Resource-Aware Federated Hybrid Profiling for Edge Node Selection in Federated Patient Similarity Network

Alramzana Nujum Navaz 1,*, Hadeel T. El Kassabi 2, Mohamed Adel Serhani 3 and and Ezedin S. Barka 4

1 Department of Computer Science and Software Engineering, College of Information Technology, UAE University, Al Ain P.O. Box 15551, United Arab Emirates
2 Faculty of Applied Sciences & Technology, Humber College Institute of Technology & Advanced Learning, Toronto, ON M9W 5L7, Canada; hadeel.el-kassabi@humber.ca
3 College of Computing and Informatics, Sharjah University, Sharjah P.O. Box 27272, United Arab Emirates; mserhani@sharjah.ac.ae
4 Department of Information Systems and Security, College of Information Technology, UAE University, Al Ain P.O. Box 15551, United Arab Emirates; ebarka@uaeu.ac.ae

* Correspondence: 201570182@uaeu.ac.ae

Abstract: The widespread adoption of edge computing for resource-constrained devices presents challenges in computational straggler issues, primarily due to the heterogeneity of edge node resources. This research addresses these issues by introducing a novel resource-aware federated hybrid profiling approach. This approach involves classifying edge node resources with relevant performance metrics and leveraging their capabilities to optimize performance and improve Quality of Service (QoS), particularly in real-time eHealth applications. Such paradigms include Federated Patient Similarity Network (FPSN) models that distribute processing at each edge node and fuse the built PSN matrices in the cloud, presenting a unique challenge in terms of optimizing training and inference times, while ensuring efficient and timely updates at the edge nodes. To address this concern, we propose a resource-aware federated hybrid profiling approach that measures the available static and dynamic resources of the edge nodes. By selecting nodes with the appropriate resources, we aim to optimize the FPSN to ensure the highest possible Quality of Service (QoS) for its users. We conducted experiments using edge performance metrics, i.e., accuracy, training convergence, memory and disk usage, execution time, and network statistics. These experiments uniquely demonstrate our work’s contribution to optimizing resource allocation and enhancing the performance of eHealth applications in real-time contexts using edge computing.

Keywords: federated learning; federated resource profiling; deep learning; edge computing; workload profile; PSN; FPSN; eHealth

1. Introduction

Healthcare applications and devices collect vast amounts of patient data from sources such as sensors and medical equipment. These data have grown exponentially in recent years, making it difficult for providers to efficiently manage the data and associated resources. Edge computing is essential for utilizing automation and machine learning technologies to process and store massive amounts of data at the network’s edge, close to the point of collection. This will allow doctors to take immediate actions to help patients avert medical emergencies. The healthcare sector may become more streamlined and efficient, thanks to edge computing.

The concept of Federated Learning (FL) [1–3] has recently gained popularity in the healthcare industry as a means of solving the issues of data privacy and security through the cooperative training of algorithms that does not require the sharing of individual data sets, while it can be challenging to obtain large amounts of medical data, FL helps solve this problem by facilitating learning through collaboration rather than data centralization.
Without transferring patient data outside of their current institutions’ firewalls, FL enables interactive insight-gathering, for example in the form of a consensus model. Instead, ML is performed locally at each participating institution, and only the characteristics of the models themselves (such as parameters and gradients) are shared. Researchers have found that FL-trained models outperform those that have access to only data from a single institution or node.

When the data defining the optimizer is dispersed unevenly among a very large number of nodes, ML presents another important context for distributed optimization. The goal was to minimize the number of communication rounds while maintaining highly efficient communication in order to train a high-quality centralized model known as Federated Optimization [4], which can provide on-device intelligence. With the increasing edge computing challenges of processing, storage, and network I/O, federated resource profiling is a promising approach that can assess local computation capacities and enhance the resource utilization capabilities of a distributed computing environment.

FL supported by resource profiling will have a minimal overhead because it will only evaluate the measurements and variance of the edge node’s resources. Quality of Service (QoS) and Quality of Experience (QoE) are key metrics in evaluating telecommunication services. QoS primarily concerns the technical aspects and performance metrics of a service, such as latency and bandwidth, while QoE focuses on the subjective user experience, encompassing aspects like satisfaction and usability. Together, these metrics provide a comprehensive understanding of service effectiveness from both technical and user-centric perspectives. Given that health tasks require a significant amount of computing power, the system must ensure strict QoS and QoE requirements. Moreover, each edge node can only perform computations up to its maximum capacity. A data-driven federation offers a trifecta of benefits, including the ability to optimize the consumption of data resources, reduce transportation costs, and enforce data privacy.

The proposed Federated Resource Profiling (FRP) model is intended to be used in the context of Patient Similarity Networks (PSNs) [5–8] and Federated PSN (FPSNs) [9], a relatively new network-based approach in precision medicine in which patients are grouped or categorized according to their similarities in a variety of characteristics. Similar instances may be grouped together to reveal common patterns in the progression of illnesses [10]. FPSN models distribute processing across federated nodes and then fuse the PSN matrices in the cloud, highlighting the need for a balance between computational speed and accuracy. This research provides valuable insights and practical solutions to address the challenges associated with resource limitations and network latency, contributing to the advancement of edge computing in the eHealth domain.

