

Edge-Cloud Collaborative Framework for Smart IoT Applications

An ML-based TSCO Framework for Energy-Efficient Fog Computing in IIoT Smart Applications

Safir Jameel Manghat

Master's Degree Thesis in
Computer and System Science

VT2025

Supervisor: Sr. Lecturer Dr. Praveen Kumar Donta

Department of Computer
and Systems Sciences



Abbreviations

Abbreviation	Description
DCs	Data Centers
CS	Cloud Server
IoT	Internet of Things
IIoT	Industrial Internet of things
IBTS	Index Based Transmission Scheduling
DMO	Device Matching Order
DSR	Design Science Research
MCU	Microcontroller Unit
QoS	Quality of Service
IEEE	Institute of Electrical and Electronics Engineers
PBCO	Priority-Based Computation Offloading
RCO	Random Computation Offloading
TSCO	Transmission Scheduling and Computation Offloading
OPEX	Operating Expense
SLR	Systematic Literature Review
DDTP	Deadline-aware dynamic task placement
EETO	Energy-Efficient Task Offloading
DPTO	Delay-dependent priority-aware task offloading
CORL	Computation Offloading using Reinforcement Learning

Table of Contents

1	Introduction	5
1.1	Background.....	7
1.2	Problem Statement.....	8
1.3	Research Question	9
1.4	Delimitations	9
2	Extended background	10
2.1	Overview	10
2.2	Literature Review	12
2.3	Computation Offloading strategy:	13
3	Methodology	16
3.1	The stage of explicate problem	16
3.2	Define Requirements	18
3.3	Design and Development.....	18
3.4	Evaluate Artefact	19
4	Conclusion.....	26
5	References	26

List of figures

Figure 1	Edge-Cloud hierarchy and processing/response time	6
Figure 2	Task offloading in IIOT-fog-cloud architecture	7

Figure 3 The Internet of Things and Fog Computing	10
Figure 4 Root Cause analysis on energy.....	17
Figure 5 machine learning model	20
Figure 6 hierarchical resource allocation	21
Figure 7b. distribution of delay-sensitive tasks	21
Figure 8 distribution of IIoT tasks across devices	21
Figure 9 task categorization.....	22
Figure 10 tasks distribution in hierarchical structure.....	22
Figure 11 energy consumption comparison.....	23

List of Tables

Table 1 Symbols used in formula.	15
Table 1 Symbols used in formula.	15

Abstract—With the rapid expansion of Industrial Internet of Things (IIoT) applications, energy consumption and delay-sensitive task offloading remain critical challenges, particularly in fog computing environments. This research addresses the critical and growing challenges of energy consumption and task offloading within the expanding realm of Industrial Internet of Things (IIoT). Despite the advancement in fog and cloud computing, a significant gap exists in current approaches. Traditional cloud-based solutions fail to meet stringent delay constraints due to long-distance data transmission. There is also need for more intelligent mechanisms for task prioritization, efficient transmission scheduling, and dynamic adaptation to fluctuating network conditions. To address these limitations, the primary objective of this research adopted a Design Science Research (DSR) methodology to develop an energy optimization framework, integrating a heuristic-based transmission scheduling strategy and a ML-based task-offloading. An intensive computer simulation demonstrates that the proposed framework achieves a 10–15% improvement in energy efficiency for IIoT task offloading compared to baseline models. The key findings of this paper obtained through intensive computer simulations demonstrate the effectiveness of ML-based intelligent framework.

The implications of these findings are substantial for the design and deployment of future IIoT systems. This research contributes valuable insights into creating "green" IIoT systems that improves Quality of Service (QoS) by meeting stringent energy constraints by illustrating the effective integration of machine learning models for intelligent resource management and task offloading in fog-cloud environments, this work provides a foundation for developing more sustainable and efficient industrial automation and smart environment solutions.

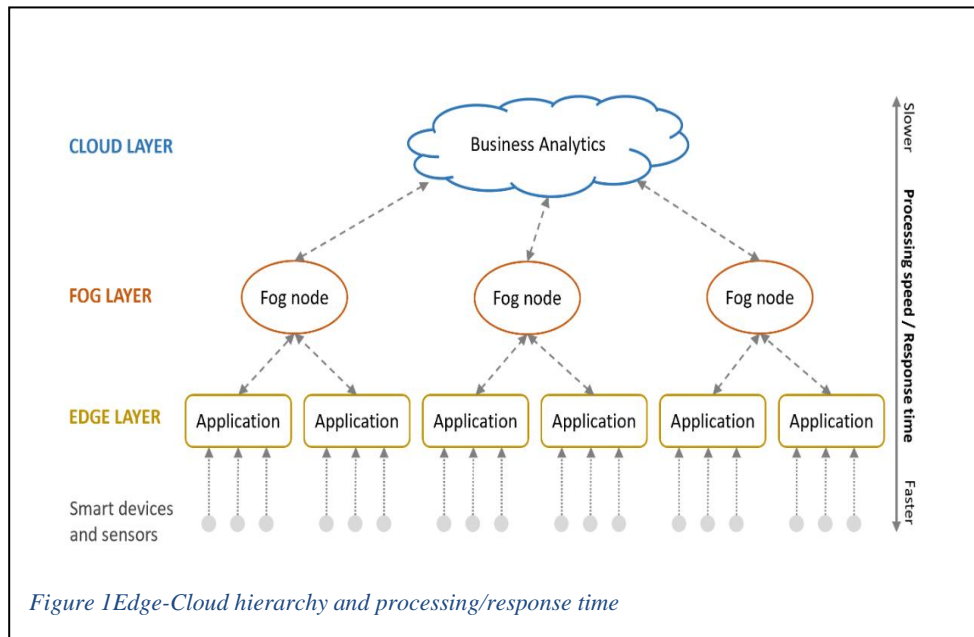
1 Introduction

This chapter introduces the area of IIoT along with the key factors influencing their performance, such as latency, energy consumption and scalability. This paper primarily focuses on developing an intelligent framework that enhances latency reduction and energy consumption in IIoT applications. By leveraging a hierarchical fog-cloud computing model, the proposed framework aims to optimize task allocation and minimize energy consumption. Through a comprehensive evaluation this research aims to contribute to the development of efficient, scalable IIoT systems, addressing critical challenges in modern industrial automation and smart environments.

The Internet of Things (IoT) has rapidly transformed various industries, driving advancements in smart cities, healthcare, transportation, and industrial automation ([Hazra et al., 2023](#)). IoT refers to a network of interconnected devices that collect, transmit, and process data to enable real-time decision-making. Among the various segments of IoT, the Industrial Internet of Things (IIoT) has gained significant attention due to its potential to revolutionize manufacturing, supply chain management, and industrial

processes. IIoT extends the benefits of IoT to industrial applications, enabling automation, predictive maintenance, and improved operational efficiency. As noted by [Donta et al. \(2021\)](#), Industrial IoT refers to the integration of IoT devices with manufacturing, automation, or control systems to collect and analyze data for enhanced machine efficiency and productivity. The demand for IIoT solutions is driven by the need for increased productivity, reduced downtime, and optimized resource utilization in industries such as manufacturing, energy, and logistics ([Sarkar et al., 2022](#)).

With the evolutionary development of the latency-sensitive industrial Internet-of-Things (IIoT) applications, delay restriction becomes a critical challenge, which can be resolved by distributing IIoT applications on nearby fog devices ([Hazra et al., 2023](#)). Traditional cloud-based architecture struggles to handle the massive volume of data generated by IIoT devices, leading to increased network congestion and delays in decision-making. In such circumstances, transferring a portion of data to a resource rich remote computing device, called computational offloading, which considered a suitable solution for handling sensitive IIoT applications ([Wang, Q., et al 2019](#)).



