [44], a hybrid optimization technique combining Particle Swarm Optimization (PSO) and Chemical Reaction Optimization (CRO) to allocate fog resources efficiently. Their model significantly improved delay and cost metrics under simulated workloads. However, the algorithm incurs high computational overhead and lacks deployment in actual fog-IoT environments. The authors [45] examined the broader context of AIoT, IIoT, and IoT integration, highlighting how heterogeneity, scalability, and interoperability remain systemic barriers to efficient service discovery. The paper provides a valuable overview of current implementation challenges but does not present a practical discovery model or algorithm. The paper [46] introduces a layered security model for the Internet of Medical Things (IoMT) that integrates dynamic, adaptive deep reinforcement learning (DRL) with blockchain to protect medical data and device communications. In this design, DRL continuously adjusts security decisions such as intrusion detection, access permissions, and routing strategies in response to evolving traffic patterns and threats, while blockchain is used as a tamper-resistant, shared ledger to record transactions, verify identities, and preserve data integrity. Applied jointly across sensing, network, and application layers, this combination is intended to strengthen core security goals in IoMT, including confidentiality, integrity, availability, and overall system trustworthiness. A key limitation is the high computational and energy cost of combining DRL and blockchain, which may not be practical for resource-constrained medical sensors and edge devices without strong offloading or hardware support. The literature carried over in this section details the main problematic issues and limitations that exist in the previous work, which is overwhelmed by the proposed solution using novel and convergence algorithms that can achieve better results.

Table 1 illustrates the overall research gaps that are identified from the in-depth literature, which are solved by the proposed solution, and the gaps are filled with the most optimal solutions for service discovery. In [47], a dynamic ontology modeling for an IoT System with a set of 31 reasoning rules was defined for whether the resource (Sensor) exists in the IoT system and whether it was a fault or not. If there were large variations in the temperature, then the device would be controlled automatically. In this work, the testing was performed only for temperature-based applications, i.e., air conditioning and refrigerators. From this work, the key problems identified are: (1) The rules are operated with manual input, which is not as autonomous. However, the temperature is controlled using a semantic ontology; there is a need for manual input of temperature. If so, even the appliances could be operated manually. Also, the manual values will not be accurate in the prediction of temperature. On the other hand, the manual building of rules cannot be dynamic, and it is not possible to create new rules for a new ontology element. (2) This system is suitable only for small scale systems since it defines rules only for temperature (Homogeneous) based appliances, and if any sudden change occurs. Then it cannot be identified.

Table 1: Research gaps of existing methods

Process handled	Research gap
Clustering [40–45]	• Clusters all the data based on similarity, temporal data aggregation was not performed. • Elimination of redundancy is not focused, which can be solved by clustering.

(Continued)

Table 1 (continued)

Process handled	Research gap
Ontology for Service Discovery [21–32]	• Graph-based rule generation from ontology requires larger resources for updating heterogeneous data. • Construction of individual ontology for different domains of data requires to be combined into a single ontology for the discovery of service. • The semantic relationship was only considered to match the services which failed to improve significant QoS in service discovery.
CoAP Protocol [34–38]	• Semantically enriched CoAP protocol was designed static with the QoS parameter which is higher in all the requests. • The matching of URI in the ontology is not the only important metric to improve accuracy in service discovery.

In [48], the rule for service discovery was determined, and the threshold was predicted dynamically. For the prediction of the threshold (based on feature value), the regression tree using the XGBoost algorithm was presented. In this XGBoost, the greedy algorithm was also used to split candidates. Then, the Deep Learning (DL) method that is built with a two-way Gated Recurrent Unit (GRU) was developed to discover new inference rules. The first two separate ontologies were created for concepts based on vehicle and traffic information. Finally, they are combined by using DL. The problematic issues in this work are: (1) The conventional problem of XGBoost is slower in processing, and hence it fails to achieve scalability. On the other hand, the Regression tree cannot handle non-numeric data. But here, the threshold for the inference rule will not only include numeric data. It also includes sensor descriptions. (2) A new inference rule is predicted for two concepts that are not too far in their concepts, both of which are related to transportation information. Also, this can be constructed as a single ontology. Since both concepts will come under the same domain in the ontology. Hence, this will degrade the performance in terms of management when the concepts are different. The COAP protocol with a modified CORE link format to match the semantic information was provided in [49]. In this work, a resource directory was maintained in which all the resources for a user request are present. The COAP servers are clustered, and a cluster head was a selection from it by which the client accesses the resource. The COAP core link format includes ontology attribute, URI of ontology, location, resource type, QoS, and application entity. From this information, the semantic matchmaker will match and give a response to the client. The defined problems are: (1) Here, the QoS in COAP is specified, ranging from 0.9:1, but in general, all the clients will require high QoS. When multiple requests arrive at a time with the same QoS constraint, it is difficult for the system to attain it. Also, the QoS is given without knowledge of environmental factors. (2) In the COAP protocol, the application entity field denotes the name, which also causes conflict and brings incorrect service when the name is like others. Here, the location information is also included; hence, it needs periodic updating in the ontology, else it degrades the result. From this, the problem defined in this research area is solved with the proposed solution that is assured to improve results over the previous research works. The problems stated in this section are solved with the most optimal solutions for IoT service discovery.

The overall problems are illustrated as: (1) Ontology for heterogeneous data is tedious, and hence it introduces a lightweight ontology; however, direct matching of a query from the entire ontology is complex. Since heterogeneous data deals with massive categories of data, this will certainly increase the size of the ontology. (2) The response time increases when an excessive number of requests are waiting for a longer time to process. Also, the key reason to increase response time is that all the searching and matching is enabled at one end. (3) Even though the system supports heterogeneous environmental data, the space and complexity in data management need to be optimal. Also, semantic information requires updating without altering any major part of the system. The property of interoperability between data management and searching was not concentrated. It either focuses on searching or data storage.

3 Proposed Lightweight IoT System

In this proposed system, the heterogeneous lightweight IoT architecture is designed with the achievement of interoperability that merges distributed data management and inference engine. The heterogeneous IoT data is huge, and so it is managed based on the process of clustering, and the redundancy is minimized in the collected data. Here, lightweight CoAP is used, in which the core link format is defined. The main goal of this architecture is to minimize response time in searching for semantic information since it requires taking immediate actions. This work is designed to process heterogeneous data, which includes sensing information from different applications, for example, healthcare, agriculture, smart home, etc. All the data is stored on the CoAP server, which can be accessed by IoT users. The heterogeneous data from the IoT devices are clustered using M-OPTICS, which eliminates redundant data as well as allows for the construction of a lightweight ontology. Further, concerning the lightweight IoT ontology, semantic rules are generated. The matching of service is handled by considering the QoS and the semantics. The algorithms that are used in this proposed work are illustrated. The detailed solution of this research is discussed in the following sub-sections. The proposed lightweight IoT architecture is shown in Fig. 2. As shown in Fig. 2, the proposed heterogeneous IoT is operated with distributed data management and an inference engine.

The methods used for management and inference engines are depicted. The overall system consists of data collected from heterogeneous IoT, CoAP servers, IoT users, and inference engines. Data aggregated on different applications is extracted as services from the CoAP server, with the search operation performed on an inference engine. The data repository is responsible for storing data from IoT which have arrived from similar and dissimilar types of IoT devices. The measurement of environmental changes may not have wide variation, and so it is supposed to have redundancy in data. The redundancy will occupy unnecessary storage space in the server. From this proposed system of data management and inference engine, the ontology is updated in the CoAP server, and the rules are generated from the updated ontology in the inference engine. Hereby, the discovery as a service is a preference in the inference engine, while the CoAP server collects the data from IoT devices, and then it just updates it into the ontology. Based on this proposed system, the overall system is tested using a set of IoT devices like sensors, actuators, and tags in an environment that collects different information as per their functionality. Let the data from a heterogeneous environment be stored in a server, and then IoT users are allowed to discover the service and extract the information of IoT devices and the timely measurements that are sensed by the device at different timestamps. It also includes any actions that are being taken in the system regarding the value of IoT devices.

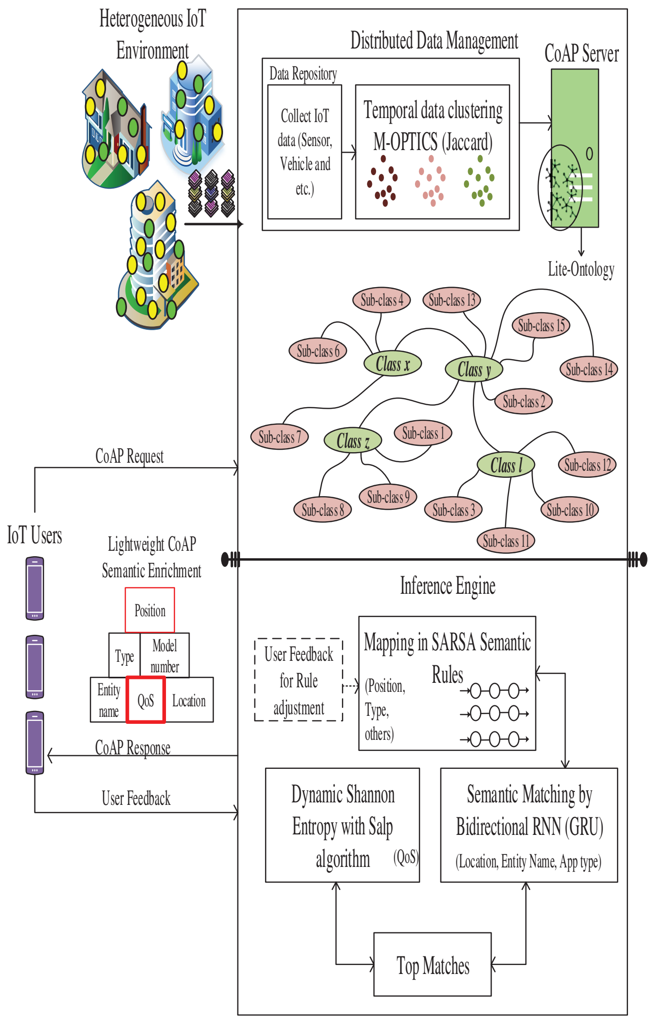


Figure 2: Proposed lightweight IoT ontology architecture

3.1 Distributed Heterogeneous Data Management

This is a distributed heterogeneous data management system in which the CoAP server is present. In this data management, the data from heterogeneous IoT devices are clustered using M-OPTICS, and then they are involved in the construction of a lite-ontology. The M-OPTICS clusters the arrived data by taking into account the temporal changes, which do not create a dataset in a clustered manner, and produces its augmented form, which shows its clustering structure. Here, the clustering is performed for temporal data since the IoT devices submit sensory data, which is highly similar, and hence, data redundancy happens very commonly. When a large number of IoT devices are added to an IoT environment, M-OPTICS performs effectively. The M-OPTICS based clustering of data achieves the segregation of redundant data, thereby considering only the non-redundant data, and correlation between the data is properly utilized by M-OPTICS. The similarity between the data is determined using the Jaccard distance measure. The notations and their descriptions for this article are given in the Appendix. The complete formulation corresponding to equations from (1)–(19) is given in the Appendix.

