SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V10I8P109

Volume 10 Issue 8, 85-97, August 2023

Original Article

Intelligent Resource Provisioning and Optimization in Fog Computing using Deep Reinforcement Learning

S. Aiswarya[1], Angelina Geetha[2], K. Ramesh[3]

[1,2]*Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, India.*
[3]*Department of Computer Science and Engineering, Sri Krishna College of Engineering and Technology, Coimbatore, India.*

[1]*Corresponding Author : Aiswarya.sudha@outlook.com*

Received: 14 June 2023 Revised: 17 July 2023 Accepted: 13 August 2023 Published: 31 August 2023

Abstract - *The number of devices linked to the Internet is continuously rising along with the development of the Internet of Things (IoT). The IoT and the expanding volume of data it communicates place constraints on cloud-based data processing and storage. Both fog and cloud computing allow users to store apps and data, but Fog has a broader geographic reach and is closer to the end user. Managing rapidly changing resource provisioning and allocation of resources in fog computing will create new challenges when developing IoT applications and satisfying user requests. To control resource consumption and Service Level Agreements (SLA), flexible and often autonomous systems must choose the appropriate virtual resources. This work presents a Deep Reinforcement Learning (DRL) based structure for resource provisioning for improving resource management efficiency in IoT ecosystems. A Deep Neural Network (DNN) is used for assessing value functions, and it allows for better compliance to diverse conditions, learning from prior sensible approaches, and acting as a self-learning adaptive system. Using the DRL algorithm and the Proximal Policy Optimization (PPO), IoT services can be established by reducing average consumption of energy and latency, cutting expenses, and wisely utilising and allocating resources. Simulations with the iFogSim show that the PPO policy increases utilization, reduces delay rates, and maintains acceptable service quality while reducing energy consumption and increasing utilization under varying loading rates.*

Keywords - *Deep learning, Energy utilization, Proximal policy optimization, Neural network, Resource provisioning, Reinforcement learning.*

1. Introduction

Fog computing differs from classical computing in that it utilizes every online service. On the other hand, the traditional strategy depends on established infrastructure. The conventional method dictates that businesses spend money on hardware and software to provide services based on the demands of their clients. In short periods, predicting customer packages or dynamic workloads is challenging [1]. Quality of service may encourage potential clients to approach the organizations.

Fog computing deployments involve various challenges such as scaling, load balancing, resource scheduling, energy consumption, service availability, security, and quality of service management. Fog resource management can provide several advantages, such as increased energy efficiency, decreased network load, increased revenue, load balancing, and fewer Service Level Agreement (SLA) breaches. A scheduling approach reduces the risk of SLA violation and optimizes revenue allocation [2]. After using Fog Computing (FC) and Cloud Computing (CC) for a while and accumulating data, Machine Learning (ML)-based approaches have gained popularity [3]. When applying predictive solutions, it is essential to consider the complexity of these methods. Resource provisioning, job offloading, resource scheduling, load balancing, and application placement are the six scopes of recourse management [4].

The main objective of static problems is to discover or approximate the optimal points. For dynamic problems, it is essential to satisfy the static state's main goal and determine the optimal point/points as soon as possible. As a result, these issues are considered complex issues because they present more difficulties than static issues [5].

Therefore, the proposed algorithm should perform well in situations with uncertainty in the environment and track the optimum variable. The Algorithms that can adapt to the changing environmental conditions should be used in such cases. This study's primary goal is to lower latency and service provisioning costs for IoT requests and boost adoption in the fog setting. The aim is to create a novel resource organization method that uses neural networks and DRL approaches [6] to manage IoT applications. Using

previous experiences, this method can automatically automate scheduling overtime. Considering IoT network's dynamic nature and the complexity of their modelling, it is necessary to use a connected and flexible method for resource provisioning, which falls under the decision-making domain. This makes investigative and law-based approaches inapplicable. ML can use data without applying predetermined rules to estimate models with acceptable accuracy for complex systems.

A reinforcement learning approach involves an agent learning to select the best possible actions to maximize their cumulative reward by continuously interacting with their surroundings. In addition to being an essential sub-area of ML, deep learning represents the fundamental relationships between inputs and outputs, which makes it a good choice for online learning [7, 8]. The DRL technique is, therefore, a reasonable and appropriate choice for resource provisioning at the IoT instance level of fog computing involving IoT applications.

The new aspects of this work also include load balancing in various networks to improve service quality, choosing the best server with the fastest response time, and optimizing bandwidth usage to raise network proficiency and resource consumption. An outline for dynamic resource provisioning employing DRL is provided to process node resources effectively. The results of the simulation testing demonstrate that the suggested technique in this work outperformed the three primary algorithms related to average waiting and reaction times, job completion, and efficient use of resources through task allocation.

The remaining part of this article is structured as follows: Review the connected studies about resource provisioning in fog computing in section 2. The proposed solution is explained in section 3. The performance evaluation is presented in Section 4, the simulation results are shown in Section 5, and the conclusion is presented in Section 6.

2. Related Works

Several factors influence the performance and behaviour of heterogeneous fog environments, according to Stavrinides et al. [9]. A balanced schedule was achieved by considering deadline limitations and scheduling challenges and balancing IoT workloads. Dynamic programming improves system performance by analyzing input data probability locations. Assigning virtual machines and selecting tasks are done in two stages. A ranking method was developed by Naha et al. [10] to allot resources dynamically to fulfil the requirements of QoS constraints in fog devices.

The study examined various situations in fog devices, nodes, and cloud servers to determine their limitations and behaviours. A resource allocation algorithm automatically prioritizes resources based on user behaviours while minimizing processing duration, latency, and expenditure.

Fog systems and system distributions are limited in terms of resources, so this article emphasizes time-sensitive applications in which every application needs assistance from the fog environment due to limited processing power in the cloud environment. It ranks resources according to their time limits to help users find the information they need more quickly. It also prioritizes applications that need processing to ensure they are found as quickly as possible. Combining fog-cloud servers and devices is possible if more than enough fog devices are required to process data.

A model of a computation system developed by Dinh et al. [11] allows multiple fogs and cloud nodes to be leased. As part of this model, resources are allocated, edge processing costs and cloud leasing options are considered. It was suggested that their proposed offline algorithm would take advantage of future request information.

Arkian et al. [12] established an effective resource provisioning model based on a mixed-integer linear programming framework. Integer Non-Linear Programming (INLP), from which Mixed Integer Linear Programming (MILP) is formed. A part of their proposed model for fulfilling the job included task distribution, data customer relationships, and virtual machine assignment.