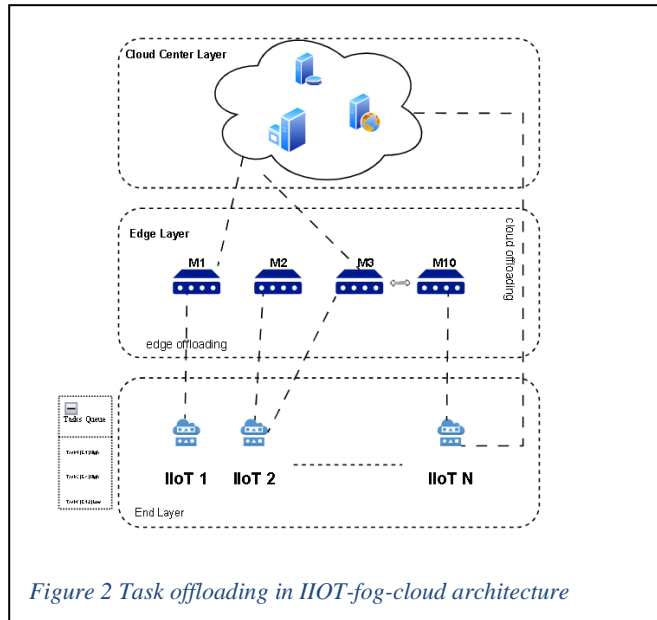
In the context of smart IoT, data processing and decision making are distributed across multiple layers of a hierarchical structure shown in [Fig. 1](#). This hierarchy typically consists of IoT devices, edge nodes, fog nodes and cloud servers. Each layer plays a critical role in processing data and reducing response time which is critical for real-time IoT applications. These real-time applications require adequate computing resources for processing and analyzing the incoming tasks. As a result, the local

IoT devices cannot meet the requirements of the incoming tasks due to their limited resource capacity ([Sarkar et al., 2022](#)). This demands the requirement from external resources from remote computing devices with efficient offloading strategies.

1.1 Background

The traditional cloud-centric IIoT model involves transmitting all sensor data to cloud servers for processing, storage, and analysis. While this model works well for applications that are not time-sensitive, it is impractical for real-time industrial applications such as predictive maintenance, robotics, and smart grid management ([Badidi, E. 2019](#)). The delay caused by the physical distance between IIoT devices and cloud servers can be detrimental in scenarios where immediate responses are required. In Addition, the continuous transmission of large volumes of data places a strain on network bandwidth and increases energy consumption ([Bonomi, F., et al 2012](#)).

A more efficient alternative is a hierarchical fog-cloud environment. In this model, fog computing provides intermediate processing between the IIoT devices and the cloud ([Bonomi, F., et al 2012](#)). Fog nodes, which can be routers, gateways, dedicated edge devices, process latency-sensitive tasks near the data source, while computationally intensive tasks are offloaded to the cloud. This hybrid approach minimizes latency, reduces energy consumption, and optimizes network resource utilization.



Several key concepts are central to understanding the development of an intelligent edge-cloud collaborative framework:

Computation Offloading: This technique involves transferring processing tasks from resource-limited IIoT devices to more powerful computing entities, such as fog nodes or cloud servers ([Wang et al. 2019](#)). Offloading reduces the energy consumption of IIoT devices and enhances processing efficiency. [Fig.2](#) shows a hierarchical fog-cloud architecture where task scheduling and computational offloading have been carried out ([Hazra et al. 2023](#)).

Transmission Scheduling: Prioritizing data transmission from IIoT devices based on factors such as network conditions and data importance helps manage network congestion and ensures timely delivery of critical information ([Hazra et al. 2023](#)).

Task-Device Matching: Efficiently allocating tasks to the most suitable fog nodes or cloud servers optimizes resource utilization and enhances system performance ([Avasalcai et al. 2021](#)).

Fog Nodes: These are intermediate computing devices that provide localized processing and storage, bridging the gap between IIoT devices and cloud servers ([Badidi, E. 2019](#)).

1.2 Problem Statement

Despite the advancements in fog and cloud computing, several challenges remain unaddressed in the context of IIoT applications. Existing frameworks often optimize either energy consumption or latency, but not both simultaneously. This leads to suboptimal performance in industrial environments where both metrics are critical. Current approaches lack efficient mechanisms to prioritize tasks based on their importance and deadlines, resulting in increased processing delays for critical applications.

The absence of an optimal device-matching strategy leads to inefficient resource utilization, as tasks are not always assigned to the most suitable computing devices (fog or cloud).

Industrial environments are characterized by dynamic network conditions, such as varying traffic loads and resource availability. Existing frameworks often fail to adapt to these changes, leading to performance degradation.

These challenges highlight the need for a holistic framework that jointly optimizes energy consumption and latency while addressing the limitations of existing approaches. The proposed TSCO framework aims to fill this gap by introducing innovative strategies for task prioritization, transmission scheduling, and computation offloading in fog-cloud environments.

1.3 Research Question

While existing research has made significant advancements in edge-cloud collaboration, several gaps remain that require further investigation:

Latency Optimization: Many existing frameworks focus on reducing latency but do not provide comprehensive mechanisms for dynamically adjusting based on real-time network conditions.

Although computation offloading reduces device energy consumption, optimal strategies for balancing workload distribution between fog nodes and the cloud remain underexplored.

Ensuring secure data transmission and processing within a heterogeneous fog-cloud environment presents a significant challenge. The lack of universally accepted standards hinders seamless integration and deployment across various IIoT ecosystems.

Based on these gaps, the following question emerges:

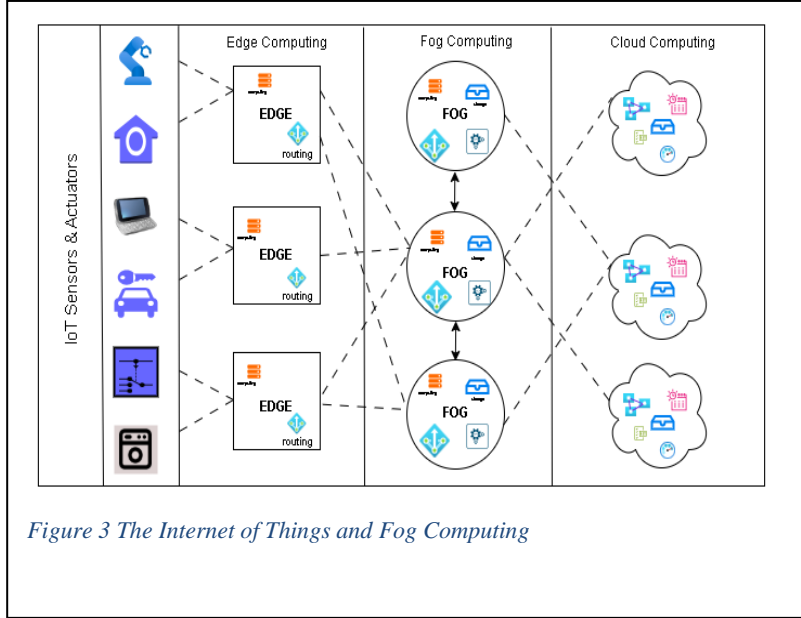
1. *How to implement a cooperative framework that offloads tasks to fog and cloud resources to minimize energy consumption in IIoT systems?*
2. *How does the collaboration between fog and cloud resources minimize the energy consumption in IIoT systems?*

1.4 Delimitations

This research focuses on developing an AI-driven edge-cloud collaborative framework for IoT applications that require real-time processing and energy-efficient task scheduling. The study is limited to synthetic data simulations and evaluates the framework based on total energy consumption and task completion rate. While latency is a crucial factor for real-time applications, it has not been explicitly modeled or evaluated in this study. The framework is compared with Random Computation Offloading (RCO) and Priority-Based Computation Offloading (PBCO). Other aspects of IoT systems, such as hardware design, network protocols, and security, are beyond the scope of this study. Additionally, the research does not address region-specific or time-bound IoT applications, nor does it consider dynamic scaling of resources. Future work will address this limitation by incorporating a latency model and developing offloading strategies with advanced AI/ML model that optimize for both energy and latency.

2 Extended background

This chapter aims to provide an extended theoretical background to the study and introduce related research. The first sections present the foundational concepts of edge-cloud computing, fog networks, and the challenges of energy consumption and latency in Industrial Internet of Things (IIoT) applications.



Based on a review of existing literature, an overview of current knowledge in the field is provided, highlighting the limitations of existing frameworks such as Random Computation Offloading (RCO) and Priority-Based Computation Offloading (PBCO). Finally, the chapter discusses the machine learning algorithms and mathematical modeling techniques used in the study to optimize task scheduling and

computation offloading in the proposed Transmission Scheduling and Computation Offloading (TSCO) framework.