In M-OPTICS, three significant constraints are computed, such as maximum radius, number of cluster points, and distance. It includes the core point, core distance, and reachability distance. In M-OPTICS, the point p is said to be a core point only when the point is on $MinPts$, i.e., p is present in the known range of ε-neighborhood. Hereby, the core distance $CD(o)$ and reachability distance $RD(p,o)$ is given as in Eqs. (1) and (2).

$$CD(o) = \begin{cases} \infty & , |o, \varepsilon| < MinPts \\ MinPts - D(o), & otherwise \end{cases} \tag{1}$$

$$RD(p,o) = \max(CD(o), D(p,o)) \tag{2}$$

where o is the center point and p is the object. The core distance is defined as the minimum value that is the radius, from this distance, the core point is differentiated. Then the reachability distance is estimated as the maximum of the core distance. Here ε is what denotes the radius in the n-dimensional data. Based on the estimation of the reachability distance, the clusters are separated. First, the similarities between data points are measured using Jaccard distance JD for determining redundancy of data, which is computed as follows in Eqs. (3) and (4).

$$JD = 1 - J_\mu(A,B) \tag{3}$$

$$J_\mu(A,B) = \frac{\mu(A \cap B)}{\mu(A \cup B)} \tag{4}$$

where A and B are the two points that exist in the collected data for the time, t_1 and μ are the Jaccard constant. Hereby, the arriving data is clustered as per the growing time, from which the redundancy data is larger in number, and hence similar data points are eliminated in the clusters. So that this will not occupy higher storage space, and it is easier to update the lite ontology. This time-based clustering minimizes the redundancy and storage, which fills the research gap that is identified from the literature. In the clustering, the data from the IoT devices are clustered, i.e., even a cluster can have multiple measurements of a particular IoT device at every millisecond or second variation. Hereby, we are clustering the data, which is to be stored in the CoAP server, so only the similarity between the values is considered. The construction of clusters is depicted in Fig. 3, which estimates distances and then the clusters. Once the clusters are created, the lite ontology is generated in the CoAP server.

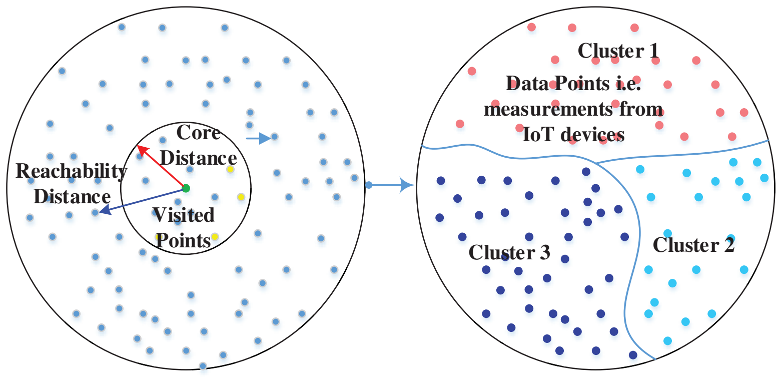
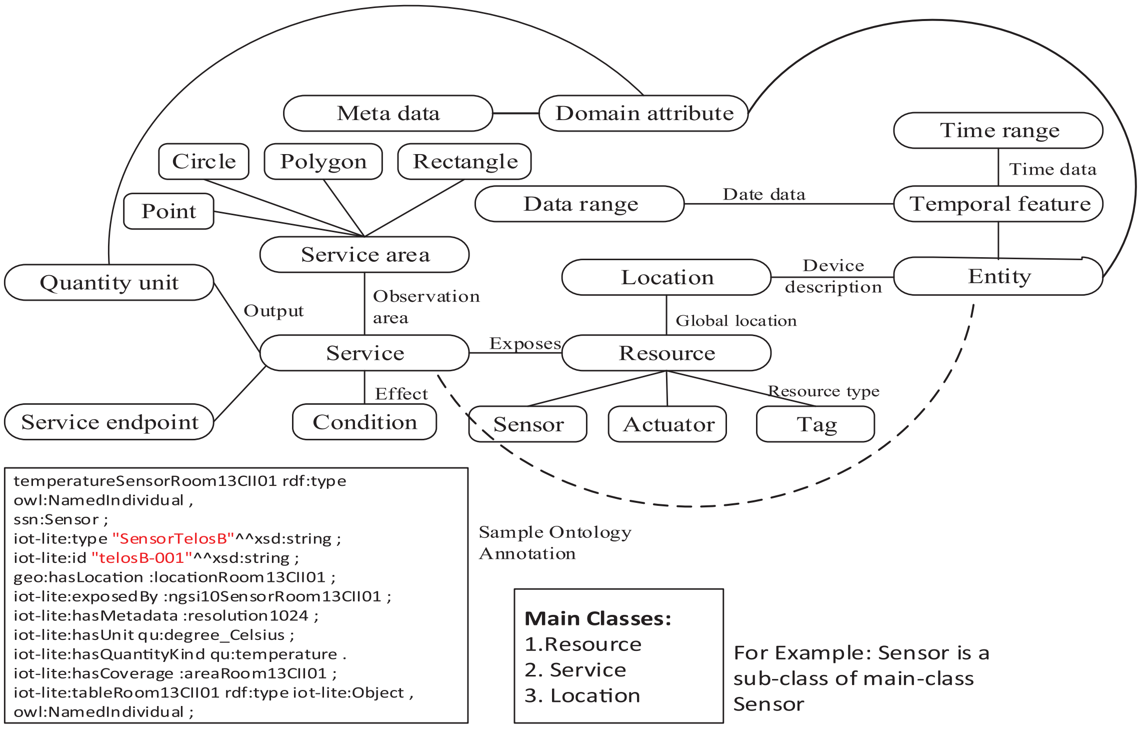


Figure 3: M-OPTICS based clustering of heterogeneous IoT data

The lite ontology composes information on IoT devices, for instance, if it is a sensor, then it will have the ability to sense [31]. So, this presents with the quantity unit as Celsius, which returns the temperature. Based on the type of IoT device, the unit and model number are adapted in a lite ontology. The lightweight

IoT ontology is created, and the annotations are given for each semantic relationship in the ontology. The construction of lite-IoT ontology is shown in Fig. 4 [31]. Ontology syntax composes annotations and axioms as given below.

Annotation: (Ontology ID, URI references, Property ID)
Axiom: (Classes, Properties, Individuals)
Syntax: rdfs:label "*List*";
rdfs:comment "*The class of RDF Lists.*";
rdfs:*subClassOf*
rdfs:*Resource.*
rdf:*nil* a
rdf:*List*;
rdfs:is*DefinedBy*



```
temperatureSensorRoom13CII01 rdf:type
owl:NamedIndividual ,
ssn:Sensor ;
iot-lite:type "SensorTelosB"^^xsd:string ;
iot-lite:id "telosB-001"^^xsd:string ;
geo:hasLocation :locationRoom13CII01 ;
iot-lite:exposedBy :ngsi10SensorRoom13CII01 ;
iot-lite:hasMetadata :resolution1024 ;
iot-lite:hasUnit qu:degree_Celsius ;
iot-lite:hasQuantityKind qu:temperature .
iot-lite:hasCoverage :areaRoom13CII01 ;
iot-lite:tableRoom13CII01 rdf:type iot-lite:Object ,
owl:NamedIndividual ;
```

Figure 4: Representation lightweight IoT ontology

This ontology is lightweight due to its faster processing time and annotation of the created ontology. The matching of the ontology is performed with the annotation. MathML is a mathematical expression that is enabled to map to the values given by IoT devices. It also includes the nearby locations; from these expressions, the ontology is updated according to the arrival of the new heterogeneous IoT data. As per the semantic annotations from MathML, the ontology is lighter for including new entries, and hence it is scalable for a large-scale environment. The semantics of the constructed ontology are given using MathML. Due to this, it can search for the service from the inference engine.

3.2 QoS and Semantic Rule-Based Service Discovery

CoAP server generates the lite ontology from which semantic rules are generated using the reinforcement learning algorithm of SARSA, which is also efficient in mapping for the arriving IoT request. The generation of rules is based on Position (P), Geo-Location (GL), Type (TY), Semantics (*SO*), URI, and Model Number (MN). The CoAP server maintains a lightweight ontology, which is also forwarded to the inference engine. This engine is responsible for generating rules using a reinforcement learning algorithm. In this algorithm, the state denotes the class and subclass information as entity name, application name, and their corresponding actions denote the list of services that are apt for the given states. SARSA is an algorithm that learns the environment from agents using a value function concerning the recent action that is performed based on the policy. The SARSA algorithm provides faster learning of the environment, which results in the dynamic generation of rules for the purpose of service discovery. This procedure in SARSA follows an on policy in the system, and the policy update is handled by the following mathematical expressions, such as in Eq. (5).

$$Q\left(S_T A_T\right) = Q\left(S_T, A_T\right) + \alpha\left(R_{T+1} + \gamma Q\left(S_{T+1}, A_{T+1}\right) - Q\left(S_T, A_T\right)\right) \tag{5}$$

where the states and the action in time T and $T+1$ is given as $\left(S_T, A_T\right)$ and $\left(S_{T+1}, A_{T+1}\right)$ respectively, and the reward is R_{T+1}, γ is the learning rate. The actions in SARSA are to match the corresponding states. The policy is developed in the SARSA algorithm (Algorithm 1) using the epsilon-greedy method that uses Bellman Expectation. The complexity of the SARSA algorithm is $O(n|A|)$, here A is the set of actions.