The rest of the paper is organized as follows. The next section surveys existing work on resource-aware FL and FL-inspired resource profiling. Section 3 provides a classification of the edge resources while Section 4 introduces the proposed resource-aware profiling model with a detailed formulation of the edge workload profile and node selection. Section 5 discusses the challenges encountered in the implementation of the proposed model and presents feasible solutions. In Section 6, the model is evaluated using the fetal health monitoring dataset distributed over various edge nodes. Section 7, ‘Future Directions’, elaborates on ongoing efforts to enhance FL systems, focusing on various advancements. Finally, Section 8 concludes the paper by summarizing the key findings and contributions of the study.

2. Related Work

Although some studies highlight resource-aware FL, very few have explored FL-enabled resource profiling that aids in node selection for optimizing the overall performance of the federated learning context. The following subsections review existing works that considered the resource characteristics of FL, FL-inspired resource profiling, and FL-driven client selection.
(A) **Resource-aware Federated Learning**

Recently, several research works related to FL have focused on developing algorithms for resource optimization based on one or more criteria, including cost, time, energy consumption, CPU power, and memory, using different algorithms. In [11], the authors developed a deep reinforcement learning algorithm using an optimization function for minimizing the cost based on training time and the consumption of energy. The objective was to increase the speed of training while conserving the energy consumption of mobile devices by controlling the CPU cycle frequency. Furthermore, the authors of [12] proposed an optimization framework based on reinforcement Q-Learning. Their goal was to minimize the communication rounds and maximize the accuracy. A hierarchical game framework was proposed to allow the dynamic edge association and resource allocation in self-organizing Hierarchical FL networks [13]. The authors of [14] introduced the q-Fair optimization objective to emphasize a fair distribution among edges in FL according to performance which is based on the model’s accuracy. To minimize the latency of FL training, the authors of [15] proposed a Multi-Armed Bandit algorithm based on training latency, availability, and fairness constraints.

A notable contribution in resource allocation and optimization in dynamically evolving network environments is proposed by Huang et al. [16]. Their research investigates the complexities of resource allocation in FL systems assisted by intelligent 104 reflecting surfaces (IRS), particularly under the constraints of imperfect channel state information (CSI). By employing advanced optimization techniques like semi-definite relaxation (SDR) and the constrained concave convex procedure (CCCP), they present a method to significantly reduce training time in these systems ultimately reducing the resource allocation challenges.

A framework proposed in [17] describes a resource and data-aware FL approach that characterized the agents/resources according to certain criteria. Initially, it uses the local data size to classify the resources. Other criteria can also be used for resource selection to more accurately classify the resources used in a priority-based decision-making model such as Multiple-Criteria Decision Making Model (MCDM). Experiments showed improvements in the accuracy with less training time over the standard FL baseline. The authors of [18] studied the effects of resource and dataset heterogeneity on the training time of FL deployed on AWS EC2. Their experimentation implied a severe impact of resource heterogeneity, in terms of response time and data quality, and the training time of FL.

(B) **Federated Learning inspired resource profiling**

Few resource-aware FL frameworks have adopted resource profiling to improve the performance of the FL model. For example, EIFISH is a resource-aware FL framework that adopted resource profiling according to the edge computation capabilities to attain synchronized collaboration [19]. Experimentation showed an improved training accuracy of up to 4% and double the speed over conventional FL schemes. The authors of [20] proposed a Tier-based FL system that classified the clients adaptively into groups according to their performance, then selected only the nodes within the same group to avoid straggler problems and delays caused by the slowest devices resulting from resource heterogeneity. The results from their experiments showed that the proposed approach improved the training speed while keeping the accuracy the same or slightly increased. Furthermore, the authors of [21] adopted a low-complexity profile-based resource-aware allocation strategy in FL called Dispersed Federated Learning (DFL). They implemented an optimization problem for resource allocation and device association by defining the preference and resource profiles. Preference profiles were used to allocate resource blocks to devices having a lower cost. Alternatively, the authors of [22] proposed a FL time minimization problem to control the data size used in each device using the makespan minimization problem and assigned identical tasks to
heterogeneous resources. They adopted an optimal polynomial-time scheduling algorithm called OLAR.