2.1 Overview

Cloud computing is becoming increasingly popular these days due to its features like elasticity, scalability, and inexpensive [\(Wu, Wang, & Wolter, 2013\)](#). The “pay-as-you-go” cloud computing model is an efficient alternative to owning and managing private data centers (DCs) for customers facing Web application and batch processing [\(Bonomi, F., et al 2012\)](#). Bonomi points out that Several factors contribute to the economy of scale of mega DCs, higher predictability of massive aggregation, which allows higher utilization without degrading performance; convenient location that takes advantage of inexpensive power; and lower OPEX achieved through the deployment of homogeneous compute, storage, and networking components.

Here, cloud not only refers to the regional data centers but also the edge-cloud servers in the proximity of mobile service subscribers. With the proliferation of IoT, more data are created by widespread and geographically distributed mobile and IoT devices, and probably more than the data generated by the

mega-scale cloud datacenters ([Deng, S et al. 2020](#)). According to the prediction by Ericsson, 45% of the 40-ZB global Internet data will be generated by IoT devices in 2024 ([Ericsson, 2019](#)).

Considering such context, more real-time technologies, such as gaming, video-streaming, human-computer interaction. etc., have arisen and gained significant interest. These real-time applications require computing resources to process and analyze the incoming tasks. As a result, the local IoT devices with Node-MCU cannot meet the requirements due to their limited resource capacity ([Sarkar et al., 2022](#)).

However, several challenges are admitted due to physical distance and unreliable connection between IoT devices and CS. To overcome these issues, an emerging computing paradigm is introduced by CISCO, known as *fog computing*. The standard fog infrastructure consists of a set of local fog/edge devices that bring the cloud resources at the edge of the network and centralized CSs for backup purposes and secure data storage ([Hazra et al. 2021](#)). Thus, IoT devices tend to process latency-sensitive tasks locally using Node-MCU or offload them to nearby fog/edge devices to reduce latency and enhance reliability ([Zhou et al., 2021](#)). As a result, fog service providers deploy a network of fog and edge devices in a region to address the specific needs of IoT applications. However, each provider must manage and utilize its own resources, leading to a resource-constrained fog infrastructure that struggles to support large-scale IoT applications on local fog devices while still meeting diverse Quality-of-Service (QoS) requirements ([Sarkar et al., 2022](#)). Therefore, to meet QoS goals and manage emergency applications, a hierarchical fog–cloud setup is more advantageous for Industrial Internet of Things (IIoT) applications. In this model, cloud servers can handle resource-intensive tasks, while fog devices process time-sensitive applications simultaneously ([Zhou et al., 2019](#)). This raises concerns about the massive task processing within the fog network, which could lead to issues related to latency and energy consumption.

In recent decades, researchers have focused on edge-cloud collaboration and optimization techniques within fog layer. ([Li et al., 2018](#)) suggested that, while fog computing provides an account of advantages for IIoT, this geo-distributed computing architecture also brings several challenges, additionally due to the decentralization nature of fog computing, Li identified that there is no state observer to globally monitor and control the computing states of each fog-enabled IIoT service. Thus, fog computing urgently needs to be equipped with a novel function that can automatically partition computation resources for popular IIoT services.

To better understand the mechanisms of fog computing, and its effects ([Mukherjee et al., 2019](#)) demonstrated computational offloading becomes an important and essential research issue for the delay-sensitive task completion at resource-constraint end-users. Fog computing that extends the computing and storage resources of cloud computing to the network edge emerges as a potential solution towards low-latency task provisioning via computational offloading. Fig.4 shows how fog networks support low-latency communication scenarios via task offloading.

The integration of edge computing and cloud services has emerged as pivotal strategy for enhancing the performance and efficiency of IoT applications, enabling real-time data processing and analysis while minimizing latency. This collaborative framework not only optimizes resource utilization but also addresses the challenges of scalability and security, making it an attractive solution for various industries looking to leverage IoT technologies. The synergy between these two technologies further enhances the user experience and operational agility.

2.2 Literature Review

To better understand the current state of research gap and significance of TSCO framework, a SLR was conducted. The SLR analyzed and compared articles published between 2014 and 2024, focusing on existing frameworks for fog-cloud task offloading and highlights the advancements in TSCO framework.

Research papers for this study are selected from reputable databases such as Google Scholar, IEEE Xplore, ACM Digital Library, ResearchGate, and Springer. Papers that focus on task offloading and latency optimization are prioritized. Only peer-reviewed papers related to IIoT, and fog-cloud computing are included, while unrelated papers are excluded.

Over the last few years, numerous research efforts have been made to address issues related to computation offloading in the multitier fog–cloud architecture for handling various delay-restricted IIoT applications ([Adhikari et al. 2020](#)). With the introduction of fog computing years ago, numerous opportunities and challenges have emerged. This paradigm extends cloud computing capabilities closer to the edge, enabling low-latency processing and improved efficiency for IoT and real-time applications ([Sarkar et al. 2022](#)). However, it also brings challenges related to security, resource management, and scalability, requiring innovative solutions to fully harness its potential.

By providing the flexible and shared computing and communication resources along with the cloud services, the fog computing became an attractive paradigm to support delay-sensitive tasks in the IoT ([Adhikari et al., 2020](#)). According to ([Aazam et al. 2021](#)) IoT applications demand extensive storage and faster response to ensure seamless and interoperable communication. Hence, Edge Computing and data/task offloading among the edge or cloud servers become promising, while also posing critical research challenges for edge-enabled large-scale IoT ecosystems. Task offloading in fog computing for IoT applications has gained attention as an efficient alternative to cloud computing, offering reduced latency and improved quality of service ([Hussein & Mousa, 2020](#)).

In recent years, research has advanced with the development of novel frameworks aimed at addressing the challenges of energy and delay constraints in fog environments. ([Karagiannis et al., 2023](#)) developed a context-aware routing mechanism for fog computing systems that sends data directly to the

closest node that tends to accept data of the same context, in order to reduce communication latency and the number of hops traveled. (Karagiannis) proposed routing mechanism sends data directly to the closest node that tends to accept data of the same context, reducing the number of forwarding nodes and communication latency. ([Avasalcai et al., 2022](#)) introduced an EdgeFlow framework that assists developers in the development and deployment of latency-sensitive IoT applications on edge computing platforms by extending the flow-based programming paradigm and providing a resource allocation technique for deployment and validation. ([Mukherjee, M., et al. 2020](#)) proposed a latency-driven parallel task data offloading strategy in fog computing networks for industrial applications. ([Sarkar et al., 2022](#)) developed a Deadline-Aware Dynamic Task Placement (DDTP) strategy to offload and place tasks on suitable computing devices while meeting task deadlines. ([Adhikari et al., 2020](#)) proposed a new (DPTO) strategy that assigns priority to tasks based on deadlines, uses a multilevel feedback queue to reduce waiting time and starvation, and selects optimal computing devices to minimize overall offloading time while meeting task deadlines. ([Hazra, A., et al 2020](#)) introduces an (EETO) policy combined with a hierarchical fog network to handle the energy-performance trade-off by jointly scheduling and offloading real-time IoT applications using a heuristic priority assignment and a stochastic-aware data offloading approach with Lyapunov optimization. ([Yadav et al., 2021](#)) proposed a (CURL) scheme can save energy, minimize latency, and maximize resource utilization in edge-enabled sensor networks.

Most of the current strategies focus on optimal scheduling and computation offloading separately for reaching various QoS objectives ([Deng et al., 2020](#); [Avasalcai et al., 2022](#)). (Hazra et al 2023) argues that previous studies of task scheduling and offloading have not considered the importance of transmission scheduling and device matching strategy in the industrial for networks. even though the optimal device-matching strategy helps to offload tasks in suitable computing devices and utilize fog resources more efficiently ([Hazra et al 2023](#)).

These challenges encourage us to design a cooperative transmission scheduling and computation offloading framework that can minimize overall delay and energy consumption, where critical tasks can be prioritized based on network conditions ([Hazra et al., 2023](#)). However, a key challenge in designing this framework lies in two main aspects. First, determining an efficient transmission scheduling strategy for delay-sensitive IIoT applications. Second, defining an optimal task-device matching strategy in fog networks to ensure that IIoT-generated tasks are effectively offloaded to suitable devices.