Algorithm 1: Rule Generation and Updation by SARSA

```
Input:$\{S_1, S_2, S_3 \ldots\ldots\} = \mathbb{S}$,
Output: $\{A_1, A_2, A_3 \ldots\ldots\} = \mathbb{A}$
1. begin
2. initialize $Q\left(S, A\right), \forall S \,\varepsilon\, \mathbb{S}, A \in \mathbb{A}(s)$
3. for all states $S$
4. initialize $S \rightarrow \left(P, GL, TY, URI, MN\right)$ //Initialize each state with these constraints
5. select the action $A$ for the state $S$ from the policy defined as $Q(S, A)$
6. generate rules $\{R_1, R_2, \ldots\} \rightarrow \{(S_1, A_1), (S_2, A_2)\ldots\}$
7. for iteration = 1
8. perform action $A$ for the state $S$
9. detect rewards $R$ and new state $S_{T+1}$
10. select $A_{T+1}$ and $S_{T+1}$ from updated policy $Q$ using (5)
11. identify $S_{T+1} \rightarrow S$, $A_{T+1} \rightarrow A$
12. terminate at maximum iteration
13. if new IoT data as $S_{new}$
                {
                      perform steps 5 to 11
                      update $A_{new}$ //new rule updated in the inference engine
                  else
goto step 14
                 }end if
14. end
```

As per the above pseudo-code, the SARSA algorithm is executed, in which each state composes the ontology metadata as P, GL, TY, *SO*, URI, and MN. The semantic rules are initially generated as $(R_1, R_2, \ldots)$, and from the ability to learn from IoT user feedback. The feedback is either positive or negative based on the response given to the service. If the service is satisfied, then that rule is maintained; else it is updated. Also, the arrival of new data from IoT generates new rules and updates in the inference engine. Further, these rules, the matching of QoS and the semantic relationship. The information about QoS and semantics is included in the CoAP protocol. Especially in semantic information, it composes of position, type, model number, entity name, and location in the lightweight CoAP semantic enrichment message format. In this proposed system, a lightweight CoAP semantic enrichment protocol is used by the IoT users to request the required service. For matching the QoS requirements using dynamic entropy with the salp algorithm and bidirectional Recurrent Neural Network (RNN) using Gated Recurrent Unit (GRU). The salp algorithm is a swarm-based optimization algorithm. The advantage of salp algorithms is that it maintain a better tradeoff between exploration and exploitation. The salp chain performs a search for an optimal food source here it searches for optimal QoS. Initialize the population that denotes all the QoS constraints. Let X be the swarms for n number of slaps that is given as in Eq. (6).

$$X_i = \begin{bmatrix} x_1^1 & x_2^1 & \ldots & x_d^1 \\ x_1^2 & x_2^2 & \ldots & x_d^2 \\ \vdots & \vdots & \ldots & \vdots \\ x_1^n & x_2^n & \ldots & x_d^n \end{bmatrix} \tag{6}$$

where x_i^1 denotes the first salp position with $i^{th}a$ dimension, which is mathematically expressed as in Eq. (7).

$$x_i^1 = \begin{cases} y_i + r_1\left(\left(UB_i - LB_i\right) r_2 + LB_i\right), r_3 \geq 0 \\ y_i + r_1\left(\left(UB_i - LB_i\right) r_2 + LB_i\right), r_3 < 0 \end{cases} \tag{7}$$

here UB_i and LB_i represent the upper bound, and lower bound values, r_2 and r_3 denote random numbers that are chosen between the ranges $[0, 1]$. The term y_i denotes food position on i^{th} dimension the food position denotes the target QoS, i.e., optimal. Then r_1 is formulated as in Eq. (8).

$$r_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \tag{8}$$

where the l is the iteration in which the process is being executed, and L denotes the total number of iterations. Then the update of position is performed by the following equations such as Eqs. (9)–(11).

$$x_i^j = \frac{1}{2}\lambda t^2 + \delta_0 t \tag{9}$$

$$\lambda = \frac{\delta_{final}}{\delta_0} \tag{10}$$

$$\delta_0 = \frac{x - x_0}{t} \tag{11}$$

In the above equation, x_i^j is the position in i^{th} dimension for j^{th} salp, then δ_0 is the speed, and t represents iteration. In case, if $\delta_0 = 0$, the compute the position using the following equation Eq. (12).

$$x_i^j = \frac{1}{2}\left(x_i^j + x_i^{j-1}\right) \tag{12}$$

The fitness function is defined as significant QoS parameters like delay, throughput, and jitter. The fitness function is given in appendix having Eqs. (13) and (14).

$$F = (QoS_i) \tag{13}$$

$$QoS = Th_{PT} \cdot D_y \tag{14}$$

where Th_{PT} and D_y are throughput and delay in QoS. The coefficient value of r_1 computed from Eq. (8) is used to balance the process of exploration and exploitation. Then the position of the salp position is updated and if any of the salps goes beyond the search space then those points are included in the search space using Eq. (15).

$$x_i^j = \begin{cases} l^j & if x_i^j \leq l^j \\ u^j & if x_i^j \geq u^j \\ x_i^j & otherwise \end{cases} \tag{15}$$

where x_i^j represents the position in i^{th} dimension with j^{th} salp, i.e., a QoS requirement. In this salp algorithm, Shannon entropy is applied which is calculated dynamically as per the environment. The Shannon entropy SE is formulated in the following Eq. (16).

$$SE = -\sum_{i=1}^{N} P(x_i) \log P(x_i) \tag{16}$$

From the entropy function, the threshold is computed dynamically since the QoS of the environment is dynamic and the requirements of each user differ. Then $P(x_i)$ denotes the probability values where $i = 1, 2, \ldots, N$. This threshold is computed for a set of IoT user requests, i.e., it is not computed for each request, it has been calculated for a time based arrived CoAP request. The QoS matching is performed from the salp algorithm by estimating the QoS of the arrived requirements from the semantic enrichment in CoAP. The semantic enrichment CoAP enables the provision of semantic solutions that are involved in selecting significant constraints using the CORE link format. The fields that are included in CoAP are defined below,

- Position: The position as PO that denotes whether the IoT device is used outdoor or indoor. This is essential since some IoT devices are used for both purposes. For instance, the temperature measuring IoT device will be used in the smart industry as well as smart city too. Hence this is considered one of the important metrics in CoAP.
- Type: The type is TY that denotes the type of the device, i.e., as a sensor, surveillance, or some other IoT device.
- Model number: The model number MN denotes the manufactured IoT device number.
- Entity name: EN be the entity name that denotes the physical object that is derived from the IoT device as a sensor or actuator.
- QoS: The QoS value defines the Quality-of-Service requirements of each IoT user.
- Location: The LOC defines the locality of the device which gives information of latitude and longitude at that point.
- Semantic: The SO is the reference ontology that is enabled to match the services service entity and application entity from the CoAP server.

The pseudo-code of this proposed optimization algorithm for QoS matching is illustrated in the following Algorithm 2.

Algorithm 2: QoS matching in Salp algorithm

```
Input: QoS requirement of IoT user, Output: Optimal QoS
1. begin
2. initialize population $x_i$
3. initialize $UB_i$ and $LB_i$
4. while (terminate condition not reached)
Do
5. compute fitness for each IoT user
6.    search the best solution from $F$
7. update the value of $r_1$
8. for each $(x_i)$
        do
          if $(i == 1)$
                {
                  then
                        update position of leading salp
                  else
                        update position of follower salp
                  } end if
    end for
9. cross-check position salp with respect to $UB_i$ and $LB_i$
10. end while
11. return best $F$
12. end
```

From these CoAP CORE link formats, the inference engine in the proposed system is used to discover the service. The CORE link format provides meaningful information, using which the semantic matchmaker module matches the semantic parameters. The use of new fields such as *TY, EN*, and *SO* enables us to understand semantics to discover a service. The CoAP request is submitted as,

```
coap://localhost:5683/.PO = "Indoor/outdoor" TY = "RFID Tag" MN = "675,432" QoS = "0.98" LOC =
"13.0827°, 80.2707° "SO = "12,367"
```

The ontology reference in the CoAP CORE link format enables semantic relationships. To match the semantic information bidirectional RNN is used with Soft-GRU. This proposed a bidirectional RNN is composed of two RNN, layers that are stacked, which can process simultaneously. The reason behind implementing GRU is that it possesses faster training than other RNN models. Let Y be the input, and Y_k be different inputs that have arrived from different timestamps. The processing is simultaneously, but the layers are arranged sequentially, and hence the input is processed one after the other layer. In the hidden layer, two hidden states are estimated since two RNN layers are present; this is determined for each step. The hidden states are joined into a single layer by concatenating the two inputs using a simple operator such as an addition. This is a fully connected neural network in which each neuron in the system is executed as a bidirectional RNN. This semantic matching can consider both the output elements and the prior input that are processed in consecutive time steps. Soft-GRU in RNN is low-complexity that processes the input $\tilde{x}_{k,n}$ and uses history as $h_{k-1,n}$. This is low complex due to the use of soft plus as activation function and the soft

GRU is given as in Eqs. (17)–(19).