**C. Federated Learning-driven edge node selection**

In FL, the number of clients may be large, but the bandwidth available for model distribution and re-uploading is limited, making it advisable to only involve only a few of the edge nodes in the training process at any given time. This is a challenge that must be overcome to ensure that the model training process runs smoothly and without delay. With regards to the training efficiency, the quality of the final model, and its fairness, a client selection policy is crucial to a FL process. In [23], the authors proposed a model wherein the Lyapunov optimization problem of selecting clients with the highest degree of fairness was handled. Although FL has the potential to safeguard privacy in distributed machine learning, its limited applicability is due to inefficiencies in its communication [24]. With hierarchical FL (HFL), workers who own the data first send their updated model parameters to the edge servers for intermediate aggregation before sending everything to the central server for global aggregation. For the effective HFL deployment, it is necessary to address the issues of edge association and resource allocation for non-cooperative parties such as edge servers and model owners. Using the edge server’s predicted bandwidth allocation management approach in self-organizing HFL networks, a hierarchical game framework [13] is proposed to analyze the dynamics of edge association and resource allocation, while collecting more local data from the clients enables more precise global model building; the resulting heavy data arrivals at the edge could be counterproductive to queue stability. The authors of [25] proposed an algorithm that varied the number of clients chosen based on their current resource usage for transmission in order to maximize the time-averaged FL accuracy while maintaining the queue’s stability.

In a traditional wireless network, an FL setup involves clients sharing a wireless connection to a central server where the federated models are trained. Due to the fact that both radio energy and the client’s energy are limited [26], the learning performance is dependent on how the clients are selected and how the bandwidth is distributed among the selected clients in each learning round. To formulate a stochastic optimization problem for joint client selection and bandwidth allocation under long-term client energy constraints, the article [27] developed a new algorithm that used only the currently available wireless channel information but still guaranteed long-term performance.

**D. QoS and QoE in Federated Learning Systems**

QoS and QoE are fundamental to the effectiveness of FL systems, especially in edge computing environments and is particularly crucial in sectors like healthcare [28], where the precision and speed of data processing are essential. QoS, as defined by ITU-T E.800, encompasses critical service characteristics like latency and bandwidth that directly impact the performance of FL systems [29]. QoE, on the other hand, focuses on the user’s perception and satisfaction with the service, as outlined by ITU-T P.10 and Qualinet [30]. Recent studies emphasize the importance of QoS and QoE in heterogeneous networks. Poryazov et al. [31] address QoS estimation challenges, proposing a normalization approach for telecommunication systems. Huang et al. [16] explore resource allocation in FL systems, contributing to the understanding of QoS under imperfect channel conditions. These insights are vital for optimizing both QoS and QoE in FL systems.

Ensuring high QoS and QoE in FL systems, particularly in data-intensive applications like healthcare, involves addressing challenges such as resource heterogeneity and network variability. Adhering to standards like ITU-T Supp. 9 of the E.800 Series [30] ensures consistency and interoperability in different computing scenarios. Optimization of QoS and QoE in FL requires adaptive strategies that consider the dynamic nature of edge computing. Simulation tools like iFogSim and FogNet-
Sim++ are instrumental in modeling network characteristics and resource utilization, facilitating the development of strategies that enhance both QoS and QoE [32]. The integration of QoS and QoE considerations is crucial for the development of efficient, reliable, and user-centric FL systems in edge computing. By focusing on these quality metrics and leveraging advanced simulation tools, FL systems can be optimized to meet diverse user needs and network conditions, enhancing the overall effectiveness of edge computing services.

There is a gap in the literature concerning the significance of measuring performance metrics in optimizing edge performance, as well as the appropriate classification of edge resources and the relevant metrics. To the best of our knowledge, previous works on FL resource profiling have not combined both edge static and dynamic resource quality profile attributes with edge reputation information to ensure optimal edge node selection. With this in mind, we proposed a hybrid resource-aware FPSN solution that integrated multi-criteria resource quality evaluations and edge's reputation to decrease the response times and improve the prediction performance.

3. Edge Resources Classification

Efficient optimization of the edge resources can enable real-time analytics and business intelligence, which can be used to gain a competitive advantage. One of the first steps in creating adaptive resource-management systems is determining the various application-specific performance characteristics and metrics of edge-based resources [33,34]. It is challenging to gather different performance metrics in a timely and accurate manner due to the abundance of deployed hardware configurations with various architectures and versions [35].

Resource profiling is the first step to determine how resources are currently being used for the best management of the hardware, and it is necessary to identify efficient methods for controlling hardware and obtaining the best performance out of it. Hence, prior to resource profiling, a selection of the edge resources and their performance characteristics are identified, as depicted in Figure 1 and Table 1, containing the respective description and metrics. The psutil (process and system utilities) Python library [36] provides access to system and process information on any supported platform such as CPU, memory, disks, network, sensors, etc. The categorization is largely based on such utility functions that can monitor edge nodes to help create the resource profile, control access to system resources, and manage processes in progress.

![Figure 1. Edge resource metrics classification.](image-url)
Table 1. Edge resource metrics.