2.3 Computation Offloading strategy:

This section introduces two key strategies developed to achieve the least possible energy consumption in fog networks.

2.3.1 Index Based Transmission Scheduling (IBTS)

According to (Hazra et al .,2023) identifying index of each IIoT device and allow them transmission is the processing step of proposed transmission scheduling strategy called (IBTS). Author defined the task generated via IIoT device are delay sensitive tasks, if device priority index is less than or equal to threshold. Otherwise, tasks are classified as resource intensive. Device priority index is calculated based on following formula:

$$\mathcal{A} = \arg \max_{a \in A} \left(\frac{\kappa_k^{in}(t)}{(\chi(\kappa, a) \times \lambda_\kappa)} \right) \beta_a$$

2.3.2 Device-Matching Order (DMO)

(Hazra et al., 2023) developed a (DMO) policy for distributing all the scheduled tasks among suitable fog device or cloud servers. This strategy trade off energy and delay for obtaining suitable computing devices from the device pool. Initially a $(k \times l)$ matrix is constructed where k represents rows and l represents columns. Each entry in the matrix denoted as e_{kl} a nonnegative heuristic information for task k to assign the resource l. The matrix E_{kl} computed using the following formula:

$$\text{minimize } \max_{k \in \mathcal{K}} \sum_{l=1}^{S_l} \chi(k, l) \cdot e_{kl}$$

Table 1 Symbols used in formula.

Symbols	Definition
\mathcal{K}	Set of IIoT tasks in the fog networks
\mathcal{A}	Set of IIoT devices in the fog networks
\mathcal{M}	Set of fog devices in industrial networks
\mathcal{N}	Set of cloud servers in the fog networks
\mathcal{G}	Set of IIoT gateway devices in the fog networks
λ_{κ}	Task arrival rate in the IIoT device A.
Γ_m^{CPU}	Computation frequency of fog device.
$\mathcal{X}(\kappa, \ell)$	Binary computation offloading matrix.
ϵ_{ℓ}^{max}	Maximum energy consumption threshold
$E_{\kappa\ell}^{total}$	Total energy consumption on computing device l.
$T_{ka}^{process}$	Task processing time of a th IIoT device.
β_{am}^{up}	Transmission bandwidth between IIoT device and fog device.
$\mathcal{K}_{\kappa}^{in}$	$(\mathcal{K}_{\kappa}^{CPU}, \mathcal{K}_{\kappa}^{freq}, \mathcal{K}_{\kappa}^{exe})$

The following steps are performed to achieve the goal of DMO policy:

Step 1: Order the e_{kl} values in nondecreasing order

Step 2: Find the minimum e_{kl} rank as an element (r) in the kth row and lth column E in increasing order until each column and row contains at least one element.

Step 3: Replace the entries e_{kl} of E according to: $e_{kl} = 0, \text{ if } e_{kl} \leq r \mid e_{kl} = e_{kl}$

Step 4: Consider a column (l) which had less number of zeros and assign all the tasks (k) whose associated values are 0 to the particular resource (either fog/cloud).

Step 5: Repeat step 4 until all the tasks are offloaded.

To clearly state the novelty of the TSCO framework, it is important to highlight the existing framework often focus on specific challenges, such as latency, energy consumption, without providing a holistic solution. TSCO jointly optimizes energy consumption and latency, addressing the limitations of existing approaches that focus on either energy efficiency or latency optimization, but not both simultaneously.

Dynamic Adaptation: TSCO's transmission scheduling strategy considers network dynamics to reduce transmission overhead. The TSCO framework includes innovative strategies for task prioritization ([Hazra et al., 2023](#)).

Fog-Cloud Collaboration: TSCO is designed to utilize both fog and cloud resources simultaneously, prioritizing emergency tasks based on network conditions. Fog devices process delay-sensitive applications, while cloud servers handle resource-intensive applications ([Hazra et al., 2023](#)).

In summary, the novelty of the TSCO framework lies in its holistic approach to jointly optimizing energy consumption and latency in dynamic network conditions, addressing the limitations of existing baseline algorithm and frameworks that often focus on specific aspects without providing a comprehensive solution.

3 Methodology

In this chapter, a detailed discussion is presented in the stages of DSR development for producing the artefact in the form of TSCO framework. The results provided in this chapter show the most suitable research strategies and the knowledge base needed during the development of artefact in every stage in Fig. 3, starting from state of explicated problem, defining requirements, designing and developing the artefact, demonstrating it and evaluating it.

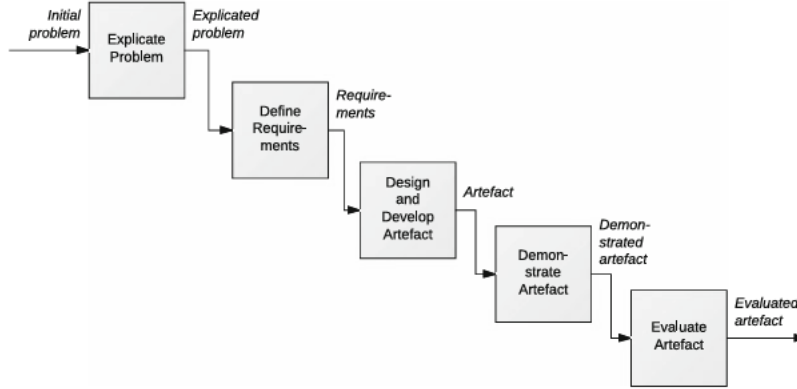


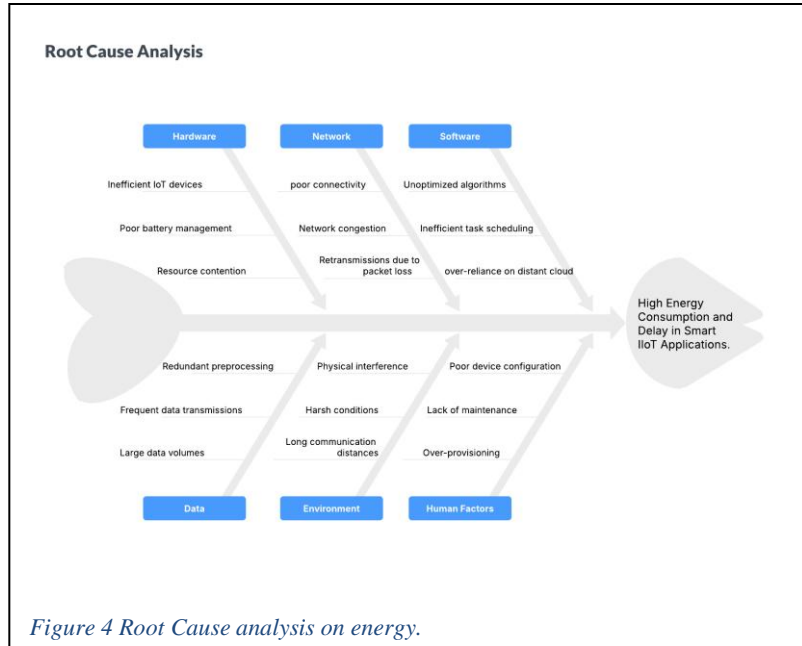
Fig. 3. The stages for conducting design science research using [\(Johannesson & Perjons, 2014\)](#)

The TSCO (Transmission Scheduling and Computation Offloading) framework is designed to address energy consumption challenges in delay-sensitive Industrial Internet of Things (IIoT) applications within fog networks. The framework is developed using the Design Science Research Methodology (DSR), which consists of five key activities: Explicate Problem, Define Requirements, Design and Develop Artefact, Demonstrate Artefact, and Evaluate Artefact.

3.1 The stage of explicate problem

This step identifies the main issue, which is the high energy consumption of IIoT applications in fog networks that are delay sensitive. This issue is particularly critical in scenarios involving emergency tasks, where efficient resource utilization and timely processing are essential. The significance of this problem lies in the need to balance energy efficiency and low-latency requirements in fog environments.

The problem appears in the practice of delay-sensitive Industrial Internet of Things (IIoT) applications within fog networks. Existing systems often struggle to balance energy efficiency and low-latency requirements in fog environments. The problem is that current frameworks either optimize energy consumption or latency, but typically not both. Moreover, existing frameworks often fail to adapt to dynamic changes in network conditions, traffic loads, and resource availability, leading to performance degradation.