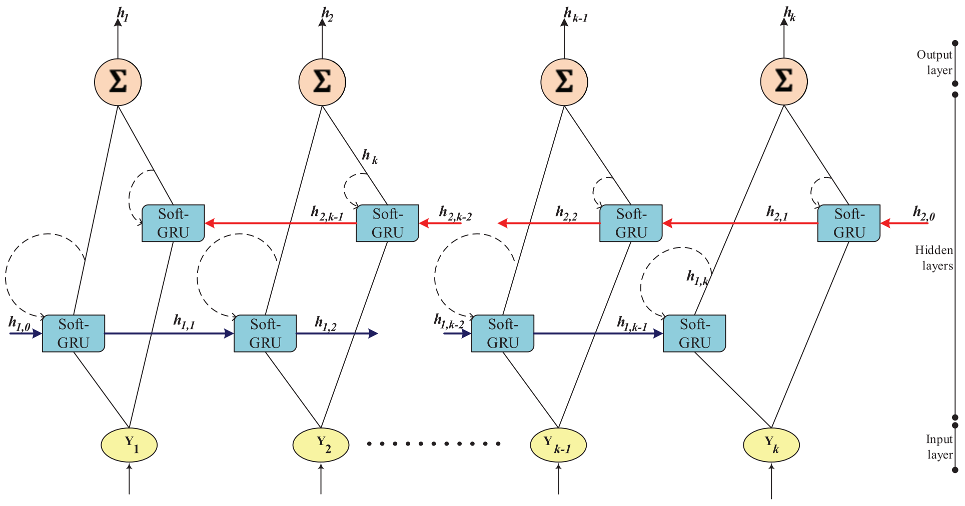
$$h_{k,n} = (1 - z_t) \odot h_{k-1,n} + z_t \odot \tilde{x}_{k,n} \tag{17}$$

$$\widetilde{x}_{k,n} = \pi\left(W_x x_{k,n} + b_x\right) \tag{18}$$

$$z_t = \sigma\left(W_x x_{k,n} + U_z x_{k-1,n} + b_z\right) \tag{19}$$

The term $\pi(x)$ is the soft plus function, i.e., computed as $\log(1 + e^x)$, σ is the sigmoid function, and b_x, b_z are biases, respectively. W_x, W_z, U_z denote the GRU weights.

The proposed system with semantic matching is performed in a bidirectional RNN, as shown in Fig. 5. The GRU can operate for natural language processing, which uses the context information that exists in the created lite ontology. From the matched semantic and QoS based on the CoAP request, it discovers the top matched services that are present in the CoAP server. According to the rules constructed for the services, they are discovered by the IoT user. As per the arrival of data from the heterogeneous IoT devices is updated in the semantic rules and so the request for a new service will also be satisfied in this system. The time complexity of the proposed approach is presented in Table 2.



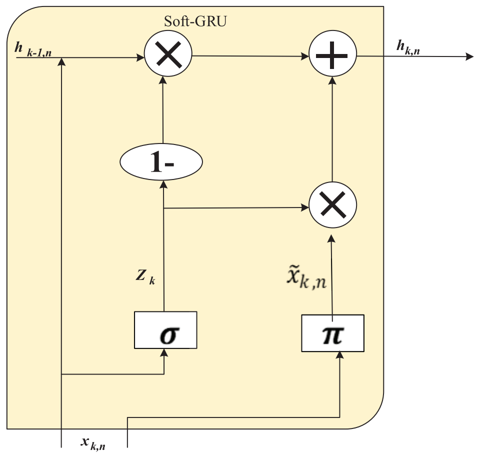


Figure 5: Bidirectional GRU for QoS matching

Table 2: Time complexity for the proposed method

Process	Time complexity	Description
M-optics based clustering	O(N log N)	N denotes the number of iterations
Lite ontology	$O(v \times \kappa \times \rho)$	v and $kappa$ denotes the relationship between the concepts and ρ denotes iterations
Rule generation and updation	$O\ (\mathrm{N}\|A\|)$	N denotes the number of iterations
QoS and semantics matching	O(N × K)	N denotes the number of iterations and K denotes the attributes

4 Experimental Result Analysis

The proposed has experimented with the proposed algorithms for service discovery using CoAP. This section deals with testing the system and evaluating this proposed system using significant metrics that measure the performance of this proposed work. The proposed work is compared with previous work to ensure the better efficiency of the proposed system. This section is categorized into three sub-sections: experimental system development, comparative results, and research highlights that elaborate on the solutions proposed in this system. The improvements of this proposed lightweight IoT ontology are completely studied in this section.

4.1 Experimental Setup

The proposed lightweight IoT system is implemented on NS 3.35 using Python programming under the Ubuntu operating system. The software is installed and developed with the proposed algorithms for the discovery of the service. The implemented environment of the proposed IoT lightweight ontology is shown in Fig. 6, which involves the system specification as discussed in this section. The implementation is performed for service discovery with the design of data management and rule generation in an inference engine. The enhanced CoAP semantic enrichment is a lightweight protocol in IoT that is also designed to match QoS as well as semantics in the services. To ensure experimental transparency and facilitate reproducibility, the proposed framework was implemented using NS-3.35 running on Ubuntu 20.04 with Python-based simulation control. The network was configured with different heterogeneous IoT nodes, employing the CoAP protocol over UDP, while semantic processing and inference modules were executed at the edge server. The M-OPTICS clustering parameters, including minimum points, reachability distance, and temporal window size, were fixed across all experiments and are explicitly provided in the heterogeneous data management section of this paper. Energy consumption was modeled using the NS-3.35 energy framework. All experimental results are obtained using synthetically generated IoT data under realistic network and traffic conditions. The experimental evaluation is conducted using a synthetically generated dataset produced by virtual IoT sensors within the simulated environment. The initial raw dataset consists of continuous sensor streams representing key network, device types, contextual parameters, and service quality parameters, including throughput, delay, jitter, packet loss, and other QoS-related attributes. Since raw sensor streams often contain unstable readings, duplicated patterns, and noise due to continuous sensing behavior, a data refinement stage is performed prior to evaluation. In this phase, threshold-based filtering combined with M-OPTICS clustering using the Jaccard distance method is applied to remove highly similar, redundant, and noisy samples. This process ensures that only distinct, meaningful, and temporally relevant observations are retained for analysis. After preprocessing, a refined dataset comprising 40,000 samples is obtained and used for all performance evaluations. The number of generated samples scales dynamically with the increase in the number of IoT devices and sensors, thereby reflecting realistic variations in network load and sensing intensity. This preprocessing strategy improves the reliability and validity of the experimental results by preventing performance bias caused by duplicated data and unstable sensor fluctuations. Each synthetic data sample is represented using a multi-dimensional feature vector that combines device attributes, contextual information, QoS parameters, energy metrics, semantic descriptors, clustering metadata, and inference outputs. This comprehensive feature structure enables accurate evaluation of clustering quality, semantic reasoning, QoS-aware discovery, and energy efficiency. Instead of adopting an extensive hyperparameter tuning process, this work employs a targeted parameter-selection strategy focused on the key components of the proposed framework. The objective is to ensure stable clustering behaviour, efficient ontology representation, and low-latency semantic reasoning, while maintaining compatibility with resource-constrained IoT devices. For the M-OPTICS clustering module, the values of ε, MinPts, and the reachability threshold were selected empirically through small-range parameter sweeps to maintain consistent cluster formation across heterogeneous node densities. The dimensionality of the lightweight ontology vectors was chosen by balancing semantic discriminability with memory and computation constraints on embedded IoT platforms. In the semantic scoring mechanism, the weighting coefficients governing the integration of semantic similarity and QoS attributes were selected to stabilise ranking behaviour without increasing reasoning latency. Protocol-level configuration parameters, such as CoAP retransmission limits and acknowledgment timeouts, were set according to experimentally verified operational ranges suitable for lossy and low-power networks. Importantly, the overall parameter-selection process remains lightweight, reproducible, and aligned with practical IoT deployment conditions, as the proposed framework does not involve computationally intensive deep-learning-based tuning.

The heterogeneous data from the IoT devices are clustered, and a lite ontology is created on the CoAP server. From this ontology, the rules are generated using a reinforcement learning algorithm. On the other hand, the arrived CoAP requests with QoS and semantic service requirements match using Salp Optimization and bidirectional RNN algorithm. The development of these algorithms into programmable code can process the ontology and evaluate the results. Table 3 depicts the main requirements that are used for this implementation. The lightweight ontology is constructed from the collected data of IoT devices that are defined based on the type and its attributes. From the attributes of RDF files using OWL, the rules of this proposed system are generated, from which services are discovered for IoT users. As per the semantic matching from the relationship, it discovers the services by performing matching using a bidirectional RNN. According to the developed lightweight ontology, Table 4 depicts specific constraints that are used in this proposed IoT for testing the system. The developed ontology is used to derive the relationship between sensing devices, actuating devices, and tag devices. These three devices are commonly said to be IoT devices. It relates the IoT concepts to the main relationships using which the rules are defined, and then they are utilized to discover services. The further testing process is handled in this ontology.

