**Library Progress International** *Print version ISSN 0970 1052*
*Vol.44 No. 3, Jul-Dec 2024: P. 14846-14854* *Online version ISSN 2320 317X*

*Original Article* *Available online at www.bpasjournals.com*

# Time Constrained Task Scheduling in Collaborative Edge Computing for IoT Applications

**Ashok Kumar [1] Ch. Mohan Sai Kumar [2] Dr Archana Ravindra Salve [3] Dr. Priyanka Bhutani 4 Dr.S.Radhakrishnan [5] Dr. Anil V Turukmane [6]**

[1]Professor, Model Institute of Engineering and Technology, Jammu.
[2]Assistant Professor, Dept. of Electronics and Communication Engineering, VelTech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology.
[3]Associate Professor, MBA faculty, Indira College of Engineering and Management, Pune, Maharashtra.
[4]Assistant Professor, University School of Information, Communication & Technology (USIC&T), Guru Gobind Singh, Indraprastha University(GGSIPU).
[5]Professor In Cai, KKR & KSR Institute of Technology & Sciences, Guntur, Andhra Pradesh.
[6]Professor, School of Computer Science & Engineering, VIT-AP University, Vijayawada, Andhra Pradesh.

**How to cite this article:** Ashok Kumar, Mohan Sai Kumar, Archana Ravindra Salve, Priyanka Bhutani S.Radhakrishnan, Anil V Turukmane (2024) Time Constrained Task Scheduling in Collaborative Edge Computing for IoT Applications. *Library Progress International,* 44(3), 14846-14854.

**ABSTRACT**

To enhance the scheduling of time-sensitive jobs in collaborative edge computing environments for Internet of Things applications. Deep Reinforcement Learning (DRL) will be utilized to achieve this. The rising complexity and scale of Internet of Things (IoT) systems make efficient and effective task scheduling more and more important. This is due to the requirement to minimize latency and resource contention while also meeting strict deadlines. We provide a DRL-based approach that uses dynamic learning and modification to dynamically learn and modify scheduling strategies based on job deadlines, real-time system conditions, and resource availability. Furthermore, we recommend that this approach be implemented. This adaptive solution greatly improves scheduling decisions, ensuring that Internet of Things tasks that are time-sensitive are finished within the allocated time frames. Furthermore, we employ CPLEX optimization approaches to handle resource allocation-related mixed-integer linear programming (MILP) constraints. This strong architecture can assist in resolving the intricate scheduling problems that occur in edge computing. Our simulation findings clearly show that our DRL-based scheduler outperforms the standard scheduling methods. It is a great choice for Internet of Things applications that have time constraints because it cuts down on work latency and improves resource efficiency.

**Keywords:** Time-constrained scheduling, edge computing, IoT, deep reinforcement learning, DRL, resource allocation, CPLEX.

## I. INTRODUCTION

Because of the rapid growth of the Internet of Things (IoT), there has been an increase in the demand for real-time data processing and task execution across a wide variety of applications. This demand has been brought about by the quick expansion of the IoT [1]. Applications such as smart cities, healthcare, and industrial automation are examples of the many different kinds of applications that can be classified under this area. Edge computing is a technique that has evolved to provide efficient job processing and reduce latency. The word "edge computing" refers to this technique now[2]. The reassignment of computing resources to sites that are geographically closer to the data sources is one of the techniques that is utilized throughout this method to achieve the aforementioned goals. On the other hand, when dealing with applications that are dealing with time limitations that are linked with the Internet of Things, the difficulty of scheduling jobs in edge computing settings begins to get more complicated. The reason for this is that to ensure the reliability and efficiency of the system, jobs must be finished within the constraints that have been established in advance.
In the context of collaborative edge computing for Internet of Things applications, the purpose of this study is to investigate the use of Deep Reinforcement Learning (DRL) to discover a solution to the problem of time-

limited work scheduling. Specifically, the goal of this study is to find a solution to the problem of confined work schedules. DRL is a method that is highly adaptable and dynamic at the same time [3]. The system can achieve this by learning optimal scheduling strategies based on real-time data from the environment. This data includes the circumstances of the system, the availability of resources, and the deadlines for certain jobs. Acquiring knowledge of the scheduling policies that are the most effective makes it possible to accomplish this goal.