Unoptimized algorithms and ineffective tasks offloading mechanisms that force resource-intensive computation and needless cloud dependencies are among the major contributing factors to excessive energy according to the root cause analysis.

Addressing this problem is important because energy consumption is a serious challenge with the continuous development and diversity of IIoT applications in fog networks. Conventional

cloud technology cannot adhere to the delay requirement of sensitive IIoT applications due to long-distance data travel. An even better strategy of fog association, transmission scheduling, and computation offloading not only minimizes the energy consumption rate but also increases the QoS for delay-critical IIoT applications. The problem is significant for stakeholders who want to find a solution.

The problem is of interest beyond a local practice because the Internet of Things (IoT) has rapidly transformed various industries, driving advancements in smart cities, healthcare, transportation and industrial automation. A framework that enhances latency reduction and energy consumption in IIoT applications will contribute to the development of efficient, scalable IIoT systems, addressing critical challenges in modern industrial automation and smart environments.

The problem is solvable by designing a novel energy optimization framework called ML-based transmission scheduling and computation offloading , while maintaining energy constraints in the fog environment. This involves using a heuristic-based transmission scheduling strategy to transfer IIoT-generated tasks based on their importance and a graph-based task-offloading strategy using constrained-restricted mixed linear programming to handle high traffic in rush-hour scenarios.

The sources of the problem are identified in existing literature and through the experiences of stakeholders dealing with IIoT systems. The limitations of existing frameworks such as Random Computation Offloading (RCO) and Priority-Based Computation Offloading (PBCO) have been highlighted.

The problem has been explicated through a review of existing literature, which revealed the limitations of current approaches in addressing both energy consumption and latency in dynamic IIoT environments. Stakeholders' involvement would help to define the requirements for the TSCO framework, ensuring it aligns with real-world needs and constraints

3.2 Define Requirements

According to [\(Johannesson & Perjons, 2014\)](#), this step involves identifying the type of artifacts needed and eliciting both functional and non-functional requirements. Functional requirements include efficient task offloading and transmission scheduling, while non-functional requirements could focus on scalability, reliability, and security.

The TSCO framework will be an algorithm designed to optimize collaboration between edge and cloud computing resources in IoT applications. It is a novel energy optimization framework called transmission scheduling and computation offloading (TSCO), while maintaining energy and delay constraints in the fog environment.

Formulated Requirements:

- The framework shall efficiently offload tasks from IIoT devices to fog nodes or the cloud based on network conditions and task priorities in comparison to baseline algorithms.
- The framework shall intelligently predict delay-sensitive and resource intensive tasks generated from IIoT.
- The framework should be able to handle a growing number of IIoT devices and increase data volumes without significant performance degradation.
- The system should ensure reliable task execution and data transmission, even in the presence of network failures or node outages.
- The framework should minimize energy consumption comparatively by optimizing task offloading and transmission scheduling decisions.
- The framework should ensure low-latency task execution comparatively to meet the stringent delay requirements of IIoT applications.
- The framework should adapt to dynamic changes in network conditions, traffic loads, and resource availability.

The requirements are derived from Literature on energy consumption challenges in IIoT applications and the limitations of existing frameworks. An interview with a publisher, who have contributed to defining the needs for efficient task offloading, transmission scheduling, scalability, reliability, energy efficiency, low latency, and adaptability.

3.3 Design and Development

Development of the framework approach is based on the specified needs. This entails creating a transmission scheduling policy that takes into account variables like transmission rate and data size.

Transmission Scheduling Module:

- *Functionality:* Prioritizes and schedules the transmission of tasks generated by IIoT devices based on their importance and urgency. This ensures that critical tasks are processed promptly, while less urgent tasks are scheduled to optimize resource utilization.
- *Structure:* This module uses a heuristic-based algorithm to determine the order in which tasks are transmitted. The algorithm considers factors such as task priority, deadline, and network conditions to make scheduling decisions.

Developing a device-matching strategy to ensure efficient task offloading between fog nodes and IIoT devices.

Computation Offloading Module:

- *Functionality:* Determines whether tasks should be processed locally at the fog node or offloaded to the cloud based on network conditions, task requirements, and resource availability. This module balances the need for low-latency processing with the benefits of cloud computing.
- *Structure:* The module employs a graph-based task-offloading strategy using constrained-restricted mixed linear programming to handle high traffic scenarios. The graph represents the network topology, task dependencies, and resource constraints.

Machine learning model:

- *Functionality:* A machine learning model employed to predict delay-sensitive and resource constraints tasks, this model calculates the energy required for tasks and compares it with frequency of processing devices.
- *Structure:* Logistic regression model used for classification, a supervised classification model suitable for binary and multi-class classification tasks.

Consider a scenario where an IIoT device generates a task for real-time monitoring of a manufacturing process. The machine learning model analyzes the task's characteristics (e.g., small data size, high priority) and predicts that it is delay-sensitive. Based on this prediction, the Computation Offloading Module decides to process the task locally on a fog node to minimize latency. Simultaneously, the model estimates energy consumption and confirms that local processing is also energy efficient.

The ultimate goal is to create a framework that optimizes both energy consumption and delay for IIoT applications.

3.4 Evaluate Artefact

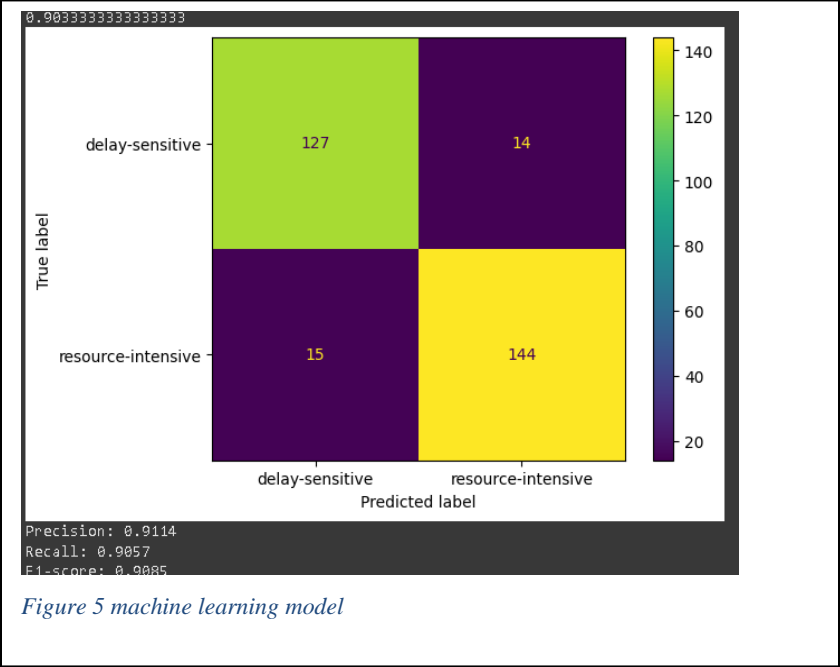
The final step involves assessing the degree to which the framework meets the specified requirements. This is accomplished by comparing the developed artefact with similar existing ones and examining any potential side effects associated with its use. A formative evaluation is used, as it enables assessment

during the framework’s development phase. An ex-ante evaluation is preferred over an ex-post evaluation due to its cost-effectiveness, efficiency, and suitability for formative evaluation.

The evaluation setting can be artificial, utilizing computer simulations to demonstrate the performance of the developed artefact in comparison to similar frameworks. The key metrics considered in the evaluation include energy consumption – The total energy consumed by the system during task execution.

The evaluation process provides valuable insights into the framework’s performance and highlights areas that require further refinement.

Figure.6 demonstrates a balanced model capable of predicting the tasks offloading efficiently.



In order to assess the efficacy of our proposed task of offloading and resource allocation framework, we developed a synthetic data generation mechanism using Python. The simulation function produces a heterogeneous IIoT environment comprising 20 devices with varying computation capabilities and communication parameters. For each device, the function generates tasks with randomly assigned attributes such as required CPU, frequency, execution deadline, this synthetic dataset, encompassing 1000 tasks, is then used to train and evaluate a Logistic Regression model for

task classification. The model achieves high accuracy and precision, exceeding 91% in both metrics when tested on 300 independent test cases, demonstrating its capability to effectively differentiate between task types and guide intelligent offloading decisions.