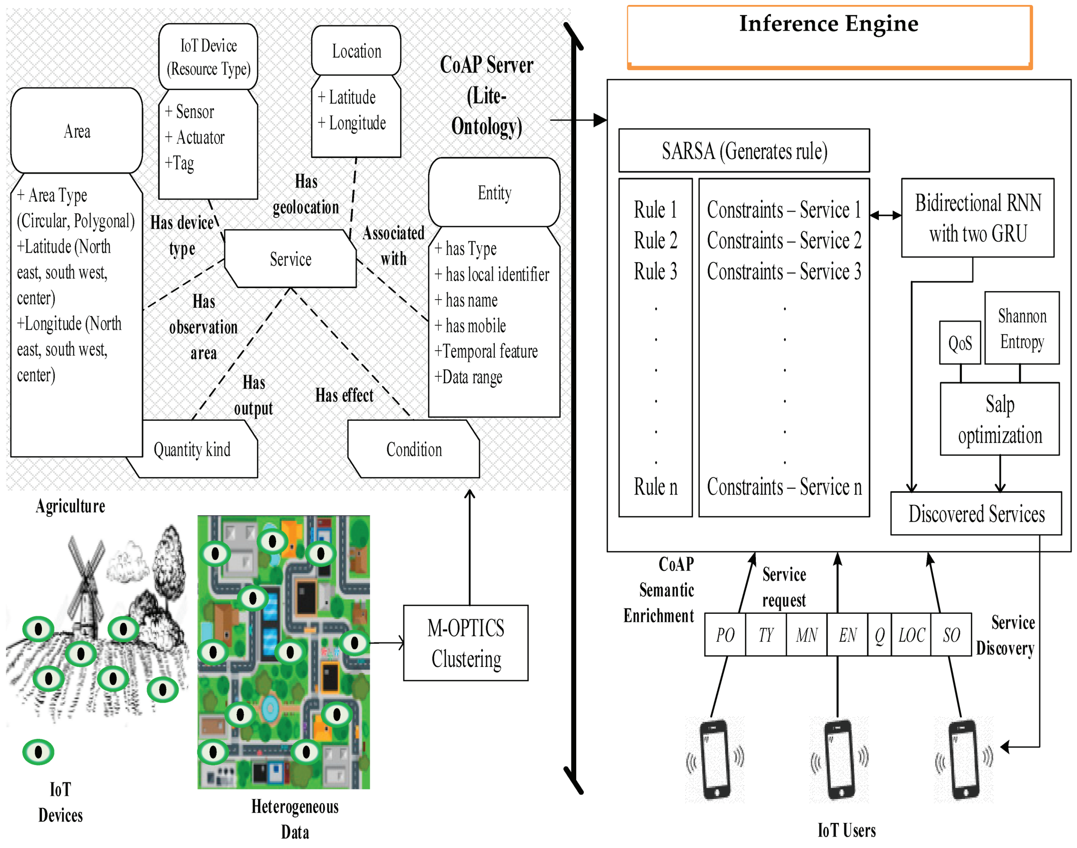


Figure 6: Proposed lightweight ontology pipeline architecture

Table 3: System specification for the proposed method

Service discovery system	
Parameter	Range/Version
Number of IoT users created	30
Number of CoAP requests	35 requests per second
Simulation time	300–400 ms
Bidirectional RNN	
Number of hidden layers	2
Encoder	Bidirectional
RNN cell variant	Soft GRU cell
Learning rate	0.01
Batch size	1
SARSA algorithm	
Epsilon	1
Gamma (γ)	0.9
Alpha (α)	0.9

(Continued)

Table 3 (continued)

Decay	0.5
Software/Hardware requirements	
Network simulator	NS-3.35
Operating system	Ubuntu 22.04
Processor	2.5 GHz or Above
RAM	4 GB
Operating speed	3 GHz
Tool	Python 3.11.7

Table 4: Lite-ontology specifications

Parameter	Specification
Classes	10
Object properties	10–15
Datatype properties	9
Number of nodes	34
Gateways	1
IoT devices	25 Sensors
CoAP server	1
Base station	1

4.2 Comparative Analysis

This work of service discovery from a lightweight IoT environment is compared with the previous work of multi-ontology that is constructed following a deep learning approach [48]. This work is compared to previous work that develops knowledge-based reasoning from which a knowledge graph is constructed for the discovery of services. To evaluate the performance of the proposed system, a set of significant parameters is computed, they are response time, execution time, service delay, precision, and recall. These metrics are compared with prior work in which deep learning is used for the generation of rules that are used to discover the services. Also, the common comparison of the potentialities in IoT service discovery is illustrated.

Table 5 depicts the overall comparison of the proposed work concerning the previous work and the process that is handled in those works. The proposed work is presented with all four processes for data, management and service discovery. While in previous work, the clustering is performed to manage data; the service discovery based on ontology is performed.

Table 5: Comparison of proposed and existing works

Reference	Method used	Clustering	CoAP protocol	Ontology	Rule generation
[40]	Improved clustering algorithm	√	×	×	×
[41]	Optimized clustering-based discovery framework on IoT	√	√	√	×

(Continued)

Table 5 (continued)

Reference	Method used	Clustering	CoAP protocol	Ontology	Rule generation
[42]	Multidimensional model-based approach and density-peaks based clustering	√	×	×	×
[43]	Self-organized map clustering and deep autoencoder	√	×	×	×
[47]	Pellet reasoning engine with dynamic ontology modeling	×	×	√	√
[48]	Multi-ontology using two-way GRU	×	×	√	√
[49]	Semantic enrichment of CoAP	√	√	√	×
Proposed	Bidirectional RNN-GRU, SARSA, M-OPTICS and CORE Link format, Salp optimization	√	√	√	√

Note: √—Present, ×–Not Present.

4.3 *Evaluation of Time Metrics*

The metric time plays a vital role in the service discovery system. The shorter time taken ensures that the system can discover services within a short period. This metric is computed as response time and execution time. The response time defines the time taken for processing the arrived request and discovering the required service is response time. As for the increase in the number of users, the response time increases accordingly. Figs. 7 and 8 depict the performance of response time with respect to the increasing number of users and CoAP requests. The measurement of this metric shows a reduction in response time than the existing work. In previous research work, knowledge graphs were constructed and rules were defined from deep learning algorithms, while multi ontology is generated in this work. In multiple ontology, multiple information about IoT, which is highly available and hence reduces searching time, extraction of information is more effective and gives a different perspective on the information. In this case, if new arriving data requires a larger time for updating and hence the response time increases in the previous work. The IoT users can submit more than one service request to the system; however, as the number of requests increases and this proposed lightweight IoT system obtains a gradual increase in response time. With the increase in the number of IoT requests, the average response time is 27.175, 29.085 and 14.157 ms in the existing and proposed systems, respectively. The difference is 13.018 ms, which will be higher when the number of users increases. As per the increase in response time, the performance also degrades when the users in the system increases. Hereby, the increase in response time will certainly reflect on other parameters. The main reasons to minimize response time are as follows:

- Updation of rules based on SARSA would increase the response time, so that the lite ontology can discover newly updated services from the heterogeneous IoT environment. While the state of the works does not focus on a heterogeneous environment and is constructed as a single ontology, this degrades the performance.
- The bidirectional RNN is enabled to process multiple requests in parallel, which reduces the computation time simultaneously due to its design of the number of neurons in the network, and also it learns from

the output, hence it performs faster with respect to semantics matching, which reduces the response time, execution time, and service delay, respectively. The number of neurons is defined based on the number of user requests.

- The matching of optimal QoS for the arrived request is better in convergence which reduce the matching process latency, response time and increase the QoS efficiency, since it uses a salp algorithm which is meta-heuristics method and consider only input and output would increase flexibility with dynamic entropy that suits with the environmental QoS as well as it is able to match with the optimal QoS of the services.

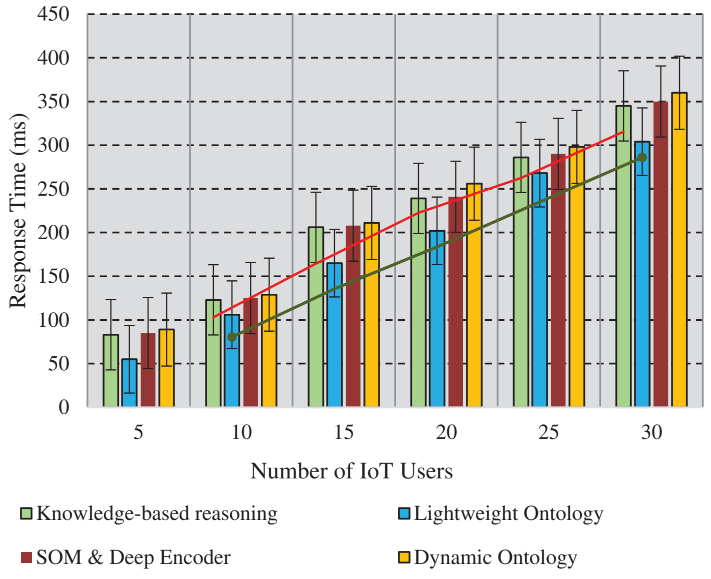


Figure 7: Comparison of response time w.r.t to IoT users

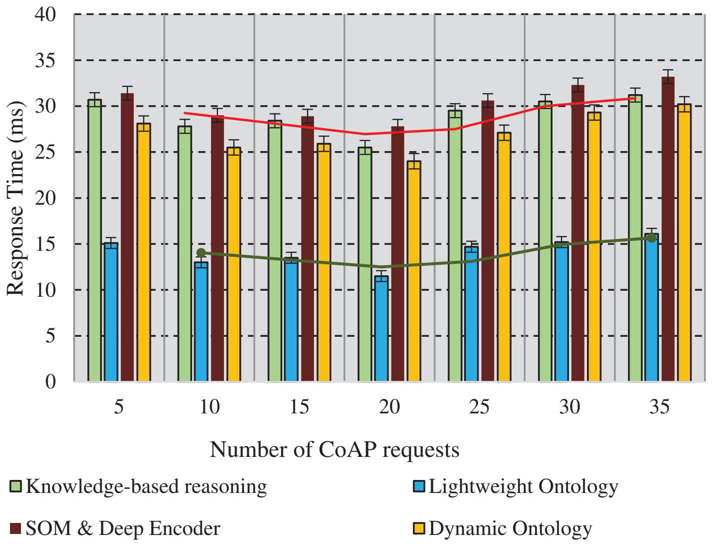


Figure 8: Comparison of response time w.r.t to requests

Fig. 8 shows that the response time would saturate at some point when the number of CoAP requests increases, which shows that the algorithm used in the proposed Lightweight-Ontology is not power consuming, increases the overall efficiency of the proposed method, and is able to tolerate any No. of requests. Based on these reasons, the response time is comparatively lower than the existing work, which only uses knowledge graphs and ontology for service discovery in the system. However, the rules are defined from deep learning; they are defined only for the combined ontology, which is not generated for the new data arrival. Due to the degradation of response time, it is also subjected to reduce the service delay of the system. According to the increase in the number of CoAP requests, the service delay increases due to the processing of a higher number of requests at each time step.

