International Journal of Computer Science and Engineering Development
Volume 1, Issue 5 | November - December 2026 | www.ijcsed.com
ISSN: **3139-0862**

A Comparative Study of Load Balancing Techniques in Edge-Based Computing Networks

Davies Isobo Nelson[#1], Cookey Ibiere Boma[*2], Taylor Onate Egerton[#3]
[#123]*Department of Computer Science, Rivers State University, Nigeria*
[1]isobo.davies@ust.edu.ng , [2]cookey.ibiere@ust.edu.ng , [3]taylor.onate@ust.edu.ng

Abstract—

The rapid proliferation of Internet of Things (IoT) devices, autonomous systems, and latency-sensitive applications has driven the adoption of edge-based computing networks as a viable alternative to centralized cloud infrastructure. In these distributed environments, effective load balancing is paramount to ensuring optimal resource utilization, minimizing task completion latency, and enhancing overall system throughput. This paper presents a comprehensive comparative study of load balancing techniques applicable to edge-based computing networks, encompassing static, dynamic, and intelligent approaches. Specifically, the study analyzed Round Robin, Least Connection, Weighted Least Connection, Min-Min, Max-Min, Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Deep Reinforcement Learning (DRL)-based methods. Each technique is evaluated across key performance metrics including latency, resource utilization, scalability, energy efficiency, and fault tolerance. The study further discusses architectural challenges unique to edge environments including node heterogeneity, intermittent connectivity, and constrained computational resources, and examine how state-of-the-art approaches address these constraints. Results from the surveyed literature indicate that DRL-based load balancing consistently outperforms classical algorithms in dynamic, heterogeneous edge environments, yet incurs greater training overhead. Taxonomy is proposed to guide practitioners in selecting appropriate strategies aligned with specific deployment contexts. Finally, this paper identifies open research challenges and future directions, including the integration of federated learning, energy-aware scheduling, and 6G-enabled edge computing.

Keywords—Edge Computing, Load Balancing, Mobile Edge Computing, Deep Reinforcement Learning, Task Offloading, Resource Management, IoT, Quality of Service.

I. INTRODUCTION

The exponential growth of connected devices and computation-intensive applications has placed unprecedented demands on traditional cloud computing infrastructure. Modern latency-critical applications such as autonomous vehicles, augmented reality, real-time industrial automation, and remote healthcare, require end-to-end communication delays of less than 10 milliseconds. Such rigorous performance requirements exceed the capabilities of conventional cloud data centers, whose physical distance from end users often introduces unavoidable network delays that hinder ultra-low-latency service delivery [1]. However, in recent years, edge computing has emerged as a transformative paradigm that brings computational and storage resources to the periphery of the network, usually closer to where data is generated and also consumed. By processing data at or closer the source, edge-based computing networks dramatically reduce latency, conserve backhaul bandwidth, and improve the responsiveness of latency-critical services [2].

Additionally, a key instantiation of the edge computing paradigm is the Mobile Edge Computing (MEC). It extends cloud-like capabilities to base stations and access points within mobile networks. The European Telecommunications Standards Institute (ETSI) standardized MEC to enable low-latency application execution at the radio access network (RAN) edge [3]. Alongside MEC, fog computing extends cloud intelligence to intermediate network tiers between end devices and the cloud core, thereby creating a continuum of computing resources that can be orchestrated for optimal service delivery [4].

However, despite these advances, edge-based computing networks introduce unique resource management challenges. Unlike cloud environments that offer virtually unlimited, homogeneous resources, edge nodes are typically constrained in compute, memory, and energy capacity. Workloads arriving at edge nodes are dynamic and often bursty, creating load imbalances that degrade performance and user experience. Without effective load balancing, individual edge nodes may become bottlenecks while neighboring nodes will eventually remain idle, leading to service level agreement (SLA) violations and increased energy waste [5].

In edge computing, the term load balancing is used to refer to as an intelligent distribution of incoming tasks or service requests across available edge nodes to optimize resource utilization, minimize response time, and maximize system throughput. Given the distributed, heterogeneous, and dynamic nature of edge environments, conventional cloud load balancing algorithms which are designed for stable in high-capacity environments often perform sub-optimally. Therefore, specialized techniques that account for node heterogeneity, network dynamics, and real-time workload fluctuations are essential [6].

This paper present a systematic comparative study of load balancing techniques in edge-based computing networks. The comparison encompasses classical static methods, dynamic heuristics, meta-heuristic optimization algorithms, and modern artificial intelligence (AI)-driven approaches. The goal is to:

i. Provide a structured taxonomy of load balancing techniques for edge computing, categorizing them by algorithmic approach, operational characteristics, and suitability for specific edge deployment scenarios.

ii. Analyze and compare eight representative load balancing algorithms such as spanning static,

ISSN: **3139-0862**

dynamic, meta-heuristic, and AI-based categories, across five performance metrics which includes latency, resource utilization, scalability, energy efficiency, and fault tolerance.

iii. Synthesize finding from the recent literature to identify performance trends, algorithmic trade-offs, and deployment considerations for practitioners.

II. LITERATURE REVIEW

A. Edge Computering Architecture

Edge-based computing networks are tightly organized as multi-tier distributed systems consisting of end devices, edge nodes, and cloud data centers. End devices such as sensors, smartphones, embedded controllers, Internet of Things (IoT) smart devices and industrial machines serve as the primary sources of data generation and computational requests. These devices continuously collect environmental and operational data while initiating latency-sensitive tasks that require immediate processing which requires cloud data centers to provide backup compute capacity, global coordination, and long-term storage [7].

Within this architecture, three tiers of edge infrastructure are commonly distinguished. The near-edge tier, also called the device layer, includes gateways and micro-servers co-located with end devices. The far-edge tier comprises MEC servers at cellular base stations or network access points. The cloud-edge tier encompasses regional data centers that extend cloud resources toward the network periphery [8]. Each tier operates under different resource constraints, communication latencies, and workload characteristics, necessitating adaptive load balancing that can operate across heterogeneous layers.