In the MILP formulation, cost minimization is defined as the distribution of activities and the location of virtual machines. Fog computing's results demonstrate a better-provisioned quality of services. This method should have considered Fog-layers sensitivity and privacy, making the plan less complete and practical.

According to Bahreini et al. [13], resource allocation and provisioning should be optimized to minimize energy costs to maximize provider profitability. They use edge computing to implement their heuristic-based method for addressing the MILP issue, which is inherently NP-hard.

Various issues with various sizes and configurations were examined to evaluate the method's performance. Based on the study's results, it is evident that the proposed algorithm is efficient regarding both execution time and solution quality.

With flying fog, Madan et al. [14] have done work for smart cities that will provision mobile network resources based on demand. To accomplish this, fog units are used to allocate and provision resources. Based on the preventive resource provisioning model, lease periods for allocated resources are determined. A fog-based micro data centre autoscaling model was proposed by Abdullah et al. [15] to increase resource organization and accelerate response times.

Their rule-based model preprocesses and postprocesses the training dataset to produce the relevant predictions. A decision tree regression model is then developed using the trained dataset.

Bahreini et al. [13] sought to minimize energy consumption and resource allocation in edge computational structures to maximize the turnover of the service provider. A meta-heuristic algorithm [16] is used to evaluate the MILP problem using extensive experimental analysis of the samples.

On-demand resource provisioning was proposed by Guo et al. [17]. Their load prediction solution utilized back propagation neural networks and an Auto-Regressive Integrated Moving Average (ARIMA) model to minimize assessed errors. In this model, user data must be transferred to further occupied nodes before liberating node resources to prevent data loss. As a result, they presented a method for migrating data by workload predictions. When the proposed method is applied, the service expenditures are reduced, and load balancing is improved.

The perception of service performance chains in FC and CC nodes was studied by Siasi et al. [18] to provision a fog-cloud hybrid architecture using SFC-based provisioning. Using a heuristic search method, they managed the basic parameters to meet the various latency requirements.

Li et al.'s [19] method of cost-effective resource scaling was put out for edge-cloud situations with fluctuating load. They created a scaling policy based on reservations and on-demand planning using deep belief networks and dual-phase planning. Integer programming is planned to solve the former, a stochastic problem turned deterministic.

According to Tadakamalla and Menace [20], fog servers have a limited processing capacity compared to cloud servers. Additionally, they pointed out that cloud resources are more expensive than fog servers [21]. The researchers examined the modelling challenges associated with IoTs in fog computing. A queueing network-based technique, FogQN, was then presented to analyze the cost and effectiveness of Fog using the cloud.

Based on the results, the controller may be handy when the arrival rate of requests varies widely. Using distributed computation, Faraji et al. [22] proposed a framework for managing resources autonomously using reinforcement learning and backup vector regression in a fog computing environment for time-varying workflows.

Abdullah et al. [15] have created an autonomous predictive scaling solution that utilizes Fog MDC to fulfil the SLO turnaround time requirement for microservices. A reactive rule-based algorithm is used in the proposed approach to gather training data sets automatically. As a result, a predictive automatic scale model can be constructed by preprocessing and postprocessing. It uses tree regression to develop a predictive model for automated scalability by increasing artificial workloads.

According to Mohammad Faraji et al. [23], As part of the planning phase, the proposed system uses learning automata, while in the analysis stage, a time series prediction model is used. Comparing the simulation results with other approaches, the simulation showed a minimum delay in service provisioning, lower overall costs, and lower SLA violations. As a result, fog computing can ensure Quality of Service (QoS).

The study by Liu et al. [24] sought to manage elastic resources by using error-correction load forecasts in edge/cloud environments in response to user requests. Their approach centred around workload forecast (ARMA and ENN) and workload migration, combining error correction and workload migration models to advance prediction precision and reduce migration time [25].

Al-Makhadmeh et al. [26] Developed an efficient and scalable resource provision model based on DL. Additionally, it devised a practical structure for balancing resource demands and user density. The result is high reliability and customer satisfaction with IoT services.

A framework for managing cloud resources in intelligent homes, called ROUTER, was proposed by Gill et al. [27]. They focused on performance improvement for overloaded edge environments with high arrival rates. Examining the interaction between a fog data server and IoT devices, as well as between the particle's optimization algorithm and the cloud data server, is part of the design model for a smart home. The toolkit iFogSim was used to test and evaluate the IoT-based innovative home automation approach they developed. Their analysis found that it reduced network bandwidth by as much as 12%, response time by as much as 10%, latency by 14%, and energy consumption by 12.35%.

Yousefpour et al. [28] examined QoS in delay-sensitive requests. They devised the FOGPLAN structure as a solution and looked at how it would help Fog Service Providers (FSP) regarding cost-savings and QoS. To examine dynamic FSPs with QoS considerations, the researchers used two greedy methods, Min-Viol and Min-Cost, using an INLP formulation. According to our results, the Min-Viol algorithm performs better but is slower.

Both algorithms are nearly equal in asymptotic complexity. Table 1 contrasts the study on resource supply in fog computing based on the suggested classification.

Table 1. Survey on various approaches to resource provisioning technique

Reference	Techniques	Performance Parameters	Benefits
Georgios L. Stavrinides et al. [9]	Queuing Model	Response Time	For High-Priority Tasks, A Priority Policy is Available
Ranesh Kumar Naha et al. [10]	Heuristic Method	Processing Time, Delay, Cost	Low Cost and Processing Time
Thinh Quang Dinh et al. [11]	Polynomial Approximation	Cost	Excellent Mathematical Presentation for the Allocation Algorithms
Hamid Reza Arkian et al. [12]	Integer Linear Program	Cost, Power Usage, and Operation Delay	Reduced Power Usage and Service Latency, Support for Quick Responses at Various Levels
Tayebeh Bahreini et al. [13]	Integer Linear Program	Energy Usage and Expense	High scalability and fast execution
Naman Madan et al. [14]	Heuristics (Distance-Based)	Energy, Latency	Appropriate system model
Muhammad Abdullah et al. [15]	Decision Tree Algorithm (DTR)	Response Time, SLO-Violations, Rejected Requests	Enhanced Resource Provisioning and Prediction
Tayebeh Bahreini et al. [13]	Heuristics (Iterative Based)	Energy, Profit	Simple System Model with Scaling
Jingjing Guo et al. [17]	ARIMA Model, BP Neural Network	Relocation Time and Cost	Reduced Service Costs and a Cluster that is Load-Balanced
Nazli Siasi et al. [18]	Heuristics (Graph-Based)	Energy, Delay, Cost	Effective Organization, In-depth Analysis
Chunlin Li et al. [19]	Integer Programming, Genetic Algorithm	Cost	Relevant Algorithms with a Suitable Level of Complexity
Uma Tadakamalla, Daniel A. Menascé [20]	Autonomic Computing, Queuing Model	Response Time, Cost	The Utility of this Method is High when Request Arrival Rates vary Widely
Mohammad Faraji Mehmandar et al. [22]	ML (RL, SVR)	Cost, Delay Violation	A Thorough Illustration of the MAPE-K Loop Model
Muhammad Abdullah et al. [15]	Machine Learning (DTR)	Response Time	Comprehensive Analysis
Mohammad Faraji et al. [23]	Autonomic Computing	Cost, SLA Violation, Service Delay	Formulating Problems Effectively for Cost and Delay a Thorough Self-Management Model
Boyun Liu et al. [24]	Machine Learning (ARMA, ENN)	Prediction Precision and Migration Time.	Improved Prediction Precision and Reduced Migration Time.
Al-Makhadmeh et al. [26]	Machine Learning (Deep Learning)	Delay, Response Time	Maximizes the Consistency of IoT Services and User Satisfaction
Sukhpal Singh Gill et al. [27]	Particle Swarm Optimization	Response Time, Network Throughput, Power Usage, and Latency	Reductions in Energy Use, Network Capacity, and Delay
Ashkan Yousefpour et al. [28]	Greedy Algorithms, Framework-Based, Optimization Problem	Delay, QoS, Cost	Improved QoS for Time-Sensitive Apps