Fig. 9 demonstrates the performance of service delay for the proposed and existing. From this graphical plot, the service delay increases than the proposed work due to the use of lightweight CoAP semantic enrichment that considers QoS constraints. Delay is one of the key parameters in QoS that is evaluated in the proposed work, so the matching of optimal QoS in this proposed lightweight IoT ontology reduces service delay than the previous work. Approximately 2 to 3 s of difference exist between knowledge-based reasoning and lightweight ontology. This happens due to the limitations in the existing work,

- It considers evaluating semantic relationships alone, and hence, the service is higher than the proposed one, which also considers QoS.
- It creates multiple ontologies and then generates the rules for the ontology to combine the ontologies, but it does not define a rule for new data, and hence, the service delay increases in a knowledge-based reasoning system.

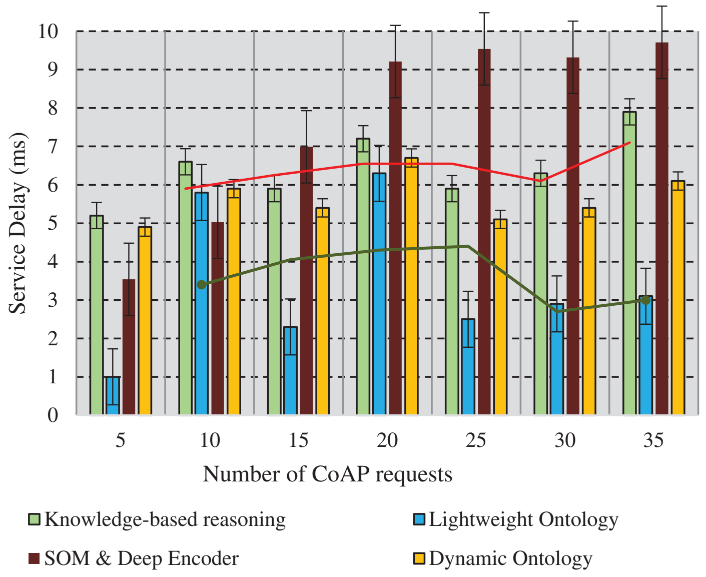


Figure 9: Comparison of service delay

Execution time indicates the time taken for processing all the arrived requests from the users, as well as the time taken for data management in the system. Based on the input data, this execution time either increases or decreases in the system. Even though the input is high, it needs to be processed in a short time, and so the scalability of the system is assumed to be better than the existing work. The execution time is evaluated concerning the increase in the number of CoAP requests, which submit requests for different services in the system. Since this system generates an ontology for heterogeneous IoT data, the request services are also applicable to different entity names and application types from the system. Parallel processing of semantic matching is presented with a bidirectional RNN. Also, the matching of QoS and semantics is performed in parallel, and finally, the top matches from it are used for service discovery and response to the IoT user.

The experimental result of execution time is illustrated in Fig. 10 concerning the number of CoAP requests. As per the obtained result, this proposed work is less in execution time than the previous work due to the use of better algorithms that can be learned from the previous results. Hence, the performance of the proposed lightweight ontology is efficient in terms of time-based metrics when compared with the knowledge graph method.

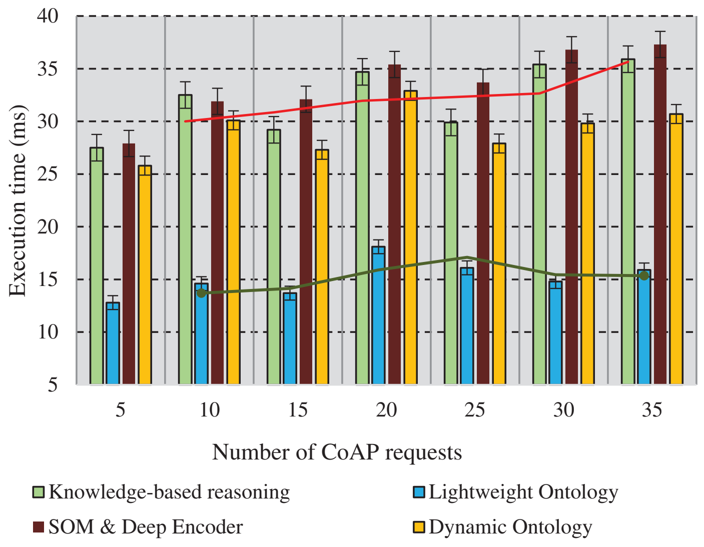


Figure 10: Comparison of execution time

Table 6 illustrates the performance of time-based parameters such as response time, service delay, and execution time. On comparison of this result, the proposed and existing system is compared, and the result shows that the proposed system is better than existing work in time analysis. Hence, the time analysis results can improve the scalability parameter since the time is less comparatively to the existing work with respect to the increase in the number of users' requests.

Table 6: Comparison of proposed lightweight ontology time analysis

Time-based parameters	Dynamic ontology	SOM & deep encoder	Knowledge-based reasoning	Lightweight ontology
Response time (ms)	27.175	30.457	29.085	14.157
Service delay (ms)	5.642	8.690	6.428	3.414
Execution time (ms)	29.214	33.585	32.157	15.142

4.4 Evaluation of Precision and Recall

Precision and recall are the two important result metrics that define the efficiency of the discovery of services for each submitted IoT user request. These two metrics are computed in the ration which covers within 0–1. The mathematical expressions for precision as P and recall as R are given in Eqs. (20) and (21).

$$P = \frac{T_p}{T_p + F_p} \tag{20}$$

$$R = \frac{T_p}{T_p + F_n} \tag{21}$$

where T_p, F_p and F_n is true positive, false positive, and false negative, respectively. Precision and recall metrics are mathematically computed using the rates of true positive/negative and false positive/negative. These rates are known to represent the correctness in the discovered service is as expected or it completely differs from the given request. In simple, the service discovery should be the same as that of the given service request of IoT users. As a comparative analysis of precision, the proposed lightweight ontology is efficient since it performs with simpler mathematical formulation to minimize the execution time, and it can match with the semantic relationship between the concepts that are involved in the IoT environment. The true positive defines that the services discovered are accurate and it is as expected by the IoT user which negative impacts in poor prediction of the services for the submitted request. The performances of precision and recall are depicted in Figs. 11 and 12, in which the proposed lightweight IoT ontology shows better results than the existing deep learning approach on a knowledge-based reasoning system. The precision and recall measures are comparatively less in deep learning than proposed due to the matching of a semantic relationship without considering QoS. Also, the matching of semantics using a bidirectional RNN with GRU ensures optimal service discovery.

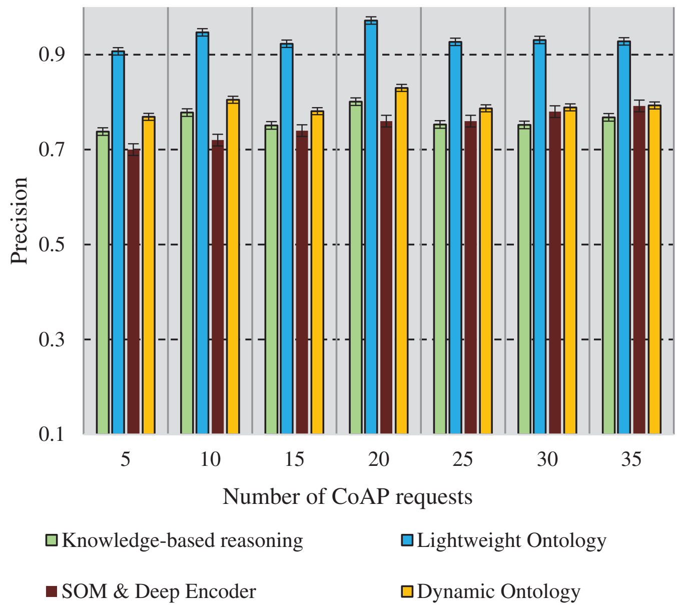


Figure 11: Comparison of precision

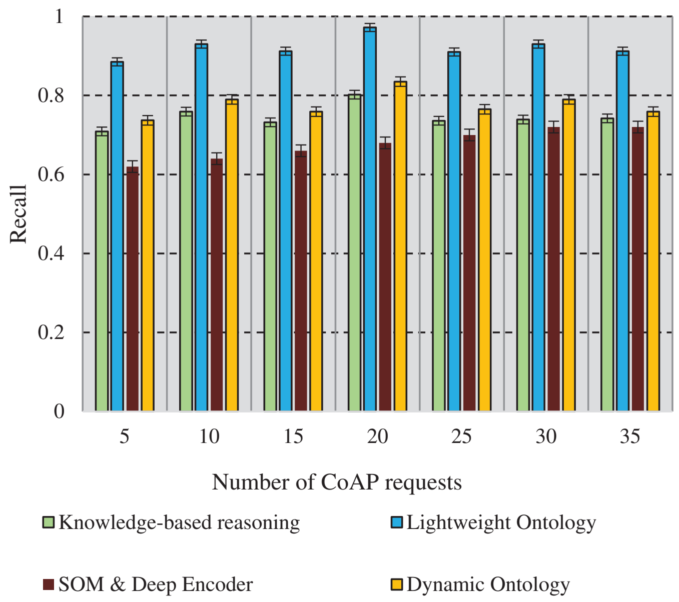


Figure 12: Comparison of recall

From the evaluation of all the parameters, the proposed lightweight IoT ontology illustrates that it is better than the existing in-service discovery by data management and the generation of the rule. As per the generated rule, the semantics are matched from the RNN, and the services are discovered for the submitted IoT users with CoAP requests. Precision and recall are also used to evaluate the accuracy in terms of computing the F-score from the measured values of precision and recall. The computation of F-score is mathematically formulated in Eq. (22).

$$\text{F} - \text{score} = 2 \times \frac{P \times R}{P + R} \tag{22}$$

The terms P and R represent the precision and recall of the system. The comparison for F-score is depicted in Table 7, where the performances of existing and the proposed work are evaluated and shown. The increase in the F-score shows that the system has higher correctness in the services discovered. Table 7 illustrates the performance of the F-score, and the results show that the proposed work is better than existing work. So, the proposed lightweight ontology is assured to bring positive results for the service requests.