The ETSI MEC reference architecture specifies standard interfaces for deploying applications at the edge, including the MEC platform, virtualization infrastructure, and application lifecycle management [3]. In practice, edge environments vary widely from small Raspberry Pi-class devices running containerized micro services to multi-core server clusters hosted at telecom base stations. This heterogeneity complicates resource management and underscores the need for adaptive load balancing algorithms [2].

B. Importance of Load Balancing in Edge Computing

Technically, load balancing in edge computing serves multiple objectives simultaneously. First, it aims to reduce the end-to-end service latency experienced by end-user applications by routing tasks to the most capable available node. Second, it improves resource utilization by preventing overloading of hot-spot nodes while underutilized nodes remain idle. Third, effective load balancing enhances energy efficiency by distributing workloads to avoid unnecessarily activating power-hungry resources. Fourth, it improves fault tolerance by redistributing tasks when nodes fail or become unreachable [5].

The dynamic nature of edge workloads is driven by spatial and temporal variations in demand, end-device mobility, and unpredictable connectivity, making static pre-assignment strategies insufficient [9]. Research has consistently demonstrated that adaptive load balancing algorithms can achieve significant gains in throughput and latency reduction compared to naive approaches. For instance, studies employing the EdgeUP mechanism which is a priority-aware dynamic balancer, reported outperforming classical Round Robin by a factor of 5.6 in CPU utilization efficiency on real Raspberry Pi edge deployments [10].

C. Related Surveys and Gap Analysis

In recent years, several surveys have examined load balancing and task scheduling in cloud and edge environments. [11] provided a foundational survey of load balancing algorithms in cloud computing, cataloguing heuristic and meta-heuristic approaches. More recently, [12] conducted an analytical review of load balancing in edge computing platforms, covering developments through 2024. A systematic literature review by [6] examined load balancing and task scheduling in cloud environments, noting the increasing relevance of edge paradigms. [13] surveyed AI-based resource management in fog computing and concluded that AI-driven methods offer measurable advantages over classical approaches. [14] investigated load balancing for heterogeneous serverless edge computing, demonstrating the importance of empirical benchmarking alongside theoretical analysis.

However, despite this body of work, no recent study provides a structured, head-to-head comparison of load balancing techniques spanning static, dynamic, meta-heuristic, and deep learning categories, evaluated uniformly across a consistent set of performance metrics for edge-specific environments. This paper addresses that gap with a focus on the period 2024 to 2026

III. TAXONOMY OF LOAD BALANCING TECHNIQUES

Load balancing techniques in edge-based computing networks can be classified along several dimensions such as the timing of load assessment (static vs. dynamic), the centralization of decision-making (centralized vs. distributed), the algorithm type (deterministic, heuristic, meta-heuristic, or learning-based), and the optimization target (latency, energy, throughput, or a combination thereof). Fig 1 conceptually illustrates the taxonomy of load balancing techniques in edge computing environment.