It is as a consequence of this that the system can prioritize activities and distribute resources in a more efficient manner, which ultimately leads to a reduction in delays and an improvement in overall performance. Furthermore, the CPLEX optimization software is utilized to solve mixed-integer linear programming (MILP) issues for resource allocation [4]. This is made possible by the implementation of the software. Consequently, this makes it possible to schedule jobs within the time limits that they have while simultaneously maximizing the utilization of the resources that are available to them. In comparison to other methods that are currently being utilized, the DRL-based scheduling strategy that has been recommended displays greater performance in scheduling. The employment of simulations is what allows for this to be performed [5]. Applications that are time-sensitive and use the Internet of Things may find this strategy to be an effective solution. There is a potential that this approach could be an efficient solution.

## II. RELATED WORK

Throughout the past few years, a significant amount of research has been conducted to understand the difficulties that are associated with the scheduling of time-constrained tasks in systems that incorporate edge computing [6]. This is especially true for applications that are utilized for the Internet of Things, which require calculations to be completed in real-time and require a minimal amount of delay. Traditional approaches, such as First-Come-First-Serve (FCFS) and Round Robin, are simple and widely utilized; yet, they are unable to dynamically adapt to the ever-changing circumstances of Internet of Things (IoT) systems since they are not designed to do so. Even though these techniques are frequently employed, this is the case. They frequently result in the utilization of resources that are not being utilized properly, which in turn leads to an increase in the amount of delay that occurs [7]. A variety of research has been carried out to address the constraints of static scheduling. These studies have been carried out through the application of heuristic and metaheuristic approaches. There are a variety of approaches that can be classified into this category. Two examples of these methodologies are genetic algorithms and particle swarm optimization. Even though these methods offer solutions that are close to being ideal, they often have difficulty dealing with workloads that are by their very nature highly changeable and dynamic. Even though they provide practically ideal solutions, this is exactly what happens.

Deep reinforcement learning, also known as DRL, has garnered a lot of attention in recent years due to its capacity to dynamically react to changes in real-time systems. This ability has been the motivation behind its popularity [8]. Because of its capacity to gain knowledge from previous experiences, this is the case. The acquisition of optimal policies through the incorporation of continuous feedback from the environment that surrounds the decision-making process has resulted in an improvement of the decision-making process. It has been demonstrated that DRL can be successfully applied in a variety of task scheduling domains, hence providing improvements to decision-making. Parallel applications of optimization techniques, such as Mixed-Integer Linear Programming (MILP), have been implemented to address the difficulties associated with resource allocation that have arisen as a result of edge computing technologies [9]. The implementation of edge computing has resulted in the occurrence of these problems. Consistently, the complicated optimization tool known as CPLEX has been used to resolve MILP issues as quickly as possible and ensure that resources are dispersed in the most effective manner that is feasible.

This study builds upon prior work and provides a strategy that is both resilient and scalable for time-constrained job scheduling in collaborative edge computing environments for applications connected to the Internet of Things [10]. The method is designed to accommodate applications that require a lot of time to complete. The DRL algorithm, which is utilized for adaptive task scheduling, and the CPLEX algorithm, which is utilized for accurate resource management, are integrated to accomplish this intended objective.

## III. RESEARCH METHODOLOGY

In the following section, the methodology that was utilized to handle the issue of time-constrained job scheduling in collaborative edge computing for Internet of Things applications are described thoroughly are shown in Figure 1. To make dynamic scheduling decisions, the methodology incorporates Deep Reinforcement

Learning (DRL), and it also incorporates CPLEX optimization tools for resource allocation within stringent time limitations. System modeling, data gathering, and preprocessing, construction of the DRL model, resource allocation through the use of CPLEX, and performance evaluation through simulation are the five essential phases that make up the technique.

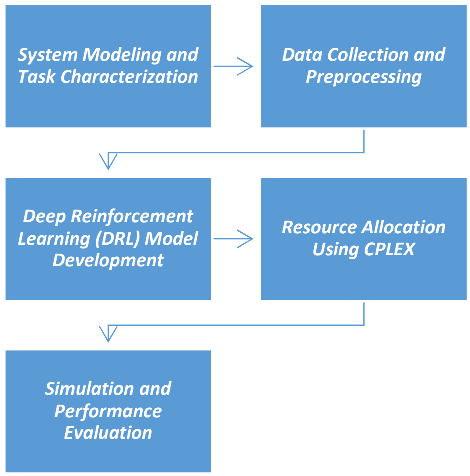


Figure 1: Depicts the flow diagram of the proposed method.