Table 7: Comparison of proposed lightweight ontology F-score

Number of CoAP requests	Knowledge-based reasoning	Dynamic-ontology	Lightweight ontology
5	0.72	0.75	0.89
10	0.76	0.79	0.93
15	0.74	0.77	0.91
20	0.80	0.83	0.97
25	0.74	0.77	0.91
30	0.74	0.78	0.93
35	0.75	0.77	0.91

4.5 Evaluation of Overhead

In an IoT environment, overhead leads to high energy consumption and low storage capacity. However, these two requirements are necessary to improve the QoS in IoT. Most of the works do not concentrate on overhead while designing interoperable and semantic IoT architecture. So, by considering overhead as a parameter, in this paper, the proposed method is compared with existing methods, and a graph is shown in Fig. 13. As shown in the graph, the proposed Lightweight Ontology overhead is less when compared to existing methods, even though it is in a heterogeneous IoT environment. By reducing overhead in the Lightweight Ontology method and clustering for heterogeneous data, we achieve low energy consumption and high storage capacity, which improves the overall efficiency of the proposed method.

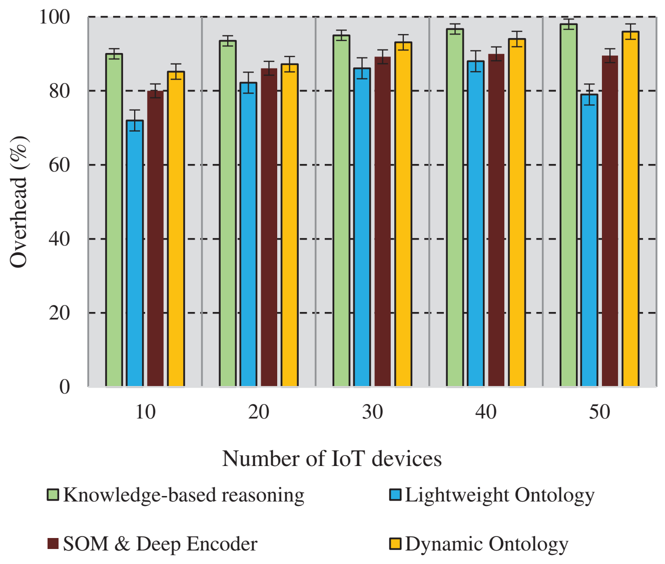


Figure 13: Comparison of overhead

4.6 Use Case of Lightweight Ontology

This proposed lightweight IoT environment is suitable for smart home applications in which the testbed can be deployed with different indoor and outdoor IoT devices. For instance, a set of indoor sensors is positioned to monitor the temperature, sound, vibration, and light. While the outdoor of a smart home will be the gardening area in which the pollution sensors and temperature sensors are used. Hereby, the measurements from each sensor are different in their units. Also, their geo-location depends on the latitude and longitude position at home. All the IoT devices relate to an IoT gateway that is present inside the smart home. Once the atmosphere is measured, it is sent to the end server through the IoT gateway. In this way, the server collects data from multiple smart homes. However, the IoT devices perform continuous sensing, and hence, there occurs higher redundant data, which is also solved in the proposed system. The smart homeowners, as IoT users, have smart mobile phones to access the service from the server. Here, the services are to access the current and history of measurements of a particular IoT device from their smart home environment. The proposed lightweight IoT is applied as in Fig. 14 for a smart home. Similarly, it can be applied to other IoT applications that measure environmental changes that enable the monitor a particular area from a remote location. Hereby, this can be used for the agriculture field too, where the field is analyzed, and so the crops are seeded in a particular region. Although the proposed framework demonstrates strong performance in terms of discovery latency, energy efficiency, accuracy, and scalability, several limitations should be noted. First, all experiments are conducted in a simulation-based environment, and the system has not yet been validated through large-scale real-world deployment, where hardware constraints, wireless interference, and unpredictable network dynamics may affect performance. Second, the evaluation relies on a synthetic dataset generated by virtual IoT sensors, which, although designed to mimic realistic sensing behavior, may not fully capture the complexity and irregularity of real sensor data. Third, the proposed lightweight ontology adopts a simplified semantic representation, which reduces computational overhead but may limit expressiveness for highly complex service descriptions. Finally, the computational and memory requirements of the clustering and inference modules may still pose challenges for extremely resource-constrained devices. These limitations provide important directions for future work toward real-world validation and adaptive resource-aware optimization.

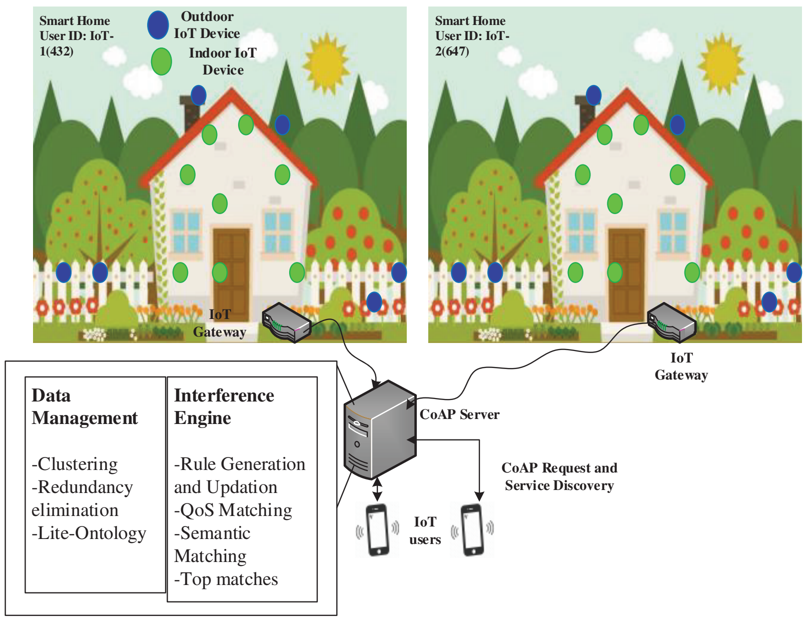


Figure 14: Smart Home as a use case for proposed method

5 Conclusion

In this paper, a lightweight IoT architecture is designed for service discovery. This system is composed of two processes, namely data management and service discovery. In data management, the heterogeneous data from IoT devices is aggregated and clustered to eliminate redundancy, and then the server constructs a lightweight ontology. By doing so, the effective management of heterogeneous data is achieved. The inference engine is responsible for generating rules from the constructed ontology. For rule generation, the SARSA algorithm is used, which can teach the environment, and hence it can update as per the arrival of a new entry into the ontology. As per the rules generated, semantic relationship matches are made using a bidirectional RNN algorithm that uses two RNN layers and concatenates. These two layers are GRU, which can identify the semantics between the arrived request and the rules. So that the perfect semantic-based services are identified using this method. The QoS also plays a vital role in discovering services, and hence the salp optimization algorithm is addressed with the dynamic computation of Shannon entropy for matching the QoS constraints in the service request. Through this, the QoS in service discovery is accomplished by our approach. Further, to make the requests lightweight, CoAP semantic enrichment is developed with a new CORE link format, which is submitted to the system for service discovery. In the future, this proposed work is planned to include IoT user authentication, which allows only legitimate users to access the services from the system.

Acknowledgement: Not applicable.

Funding Statement: The authors received no specific funding.

Author Contributions: The authors confirm contribution to the paper as follows: Research conceptualization and design methodology: Suman Sukhavasi, Thinagaran Perumal; Data generation: Suman Sukhavasi; Software: Suman Sukhavasi; Analysis and interpretation of results: Suman Sukhavasi, Thinagaran Perumal, Norwati Mustapha, Razali Yaakob; Manuscript preparation: Suman Sukhavasi; Editing and proof reading of manuscript: Thinagaran Perumal, Norwati Mustapha, Razali Yaakob. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix

Notations	Description
$CD(o)$	Core distance
$RD(p,o)$	Reachability distance
JD	Jaccard distance
μ	Jaccard cosntant
S_T	State at time T
A_T	Action at time T
R_T	Reward at time T
Q	Policy value
γ	Learning rate
X_i	Salp position
y_i	Food position
UB_i	Upper bound
LB_i	Lower bound
r	Random number
δ_0	Speed of swarm
F	Fitness function
Th_{PT}	Throughput
D_y	Delay
SE	Shannon entropy
PO	Position of the device
TY	Type of the device
MN	Model number of device
EN	Entity name of the device
LOC	Locality of the device
SO	Semantics
$\tilde{x}_{k,n}$	Input function of GRU
$h_{k,n}$	History attribute
$\pi(x)$	Soft plus function
σ	Sigmoid function
b_x, b_z	Biases
W_x, W_z, U_z	GRU weights