Figure 7 illustrates the varying computational capacities and frequencies across the three tiers of our IIoT system: IIoT devices, fog servers, and cloud servers. The proposed task offloading algorithm leverages these heterogeneous resources by intelligently assigning tasks based on priority, device capacity, and the current load at each tier. Consider a scenario with 100 tasks queued in the system. The algorithm first evaluates the priority of each task, as determined by the machine learning model's prediction. Concurrently, it assesses the frequency capabilities of available IIoT devices. Tasks whose frequency requirements are compatible with an available device's frequency are executed locally, minimizing latency and energy consumption. However, if a task's frequency demands exceed the capabilities of local IIoT devices, it is offloaded to the next tier – the fog servers. Fog servers, with their enhanced computational resources, handle a broader range of tasks. Tasks that still cannot be accommodated at the fog level are further offloaded to the cloud servers, ensuring all tasks are eventually processed. This hierarchical approach optimizes resource utilization by matching tasks with the most suitable processing environment based on their priority and computational demands.

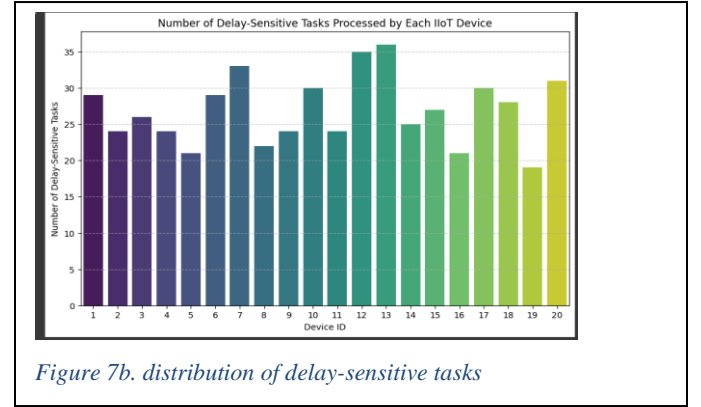
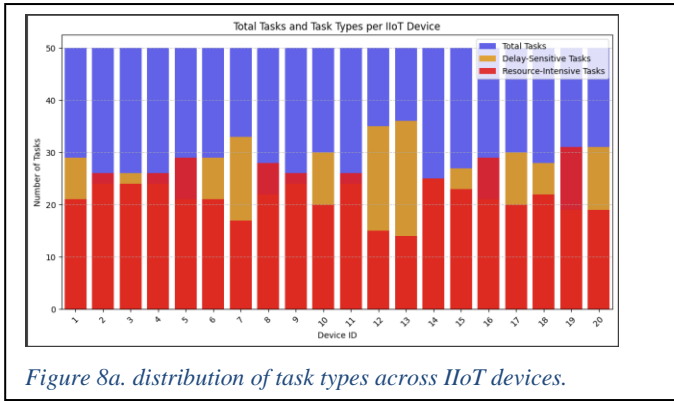
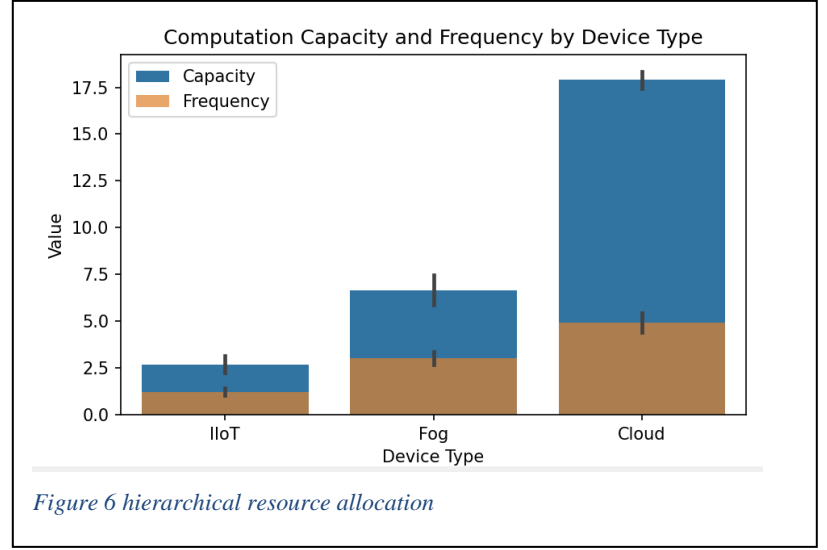
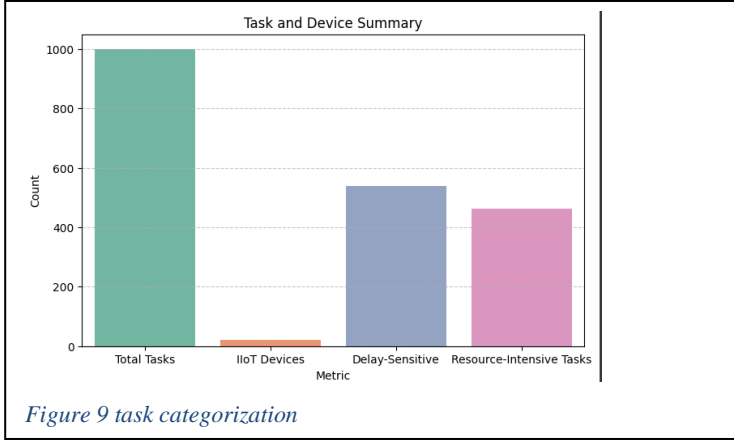


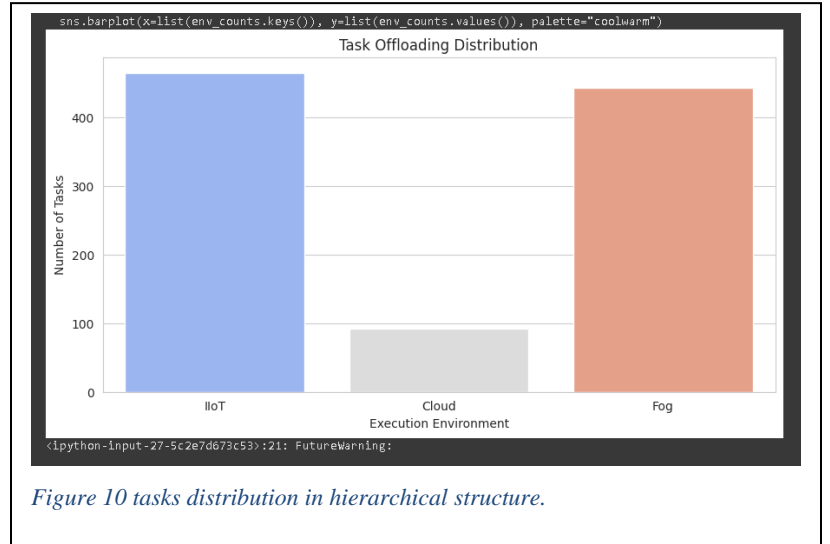
Figure 8.a and 8.b illustrates the distribution of tasks on various devices. To analyze the distribution of tasks across the IIoT devices, we employed a visualization technique using Python libraries seaborn and matplotlib (McKinney et.al 2023). Task data, including classifications as either delay-sensitive or resource-intensive, was loaded from a CSV file generated during the simulation. This

data was then grouped by device ID, calculating the total tasks assigned to each device and the distribution of task types. The resulting information was visualized using stacked bar plots, with the total task count represented by blue bars and further broken down by orange bars for delay-sensitive tasks and red bars for resource-intensive tasks. This visualization allowed for a comprehensive understanding of the workload distribution across the IIoT environment, revealing the number and types of tasks assigned to each individual device.