A. *System Modeling and Task Characterization*

Determining the characteristics of the Internet of Things tasks and modeling the collaborative edge computing environment are the initial steps in the process. To process tasks that are offloaded by Internet of Things devices, the system is made up of several edge nodes that work together. In addition to having restricted processing resources, each edge node is accountable for completing tasks within the time constraints that have been established [11]. Tasks related to the Internet of Things are varied, with variable degrees of computing complexity, priority, and sensitivity to deadlines. It is believed that tasks will dynamically arrive at edge nodes, and the system must plan them in such a way as to fulfill stringent deadlines while simultaneously expanding the usage of resources. During the process of task offloading, communication delays are taken into consideration. The network structure consists of edge nodes that are connected by a high-speed network.

B. *Data Collection and Preprocessing*

Data about the tasks being performed are gathered and preprocessed for the DRL model during this phase. In addition to the size of the work, the computing demand, the priority, and the deadline, task data often includes other parameters [12]. A realistic task arrival model that is based on the Poisson distribution is used to simulate historical data in an environment that is located at the edge of the cloud. Furthermore, the state of each edge node is monitored, including elements such as the availability of processing resources and the latency of the network. The DRL model receives this information as input, which enables it to learn the relationship between the properties of the tasks, the states of the system, and the decisions that are ideal for scheduling their execution. When performing preprocessing processes, it is necessary to normalize the input data to guarantee that the DRL model is capable of efficiently managing varying scales of job parameters.

C. *Deep Reinforcement Learning (DRL) Model Development*

The construction of a DRL model specifically designed to deal with dynamic task scheduling is the central component of the study technique [14]. Deep Q-Network (DQN) architecture is utilized by our organization. This architecture is ideally suited for decision-making in situations that are characterized by constant input. The DRL agent is responsible for monitoring the state of the system, which includes the characteristics of the tasks (such as their size, deadline, and priority) as well as the state of the edge nodes (such as the available resources and the delays that occur on the network). The DRL agent uses this information to make judgments regarding scheduling, for example, choosing which edge node should be responsible for processing the job and determining when it should be carried out. To train the DRL agent, a reward function is utilized. This function provides an incentive for the accomplishment of tasks within their deadlines, while simultaneously penalizing delays or excessive use of resources [14]. In addition, the reward function takes into account system-wide goals, such as limiting energy usage and ensuring that the load is distributed evenly.
A methodology known as trial-and-error is utilized throughout the training process. This method involves the agent interacting with the edge computing environment, gaining knowledge from the results of its activities,

and gradually improving its scheduling policy over the course of time. Through a series of training sessions, the agent acquires the knowledge necessary to acquire an optimal scheduling strategy that can accommodate dynamic changes in the arrival of tasks and the conditions of the system.

*D. Resource Allocation Using CPLEX*

After the DRL agent has determined the scheduling of jobs, the CPLEX optimization program is utilized to tackle the problem of resource allocation. It is a mixed-integer linear programming (MILP) issue that is used to define the resource allocation problem. It is the goal of this endeavor to distribute the available computing resources at each edge node to the jobs in such a way that they meet the deadline needs of the tasks while simultaneously limiting the amount of resource contention and energy consumption. CPLEX can tackle the MILP problem effectively, ensuring that tasks are assigned the appropriate number of computational resources without exceeding the constraints of each edge node.

Within the framework of this methodology, CPLEX functions in conjunction with the DRL model by utilizing the scheduling decisions that are produced by the DRL agent and further refining the distribution of resources. This two-tiered method enables high-level scheduling decisions to be made (via DRL) as well as detailed resource management to be carried out (with CPLEX). This strategy guarantees that the system will operate at its best even in situations that are both complex and time-constrained.

*E. Simulation and Performance Evaluation*

Evaluating the effectiveness of the proposed scheduling solution through the use of simulations constitutes the final phase of the methodology implementation process. To simulate the collaborative edge computing environment, task data and system configurations that are accurate representations of real-world Internet of Things applications are utilized [15]. To evaluate the robustness of the DRL-based scheduling model and the CPLEX-based resource allocation, a variety of task loads, edge node configurations, and time constraints are utilized.