References

1. Noura M, Atiquzzaman M, Gaedke M. Interoperability in Internet of Things: taxonomies and open challenges. Mob Netw Appl. 2019;24(3):796–809. doi:10.1007/s11036-018-1089-9.
2. Kousalya A, Sakthidasan K, Latha A. Reliable service availability and access control method for cloud assisted IOT communications. Wirel Netw. 2021;27(2):881–92. doi:10.1007/s11276-019-02184-3.
3. Conti M, Dushku E, Mancini LV. RADIS: remote attestation of distributed IoT services. In: Proceedings of the 2019 Sixth International Conference on Software Defined Systems (SDS); 2019 Jun 10–13; Rome, Italy. 2019. p. 25–32. doi:10.1109/SDS.2019.8768670.
4. Koo J, Oh SR, Kim YG. Device identification interoperability in heterogeneous IoT platforms. Sensors. 2019;19(6):1433. doi:10.3390/s19061433.
5. Lo SK, Liew CS, Tey KS, Mekhilef S. An interoperable component-based architecture for data-driven IoT system. Sensors. 2019;19(20):4354. doi:10.3390/s19204354.
6. Ray PP, Thapa N, Dash D. Implementation and performance analysis of interoperable and heterogeneous IoT-edge gateway for pervasive wellness care. IEEE Trans Consum Electron. 2019;65(4):464–73. doi:10.1109/TCE.2019.2939494.
7. Ranpara R. A semantic and ontology-based framework for enhancing interoperability and automation in IoT systems. Discov Internet Things. 2025;5(1):22. doi:10.1007/s43926-025-00122-8.
8. Zgheib R, Kristiansen S, Conchon E, Plageman T, Goebel V, Bastide R. A scalable semantic framework for IoT healthcare applications. J Ambient Intell Humaniz Comput. 2023;14(5):4883–901. doi:10.1007/s12652-020-02136-2.
9. Al-Osta M, Bali A, Gherbi A. Event driven and semantic based approach for data processing on IoT gateway devices. J Ambient Intell Humaniz Comput. 2019;10(12):4663–78. doi:10.1007/s12652-018-0843-y.
10. Aziez M, Benharzallah S, Bennoui H. Service discovery for the Internet of Things: comparison study of the approaches. In: Proceedings of the 2017 4th International Conference on Control, Decision and Information Technologies (CoDIT); 2017 Apr 5–7; Barcelona, Spain. 2017. p. 0599–604. doi:10.1109/CoDIT.2017.8102660.
11. Sim S, Choi H. A study on the service discovery support method in the IoT environments. Int J Electr Eng Educ. 2020;57(1):85–96. doi:10.1177/0020720918813824.
12. Ahmed AH, Omar NM, Ibrahim HM. Secured service discovery technique in IoT. J Commun. 2019;14(1):40–6. doi:10.12720/jcm.14.1.40-46.
13. Elsayed K, Abu Baker Ibrahim M, Hamza HS. Service discovery in heterogeneous IoT environments based on OCF/IoTivity. In: Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC); 2019 Jun 24–28; Tangier, Morocco. 2019. p. 1160–5. doi:10.1109/IWCMC.2019.8766488.
14. Quevedo J, Antunes M, Corujo D, Gomes D, Aguiar RL. On the application of contextual IoT service discovery in Information Centric Networks. Comput Commun. 2016;89:117–27. doi:10.1016/j.comcom.2016.03.011.
15. Badawy MM, Ali ZH, Ali HA. QoS provisioning framework for service-oriented Internet of Things (IoT). Clust Comput. 2020;23(2):575–91. doi:10.1007/s10586-019-02945-x.
16. Tabassum S, Vijay Babu AR, Dheer DK. A comprehensive exploration of IoT-enabled smart grid systems: power quality issues, solutions, and challenges. Sci Technol Energy Trans. 2024;79:62. doi:10.2516/stet/2024056.
17. Ferdousi R, Mandal PK. LOAMY: a cloud-based middleware for CoAP-based IoT service discovery. In: Proceedings of the 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP); 2019 Feb 25–28; Gangtok, India. 2019. p. 1–6. doi:10.1109/ICACCP.2019.8883000.
18. Khudoyberdiev A, Jin W, Kim D. A novel approach towards resource auto-registration and discovery of embedded systems based on DNS. Electronics. 2019;8(4):442. doi:10.3390/electronics8040442.
19. Martí M, Garcia-Rubio C, Campo C. Performance evaluation of CoAP and MQTT_SN in an IoT environment. Proceedings. 2019;31(1):49. doi:10.3390/proceedings2019031049.
20. Sethi P, Sarangi SR. Internet of Things: architectures, protocols, and applications. J Electr Comput Eng. 2017;2017:9324035. doi:10.1155/2017/9324035.

21. Guerroudj B, Siam A. The importance of semantic interoperability in the Internet of Things. In: Joint Proceedings of the Second International Workshop on Semantic Reasoning and Representation in IoT (SWIoT 2023) and the Third International Workshop on Multilingual Semantic Web (MSW 2023); 2023 Nov 13–15; Zaragoza, Spain.
22. Pattar S, Kulkarni DS, Vala D, Buyya R, Venugopal KR, Iyengar SS, et al. Progressive search algorithm for service discovery in an IoT ecosystem. In: Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData); 2019 Jul 14–17; Atlanta, GA, USA. 2019. p. 1041–8. doi:10.1109/ithings/greencom/cpscom/smartdata.2019.00180.
23. Antoniazzi F, Viola F. Building the semantic web of things through a dynamic ontology. IEEE Internet Things J. 2019;6(6):10560–79. doi:10.1109/JIOT.2019.2939882.
24. Gomes P, Cavalcante E, Batista T, Taconet C, Conan D, Chabridon S, et al. A semantic-based discovery service for the Internet of Things. J Internet Serv Appl. 2019;10:10. doi:10.1186/s13174-019-0109-8.
25. Liu X, Deng Y. A new QoS-aware service discovery technique in the Internet of Things using whale optimization and genetic algorithms. J Eng Appl Sci. 2024;71(1):4. doi:10.1186/s44147-023-00334-1.
26. Kurte R, Salcic Z, Wang KI. Decentralised global service discovery for the Internet of Things. Sensors. 2024;24(7):2196. doi:10.3390/s24072196.
27. Horvath K, Kimovski D. Efficient location-based service discovery for IoT and edge computing in the 6G era. In: Proceedings of the 2025 10th International Conference on Information and Network Technologies (ICINT); 2025 Mar 12–14; Melbourne, VIC, Australia. 2025. p. 94–101. doi:10.1109/ICINT65528.2025.11030893.
28. Phengsuwan J, Shah T, James P, Thakker D, Barr S, Ranjan R. Ontology-based discovery of time-series data sources for landslide early warning system. Computing. 2020;102(3):745–63. doi:10.1007/s00607-019-00730-7.
29. Escobar P, del Mar Roldán-García M, Peral J, Candela G, García-Nieto J. An ontology-based framework for publishing and exploiting linked open data: a use case on water resources management. Appl Sci. 2020;10(3):779. doi:10.3390/app10030779.
30. Liu F, Li P, Deng D. Device-oriented automatic semantic annotation in IoT. J Sens. 2017;2017:9589064. doi:10.1155/2017/9589064.
31. Bermudez-Edo M, Elsaleh T, Barnaghi P, Taylor K. IoT-Lite: a lightweight semantic model for the Internet of Things and its use with dynamic semantics. Pers Ubiquitous Comput. 2017;21(3):475–87. doi:10.1007/s00779-017-1010-8.
32. Elsaleh T, Enshaeifar S, Rezvani R, Acton ST, Janeiko V, Bermudez-Edo M. IoT-stream: a lightweight ontology for Internet of Things data streams and its use with data analytics and event detection services. Sensors. 2020;20(4):953. doi:10.3390/s20040953.
33. Abdelwahed SH, Hefny IM, Hegazy M, Said LA, Soltan A. Survey of IoT multi-protocol gateways: architectures, protocols and cybersecurity. Internet Things. 2025;33:101703. doi:10.1016/j.iot.2025.101703.
34. Jin X, Jung J, Chun S, Yoon S, Lee KH. SECoG: semantically enhanced mashup of CoAP-based IoT services. Serv Oriented Comput Appl. 2019;13(1):81–94. doi:10.1007/s11761-019-00254-0.
35. Djamaa B, Yachir A, Richardson M. Hybrid CoAP-based resource discovery for the Internet of Things. J Ambient Intell Humaniz Comput. 2017;8(3):357–72. doi:10.1007/s12652-017-0450-3.
36. Han SN, Crespi N. Semantic service provisioning for smart objects: integrating IoT applications into the web. Future Gener Comput Syst. 2017;76:180–97. doi:10.1016/j.future.2016.12.037.
37. Sciullo L, Gigli L, Montori F, Trotta A, Di Felice M. A survey on the web of things. IEEE Access. 2022;10:47570–96. doi:10.1109/ACCESS.2022.3171575.
38. Castro M, Jara AJ, Skarmeta AF. Enabling end-to-end CoAP-based communications for the web of things. J Netw Comput Appl. 2016;59:230–6. doi:10.1016/j.jnca.2014.09.019.
39. Ferranti N, Rosário Furtado Soares SS, de Souza JF. Metaheuristics-based ontology meta-matching approaches. Expert Syst Appl. 2021;173:114578. doi:10.1016/j.eswa.2021.114578.
40. Yao X, Wang J, Shen M, Kong H, Ning H. An improved clustering algorithm and its application in IoT data analysis. Comput Netw. 2019;159:63–72. doi:10.1016/j.comnet.2019.04.022.
41. Bharti M, Jindal H. Optimized clustering-based discovery framework on Internet of Things. J Supercomput. 2021;77(2):1739–78. doi:10.1007/s11227-020-03315-w.

42. Zhao S, Yu L, Cheng B, Chen J. IoT service clustering for dynamic service matchmaking. Sensors. 2017;17(8):1727. doi:10.3390/s17081727.
43. Ullah A, Haydarov K, Ul Haq I, Muhammad K, Rho S, Lee M, et al. Deep learning assisted buildings energy consumption profiling using smart meter data. Sensors. 2020;20(3):873. doi:10.3390/s20030873.
44. Hashemifar S, Rajabzadeh A. Optimal service provisioning in IoT fog-based environment for QoS-aware delay-sensitive application. arXiv:2301.12522. 2023.
45. Hou KM, Diao X, Shi H, Ding H, Zhou H, de Vaulx C. Trends and challenges in AIoT/IIoT/IoT implementation. Sensors. 2023;23(11):5074. doi:10.3390/s23115074.
46. Sharma N, Shambharkar PG. Enhancing Internet of medical things security: a multi-layered approach using dynamic adaptive deep reinforcement learning and blockchain. Comput Electr Eng. 2026;129:110808. doi:10.1016/j.compeleceng.2025.110808.
47. Chen G, Jiang T, Wang M, Tang X, Ji W. Modeling and reasoning of IoT architecture in semantic ontology dimension. Comput Commun. 2020;153:580–94. doi:10.1016/j.comcom.2020.02.006.
48. Liu J, Zhang X, Li Y, Wang J, Kim HJ. Deep learning-based reasoning with multi-ontology for IoT applications. IEEE Access. 2019;7:124688–701. doi:10.1109/ACCESS.2019.2937353.
49. Vandana CP, Chikkamannur AA. S-COAP: semantic enrichment of COAP for resource discovery. SN Comput Sci. 2020;1(2):88. doi:10.1007/s42979-020-0104-y.