<table>
<thead>
<tr>
<th>Edge Resource</th>
<th>Metrics</th>
<th>Description</th>
<th>Measurement Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>Processing Speed</td>
<td>The number of instructions per second the computer executes.</td>
<td>MHz</td>
</tr>
<tr>
<td></td>
<td>Utilization</td>
<td>Current system-wide CPU utilization.</td>
<td>Percentage</td>
</tr>
<tr>
<td></td>
<td>Frequency</td>
<td>Include current, minimum and maximum CPU frequencies.</td>
<td>MHz</td>
</tr>
<tr>
<td></td>
<td>Counts</td>
<td>The system’s logical CPU count.</td>
<td>Number</td>
</tr>
<tr>
<td><strong>Process</strong></td>
<td>Creation Time</td>
<td>The process’s creation time, since the epoch.</td>
<td>s</td>
</tr>
<tr>
<td></td>
<td>Execution Time</td>
<td>The time spent by the system executing the task.</td>
<td>s</td>
</tr>
<tr>
<td></td>
<td>Timeout</td>
<td>The wait for a process’s PID to terminate.</td>
<td>s</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>Available</td>
<td>The amount of RAM that can be made available to programs without forcing the operating system to swap.</td>
<td>bytes</td>
</tr>
<tr>
<td></td>
<td>Cached</td>
<td>Cache memory for numerous purposes.</td>
<td>bytes</td>
</tr>
<tr>
<td></td>
<td>Usage</td>
<td>The amount of memory used.</td>
<td>bytes</td>
</tr>
<tr>
<td></td>
<td>Size</td>
<td>Total physical memory.</td>
<td>bytes</td>
</tr>
<tr>
<td></td>
<td>Shared</td>
<td>The memory that many processes may concurrently access.</td>
<td>bytes</td>
</tr>
<tr>
<td><strong>Network</strong></td>
<td>I/O</td>
<td>System-wide network I/O statistics that includes: bytes sent</td>
<td>bytes received</td>
</tr>
<tr>
<td></td>
<td>Latency/Delay</td>
<td>The amount of time it takes to transfer data from one destination to another.</td>
<td>s</td>
</tr>
<tr>
<td></td>
<td>Throughput</td>
<td>The number of data packets successfully sent to the destination, which varies depending on the network area. Dropped packets must be retransmitted if throughput is low.</td>
<td>bits per second (bps)</td>
</tr>
<tr>
<td></td>
<td>Bandwidth</td>
<td>Maximum data transfer rate or capacity of a given network for understanding network speed and quality.</td>
<td>bits per second (bps)</td>
</tr>
<tr>
<td><strong>Sensors</strong></td>
<td>Battery Level</td>
<td>Battery power left.</td>
<td>Percentage</td>
</tr>
<tr>
<td></td>
<td>Battery Secs Left</td>
<td>Approximate seconds left before the battery runs out of power.</td>
<td>s</td>
</tr>
<tr>
<td></td>
<td>Battery power plugged</td>
<td>The AC power cord is plugged in.</td>
<td>Boolean</td>
</tr>
<tr>
<td></td>
<td>Fans</td>
<td>Hardware fan speed.</td>
<td>RPM</td>
</tr>
<tr>
<td></td>
<td>Temperature</td>
<td>The hardware’s current temperature.</td>
<td>Celsius</td>
</tr>
<tr>
<td><strong>Storage</strong></td>
<td>Disk Space</td>
<td>Total disk space.</td>
<td>bytes</td>
</tr>
<tr>
<td></td>
<td>Disk Usage</td>
<td>Disk space used.</td>
<td>bytes</td>
</tr>
</tbody>
</table>

In addition to the quantitative performance qualities indicated in Table 1, the following intangible elements can also improve edge performance.

**Agility:** This involves preparing for unknown workloads and applications, reconfiguring component systems, and having edge management tools that adapt. It relates to the velocity component of big data and the capacity to process continuously flowing streams in a timely manner [37].

**Total Cost of Ownership:** Edge providers must weigh the costs and benefits of using industry-standard hardware, storage, support, and other services, as the redundant infrastructures required to support cloud computing have a significant impact on the TCO [38]. Detaching storage components from other parts of the system may help with scalability.

**Serviceability:** Edge locations can be difficult to service due to distance, cost, and the volume of sites to service. The serviceability of a network is defined as its capacity to meet the needs of connected end devices (e.g., throughput, delay, and packet loss) [39]. In the 5G...
era, high serviceability at the edge of the network is seen as a crucial foundational criterion for a successful Internet of Things paradigm. A software-defined implementation with high availability and durability of hardware and storage media is recommended.

Security and Privacy: Content perception, real-time processing, and parallel processing are some of the ways in which edge computing differs from more traditional forms of computing, and this difference has led to new challenges in the area of data security and privacy protection [40]. There is a large attack surface on edge resources, so security countermeasures need to be effective and efficient. DoS attacks, penetration techniques, and physical break-ins are all part of this category. Since manual patches can have an impact on performance, automated remote patching is an absolute necessity from a security perspective.