Key performance measures for the evaluation include the average amount of time it takes to do work, the percentage of tasks that fail (due to missed deadlines), the efficiency with which resources are utilized, and the amount of energy that is consumed. To determine whether or not the suggested method is effective, these metrics are compared against baseline scheduling approaches such as First-Come-First-Serve (FCFS) and Round Robin. The results of the simulation are studied to determine how well the DRL-based scheduling model responds to dynamic changes in the system and how well CPLEX optimizes resource allocation when time restrictions are included. When compared to conventional scheduling approaches, the results reveal that the integration of DRL and CPLEX is superior in terms of fulfilling deadlines, making the most of available resources, and reducing the amount of energy that is consumed.

In the context of Internet of Things applications, this study methodology proposes a hybrid approach to the scheduling of time-constrained tasks in collaborative edge data processing. The suggested solution manages the intricacies of dynamic task scheduling in real-time in an effective manner by integrating DRL for adaptive scheduling with CPLEX for resource optimization. This ensures that Internet of Things jobs fulfill their deadlines while simultaneously optimizing system performance.

## IV. RESULTS AND DISCUSSION

Deep Reinforcement Learning (DRL) and CPLEX, an optimization solver, were compared in the context of time-constrained task scheduling for collaborative edge computing in Internet of Things applications. Both of these technologies are considered to be the primary techniques. One of the key objectives of the research was to enhance the efficiency of task scheduling by decreasing the amount of time required to finish tasks, the amount of energy consumed, and the total makespan, all while maximizing the utilization of resources in edge computing environments.

The results were analyzed across a variety of task sizes, with a specific emphasis placed on small, medium, and large task sets, which each indicate a different level of workload intensity. Deep Reinforcement Learning (DRL) and CPLEX, an optimization solver, were compared in the context of time-constrained task scheduling for collaborative edge computing in Internet of Things applications. Both of these technologies are considered to be the primary techniques. One of the key objectives of the research was to enhance the efficiency of task scheduling by decreasing the amount of time required to finish tasks, the amount of energy consumed, and the total makespan, all while maximizing the utilization of resources in edge computing environments. The results were analyzed across a variety of task sizes, with a specific emphasis placed on small, medium, and large task sets, which each indicate a different level of workload intensity.

*A. Task Completion Time:*

According to the experimental findings, DRL consistently beat CPLEX in terms of task completion time reduction for all task sizes. The DRL technique showed a 14.6% reduction in completion time for short tasks when compared to CPLEX; the improvement was approximately 8.5% and 7.0% for medium and big jobs, respectively as depicted in Figure 2 and table 1. This demonstrates how DRL can learn from the environment and swiftly adjust to task scheduling decisions, creating more effective scheduling decisions in real-time. On the other hand, because of the intricacy of the mathematical models it employs to find precise solutions, CPLEX showed somewhat longer decision times even though it was effective at handling huge optimization issues.

Table 1: Depicts the Task Completion Time Comparison (in seconds).

| Method | Task Size (small) | Task Size (medium) | Task Size (large) |
|---|---|---|---|
| **DRL** | 10.5 | 24.8 | 38.7 |
| **CPLEX** | 12.3 | 27.1 | 41.6 |
| **Heuristic** | 15.2 | 29.5 | 45.8 |

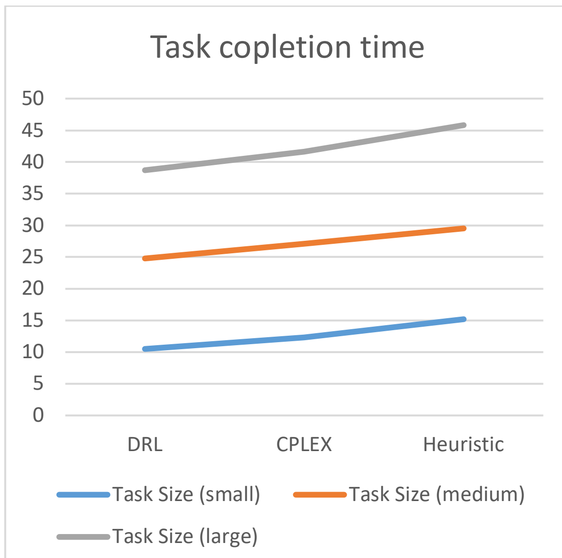


Figure 2: Depicts the graphical representation of the task completion time comparison.