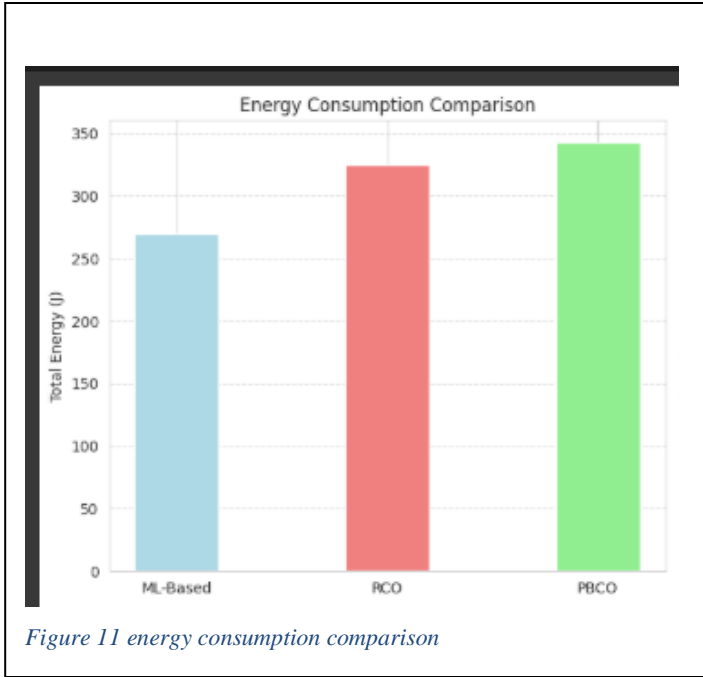


A total of 1000 tasks were generated across 20 IIoT devices in the simulation. Of these tasks, 538 were classified as delay-sensitive, while the remaining 462 were identified as resource-intensive. Figure 10 illustrates the distribution of delay-sensitive and resource-intensive tasks generated by each individual device.

This research implements a three-tier hierarchical framework for efficient task execution in an IIoT environment, encompassing IIoT devices, fog nodes, and cloud servers. Figure 11 illustrates the distribution of tasks over various environments. The task offloading strategy prioritizes local processing on IIoT devices if their frequency capacity aligns with task requirements. When local execution is insufficient, a trained machine learning model classifies tasks as either delay-sensitive or resource-intensive, leveraging features like required CPU, frequency, and deadline. This classification guides the offloading decision: delay-sensitive tasks are directed to the fog tier to minimize latency, while resource-intensive tasks are offloaded to the cloud for access to greater computational resources. This hierarchical approach optimizes resource utilization by balancing processing needs with latency and energy considerations. The visualization results, depicting



task distribution and overall energy metric, further substantiate the efficacy of this strategy in enhancing the performance of IIoT systems.



The analysis of energy consumption in the IIoT system reveals that the ML-based offloading strategy significantly reduces energy consumption approx. 10%-15% compared to RCO and PBCO. The ML model's intelligent task scheduling minimizes unnecessary offloading to remote resources (fog or cloud), where energy costs are higher due to data transmission and remote processing. While RCO offloads resource-intensive tasks to the cloud for performance

gains, it increases overall energy usage. PBCO, with its random fog/cloud offloading, exhibits the highest energy consumption due to its inefficient task allocation. Consequently, the ML-based approach emerges as the most energy-efficient strategy by strategically optimizing task execution locations to minimize energy expenditure in the IIoT environment.

3.4.1 Summary of findings

In the simulated IIoT environment, the suggested ML-based task offloading architecture showed encouraging energy efficiency outcomes. In comparison to baseline techniques like RCO and PBCO, the framework significantly reduced energy consumption by classifying workload intelligently and making well-informed offloading decisions. This is explained by the Logistic Regression model's capacity to distinguish between tasks that require a lot of resources and those that are delay sensitive. This allows tasks to be completed in the most energy-efficient processing environments, such as local, fog, or cloud. Unlike baseline techniques that do not have this task-specific understanding, the framework uses a Logistic Regression model to intelligently classify incoming tasks as either resource-intensive or delay-sensitive. This classification allows for informed offloading decisions, directing tasks to the most suitable processing environment (local device, fog server, or cloud server). The framework achieves a substantial reduction in energy consumption, ranging from 10% to 15%, compared to baseline offloading strategies. In contrast to the ML-based approach, PBCO offloads tasks

based on a fixed probability without considering the particular requirements of each task, resulting in suboptimal energy usage, while RCO offloads all tasks to the cloud, causing high energy consumption due to data transmission and remote processing. The primary benefit of employing ML models is their capacity to distinguish between tasks that require a lot of resources and those that are delay-sensitive; this ability establishes whether the tasks should be completed locally or in close proximity to fog devices. By assigning the activities to appropriate contexts, machine learning-based resources were intelligently optimised. This tactic reduces needless energy usage related to data transmission and processing. On the other hand, RCO and PBCO frequently lead to excessive energy use and inefficient resource allocation.

In summary, the ML-based task offloading framework presents a promising approach to enhancing energy efficiency in IIoT systems. Unlike traditional methods such as RCO and PBCO, it intelligently classifies tasks and makes informed offloading decisions, leading to significant energy savings (10-15%). This energy optimization is achieved by directing tasks to the most suitable processing environments, minimizing data transmission costs, and leveraging the strengths of different computing resources within the IIoT ecosystem.

3.4.2 Limitations and challenges

The latency component of task offloading is not specifically addressed in this study, which mainly concentrates on energy expenditure. In practical IIoT applications, latency is an essential component that can greatly affect user experience and performance. A thorough latency investigation was omitted due to time constraints and the difficulty of including latency calculations. This simulation offers insights into the energy efficiency of various task offloading mechanisms, however it employs synthetically generated data that could not fully reflect the features and behaviour of real-world IIoT setups. To create more thorough and practical models for IIoT system optimization, it is crucial to recognize the constraints and potential topics for further study..

3.4.3 Future research

Future studies could focus on two main aspects. At first, the research could also aim to reflect previously underrepresented latency constraint in a similar context. Secondly, the research could focus on advanced AI/ML models to enhance the energy and delay in cooperative task scheduling and computation offloading IIoT smart application.

3.4.4 Ethicality

IIoT systems frequently gather and analyze sensitive data, which raises privacy and security concerns. Future studies that use actual IIoT-generated data should think about data anonymization and encryption to safeguard private information. Bias may be introduced by the algorithms used to decide which tasks to offload, which could result in unfair outcomes for some devices. Making sure that these

algorithms are created, trained on a variety of datasets, and thoroughly assessed for fairness is crucial. AI-driven job offloading systems must have open and responsible decision-making procedures. In addition to being able to recognize and resolve any possible problems or biases, users should have a thorough comprehension of the decision-making process.

3.4.5 *Simulation Environment*

The system used for the simulation studies included an Intel Core i5-2400 CPU running at 2.42 GHz with four cores, 8 GB of RAM, and Windows installed. The Python programming language was used to create the simulation environment, utilizing pertinent libraries for data manipulation, analysis, and machine learning modelling, including NumPy, Pandas, and Scikit-learn.

A system model called IIoT System was put into place to depict the IIoT environment's hierarchical structure. Three different hierarchical environments—local (IIoT devices), fog, and cloud—are simulated by this model. Each environment is distinguished by its frequency, transmission rate, and processing capability. In order to represent the different processing capacities and communication traits of the various tiers, these factors were allocated to each environment.

A network of 20 IIoT devices, 10 fog devices for intermediate processing, and two cloud servers for resource-intensive operations was taken into consideration in the simulated studies. The 20 IIoT devices produced a total of 1000 tasks, and 40 Mbps was allocated as the network bandwidth. To decide whether a job should be offloaded based on its priority index, an offloading determination threshold of 0.5 was used. Required CPU, frequency, execution deadline, transmission time, and energy consumption were all essential task parameters.

The task type, which was classified as 0 for resource-intensive activities and 1 for delay-sensitive tasks, served as the dependent variable in the linear regression model. A Logistic Regression model was used to predict this variable based on the previously given task characteristics.

The simulation workflow comprised the following steps:

Data Generation: The `generate_labeled_data()` function was used to create a synthetic dataset of tasks, incorporating features and labels (Task_Type) based on a calculated priority index.

Model Training: A Logistic Regression model was trained using the generated dataset to classify tasks as either delay-sensitive or resource-intensive.

Task Offloading: The `task_offloading()` function simulated the offloading decisions by employing the trained model and considering device capabilities and task characteristics.

Evaluation: The simulation results were evaluated using metrics such as the distribution of tasks across execution environments (local, fog, and cloud), total energy consumption.