ISSN: **3139-0862**

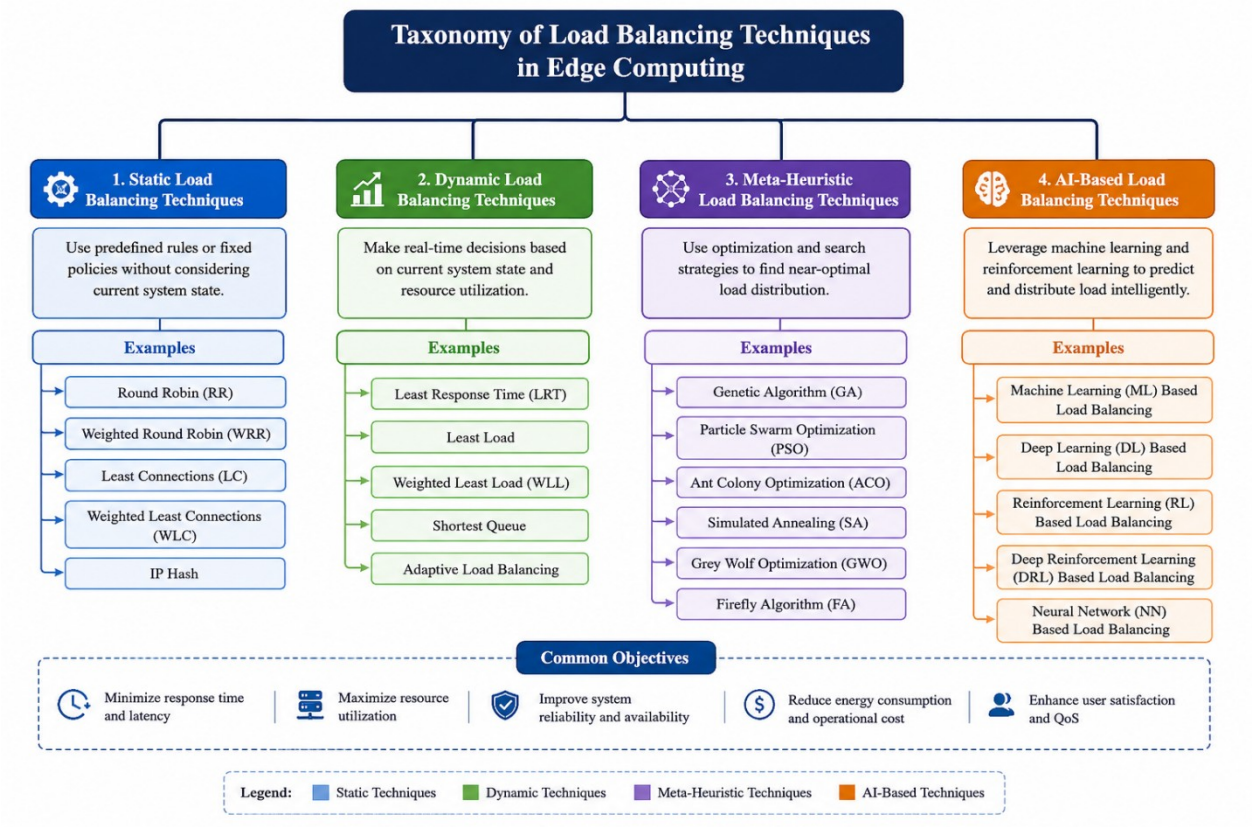

Fig 1 Taxonomy of Load Balancing Techniques in Edge Computing

Static load balancing methods assign tasks to nodes based on pre-determined rules or fixed weights without considering real-time system states [15]. They are computationally inexpensive and predictable but fail to adapt to workload fluctuations. Dynamic load balancing methods monitor system state in real time and make assignment decisions based on current resource availability, queue lengths, or processing speeds [16]. They offer adaptability at the cost of additional monitoring overhead. Meta-heuristic approaches draw on nature-inspired optimization principles to search the solution space efficiently when exact solutions are computationally intractable [17]. AI-based methods, particularly reinforcement learning and deep learning techniques, learn optimal policies from interaction with the environment and can adapt to highly complex, non-stationary edge workloads [18].

Presented in Table I is the taxonomy adopted for this study.

TABLE I TAXONOMY OF LOAD BALANCING TECHNIQUES IN EDGE COMPUTING

Category	Algorithm Examples	Decision Basis	Adaptability	Complexity
Static	Round Robin, Weighted RR	Fixed rules / weights	None	Very Low
Dynamic	Least Connection, Weighted Least Conn., Adaptive Threshold	Real-time node state	Moderate	Low–Medium
Meta-heuristic	ACO, PSO, GA, GSA	Population-based search	High	Medium–High
AI / ML-based	DRL, DQN, A3C, Federated DRL	Learned policy	Very High	High

IV. Static Load Balancing Techniques

A. Round Robin (RR)

The RR is one of the simplest and most widely deployed load balancing algorithms. In this method, tasks are assigned to edge nodes in a fixed cyclic order, regardless of the current load, processing speed, or available resources of each node. Each node receives an equal share of incoming requests over time. The algorithm is easy to implement and incurs negligible computational overhead [19].

However, the lack of awareness of node states makes the RR poorly suited for heterogeneous edge environments. If nodes differ substantially in processing capacity as is common in real edge deployments, the algorithm over-burdens slower nodes while faster nodes remain underutilized. Empirical studies report that Round Robin (RR) performs significantly worse in heterogeneous edge environments compared to adaptive approaches. The EdgeUP study by [10] demonstrated that Round Robin's imbalance in CPU utilization led to 5.6 times lower efficiency compared to a dynamic context-aware mechanism on a Raspberry Pi 400 testbed.

Nonetheless, the RR is appropriate in small-scale, homogeneous deployments with predictable, uniform

ISSN: **3139-0862**

workloads where simplicity and determinism are prioritized over adaptive performance.

B. Weighted Round Robin (WRR)

Technically, the WRR extends the basic Round Robin (RR) algorithm by assigning a numerical weight to each node, reflecting its relative processing capacity. Nodes with higher weights receive a proportionally greater share of incoming tasks. For example, a node with weight three receives three task assignments for every single assignment received by a node with weight one. This allows the WRR to account for static differences in node capabilities [20].

While WRR improves over unweighted Round Robin in heterogeneous deployments, it remains static in nature [21]. Weights are assigned at configuration time and do not reflect real-time changes in node load or availability. In dynamic edge environments with fluctuating workloads, WRR can still produce imbalances whenever actual node performance deviates from the pre-assigned weights.

V. DYNAMIC LOAD BALANCING TECHNIQUES

A. Least Connection (LC)

The Least Connection (LC) algorithm dynamically routes incoming tasks to the edge node with the fewest currently active connections or pending tasks. Here, the load balancer maintains a real-time count of active tasks per node and selects the least loaded node for each new task arrival [22]. Unlike static algorithms, LC inherently adapts to differences in task processing durations, thereby ensuring that nodes completing tasks quickly receive more new work.

However, the LC performs well in environments where tasks have variable processing times which a common characteristic of edge workloads spanning IoT data processing, video analytics, and inference requests. Its main limitation is that the active-connection count may not accurately reflect the true computational burden, particularly when tasks differ substantially in resource requirements. More sophisticated variants use weighted connection counts or incorporate CPU utilization metrics to improve accuracy.

B. Weighted Least Connection (WLC)

The Weighted Least Connection (WLC) combines the responsiveness of the Least Connection (LC) algorithm with the capacity-awareness of Weighted Round Robin (WRR) [23]. Here, each node is assigned a static weight representing its relative capacity, and the scheduling metric is computed as the ratio of active connections to the node's weight. The node with the lowest weighted connection count receives the next task [24].

The WLC is more nuanced than either of its constituent algorithms and is commonly employed in commercial load balancers such as Linux Virtual Server (LVS) and HAProxy. In edge contexts, WLC has been demonstrated to offer better balance than Round Robin or basic LC under heterogeneous, mixed-workload conditions. However, like all dynamic connection-based methods, WLC is reactive rather than predictive as it responds to load imbalances after they have partially materialized rather than anticipating them.

C. Adaptive Threshold-Based Load Balancing

The adaptive threshold-based approaches establish high and low load thresholds for each edge node and trigger task migrations when node utilization crosses these boundaries [25]. When a node's utilization exceeds the upper threshold (indicating overload), excess tasks are offloaded to less loaded nodes. Conversely, when utilization falls below the lower threshold, the node may enter a low-power state to conserve energy.

This approach directly targets two key edge computing objectives which are known as maintaining responsive service and conserving energy [26]. The challenge often lies in threshold calibration. Static thresholds may be too conservative or too aggressive depending on workload variability. Recent research incorporates workload prediction models and machine learning to dynamically adjust thresholds in response to observed patterns, yielding what are termed intelligent threshold' methods [27].