4. Resource-Aware Federated Profiling Model

In this section, an overview of our resource-aware federated profiling model, the formulation of the edge workload profiling and an explanatory example of the edge workload profile is provided. In order to make the most effective use of the distributed computing resources at the edges, the resource-aware federated profiling framework was established.

(A) Model Overview

The Federated Workload Profiling model proposed in this section addresses resource planning. Planning of the edge workload is formulated as a resource optimization problem that eliminates the nodes that do not meet the computing demands. The model as shown in Figure 2, is a hybrid resource profiling model that combines both static and dynamic profiling. Static profiling mainly depends on the physical characteristics of the edge resource’s, such as the CPU’s computation power, memory size, and transmission power. On the other hand, dynamic profiling relies on the collected real-time available resources and the performance metrics at the edge. The real-time available resources include but are not limited to CPU utilization, memory usage, and network delays. However, the edge’s performance metrics are based on data training performance, such as accuracy and convergence time of the model.

Figure 2. The Federated Workload Profiling model.
Initially, a baseline resource profile is created based on the static profiling and is collected from all edges. An Edge Workload Profile (EWP) is created based on the configuration file available at the edge and a real-time measurement of the parameters including the task, the incoming data, the outgoing data, the infrastructure (CPU/GPU, storage), the mobility of the edge device, the round-trip latency based on the current location and the reputation of the edge. The Edge Workload Profiling Module (EWPM), which is responsible for continuously updating the EWP, receives information about the dynamic characteristics of the edge resources from the Node Resource Monitoring Agent (NRMA) at the edge node. Each client creates their own edge workload profile, which is then sent to the federation server, where the Federated Resource Profile is built, in accordance with the federation’s rules. The EWP’s resource parameters and the edge node’s reputation that sustain prior edge performance, both are taken into consideration by the federation’s rules. The Federated Workload Profiling (FWP) module aggregates EWP’s from all nodes and provides inputs to the Node Selection Module (NSM) that selects the edges depending on the thresholds of the resource parameters that have been stated in advance. The thresholds for CPU, memory, network, sensors, and storage metrics for the NSM depends on the system’s requirements and resources, but a general guideline is provided in Table 2. It is important to note that these thresholds are not set rigidly and may need to be adjusted based on the specific requirements of the system and the workload being processed. The effect of different thresholds may vary depending on the number of edges and their individual resource capabilities. For example, if the threshold for network usage is set too low, it may lead to slower communication and longer training times, particularly if there are a large number of edges. On the other hand, if the threshold is set too high, it may lead to network congestion and communication failures.

Additionally, the thresholds can be set dynamically based on the workload and resource utilization patterns over time. Resource profiling will initially rely on expert-set thresholds, using their deep understanding of system capabilities and expected workloads. Over time, as the system accumulates and analyzes historical data on resource usage, these thresholds can be refined to more accurately reflect the real-time demands and efficiency requirements. This evolving process allows the system to dynamically adapt its resource allocation, enhancing performance and resource management based on actual usage patterns. For instance, in a cloud-based service use case, the system initially operates with expert-set CPU and network bandwidth thresholds. However, it observes that during peak business hours, CPU usage consistently exceeds these thresholds, leading to performance bottlenecks. By dynamically adjusting the CPU threshold based on this historical data, the system can allocate more processing power during these peak hours, thereby reducing latency and enhancing user experience. This practical utility demonstrates how adaptive resource profiling can respond to fluctuating workload demands, ensuring optimal resource utilization and system performance.

Table 2. Resources and their suggested thresholds.

<table>
<thead>
<tr>
<th>Edge Resource</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>80%</td>
</tr>
<tr>
<td>Memory</td>
<td>70%</td>
</tr>
<tr>
<td>Network</td>
<td>90%</td>
</tr>
<tr>
<td>Sensors</td>
<td>95%</td>
</tr>
<tr>
<td>Storage</td>
<td>80%</td>
</tr>
</tbody>
</table>

The client edge is discarded if the edge resources are limited. For the duration of the data streaming process, iterations are performed, and the profile is incrementally
updated. To minimize the overhead, only the profile updates are transmitted following the initial run and at every federated aggregation event. As time is often a deciding factor in PSN deployments, the objective of this process was to select the nodes with the fastest response times by facilitating the faster convergence of the ML algorithms with resource optimizations.

(B) Workload profiling model formulation

The EWP is executed at the edge, to determine if the edge satisfies the federation’s profiling requirements. The static EWP is based on $C_e$, which is the maximum computation capacity of the edge node $e$; $\beta_e$ which is the computation cost of profiling; $\mu_e$ which is the effective memory at the edge; and $S_e$, which is the data size available at the edge. However, the dynamic EWP relies on resource usage monitoring process that is responsible for collecting the edge’s real-time status in terms of resource utilization and performance. This process collects the CPU usage $C_u$, the effective spare memory $\mu_u$, and the battery level $E_e$. Furthermore, this process will initiate a training model on the existing data and collect the accuracy results to measure the performance $P_e$, as well as measure the training time $T_{train}$. The results are sent back to the server for profiling.