4 Conclusion

In summary, a key component of the system is the integration of a machine learning model for the classification of prediction tasks. providing a major benefit over conventional offloading techniques in terms of wise decision-making and huge energy savings. Through the coordinated and effective use of fog and cloud resources, this ML-driven strategy successfully answers research questions. Eventually, paving the way for the creation of IIoT systems that are more responsive and energy efficient and that can satisfy the demanding requirements of industrial applications. The suggested framework for computational offloading and intelligent transmission scheduling makes use of design science research methodology. provides a comprehensive answer to these problems. The key integration of ML model specifically logistic regression model plays a crucial role in predicting delay sensitive and resource intensive tasks with high accuracy of 91%.

5 References

--

- [1] Hazra, A., Donta, P. K., Amgoth, T., & Dustdar, S. (2023). Cooperative Transmission Scheduling and Computation Offloading With Collaboration of Fog and Cloud for Industrial IoT Applications. *IEEE Internet of Things Journal*. Doi:10.1109/JIOT.2022.3150070
- [2] Sarkar, I., Adhikari, M., Kumar, N., & Kumar, S. (2022). Dynamic Task Placement for Deadline-Aware IoT Applications in Federated Fog Networks. *IEEE Internet of Things Journal*. Doi: 10.1109/JIOT.2021.3088227.
- [3] Li, G., Wu, J., Li, L., Wang, K., & Ye, T. (2018). Service Popularity-Based Smart Resources Partitioning for Fog Computing-Enabled Industrial Internet of Things. *IEEE Transactions on Industrial Informatics*. DOI: 10.1109/TII.2018.2845844.
- [4] Wang, J., Liu, K., Li, B., Liu, T., Li, R., & Han, Z. (2020). Delay-Sensitive Multi-Period Computation Offloading With Reliability Guarantees in Fog Networks. *IEEE Transactions on Mobile Computing*. Doi: 10.1109/TMC.2019.2918773.
- [5] Yadav, R., Zhang, W., Kaiwartya, O., Song, H., & Yu, S. (2020). Energy-Latency Tradeoff for Dynamic Computation Offloading in Vehicular Fog Computing. *IEEE Transactions on Vehicular Technology*. DOI:10.1109/TVT.2020.3040596

- [6] Wang, Q., Guo, S., Liu, J., & Yang, Y. (2019). Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing. *Sustainable Computing: Informatics and Systems*, 21, 154–164. <https://doi.org/10.1016/j.suscom.2019.01.007>
- [7] C. Avasalcay, B. Zarrin and S. Dustdar, "EdgeFlow—Developing and Deploying Latency-Sensitive IoT Edge Applications," in *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3877–3888, 1 March1, 2022, Doi: 10.1109/JIOT.2021.3101449.
- [8] Badidi, E. (2019). A Fog Node Architecture for Real-Time Processing of Urban IoT Data Streams. *Computer Science On-line Conference*. DOI:10.1007/978-3-030-19807-7_32
- [9] Johannesson P, Perjons E. An introduction to design science vol. 10. Springer; 2014. <https://dx.doi.org/10.1007/978-3-030-78132-3>.
- [10] Donta, P. K., Srirama, S. N., Amgoth, T., & Annavarapu, C. S. R. (2022). Survey on recent advances in IoT application layer protocols and machine learning scope for research directions. *Digital Communications and Networks*, 8(5), 727–744. <https://doi.org/10.1016/j.dcan.2021.10.004>.
- [11] Hazra, A., Adhikari, M., Amgoth, T., & Srirama, S. N. (2023). Collaborative AI-enabled intelligent partial service provisioning in green industrial fog networks. *IEEE Internet of Things Journal*, 10(4), 2913–2921. <https://doi.org/10.1109/JIOT.2021.3110910>.
- [12] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of Things. In *MCC '12: Proceedings of the first edition of the MCC workshop on Mobile cloud computing* (pp.13-16).<https://doi.org/10.1145/2342509.2342513>
- [13] Wu, H., Wang, Q., & Wolter, K. (2013). Tradeoff between performance improvement and energy saving in mobile cloud offloading systems. In *2013 IEEE International Conference on Communications Workshops (ICC)* (pp. 728–732). IEEE. <https://doi.org/10.1109/ICCW.2013.6649329>
- [14] Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., & Zomaya, A. Y. (2020). Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet of Things Journal*, 7(8), 7457–7469. <https://doi.org/10.1109/JIOT.2020.2984887>
- [15] Ericsson. (2019). *IoT connections outlook: NB-IoT and CAT-M technologies will account for close to 45 percent of cellular IoT connections in 2024*. Ericsson. <https://www.ericsson.com/en/mobility-report/reports/june-2019/iot-connections-outlook>
- [16] Hazra, A., Adhikari, M., Amgoth, T., & Srirama, S. N. (2021). Stackelberg game for service deployment of IoT-enabled applications in 6G-aware fog networks. *IEEE Internet of Things Journal*, 8(7), 5185–5193. <https://doi.org/10.1109/JIOT.2020.3041102>
- [17] Zhou, P., Shen, K., Kumar, N., Zhang, Y., Hassan, M. M., & Hwang, K. (2021). Communication-efficient offloading for mobile-edge computing in 5G heterogeneous networks. *IEEE Internet of Things Journal*, 8(13), 10237–10247. <https://doi.org/10.1109/JIOT.2020.3029166>

- [18] Zhou, Z., Guo, Y., He, Y., Zhao, X., & Bazzi, W. M. (2019). Access control and resource allocation for M2M communications in industrial automation. *IEEE Transactions on Industrial Informatics*, 15(5), 3093–3103. <https://doi.org/10.1109/TII.2019.2903100>
- [19] Mukherjee, M., Kumar, S., Shojafar, M., Zhang, Q., & Mavromoustakis, C. X. (2019). Joint task offloading and resource allocation for delay-sensitive fog networks. In *2019 IEEE International Conference on Communications (ICC)* (pp. 1–7). IEEE. <https://doi.org/10.1109/ICC.2019.8761239>.
- [20] Hazra, A., Kalita, A., & Gurusamy, M. (2024). Meeting the requirements of Internet of Things: The promise of edge computing. *IEEE Internet of Things Journal*, 11(5), 7474–7498. <https://doi.org/10.1109/JIOT.2023.3339492>
- [21] Aazam, M., Zeadally, S., & Feo Flushing, E. (2021). Task offloading in edge computing for machine learning-based smart healthcare. *Computer Networks*, 191, 108019. <https://doi.org/10.1016/j.comnet.2021.108019>
- [22] Yan, J., Wu, D., Zhang, C., et al. (2018). Socially aware D2D cooperative communications for enhancing Internet of Things applications. *Journal of Wireless Communications and Networking*, 132. <https://doi.org/10.1186/s13638-018-1127-0>.
- [23] Adhikari, M., Mukherjee, M., & Srirama, S. N. (2020). DPTO: A deadline and priority-aware task offloading in fog computing framework leveraging multilevel feedback queueing. *IEEE Internet of Things Journal*, 7(7), 5773–5782. <https://doi.org/10.1109/JIOT.2019.2946426>.
- [24] Hussein, M. K., & Mousa, M. H. (2020). Efficient task offloading for IoT-based applications in fog computing using ant colony optimization. *IEEE Access*, 8, 37191–37201. <https://doi.org/10.1109/ACCESS.2020.2975741>.
- [25] Karagiannis, V., Frangoudis, P. A., Dustdar, S., & Schulte, S. (2023). Context-aware routing in fog computing systems. *IEEE Transactions on Cloud Computing*, 11(1), 532–549. <https://doi.org/10.1109/TCC.2021.3102996>
- [26] Hazra, A., Adhikari, M., Amgoth, T., & Srirama, S. N. (2020). Joint computation offloading and scheduling optimization of IoT applications in fog networks. *IEEE Transactions on Network Science and Engineering*, 7(4), 3266–3278. <https://doi.org/10.1109/TNSE.2020.3021792>
- [27] McKinney, W., et al. (2023). *pandas: Python Data Analysis Library* (Version 2.0.3) [Computer software]. Available at <https://pandas.pydata.org/about/citing.html>
- [28] Waskom, M. L. (2023). *seaborn: Statistical Data Visualization Library* (Version 0.13.2) [Computer software]. Available at <https://seaborn.pydata.org/citing.html>