D. Min-Mim and Max-Min Algorithms

The Min-Min algorithm assigns each pending task to the node that can complete it in the minimum time possible, then selects among all tasks the one with the minimum expected completion time and assigns it first [28]. This process will repeat until all tasks are scheduled. Min-Min tends to schedule short tasks early, achieving low average response times. However, it might potentially leaving long tasks waiting, thereby creating imbalanced node utilization when workloads include many large tasks.

Max-Min inverts this logic among all task-node assignments; it selects the task with the maximum minimum completion time and assigns it first. This approach often prioritizes large tasks, preventing them from blocking the queue but potentially increasing average latency for short tasks. Both Min-Min and Max-Min are computationally intensive for large task sets and are not designed for real-time decision-making in rapidly changing edge environments [29]. These methods are better suited for batch scheduling in relatively stable edge cloud deployments.

VI. META-HEURISTIC LOAD BALANCING TECHNIQUES

A. Ant Colony Optimization (ACO)

The Ant Colony Optimization is inspired by the foraging behavior of biological ants, which collectively discover shortest paths to food sources by depositing pheromone trails [30]. In the context of load balancing, each "ant" represents a task seeking an optimal edge node for execution. Pheromone levels on each path correspond to the attractiveness of assigning tasks to specific nodes, updated based on the quality of past assignments, typically measured by response time or resource utilization.

ISSN: **3139-0862**

Technically, the ACO offers strong exploration-exploitation balance and can find good load distribution solutions in complex, multi-objective edge environments. It has been applied to task scheduling in fog computing, where the dynamic pheromone update mechanism enables the algorithm to adapt to changing node states over time. A study by [13] demonstrated that ACO-based scheduling in fog environments reduced average task completion times by measurable margins compared to static methods. Nonetheless, the drawbacks of ACO include sensitivity to pheromone decay parameters, convergence speed, and scalability concerns in large-scale deployments with hundreds of edge nodes.

B. Particle Swarm Optimization (PSO)

Technically, the Particle Swarm Optimization (PSO) models a population of particles, each representing a candidate load distribution solution [31]. Particles navigate the solution space guided by their own best-known positions and the best-known position of the entire swarm. Velocity and position updates allow the swarm to converge toward globally optimal solutions [32].

PSO has been applied to multi-objective load balancing in edge computing, simultaneously optimizing latency, energy consumption, and resource utilization. Its strength lies in rapid convergence for moderate-sized problems. However, PSO can suffer from premature convergence to local optima and may not scale well to very high-dimensional edge scenarios with hundreds of heterogeneous nodes and thousands of concurrent tasks [33]. Hybrid approaches combining PSO with local search operators or machine learning predictors have been proposed to mitigate these limitations.

C. Genetic Algorithm (GA)-Based Scheduling

Genetic Algorithms employ operations which are inspired by biological evolution. This includes selection, crossover, and mutation to iteratively improve candidate load assignments [34]. In terms of this study, a population of scheduling solutions evolves over multiple generations, with fitter solutions (those achieving better latency, utilization, or energy efficiency) having higher probability of reproduction. Generally, GA's excel at exploring complex, multi-modal solution spaces and have been applied to multi-constraint task scheduling in edge-cloud continuum deployments [35].

In the telemedicine and healthcare IoT domain, GA-based load balancing has demonstrated effective adaptation to changing workloads at the fog level, offering scalability benefits over simpler heuristics. The primary limitations of this approach are their relatively slow convergence compared to gradient-based methods and high sensitivity to parameter settings (population size, crossover rate, mutation rate). Encoding domain-specific constraints, such as data locality requirements or security policies into the GA fitness function is non-trivial.

VII. ARTIFICIAL INTELLIGENCE-BASED LOAD BALANCING TECHNIQUES

A. Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning represents the state of the art in adaptive load balancing for edge computing environments. In the DRL framework, a load balancing agent interacts with the edge network environment, observing the system state (node loads, queue lengths, network conditions) and selecting actions (task-to-node assignments) to maximize a cumulative reward signal (e.g., negative latency, positive throughput). The agent's policy is represented by a deep neural network, which is updated via gradient-based optimization after each interaction [36].

The Deep Q-Network (DQN) algorithm, which uses a convolutional or fully connected neural network to approximate the Q-value function, has been extensively applied to task offloading and load balancing in MEC environments. [37] demonstrated that DQN-based resource allocation in MEC systems achieved substantial latency reductions compared to random and greedy baselines. More advanced architectures, including Double DQN, Dueling DQN, and Proximal Policy Optimization (PPO), have been employed to address overestimation bias and improve sample efficiency in edge scenarios [38].

The Asynchronous Advantage Actor-Critic (A3C) algorithm, which maintains multiple asynchronous agents exploring different regions of the state space, has shown particular promise in multi-node edge environments. A3C-based load balancing can exploit parallelism across the edge infrastructure to accelerate policy learning and improve adaptation to spatial load variations [39].

B. Deep Reinforcement Learning with Egde Node Prediction

A significant limitation of standard DRL approaches is the lack of prior knowledge about current edge node load states from the perspective of end devices. This challenge is exacerbated by the dynamic and uncertain nature of load variations in mobile edge environments. To address this, recent research has proposed DRL frameworks augmented with edge node activity prediction modules. By first estimating the load state of each candidate node using historical data and lightweight prediction models, the DRL agent can make more informed offloading and load balancing decisions [39].

One such approach proposes a two-stage framework: a prediction stage that estimates node loads using lightweight temporal models, followed by a DRL-based decision stage that uses predicted states to minimize long-term average task completion cost. Experimental results demonstrate significant reductions in task processing latency compared to DRL without prediction, particularly in multi-user, multi-edge-node scenarios where load state information is inherently incomplete [39].