3. Proposed Work

This section will implement the proposed approach using a resource provisioning framework. Figure 1 shows the proposed fog computing model consisting of three layers. Sensors and IoT devices make up the first layer. This layer supports intelligent devices like mobile phones, sensors, and tablets. As well as providing Ad Hoc services, Fog estimates resource consumption and allocates estimated resources. The Fog layers need to perform computation, storage, and processing operations in this layer. An access point receives a user's request initially, after which it is sent to an admission control component. When a request exceeds the threshold value, it is sent to the gateway to be processed in the cloud and is neither real-time nor latency-sensitive. The data must be transferred to the fog layer for real-time processing [29]. Fog node allocation and resource provisioning are responsibilities of the fog layer.

Virtual machines of cloud data centres are used to process massive data or non-time-sensitive requests in the cloud layer. There are three layers to the projected resource provisioning model, and the main component controls them all.

Figure 1 shows the architectural diagram of the proposed work. Several components are included in the proposed controlling component to manage the fog layers automatically and efficiently.

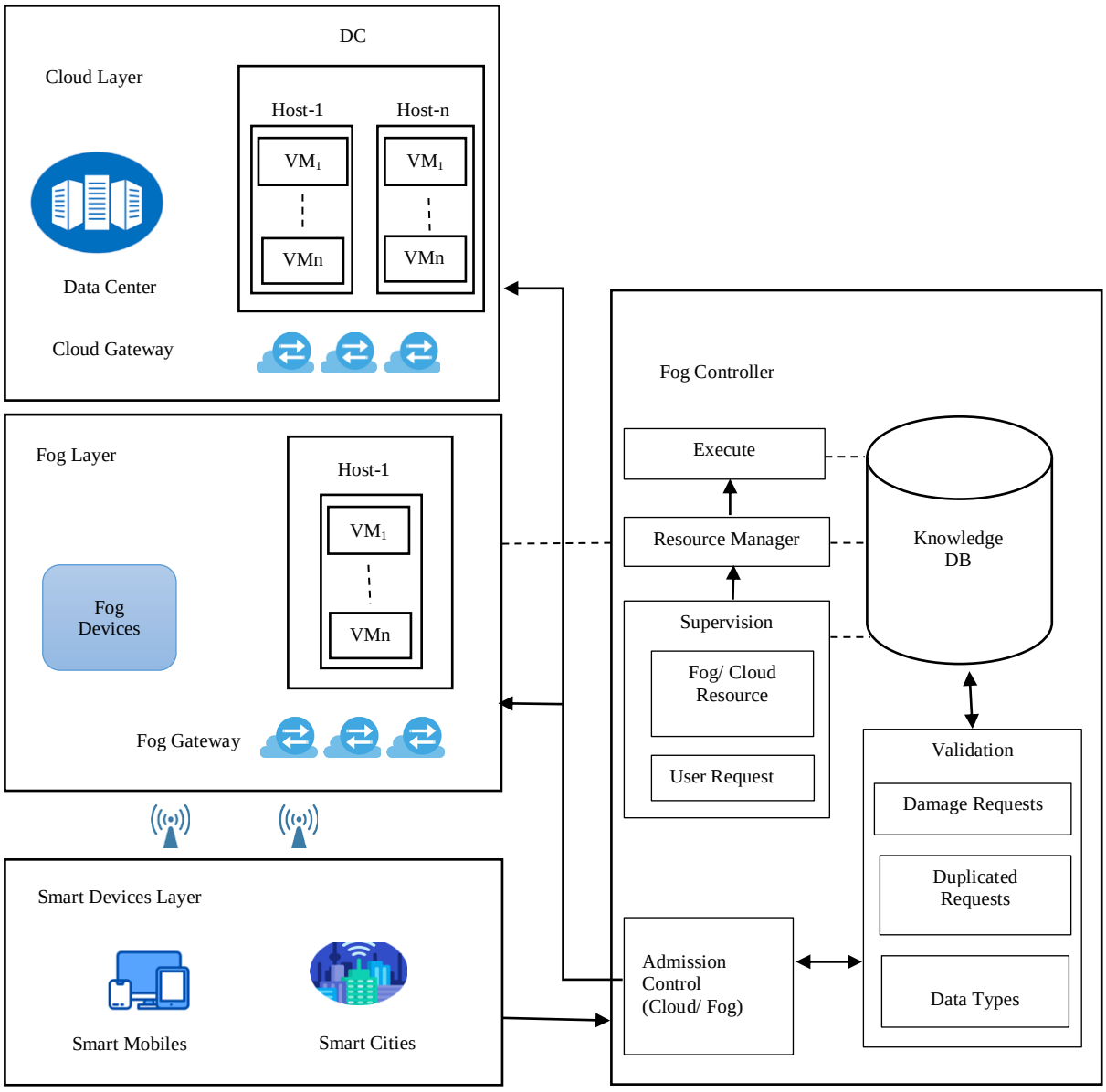


Fig. 1 Architectural diagram of the proposed approach

Supervision: Criteria associated with IoT devices and fog and cloud layers are collected by this component. The supervision component processes data by communicating with the IoT layer, which provides the necessary CPU, storage, and RAM—resource information, such as CPU efficacy storage spaces. Data centres and fog devices can gather Internet traffic and active/inactive services.