The server, specifically the FWP module collects the aforementioned parameters as well as measuring the transmission cost of the EWP message from each node $e$, denoted as $EWP_e$. Let $T_{delay}$ be the tolerable delay, and $EWP_{Te}$ be the transmission cost of $EWP_e$. The model considers the reputation of the edge as part of profiling evaluation. The reputation of an edge $e$ is determined by its mobility status and its historic performance in terms of the number of previous successful model training requests using the following formula:

$$\text{Reputation}_e = \sum \frac{\text{No of successful Training of } e}{\text{Total No of Requests to } e} - \begin{cases} 1, & \text{if Mobile Node} \\ 0, & \text{if Stationary Node} \end{cases} \quad (1)$$

The Data Flow Rate $DFR_e$ is expressed as a function of the effective distance between the edge node $i$ and the centralized cloud to the average packet forwarding rate. The FWP module at the server will generate an aggregated profile for each edge $EWP_e$ using a utility function, which calculates the weighted average of all resources’ normalized quality attributes, including the static and dynamic characteristics as follows:

$$EWP_e = \begin{cases} U(C_e, \beta_e, S_e, \mu_e, DFR_e, \text{Reputation}_e), \text{Static Profile} \\ U(C_u, E_e, P_e, \mu_u, T_{train}, DFR_e, \text{Reputation}_e), \text{Dynamic Profile} \end{cases} \quad (2)$$

s.t. $EWP_{Te} < T_{delay} \quad (3)$

This utility function is used for MCDM modeling problem, where the edge quality profile attributes are normalized to be in the same interval scale and prioritized according to the weights given by domain knowledge experts depending on the application type and its requirements.

To further explain the problem, the MCDM model is formalized as follows: Let $A$ be a set of edge quality profile attributes, where $A = \{C_e, \beta_e, S_e, \mu_e, DFR_e, \text{Reputation}_e\}$ in the case of static profiling, and $A = \{C_u, E_e, P_e, \mu_u, T_{train}, DFR_e, \text{Reputation}_e\}$ when dynamic profiling is considered. Let $n$ be the number of quality attributes used for resource profiling, and $A' = \{a'_1, a'_2, \ldots, a'_n\}$ is the set of normalized values measured for each of the quality attributes included in $A$. Let $W = \{w_1, w_2, \ldots, w_n\}$ be the set of weights given to
each of the corresponding quality attribute in \( A \) to afford flexible prioritization for each attribute according to the applications requirements and preferences.

\[
EWP_e = \sum_{i=1}^{n} w_i \times a'_i, \text{ where } \sum_{i=1}^{n} w_i = 1
\]  

(4)

If the status of the edge node given by \( EWP_e \) satisfies the federated profiling requirement and the time constraint is satisfied, the edge is selected. The \( \text{Delay}_e \) factor should be kept to a minimum for time-sensitive applications.

(C) Example of an Edge Workload Profile (EWP)