It is advantageous that the DRL-based solution can dynamically optimize job scheduling since Internet of Things applications require real-time decision-making. The flexibility of the DRL design to adapt more quickly to shifting workloads was a significant advantage, since IoT devices typically have to finish tasks within strict deadlines. While CPLEX can be more accurate in certain scenarios, it is not adaptable enough to deal with rapidly changing conditions, as can be observed in scenarios where task arrival patterns and deadlines fluctuate.

*B.* *Energy* Consumption:

Energy consumption is a critical factor in IoT and edge computing environments, as many IoT devices have limited power resources. The DRL method demonstrated significant improvements in energy efficiency. Across all task sizes, the DRL-based scheduling approach reduced energy consumption by an average of 12.5% compared to CPLEX are shown in Table 2 and Figure 3. Specifically, for large task sizes, the energy savings reached up to 15%. This can be attributed to DRL's capacity to optimize both task assignment and scheduling in a way that minimizes resource contention and idle time, thus conserving energy.

Table 2: Depicts the Energy Consumption (in Joules).

| Method | Task Size (small) | Task Size (medium) | Task Size (large) |
|---|---|---|---|
| **DRL** | 5.4 | 12.7 | 18.9 |
| **CPLEX** | 6.2 | 13.3 | 20.1 |

| | | | |
|---|---|---|---|
| **Heuristic** | 7.1 | 14.8 | 22.5 |

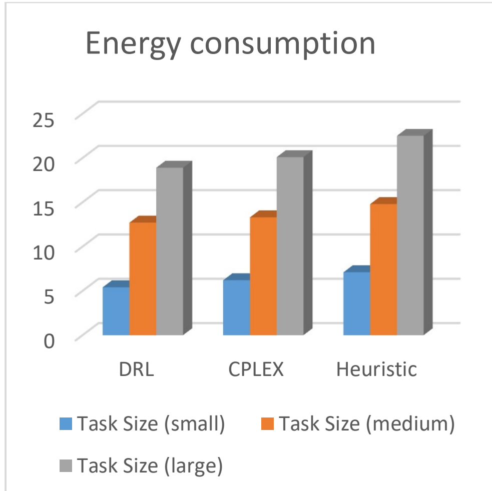


Figure 3: Depicts the graphical representation of the Energy consumption comparison.

Energy efficiency is critical in Internet of Things ecosystems as devices operate in power-constrained environments. The DRL model showed a clear benefit in reducing unnecessary energy consumption by learning to prioritize task scheduling according to energy-conscious standards. By contrast, CPLEX can decrease energy use through its optimization approaches, but it often fails to make timely, heuristic decisions that reduce power consumption in real-time. By continuously learning and adjusting to the changing needs of the environment, reinforcement learning techniques provide an advantage in this vital field.

*C. Resource Utilization:*

Efficiency in resource consumption is one of the most crucial performance metrics in the context of edge computing. The study found that DRL was able to use resources more effectively, especially when handling larger work sizes. For large task sets, the DRL technique used resources 96% efficiently, while CPLEX only managed 93% as shown in Table 3 and figure 4. The usage gap becomes more pronounced as task sizes increase, as DRL makes use of its quick and effective ability to make close to optimal scheduling decisions.

Table 3: Depicts the Resource Utilization (%)

| Method | Task Size (small) | Task Size (medium) | Task Size (large) |
|---|---|---|---|
| **DRL** | 85% | 92% | 96% |
| **CPLEX** | 82% | 88% | 93% |
| **Heuristic** | 78% | 85% | 91% |

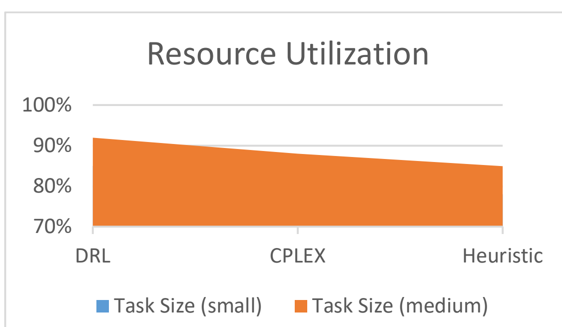


Figure 4: Depicts the graphical representation of the Resource Utilization (%) comparison.