C. Multi-Agent Reinforcement Learning (MARL)

This method extends the single-agent DRL paradigm to scenarios where multiple load balancing agents operate concurrently across distributed edge nodes. Each agent learns a local policy based on its observations and coordinates with neighboring agents through implicit or explicit communication. MARL is naturally suited to the decentralized nature of edge environments, where a single central controller would create a bottleneck or single point of failure [40].

The FLoadNet architecture [40] introduced a cooperative multi-agent actor-critic method for fog computing, demonstrating that agent cooperation leads to more balanced load distributions and lower latency than independent agent operation. More recently, research on decentralized task offloading using multi-user multi-armed bandit approaches [41] has shown competitive performance with centralized DRL while incurring substantially lower coordination overhead which is an important advantage in bandwidth-constrained edge deployments.

D. Federated Learning-Enhanced Load Balacing

Federated Learning (FL) offers a privacy-preserving approach to collaborative model training across distributed edge nodes without sharing raw data. In the context of load balancing, FL enables multiple edge nodes to jointly train a load prediction or policy model by sharing only model gradients, rather than sensitive user or task data. This is particularly valuable in multi-operator edge deployments where data privacy regulations prohibit centralized data collection [42].

Federated Deep Reinforcement Learning (FedDRL) combines the adaptability of DRL with the privacy and communication efficiency of FL. Each edge node trains a local DRL policy on its own observations and periodically aggregates policy updates with a global server. Research has demonstrated that FedDRL achieves load balancing quality comparable to centralized DRL while substantially reducing data transmission overhead and preserving user privacy [43]. The FAuNO semi-asynchronous framework [44] further addresses the challenge of heterogeneous training speeds among edge nodes by employing buffered asynchronous aggregation, accelerating convergence without sacrificing model quality.

E. SVM-Based Load Balacing

The Support Vector Machine (SVM)-based load balancing presents a supervised learning alternative that classifies incoming tasks into load categories (light, moderate, and heavy) and routes them to edge nodes accordingly. The SVM-based approach proposed by [13] demonstrated notable reductions in latency compared to heuristic and clustering methods in fog/MEC environments. Its key advantage is computational efficiency at inference time once trained; SVM-based classifiers make near-instantaneous routing decisions with low memory footprint, making them suitable for resource-constrained edge nodes. The limitation is the need for labeled training data and periodic retraining as workload characteristics evolve.

VIII. COMPARATIVE EVALUATION

A. Performance Metrics

To conduct a rigorous comparative analysis, we evaluate load balancing techniques across five standardized performance metrics commonly used in the edge computing literature including:

Latency: The end-to-end time from task submission to completion, including queuing, transmission, and processing delays.

Resource Utilization: The average fraction of available CPU and memory resources across all edge nodes, with higher utilization indicating better efficiency.

Scalability: The degree to which performance degrades as the number of edge nodes, tasks, or users scales upward.

Energy Efficiency: The total energy consumed per unit of work completed, reflecting the ability to reduce unnecessary computation and idle power draw.

Fault Tolerance: The ability to detect node failures and redistribute workloads with minimal service disruption.

B. Results of the Comparison

Table II presents a structured comparison of the load balancing techniques reviewed in this study, based on data synthesized from the surveyed literature. The performance ratings are qualitative (Low, Medium, High, Very High) derived from quantitative results reported across multiple studies.

TABLE II COMPARATIVE PERFORMANCE OF LOAD BALANCING TECHNIQUES (QUALITATIVE SYNTHESIS)

Algorithm	Latency Red.	Resource Util.	Scalability	Energy Eff.	Fault Tol.
Round Robin	Low	Low	High	Low	Low
Weighted RR	Low–Med	Medium	High	Low–Med	Low
Least Connection	Medium	Medium	Medium	Medium	Medium
Weighted LC	Medium	Med–High	Medium	Medium	Medium
Min-Min / Max-Min	Medium	Medium	Low	Medium	Low
ACO	Med–High	High	Medium	Medium	Medium
PSO / GA	High	High	Medium	High	Medium
DRL (DQN/A3C)	Very High	Very High	Med–High	High	High
Federated DRL	High	High	High	High	Very High

ISSN: **3139-0862**

C. Discussion of the Results

Static load balancing algorithms such as Round Robin and Weighted Round Robin consistently exhibit poor performance in terms of latency reduction, resource utilization efficiency, and fault tolerance. Their primary strength lies in their simplicity and low computational cost, making them suitable only for homogeneous and predictable workloads. However, such workload characteristics are becoming increasingly uncommon in modern edge computing environments, where traffic patterns and resource demands are highly dynamic.

Dynamic algorithms, including Least Connection and Weighted Least Connection, offer noticeable improvements over static approaches by considering the current state of system resources before making scheduling decisions. Although these methods respond more effectively to changing workloads, they remain reactive rather than predictive and provide only moderate scalability as the size and complexity of the edge environment increase.

Meta heuristic algorithms such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Genetic Algorithm (GA) provide a better balance between adaptability and optimization quality. These techniques are capable of discovering near optimal scheduling solutions under complex operating conditions. However, they incur considerable computational overhead because they rely on iterative population based search processes. While this overhead is acceptable for applications involving periodic or batch scheduling, it makes these methods unsuitable for real time edge computing scenarios where scheduling decisions must be completed within milliseconds. Although meta heuristic approaches generally achieve better energy efficiency than conventional dynamic algorithms, their improvements remain below those reported for learning based techniques.

Deep Reinforcement Learning (DRL) approaches, including Deep Q Network (DQN), Asynchronous Advantage Actor Critic (A3C), Double Deep Q Network (Double DQN), and Proximal Policy Optimization (PPO), consistently demonstrate superior performance in latency reduction, resource utilization, and energy efficiency. Existing studies report latency reductions ranging from 20% to 40% and resource utilization improvements between 15% and 30% when compared with traditional scheduling algorithms [36, 39]. Despite these advantages, standalone DRL models require extensive training data and substantial computational resources during the learning phase, which may limit their applicability in newly deployed edge environments. Federated Deep Reinforcement Learning extends the capabilities of DRL by improving scalability and preserving data privacy across distributed edge nodes. However, these benefits are accompanied by additional communication overhead and increased algorithmic complexity [43, 44].