Validation: Data from the supervision section is investigated by this unit. It comprises checking the data type unit, replicate and destructive requests.

Knowledge Database: Input data, such as requests made by IoT devices, fog devices, and cloud data centres, is maintained by the DB unit. After reviewing this data, the validation component updates it.

Admission Control: The admission manager unit uses database data to verify the location of IoT devices when processing requests. The cloud layer should be used if the deadline for the request exceeds a certain threshold.

The execution unit receives the request and sends it to the data resources management unit for processing and storage in the cloud data centre. If not, the request must be forwarded to the fog layer's Policy Maker component.

Resource Manager: Using the DRL technique, this component manages and decides regarding fog layer resource provisioning and processes incoming requests through the Admission Manager Unit. In response to the request, it chooses how to deliver resources.

Execute: It executes decisions made by Policy Makers. The segment specifies that the request should be managed through the admission manager in the fog or cloud layer. Additionally, the resource manager determines how many resources are required for fog layer requests. A final step in the process is to send the request to the destination for execution.

3.1. Resource Provisioning using Deep Reinforcement Learning

Learning agents in reinforcement learning receive rewards after each action is evaluated. The reinforcement learning algorithm and the environment are two decision-making elements in the dynamic resource provisioning problem.

An agent reacts to the environment according to knowledge by appropriately increasing, decreasing, or selecting resources. It determines the early number of resources for each service. The environment concurrently notifies the agent of its state and rewards it for its prior action [30].

Decision makers update their knowledge based on the reward they received for their previous actions. The cycle lasts until the agent receives the last status from the environment.

Analyzing the changes in reinforcement learning approaches results will allow us to see how our decisions will affect them. By better understanding the dynamics of the system or by trial and error, it can be determined what decisions work best.

In a fog ecosystem, resources communicate through a graph G = (N, A), where N is the set of resources, and A is the set of communication links between them. Communication link delays (in milliseconds) are affected by propagation delays, bandwidth delays, queueing delays, and network traffic conditions.

3.2. Deep Neural Network

A trained network (DNN) can approximate resource allocation policies that change over time. IoT service ecosystem status is used as input to approximate the DNN network. The output network is generated using policy parameters as weights and biases to generate the probability of selecting the necessary actions. This Figure 2 illustrates a basic model for the production of policies.

Each request in the proposed algorithm is assigned an initial policy or action-practical relationship. DNNs sense the state of a system by interacting with its environment, and they analyze what may occur in the future. Parameters are set to maximize reward.

Multi-iteration learning is used to learn the optimal policy for better provisioning resources. Resource provisioning expands by building a database, which is then utilized to advance the parameter policy parameters and give new requests. PPO uses a gradient approach to optimize the policy function that maps states to actions. A clipping operator ensures that policy updates fall within a specific range.

Additionally, it prevents policies from becoming greedy and making unwise decisions. Therefore, the maximum cumulative reward is generated during resource provisioning by obtaining optimal weights and biases after several training repetitions. The following Figure 2 shows the basic DNN policy model.

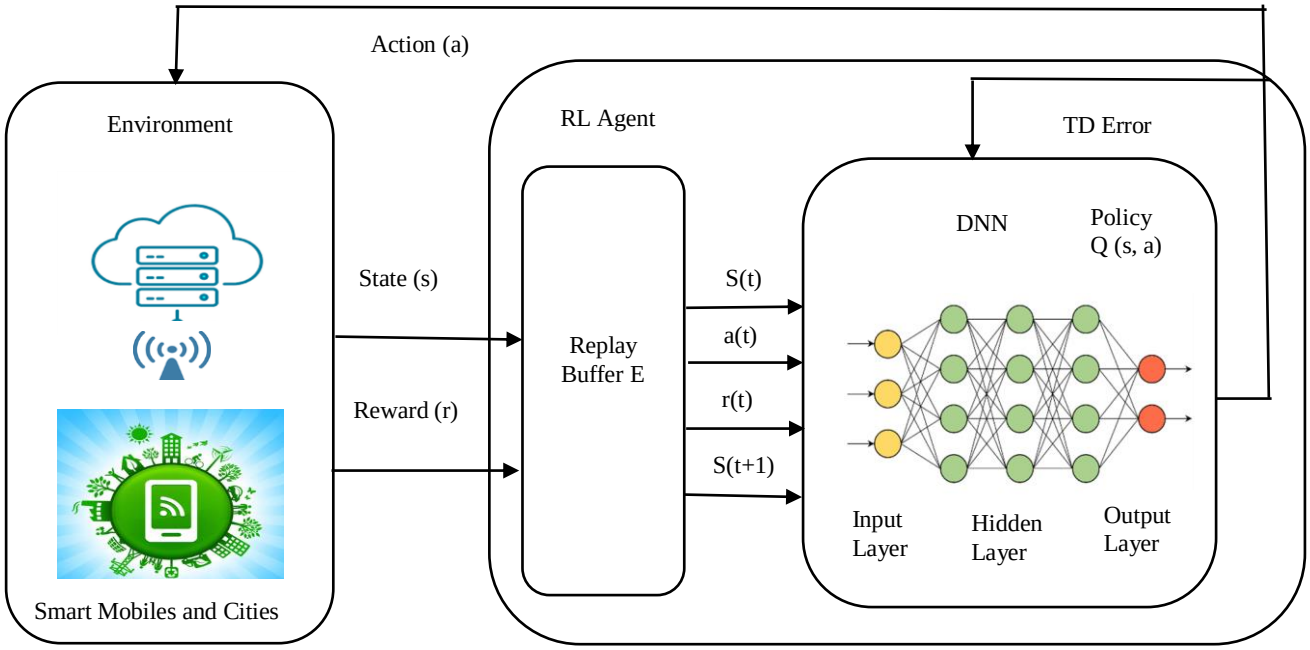


Fig. 2 DNN-based policy model

Overall, the procedure follows the Deep Q Network (DQN) framework by iteratively interacting with the environment, collecting experiences and using them to update the main DQN. By combining exploration and exploitation, the agent learns an optimal policy for exploiting collective rewards in the given environment.

DRL algorithm for Resource Provisioning

Initialization:

Initialize experience playback memory;

Initialise behaviour value function Q with random weight θ;

Initialize the target behaviour value function Q with weight $\theta_w = \theta'$.