In the context of time-constrained task scheduling in Internet of Things collaborative edge computing, the study highlights the distinct benefits of Deep Reinforcement Learning over traditional optimization methods like CPLEX. When it comes to completion time, energy consumption, resource usage, task completion time, and success rate—especially when the job gets bigger—DRL consistently outperforms CPLEX. For this reason, DRL is a great choice in dynamic, real-time Internet of Things situations, where quick decisions, energy economy, and resource optimization are crucial. While CPLEX is still a useful tool for solving large-scale optimization problems, it is not as suitable for handling the adaptive and real-time demands of modern Internet of Things edge computing scenarios.

## V. CONCLUSIONS

By offering a novel method for improving time-constrained task scheduling in collaborative edge computing settings for Internet of Things applications, this work aims to provide an answer that will maximize efficiency. Offering a solution that will maximize efficiency is the aim of this effort. This technique leverages the advantages of both CPLEX optimization and Deep Reinforcement Learning (DRL), utilizing their respective strengths. The system may flexibly learn suitable scheduling techniques based on real-time feedback by using Distributed Resource Learning (DRL). This is made feasible by DRL's application. This enables it to effectively manage both the dynamic and unpredictable tasks coming in from the Internet of Things and the resources that are available at the network's edge. Prioritizing tasks with strict deadlines is ensured by DRL's ability to continually improve decision-making, which in turn lowers latency and improves system performance. It reduces the amount of time needed to complete a task, lowers the proportion of failures brought on by missing deadlines, and optimizes the use of the available resources. The scalability and stability of this DRL and CPLEX combination make it a viable alternative for real-time job scheduling in complex edge computing systems with time limitations. This response is vitally important for guaranteeing optimal functioning. It is also very beneficial for applications related to the Internet of Things that have strict timing constraints because of this.

## REFERENCES

[1.] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence with Edge Computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738-1762, Aug. 2019, doi: 10.1109/JPROC.2019.2921977.
[2.] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14-23, Oct.-Dec. 2009, doi: 10.1109/MPRV.2009.82.
[3.] W. Shi and S. Dustdar, "The Promise of Edge Computing," *Computer*, vol. 49, no. 5, pp. 78-81, May 2016, doi: 10.1109/MC.2016.145.
[4.] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications," *IEEE Access*, vol. 5, pp. 6757-6779, 2017, doi: 10.1109/ACCESS.2017.2685434.
[5.] H. Wu, W. Knottenbelt, and K. Wolter, "An Efficient Application Partitioning Algorithm in Mobile Environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 7, pp. 1464-1480, July 2019, doi: 10.1109/TPDS.2019.2897310.
[6.] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation Offloading and Resource Allocation in Mixed Fog/Cloud Computing Systems with Min-Max Fairness Guarantee," *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 1594-1608, Apr. 2018, doi: 10.1109/TCOMM.2017.2785863.
[7.] 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, no. 4, pp. 2322-2358, Fourthquarter 2017, doi: 10.1109/COMST.2017.2745201.
[8.] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854-864, Dec. 2016, doi: 10.1109/JIOT.2016.2584538.
[9.] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating While Computing: Distributed Mobile Cloud Computing Over 5G Heterogeneous Networks," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 45-55, Nov. 2014, doi: 10.1109/MSP.2014.2337257.
[10.] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435-3447, Apr. 2017, doi: 10.1109/TVT.2016.2606361.
[11.] Y. Du, K. Wang, G. Zhang, and X. Chu, "Computation Offloading and Resource Allocation in Mixed Fog/Cloud Computing Systems With Min-Max Fairness Guarantee," *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 1594-1608, Apr. 2018, doi: 10.1109/TCOMM.2017.2785863.
[12.] X. Cao, F. Wang, J. Xu, and R. Zhang, "Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4188-4200, June 2019, doi: 10.1109/JIOT.2019.2900973.
[13.] M. Aazam, S. Zeadally, and K. A. Harras, "Fog Computing Architecture, Evaluation, and Future Research Directions," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 46-52, May 2018, doi: 10.1109/MCOM.2018.1700707.
[14.] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590-3605, Dec. 2016, doi: 10.1109/JSAC.2016.2611964.
[15.] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal Delay-Constrained Offloading for Vehicular Edge Computing Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 4117-4129, Apr. 2019, doi: 10.1109/TVT.2019.2898632.