Support Vector Machine (SVM) based approaches represent a practical compromise between conventional heuristic methods and deep learning techniques. They achieve considerably better latency performance than traditional algorithms while maintaining much lower inference time complexity than DRL models. For example, [13] reported an approximate 15% reduction in latency compared with heuristic scheduling methods. These characteristics make SVM based approaches particularly suitable for resource constrained edge nodes where computational efficiency is a critical requirement.

D. Context-Dependent Algorithm Selection

No single load balancing algorithm is universally suitable for every edge computing deployment because each approach has its own strengths and limitations. The selection of an appropriate algorithm should therefore depend on the characteristics of the deployment environment, including workload patterns, resource availability, scalability requirements, and application latency constraints.

For small scale and homogeneous edge environments with stable and predictable workloads, Weighted Round Robin and Weighted Least Connection provide satisfactory performance while maintaining low implementation complexity. In medium scale deployments where workload variability and real time responsiveness are more pronounced, adaptive threshold based techniques and lightweight machine learning models such as Support Vector Machines (SVM) offer an effective balance between performance and computational cost. For large scale and highly heterogeneous edge computing environments that support latency sensitive applications, Deep Reinforcement Learning (DRL) based approaches, particularly those enhanced with node activity prediction, provide the best overall performance in terms of resource utilization, latency reduction, and scheduling efficiency. In environments where multiple operators collaborate and data privacy is a major concern, Federated Deep Reinforcement Learning represents the most appropriate solution because it enables collaborative model training without exposing sensitive local data, although this benefit comes with additional communication overhead and increased algorithmic complexity.

IX. OPEN CHALLENGES AND FUTURE DIRECTIONS

A. Heterogeneity and Interoperability

Generally, real world edge computing environments consist of a wide variety of hardware platforms, operating systems, virtualization technologies, and container orchestration frameworks. This diversity makes it challenging to deploy load balancing algorithms that can operate efficiently across different infrastructures while maintaining compatibility with existing resource management platforms such as Kubernetes, Docker Swarm, and OpenNebula. Achieving seamless interoperability across these heterogeneous environments remains a significant concern. Although standardized application programming interfaces (APIs) and resource abstraction layers, such as those recommended by the ETSI Multi Access Edge Computing (MEC) framework, have been proposed to address these challenges, their adoption remains inconsistent across edge computing platforms [45].

B. Energy-Aware and Green Load Balacing

Energy consumption is a growing concern in edge computing, both from operational cost and environmental sustainability perspectives. Current DRL-based methods optimize primarily for latency and throughput, with energy efficiency as a secondary objective. Future research should develop multi-objective optimization frameworks that explicitly balance latency, throughput, and energy consumption, potentially incorporating renewable energy availability as a scheduling input. Edge nodes powered by intermittent renewables (solar, wind) can benefit from energy-aware load distribution that defers non-urgent tasks to periods of high renewable availability [46].

C. Security and Trust in Load Balancing

In multi-tenant edge computing environments, load balancing decisions often reveal sensitive information about workload intensity, service usage patterns, and the availability of computing resources. If such information is exposed, malicious actors may exploit it to launch denial of service attacks, resource exhaustion attacks, or inference attacks that compromise system security and performance. Consequently, there is a growing need for load balancing algorithms that are resilient to adversarial manipulation while preserving the privacy of tenants operating on shared infrastructure. Although recent studies have begun to explore privacy preserving and secure load balancing techniques, this area remains insufficiently investigated and presents significant opportunities for future research [43].

D. 6G-Enabled Egde Intelligence

The anticipated deployment of 6G cellular networks by 2030 promises terahertz communication links, near-zero latency, and tight integration of sensing, communication, and computing at the network edge. These capabilities will enable new forms of edge intelligence in which load balancing decisions are co-optimized with communication resource allocation, beamforming, and channel scheduling. Research at the intersection of 6G radio resource management and edge load balancing represents a rich and timely opportunity [47].

E. Continual Learning and Non Stationarity

Edge workloads exhibit non-stationary characteristics. Its distributions shift over time due to user behavior changes, seasonal patterns, and service deployments. Standard DRL agents trained on historical data may exhibit degraded performance when workload distributions shift significantly. Continual learning techniques, which enable models to adapt to new distributions without catastrophically forgetting previously learned policies, are needed to sustain DRL performance over long deployment horizons [36].

F. Digital Twin-Assited Load Balacing

Digital twins is a virtual replicas of physical edge infrastructure maintained in real time. It offers a promising platform for load balancing optimization. A digital twin can simulate different assignment policies offline before committing to real deployments, enabling safe exploration of novel strategies without risking service disruptions. Early work on digital twin-assisted MEC resource management [48] suggests significant performance gains over reactive approaches, though the computational cost of maintaining accurate digital twin models at scale remains challenging.

X. CONCLUSION

This study presented a comprehensive comparative review of load balancing techniques for edge computing networks, examining static, dynamic, meta-heuristic, and artificial intelligence based approaches. The findings indicate that no single load balancing algorithm is universally applicable to all edge computing environments. Instead, the most suitable approach depends on factors such as the scale of deployment, the degree of infrastructure heterogeneity, workload characteristics, and available computational resources. Static algorithms are simple to implement but lack the adaptability required for dynamic environments. Dynamic techniques improve responsiveness by adapting to changing network conditions, while meta heuristic methods provide effective optimization capabilities at the expense of higher computational cost. Artificial intelligence based approaches, particularly Deep Reinforcement Learning, consistently deliver superior performance in complex and highly dynamic edge environments. Federated Deep Reinforcement Learning further enhances scalability and data privacy, making it well suited for distributed and multi operator deployments.