Begin

1. For episode i = 1, 2, . . ., I

2: do the initial observation s_1 *is received and the preprocessing* s_1 *is taken as the start state* x_1

3: For t = 1, 2, . . ., T

4: Using random probability α, select behaviour at random;

5: Otherwise, select behavior: at = arg maxQ (x, a; θ);

6: Obtain reward rt by Executing actions in the system, observe st+1 at the next moment, and update st+1 to xt+1;

7: Playback memory is created by storing experience;

8: Obtain samples from playback memory in small random batches;

9: Calculate the target DQN Q value;

10: Minimizing the loss function L(θ) and update the main DQN;

11: Gradient descent is performed on L(θ) on network parameter θ,

12: Update the target Q value of the network.

13: End For

14: End For

15: End

4. Performance Evaluation

This section discusses the DRL-based Fog Resource Provisioning method for provisioning resources in a fog environment. The iFogSim [31] performs accurate and extensive fog simulation and helps evaluate resource provisioning and management policies in Edge, Fog, and CC. Using this tool, different situations can be handled for evaluating resource management and provisioning strategies in different computing environments. This simulation examines latency, energy consumption, operational cost, and utilization of resource management policies. In addition to supporting edge/ fog devices, network links and cloud computing, this tool also supports productivity evaluation. iFogSim supports the Sense-Process-Actuate model. Data generated by IoT is monitored in this model. Once the fog tools have executed and processed the programs, the closing decisions are moved to the data transfer space.

4.1. Simulation Settings

In stimulation, artificial workloads are changing rapidly. There are 250 requests to 1450 services in the varying workload. To create this load, several random peaks must

first be defined. A threshold is then determined based on the peak. A random procedure increases the number of services to a peak and decreases it to a certain amount similarly. There are 730 services on average in this workload. Table 2 shows the details of the simulation settings. The following Table 2 Illustrate the details of simulation settings.

Table 2. Simulation settings

Parameter	Description	Latency
T_i	The maximum time required for processing in a fog environment	10ms
QoS_i	The Quality of Service (Desired)	U (90, 99.99) %
U_{ai}	CPU demand (a_i)	U (50, 200) MI/ req
M_{ai}	Memory demand (a_i)	U (2, 400) MB
S_{ai}	Storage demand of the AI	U (50, 500) MB
U_{fj}	Processing power (fog node j)	U (800, 1400) MIPS
M_{fj}	Memory size (fog node j)	U (2, 400) MB
S_{fj}	Storage size (fog node j)	≥ 25MB
C^{U}_{FN}	The price per unit of process (fog node j)	0.002 /MI
C^{S}_{FN}	The price per unit of storage (fog node j)	0.003 Gb/sec
C^{M}_{FN}	The price per unit of the main memory (fog node j)	0.004 Gb/sec
Core, Edge link	---	10 Gbps,1 Gbps
$Pd_{(j, k)}$	Propagation delay of link (j,k)	U (15, 35) ms
$Pd_{(IoT, k)}$	Propagation delay among IoT and fog nodes	U (1, 2) ms
$Size_i^{req}$	Average service request length of i	U (10, 25) KB
$Size_i^{res}$	Average service reply length of i	U (10, 20) KB

4.2. Simulation Parameters

This evaluation uses four metrics: Average latency, cost, mean energy consumption and utilization.

4.2.1. Latency

During deployment, the propagation and transmission times are added together. This results in the mean implementation period (processing delay) and the service time (response time).

L = Uploading_ Delay (L1) + Transfer_ Delay (L2) + Processing_ Delay (L3) (1)

Where L1+ L2 is the deployment time, and L3 represents the execution time.

4.2.2. Cost

According to Eq. 2, a service's cost is assigned to the node based on interval. The total cost can be calculated using the communication, computation, deployment and penalty cost.

C = Communication Cost (CC1) + Computation Cost (CC2) + Deployment Cost (DC3) + Penalty Cost (PC4) (2)

CC1 is communication cost, CC2 is Computation cost, DC3 is Deployment cost, and PC4 is Penalty cost.

4.2.3. Energy Consumption

A combination of service latency and geographical location is estimated to determine the energy consumed by communication between fog nodes. Additionally, a linear equation determines how much computational energy a fog node consumes. Fog node CPU usage and maximum dynamic energy consumption are used for this calculation.

E = Communication_ Energy (E1) + Computation_ Energy (E2) (3)

4.2.4. Utilization

In fog nodes, utilization is derived by dividing the allocated MIPs by the available MIPs in the nodes.

$$\text{Utilization} = \frac{\text{Allocated MIPS}}{\text{Available MIPS}} \quad (4)$$

5. Result and Discussions

The proposed method is evaluated using the base algorithms Fog Resource Provisioning Learning Automata

(FRP_LA), Fog plan and Fog Resource Provisioning Reinforcement Learning (FRP_RL) methods.

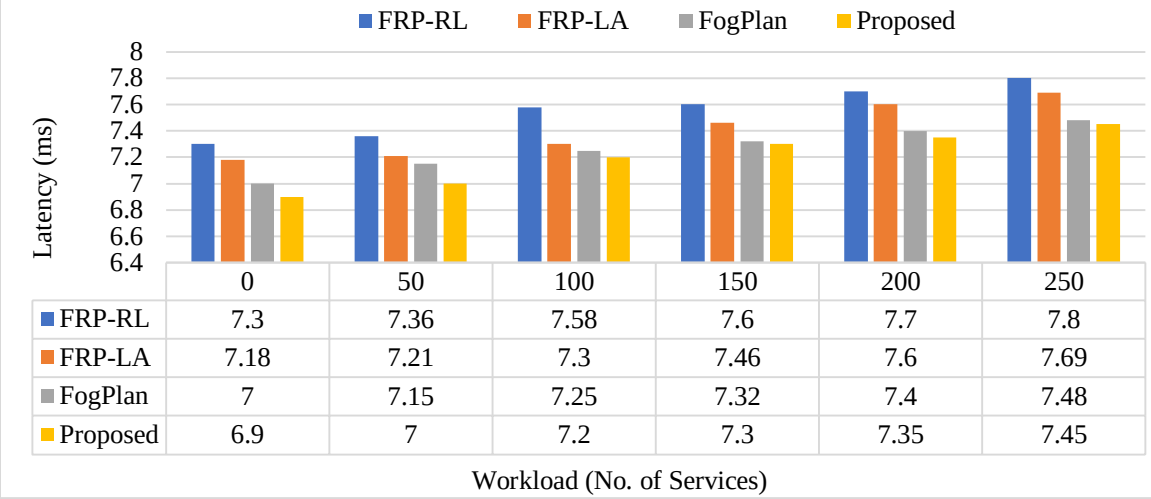
A Learning Automata approach is combined with the framework for provisioning resources proposed in this study, FRP_LA. Analyses are conducted using time series prediction methods, followed by planning in which learning automation methods are used. As one of the reinforcement algorithms, this method was chosen. Convergence times are excellent and are used in situations with variable requests.

As part of FRP_RL [32], the fog environment uses reinforcement learning to plan service allocation. The choices are made at random in the policy exploration section. A comparison of this algorithm to the proposed method will help to evaluate its performance. Performance depends on determining the agent's intellect and the effects of learning from prior experiences.

An analysis of the proposed approach is undertaken by contrasting the suggested method with the FogPlan method, which employs a greedy algorithm, and the Router method, which employs an optimization algorithm. These approaches are all container-based technologies that enable scalable service deployment by presenting a variety of applications and services.

5.1. Latency vs Workload

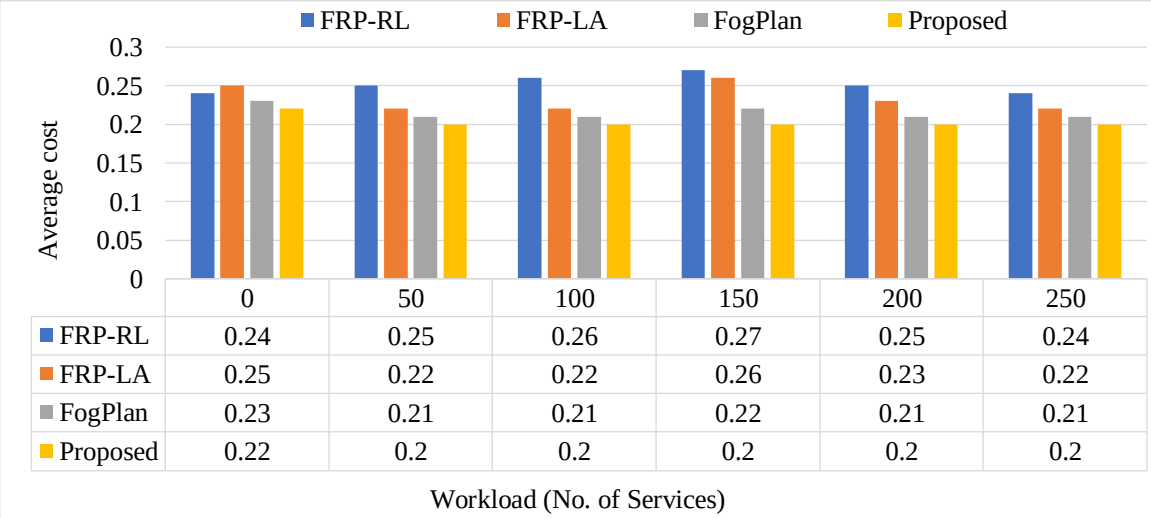
The provisioning of resources is primarily based on response time, which is one of the most effective service objectives. Resource provisioning must be done again if the requested response time cannot be completed. According to Figure 3, the variable workload method has a shorter average response time than FRP_RL, FRP_LA and FogPlan algorithms. As a result of the projected method, the desired response time is achieved because the response time equals the time between the request's arrival and its exit.



	0	50	100	150	200	250
FRP-RL	7.3	7.36	7.58	7.6	7.7	7.8
FRP-LA	7.18	7.21	7.3	7.46	7.6	7.69
FogPlan	7	7.15	7.25	7.32	7.4	7.48
Proposed	6.9	7	7.2	7.3	7.35	7.45

Fig. 3 Average latency of different workloads

5.2. Cost vs Workload



	0	50	100	150	200	250
FRP-RL	0.24	0.25	0.26	0.27	0.25	0.24
FRP-LA	0.25	0.22	0.22	0.26	0.23	0.22
FogPlan	0.23	0.21	0.21	0.22	0.21	0.21
Proposed	0.22	0.2	0.2	0.2	0.2	0.2

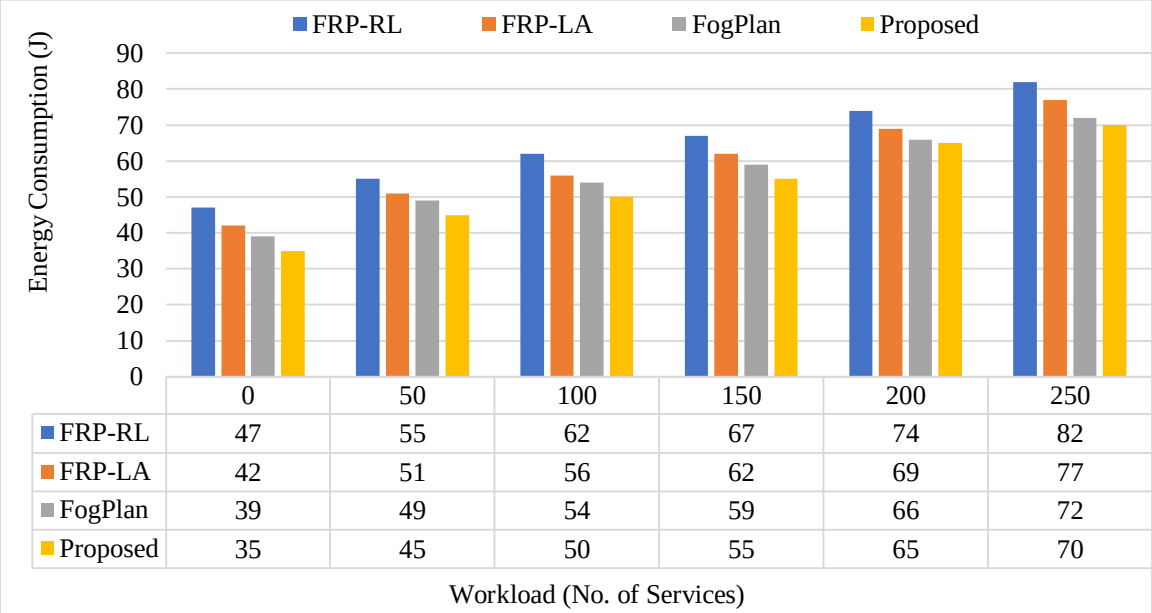
Fig. 4 Average cost of different workload

A comparison will be made between FRP_RL and FRP_LA—moreover, FogPlan algorithms to assess the cost of the new algorithm. Resource provisioning algorithms are primarily measured by their costs. Compared with FRP_RL, FRP_LA and FogPlan algorithms,

Figure 4 illustrates the cost of the new technique in different workloads. According to the results, setting appropriate resource provisioning limits and using deep reinforcement learning for decision-making will increase precision. However, using an efficient monitoring system for node information is the main reason for cost reduction. Fog resource provisioning becomes more precise as a result.