The review also identified several important research challenges that require further investigation. These include improving interoperability across heterogeneous edge infrastructures, developing energy efficient multi objective optimization techniques, strengthening the resilience of load balancing algorithms against security threats and adversarial attacks, and integrating intelligent load balancing with emerging sixth generation (6G) communication technologies. In addition, continual learning and digital twin assisted optimization have significant potential to improve the long term adaptability and operational efficiency of edge computing systems.

As edge computing continues to evolve alongside the rapid growth of the Internet of Things, fifth generation (5G) and sixth generation (6G) mobile networks, and artificial intelligence driven applications at the network edge, the need for intelligent and adaptive load balancing solutions will continue to increase. Such techniques will play a vital role in improving system performance, resource utilization, energy efficiency, and service reliability. It is expected that this review will serve as a valuable reference for researchers, practitioners, and system designers interested in advancing load balancing strategies for next generation edge computing environments.

REFERENCES

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," IEEE communications surveys & tutorials, vol. 19, pp. 2322-2358, 2017.

[2] A. Filali, A. Abouaomar, S. Cherkaoui, A. Kobbane, and M. Guizani, "Multi-access edge computing: A survey," IEEE Access, vol. 8, pp. 197017-197046, 2020.

[3] J. H. Lee, J. Lee, and J. Song, "An edge-enabled IoT framework for metaverse in smart city," in 2023 IEEE International Conference on Metaverse Computing, Networking and Applications (MetaCom), 2023, pp. 708-713.

[4] A. Maia, A. Boutouchent, Y. Kardjadja, M. Gherari, E. G. Soyak, M. Saqib, et al., "A survey on integrated computing, caching, and communication in the cloud-to-edge continuum," Computer Communications, vol. 219, pp. 128-152, 2024.

[5] Z. Nezami, K. Zamanifar, K. Djemame, and E. Pournaras, "Decentralized edge-to-cloud load balancing: Service placement for the Internet of Things," IEEE Access, vol. 9, pp. 64983-65000, 2021.

[6] N. Devi, S. Dalal, K. Solanki, S. Dalal, U. K. Lilhore, S. Simaiya, et al., "A systematic literature review for load balancing and task scheduling techniques in cloud computing," Artificial Intelligence Review, vol. 57, p. 276, 2024.

[7] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE internet of things journal, vol. 3, pp. 637-646, 2016.

[8] E. Baccour, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Reinforcement learning-based dynamic pruning for distributed inference via explainable AI in healthcare IoT systems," Future Generation Computer Systems, vol. 155, pp. 1-17, 2024.

[9] G. Famitafreshi, M. S. Afaqui, and J. Melià-Seguí, "A comprehensive review on energy harvesting integration in IoT systems from MAC layer perspective: Challenges and opportunities," Sensors, vol. 21, p. 3097, 2021.

[10] L. A. Nguyen, S. Kim, and Y. Son, "EdgeUP: Utilization and Priority-Aware Load Balancing in Edge Computing," Electronics, vol. 14, p. 565, 2025.

[11] E. J. Ghomi, A. M. Rahmani, and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," Journal of Network and Computer Applications, vol. 88, pp. 50-71, 2017.

[12] M. Khani, M. M. Sadr, and S. Jamali, "Deep reinforcement learning-based resource allocation in multi-access edge computing," Concurrency and Computation: Practice and Experience, vol. 36, p. e7995, 2024.

[13] H. M. Abdelghany, "SVM-Based Load Balancing for Efficient Edge Computing," International Journal of Telecommunications, vol. 5, pp. 1-20, 2025.

[14] A. Agiollo, P. Bellavista, M. Mendula, and A. Omicini, "EneA-FL: Energy-aware orchestration for serverless federated learning," Future Generation Computer Systems, vol. 154, pp. 219-234, 2024.

[15] N. Sadashiv, "Load balancing in fog computing: A detailed survey," International Journal of Computing and Digital Systems, vol. 13, pp. 729-750, 2023.

[16] D. Kanellopoulos and V. K. Sharma, "Dynamic load balancing techniques in the IoT: A review," Symmetry, vol. 14, p. 2554, 2022.

[17] Y. Xiao, H. Cui, R. A. Khurma, and P. A. Castillo, "Artificial lemming algorithm: a novel bionic meta-heuristic technique for solving real-world engineering optimization problems: Artificial lemming algorithm...: Y. Xiao et al," Artificial Intelligence Review, vol. 58, p. 1, 2025.

[18] E. M. Ali, J. Abawajy, F. Lemma, and S. A. Baho, "Analysis of deep reinforcement learning algorithms for task offloading and resource allocation in fog computing environments," Sensors, vol. 25, p. 5286, 2025.

[19] A. M. Alakeel, "A guide to dynamic load balancing in distributed computer systems," International journal of computer science and information security, vol. 10, pp. 153-160, 2010.

[20] S. S. Priya and T. Rajendran, "Load balancing using improved weighted round robin algorithm in cloud computing environment," International Journal of Cloud Computing, vol. 13, pp. 463-484, 2024.

[21] S. Sree Priya and D. Rajendran, "Enhanced Weighted Round Robin: A New Paradigm in Cloud Load Balancing," Indian Journal of Science and Technology, vol. 18, pp. 1220-1228, 2025.

[22] A. Junaidi and M. M. Al Haromainy, "Cloud-Based High Availability Architecture Using Least Connection Load Balancer and Integrated Alert System," bit-Tech, vol. 8, pp. 263-274, 2025.

[23] M. S. Almhanna, T. A. Murshedi, F. S. Al-Turaihi, R. M. Almuttairi, and R. Wankar, "Dynamic Weight Assignment with Least Connection Approach for Enhanced Load Balancing in Distributed Systems," 2023.