5.3. Energy Consumption vs Workload

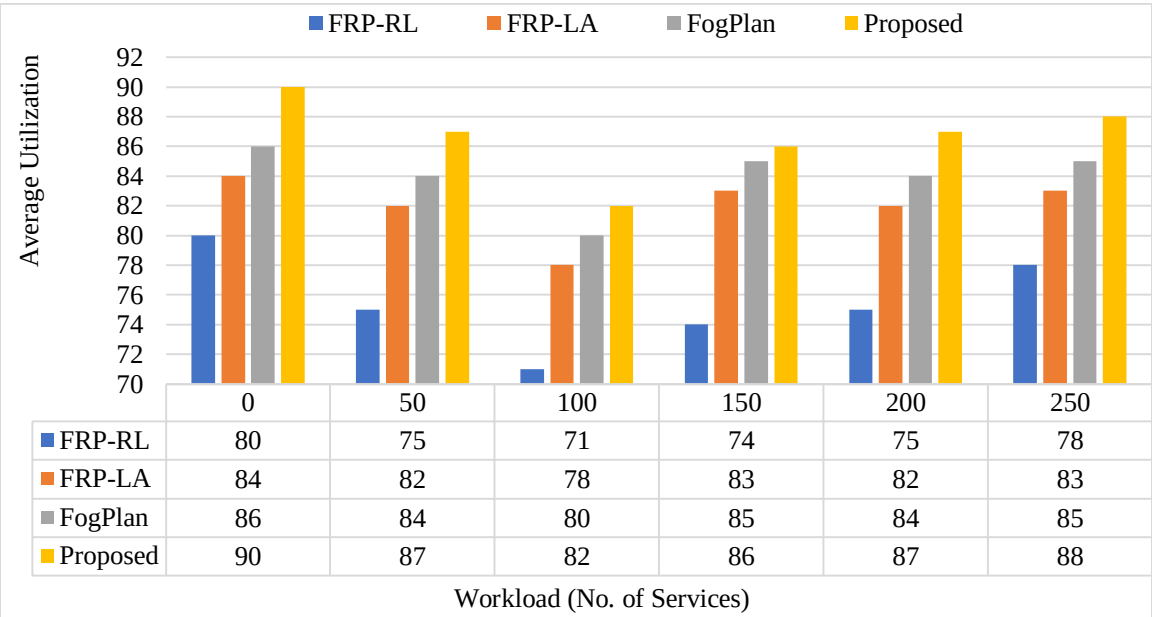
Comparing the proposed method with FRP_RL, FRP_LA and FogPlan in a variable workload structure, Figure 5. It shows its average energy consumption. The provider determines how to provide the resources based on user requests.



	0	50	100	150	200	250
FRP-RL	47	55	62	67	74	82
FRP-LA	42	51	56	62	69	77
FogPlan	39	49	54	59	66	72
Proposed	35	45	50	55	65	70

Fig. 5 Average energy consumption of different workloads

5.4. Utilization vs Workload



	0	50	100	150	200	250
FRP-RL	80	75	71	74	75	78
FRP-LA	84	82	78	83	82	83
FogPlan	86	84	80	85	84	85
Proposed	90	87	82	86	87	88

Fig. 6 Average utilization of different workload

Nodes consume energy proportionally to their operational cost, which is determined by CPU, memory, and disk consumption. In this case, operational costs are the target function for calculating energy consumption. The method developed in this work effectively distributes requests and services across virtual computers, reducing idle time, boosting resource usage, and determining the service location with the shortest latency.

In addition, the proposed method consumes less energy than the other three methods since it uses a machine with a lesser energy consumption. At the same time, timely identification and migration of virtual machines from physical machines prevent excessive overloading of these servers, lowering their energy consumption.

Figure 6 compares the new method with FRP_RL, FRP_LA and FogPlan for average utilization in a variable workload structure. In the study, the proposed method meets the desired productivity level. Detecting workloads continuously and monitoring fog node performance is essential for controlling fog cells. Provisioning is done correctly due to monitoring node status and using deep reinforcement learning decision-makers. It has always been possible to migrate virtual machines with less than three migrations at all times since overloaded physical machines have been identified and appropriate servers with a high free processor capacity have been selected to locate them.

6. Conclusion

There has been significant growth in IoT devices in recent years; they are so large data, and IoT applications need rapid responses to their requests. These types of data are increasingly being processed using edge computing. IoT devices fluctuate in foggy situations. Therefore, defining the load condition of physical equipment and, inevitably, scale resources might be helpful. It is feasible to prevent concerns with over- and under-provisioning by determining the load circumstances. We need a technique that handles more requests quickly and without delay because fog environments change and user's requests alter over time. This type of environment can be adapted to by reinforcement learning, which can provide the most accurate mapping of services to resources as a result of learning. In addition, it is capable of managing IoT services efficiently.

The study suggests a framework for resource provisioning and communications between units for IoT devices, nodes connected to the Fog, and cloud servers. It is based on deep reinforcement learning techniques and algorithms. iFogSim simulator was used to assess and compare the proposed method with three other provision methods (Fogplan, FRP_RL, and FRP_LA). Enhance efficiency and reduce response time, cost, and energy consumption. Further research can be done by using prediction techniques like neural networks to improve accuracy in the future.

Moreover, they can be combined to identify optimal physical machines and estimate input workloads. Combining deep reinforcement learning and proximal policy optimization enables the policy to be learned and optimized directly by mapping states to actions. Integration of reinforcement learning with fuzzy logic and correlation learning, automatic resource scaling with service relocation between fog nodes, and integration of automatic resource scaling with resource placement can all considerably enhance the suggested system.

References

[1] S. Aiswarya et al., "A Time Optimization Model for the Internet of Things-Based Healthcare System using Fog Computing," *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, Chennai, India, pp. 1-6, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Kaouther Gasmi et al., "A Survey on Computation Offloading and Service Placement in Fog Computing-Based IoT," *The Journal of Supercomputing*, vol. 78, no. 2, pp. 1983-2014, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[3] R. Surendiran, and K. Raja, "A Fog Computing Approach for Securing IoT Devices Data using DNA-ECC Cryptography," *DS Journal of Digital Science and Technology*, vol. 1, no. 1, pp. 10-16, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[4] Mostafa Ghobaei-Arani, Alireza Souri, and Ali A. Rahmanian, "Resource Management Approaches in Fog Computing: A Comprehensive Review," *Journal of Grid Computing*, vol. 18, no. 1, pp. 1-42, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[5] Sukhpal Singh, and Inderveer Chana, "Resource Provisioning and Scheduling in Clouds: QoS Perspective," *The Journal of Supercomputing*, vol. 72, no. 3, pp. 926-960, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Soheil Hashemi, and Mani Zarei, "Internet of Things Backdoors: Resource Management Issues, Security Challenges, and Detection Methods," *Transaction on Emerging Telecommunications Technologies*, vol. 32, no. 2, p. e4142, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] A. V. Dastjerdi et al., "Chapter 4 - Fog Computing: Principals, Architectures, and Applications," *Internet of Things, Principles and Paradigms*, pp. 61-75, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[8] A. Lakshna et al., "Smart Navigation for Vehicles to Avoid Road Traffic Congestion using Weighted Adaptive Navigation* Search Algorithm," *SSRG International Journal of Electronics and Communication Engineering*, vol. 10, no. 5, pp. 170-177, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[9] Georgios L. Stavrinides, and Helen D. Karatza, "Orchestration of Real-Time Workflows with Varying Input Data Locality in a Heterogeneous Fog Environment," *2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC)*, Paris, France, pp. 202-209, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[10] Ranesh Kumar Naha et al., "Deadline-Based Dynamic Resource Allocation and Provisioning Algorithms in the Fog-Cloud Environment," *Future Generation Computer Systems*, vol. 104, pp. 131-141, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[11] Thinh Quang Dinh et al., "Online Resource Procurement and Allocation in a Hybrid Edge-Cloud Computing System," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2137-2149, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Hamid Reza Arkian, Abolfazl Diyanat, and Atefe Pourkhalili, "MIST: Fog-Based Data Analytics Scheme with Cost-Efficient Resource Provisioning for IoT Crowdsensing Applications," *Journal of Network and Computer Applications*, vol. 82, pp. 152-165, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[13] Tayebeh Bahreini, Hossein Badri, and Daniel Grosu, "Energy-Aware Capacity Provisioning and Resource Allocation in Edge Computing Systems," *International Conference on Edge Computing, Edge Computing – EDGE 2019*, pp. 31-45, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[14] Naman Madan et al., "On-Demand Resource Provisioning for Vehicular Networks using Flying Fog," *Vehicular Communications*, vol. 25, p. 100252, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[15] Muhammad Abdullah et al., "Predictive Autoscaling of Microservices Hosted in Fog Micro Data Centre," *IEEE Systems Journal*, vol. 15, no. 1, pp. 1275-1286, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Ruchika, and Rajender Singh Chhillar, "Comparison of Meta-Heuristic Optimization Algorithms for Solving Optimized Task Scheduling Problems in Fog Environment," *International Journal of Engineering Trends and Technology*, vol. 71, no. 3, pp. 175-183, 2023. [CrossRef] [Publisher Link]

[17] Jingjing Guo et al., "On-Demand Resource Provision Based on Load Estimation and Service Expenditure in Edge Cloud Environment," *Journal of Network and Computer Applications*, vol. 151, p. 102506, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] Nazli Siasi et al., "Delay-Aware SFC Provisioning in Hybrid Fog-Cloud Computing Architectures," *IEEE Access*, vol. 8, pp. 167383-167396, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[19] Chunlin Li, Jingpan Bai, and Youlong Luo, "Efficient Resource Scaling Based on Load Fluctuation in Edge-Cloud Computing Environment," *The Journal of Supercomputing*, vol. 76, pp. 6994-7025, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[20] Uma Tadakamalla, and Daniel A. Menascé, "Autonomic Resource Management using Analytic Models for Fog/Cloud Computing," *2019 IEEE International Conference on Fog Computing (ICFC)*, Prague, Czech Republic, pp. 69-79, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[21] Harikrishnan Natarajan, and P. Ajitha, "Truthful Bidding for Cloud Resources Based on Competitive Cloud Auction, Costing and Depreciation," *SSRG International Journal of Computer Science and Engineering*, vol. 3, no. 3, pp. 9-12, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[22] Mohammad Faraji Mehmandar, Sam Jabbehdari, and Hamid Haj Seyyed Javadi, "A Dynamic Fog Service Provisioning Approach for IoT Applications," *International Journal of Communication Systems*, vol. 33, no. 14, p. e4541, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[23] Mohammad Faraji-Mehmandar, Sam Jabbehdari, and Hamid Haj Seyyed Javadi, "A Proactive Fog Service Provisioning Framework for Internet of Things Applications: An Autonomic Approach," *Transaction on Emerging Telecommunications Technologies*, vol. 32, no. 11, p. e44342, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[24] Boyun Liu et al., "Workload Forecasting Based Elastic Resource Management in Edge Cloud," *Computers & Industrial Engineering*, vol. 139, p. 106136, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[25] Nelli Chandrakala, and Vamsidhar Enireddy, "Provisioning of Defensing Mechanism against Threats during VM Migration in Cloud Environment," *International Journal of Engineering Trends and Technology*, vol. 70, no. 6, pp. 1-12, 2022. [CrossRef] [Publisher Link]

[26] Zafer Al-Makhadmeh, and Amr Tolba, "SRAF: Scalable Resource Allocation Framework using Machine Learning in User-Centric Internet of Things," *Peer-to-Peer Networking and Applications*, vol. 14, no. 4, pp. 2340-2350, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[27] Sukhpal Singh Gill, Peter Garraghan, and Rajkumar Buyya, "ROUTER: Fog Enabled Cloud-Based Intelligent Resource Management Approach for Smart Home IoT Devices," *Journal of Systems and Software*, vol. 154, pp. 125-138, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[28] Ashkan Yousefpour et al., "FogPlan: A Lightweight Qos-Aware Dynamic Fog Service Provisioning Framework," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5080-5096, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[29] Aiswarya S et al., "Internet of Health Things: A Fog computing Paradigm," *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, pp. 598-604, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[30] S. Aiswarya et al., "Latency Reduction in Medical IoT using Fuzzy Systems by Enabling Optimized Fog Computing," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 9, no. 12, pp. 156-166, 2022. [CrossRef] [Publisher Link]

[31] Harshit Gupta et al., "iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in the Internet of Things, Edge and Fog Computing Environments," *Journal of Software: Practice and Experience*, vol. 47, no. 9, pp. 1275-1296, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[32] Xiaoheng Deng et al., "Task Allocation Algorithm and Optimization Model on Edge Collaboration," *Journal of Systems Architecture*, vol. 110, p. 101778, 2020. [CrossRef] [Google Scholar] [Publisher Link]