[24] H. Hellani, L. Sliman, A. E. Samhat, and E. Exposito, "Computing resource allocation scheme for DAG-based IOTA nodes," Sensors, vol. 21, p. 4703, 2021.

[25] T. Maqsood, S. K. uz Zaman, A. Qayyum, F. Rehman, S. Mustafa, and J. Shuja, "Adaptive thresholds for improved load balancing in mobile edge computing using K-means clustering," Telecommunication Systems, vol. 86, pp. 519-532, 2024.

[26] L. Gu, W. Zhang, Z. Wang, D. Zeng, and H. Jin, "Service management and energy scheduling toward low-carbon edge computing," IEEE Transactions on Sustainable Computing, vol. 8, pp. 109-119, 2022.

[27] T. L. Duc, C. Nguyen, and P.-O. Östberg, "Workload prediction for proactive resource allocation in large-scale cloud-edge applications," Electronics, vol. 14, p. 3333, 2025.

[28] S. K. Mishra, B. Sahoo, and P. P. Parida, "Load balancing in cloud computing: a big picture," Journal of King Saud University Computer and Information Sciences, vol. 32, pp. 149-158, 2020.

[29] S. Ali, R. A. Haidri, and N. A. Farooqui, "Critical Analysis of Traditional and Adaptive Cloud Task Scheduling Algorithms: Dynamic and Heterogeneous Workload Issues," in 2026 2nd International Conference on Cognitive Computing in Engineering, Communications, Sciences and Biomedical Health Informatics (IC3ECSBHI), 2026, pp. 1035-1040.

[30] B. Cheng, F. Yang, P. Zhang, Y. Fu, and J. Sun, "An Ant-Inspired Foraging Model Based on Cellular Automata," in Chinese Intelligent Systems Conference, 2025, pp. 488-499.

[31] L. Long and J. Hu, "Multi-Objective Particle Swarm Optimization for Site Selection and Policy Subsidy Maximization of Foreign Renewable Energy Enterprises in the United States," Artificial Intelligence and Machine Learning Review, vol. 7, pp. 54-69, 2026.

[32] Y. Lalitha Kameswari, "Fundamental of swarm optimization," in Role of Nature-Inspired Algorithms in Real-life Problems, ed: Springer, 2025, pp. 95-117.

[33] L. Abualigah, "Particle swarm optimization: advances, applications, and experimental insights," Computers, Materials, & Continua, vol. 82, p. 1539, 2025.

[34] M. El-Shorbagy, "A Review of Genetic Algorithms: Principles, Procedures, and Applications in Optimization," Computer Modeling in Engineering & Sciences (CMES), vol. 147, p. 1, 2026.

[35] Y. Wang, Z. Zhao, N. Qian, W. Ding, and H. Su, "The digital twin-enabled deformation prediction and control methodologies for thin-walled parts in intelligent manufacturing: a review," Digital Twin, p. 2614885, 2026.

[36] P. Li, W. Xie, Y. Yuan, C. Chen, and S. Wan, "Deep reinforcement learning for load balancing of edge servers in iov," Mobile Networks and Applications, vol. 27, pp. 1461-1474, 2022.

[37] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," IEEE Transactions on emerging topics in computing, vol. 9, pp. 1529-1541, 2019.

[38] M. Yahya, A. Conzelmann, and S. Maghsudi, "Decentralized task offloading and load-balancing for mobile edge computing in dense networks," IEEE Communications Letters, vol. 28, pp. 1954-1958, 2024.

[39] W. Chen, S. Liu, Y. Yang, W. Hu, and J. Yu, "Dynamic Edge Loading Balancing with Edge Node Activity Prediction and Accelerating the Model Convergence," Sensors, vol. 25, p. 1491, 2025.

[40] J. Baek and G. Kaddoum, "FLoadNet: Load balancing in fog networks with cooperative multiagent using actor–critic method," IEEE Transactions on Network and Service Management, vol. 20, pp. 400-414, 2022.

[41] X. Wang, J. Ye, and J. C. Lui, "Decentralized task offloading in edge computing: A multi-user multi-armed bandit approach," in IEEE INFOCOM 2022-IEEE Conference on computer communications, 2022, pp. 1199-1208.

[42] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive

ISSN: **3139-0862**

survey," IEEE communications surveys & tutorials, vol. 23, pp. 1622-1658, 2021.

[43] K. Peng, P. Xiao, S. Wang, and V. C. Leung, "SCOF: Security-aware computation offloading using federated reinforcement learning in industrial internet of things with edge computing," IEEE Transactions on Services Computing, vol. 17, pp. 1780-1792, 2024.

[44] F. Metelo, A. Oliveira, S. Racković, P. Á. Costa, and C. Soares, "FAuNO: Semi-Asynchronous Federated Reinforcement Learning Framework for Task Offloading in Edge Systems," arXiv preprint arXiv:2506.02668, 2025.

[45] M.-A. E. Computing, "Framework and Reference Architecture, ETSI GS MEC 003 V3. 1.1," ETSI, Sophia Antipolis, France, Available online: https://www. etsi. org/deliver/etsi_gs/MEC/001_099/003/03.01. 01_60/gs_mec003v030101p. pdf.(Accessed 06 February 2023), 2022.

[46] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, "Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research," Internet of Things, vol. 12, p. 100273, 2020.

[47] G. Luo, C. Shao, N. Cheng, H. Zhou, H. Zhang, Q. Yuan, et al., "EdgeCooper: Network-aware cooperative LiDAR perception for enhanced vehicular awareness," IEEE Journal on Selected Areas in Communications, vol. 42, pp. 207-222, 2023.

[48] G. G. Devarajan, S. Thangam, M. J. Alenazi, U. Kumaran, G. Chandran, and A. K. Bashir, "Federated learning and blockchain-enabled framework for traffic rerouting and task offloading in the internet of vehicles (iov)," IEEE Transactions on Consumer Electronics, vol. 71, pp. 3817-3825, 2025.