A sample of the Edge Workload Profile which captures the static and dynamic characteristics of an edge node is presented in Figure 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<edge id="k119" name="Node1">
    <static>
        <os>Windows 10 Enterprise</os>
        <processor>Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz</processor>
        <storage unit="GB" free="35" used="439"/>
        <memory unit="MB" free="3769057280" used="13100105728"
            percent="77.7" available="3769057280"
            totalMem="16869163008"/>
        <dataset unit="MB" path="/dataset/ehealth/" size="86083"/>
        <cpu_count>12</cpu_count>
        <profilingCost>0.09</profilingCost>
    </static>
    <dynamic>
        <netio dropout="0" dropin="0" errout="0" errin="0"
            packets_recv="784985" packets_sent="331273"
            bytes_recv="1113647731" bytes_sent="132692441"/>
        <cpu_utilization percent="4.4" interval="2"/>
        <cpu_utilizationAll>{1.6, 3.1, 8.7, 1.6, 9.4, 5.5, 10.1, 4.7, 9.4, 12.5, 4.7, 2.4}</cpu_utilizationAll>
        <Battery Timeleft="1:58:32" PowerPuggedIn="False" percentage="57"/>
        <NoOfSuccessfulTraining>5</NoOfSuccessfulTraining>
        <TotalRequests>8</TotalRequests>
    </dynamic>
</edge>
```

Figure 3. Edge Workload Profile.

Dynamic parameters, such as CPU utilization, network statistics, and battery information are collected by the EWPM’s resource monitoring agent; static parameters are kept up to date via the nodes’ resource configuration files; and both sets of data are used to optimize the workloads. For the purpose of determining a node’s reputation, the EWP keeps track of the total number of training requests as well as the number of requests that have been successfully completed. The FWP module at the federation server further parses the EWP received from all the edges.

5. Model Challenges and Feasible Solutions

(A) Security and Privacy Challenges

The proposed model is concerned with maximizing resource usage and selecting the optimal edge nodes, but it also presents possible privacy concerns related to sharing sensitive information about the resources of the edge nodes. To address these concerns, the proposed model must implement robust security measures to protect the sensitive resource information from unauthorized access or misuse. For
example, the model could use encryption techniques to protect the data during transmission and storage. Access controls can also be implemented to restrict access to the resource’s information to authorized parties only. Additionally, the model must be transparent about its data handling and sharing practices, providing clear and concise privacy policies and obtaining informed consent from users before collecting and sharing their data.

The FWP model’s design incorporates a reputation-based system that assigns a reputation score to each node based on its past behavior. Nodes with a low reputation score are then excluded from the federated learning model.

Another approach is to use techniques such as differential privacy or, more specifically FL enabled personalized differential privacy [41] to protect the privacy of the data, even in the presence of malicious nodes. Differential privacy adds noise to the data before sharing it with the nodes, which makes it difficult for malicious nodes to identify the data of a specific node. Other approaches include blockchain-based [42,43] privacy information security sharing schemes.

Overall, it is important to balance the benefits of the proposed model’s resource optimization with the potential privacy risks, ensuring that adequate measures are in place to protect sensitive data and respect the users’ privacy rights. This not only applies to FL, but also to any distributed or collaborative approaches that require data sharing and processing at the edge nodes.

(B) Communication Overhead

If the edge nodes continuously update their resource information to an Edge Workflow Platform (EWP), it may incur additional communication costs. However, this can be addressed by optimizing the communication protocol and minimizing the frequency of updates. For example, the edge nodes can send updates only when there is a significant change in their resource availability or utilization.

There is a risk that an edge node may provide incorrect or misleading information about its resources. This can happen due to various reasons such as hardware or software failures, malicious attacks, or incorrect configurations.

There are several approaches that can be taken in such situations but not limited to:

**Resource verification:** The EWP should verify and validate the resources before accepting the resource information from an edge node.

**Data encryption:** To prevent malicious attacks on the data, edge encrypts its data before sending them to the server. The server can then decrypt the data and verify its authenticity. It is important to note that in the current context, the encryption is intended for resource profile data, not health datasets, and is focused on protecting personally identifiable information or other sensitive details related to the resources.

**Model aggregation:** The server can use a weighted averaging technique to aggregate the updates from all the participating edges. This method assigns more weight to information received from nodes with a better reputation and less weight to updates from edges that have previously sent erroneous data.

**Node selection:** The server will dynamically select edge nodes based on their performance history. Accordingly, nodes with a history of sending unreliable data can be excluded from the training process.

(C) Computational Overhead

Given that we are only processing the quality profile and not the entire dataset, the computational overhead remains constant regardless of the size of the data. However, profile federation overhead grows in proportion to the number of nodes. Quality profile federation is the process of aggregating multiple quality profiles into a single federated profile. The EWP contains a set of resource attributes, their tolerance values, and rules for resource optimization. The computational overhead of the FWP depends on various factors, such as the number of profiles being aggregated, the size of the profiles, and the complexity of the rules.
The computational overhead can be formulated as:

\[
\text{Computational Overhead} = \text{Number of Nodes} \times (\text{TO} + \text{PO} + \text{EO} + \text{RO} + \text{FO})
\]  

(5)

where the Transfer Overhead (TO) is the computational cost of transferring the profile data between the nodes and the server, including serialization, deserialization, and network communication. It is directly proportional to the size of the profile. The Parsing Overhead (PO) is the computational cost of parsing the input data, including XML parsing and other data transformations. The Extraction Overhead (EO) is the computational cost of extracting the relevant information from the input data, such as the DQ attributes or other features. The Rules’ Firing overhead (RO) is the computational cost of applying the DQ rules to the extracted data, including any computations or comparisons required by the rules. The Federation Overhead (FO) is the computational cost of aggregating the results of the quality rules across multiple nodes, including any additional computations or transformations required to produce the final output.

Strategies that can help with reducing the overheads include:
1. For massive XML documents, use a parser that is both lightweight and fast.
2. Save time and effort by caching parsed data.
3. Use effective algorithms and data structures to optimize the processing and aggregation of the profiles.
4. Use multithreading for parallel computation.

Note that the specific values of each overhead component will depend on the size and complexity of the input data, the number of nodes involved, and the specific details of implementing the approach. Overall, the computational overhead of EWP aggregation can be significant, when the number of nodes is significantly high, but the benefits of having a single, comprehensive Federated Workload Profile (FWP) can outweigh the costs.

6. Experimental Evaluation

This section details the experiments devised to evaluate the proposed federated resource profiling model. In the proposed approach, quality profile federation plays a crucial role, and involves aggregating multiple quality profiles, which represent the resource attributes, tolerance values, and rules for resource optimization, into a single federated profile. This federated profile provides a comprehensive view of the system’s resources and serves as a basis for making informed decisions on resource allocation and workload management. By leveraging the federated profile and employing FWP techniques, the experiment’s objective was to evaluate the FWP in terms of memory, disk space, execution time, and other resource utilization parameters at the edge nodes like network statistics and the effects of ML algorithms on tree depth, and to demonstrate that it can aid with edge node selection and FPSN. The following subsections describe the experimentation including the environment setup, the dataset used, the experimental design and scenarios. Finally, the attained results are discussed.

6.1. Dataset

The data consist of 2126 Cardiotocogram instances and 23 attributes [44,45]. These are continuous measurements of the fetal heart rate using an ultrasound transducer on the mother’s abdomen and categorized by expert obstetricians. The Instantaneous fetal heart rate (FHR) and the simultaneously communicated uterine contraction signals are the data analysis parameters. The classification results were based on fetal state labels (N = normal; S = suspect; P = pathologic).
6.2. Experiment Setup
The entire project was carried out in the open-source scientific environment Spyder IDE [46], which is a Python-based native application. The entire Fetal Health dataset is randomly split and distributed into six datasets, corresponding to the six edge nodes, for the experimental configuration. The approach divides the dataset into subsets in the current context solely for the purpose of assigning each subset to an edge node and evaluating the associated resource profiles, with no FL involved.

6.3. Scenarios
To evaluate the proposed FWP model, four experimental scenarios were implemented. By conducting these experiments, we aimed to gain insights into the capabilities and limitations of the FWP model and to validate resource profiling optimization under different scenarios.

6.3.1. Scenario 1: Federated Workload Profiling on Memory Usage
1. **Best performing ML models with respect to memory usage.**
   Initially, the effects of memory usage before and after FWP resource optimization techniques were examined using five different ML algorithms as shown in Figure 4.

![Memory usage Comparison with different ML Algorithms](memory_usage.png)

*Figure 4. ML algorithms memory usage comparison.*

After implementing the FWP resource optimization algorithms, it was found through experimentation that memory consumption dropped significantly. Additionally, it can be deduced that Random Forest Tree, followed by GBDT and Decision Tree, were the best-performing ML models. The memory usage of LightGBM was the highest of all the models evaluated. In conclusion, the experiments revealed that Random Forest Tree was the top-performing ML model in terms of memory consumption, and thus the Random Forest model was employed for the rest of the experiments.

2. **Memory usage prior to and after FWP at different edge nodes.**
   FWP optimization strategies were implemented across six diverse nodes as seen in Figure 5, revealing a significant drop in the memory requirements in comparison with conventional approaches. Although all the nodes benefited from FWP, Nodes 5, 6, and 2 saw the greatest reductions in the amount of memory they used, respectively.
6.3.2. Scenario 2: Federated Workload Profiling on Various Resource Parameters at the Node Level

1. **Disk space usage prior to and after FWP at different edge nodes.**

   The FWP experiments’ impact on disk space consumption was comparable with that of other previous experiments as evident from Figure 6. All of the nodes showed lower disk space usage after the FWP experiments, even though there was no large disparity in the usage of disk space.

![Disk Space Comparison Before and After FWP](image)

Figure 6. Impact of FWP on disk space.

2. **Network I/O statistics prior to and after FWP**

   Each edge node’s network I/O data were recorded, which included the characteristics bytes sent, bytes received, number of packets transmitted, number of packets received, total number of incoming dropped packets, and the total number of outbound lost packets. This I/O data analysis revealed crucial insights into the system’s overall performance and helped identify potential bottlenecks and issues. After applying our proposed optimization strategies, which included resource-aware data reduction methods such as compression, protocol optimization, and caching, we were able to visually
inspect the bytes sent and received and confirmed a decrease after implementing FWP, as shown in Figure 7.

Figure 7. Effect of FWP on network I/O.

6.3.3. Scenario 3: The Cumulative Impact of Federated Workload Profiling across Expanding Node Networks

Extending the Scenario 2 experiments, an assessment was conducted to evaluate the cumulative impacts of optimization strategies on disk space, memory, and network performance as the number of nodes in the federation increases. The size of a federation can vary widely, ranging from a few nodes to hundreds or even thousands, depending on the specific network or system architecture. The number of nodes in a federation often impacts various aspects of network management, performance, and resource allocation.

1. **FWP impact on cumulative memory with increasing nodes.**

   In this experiment, we explore the cumulative memory usage across ten nodes before and after implementing FWP and its impact as the number of nodes increases. Figure 8 illustrates the cumulative memory usage across ten nodes both before and after the implementation of FWP. The blue line represents the memory consumption before FWP, and it shows a steady increase as additional nodes are incorporated into the system. Conversely, the orange line, depicting memory usage after FWP, displays a similar upward trend but with consistently lower values compared to the pre-FWP scenario. This suggests that FWP effectively optimizes memory resources, resulting in reduced memory utilization as the number of nodes in the federation increases. The graph highlights the positive impact of FWP on memory management, showcasing its ability to efficiently scale memory consumption with the growing number of nodes, potentially leading to enhanced system performance and resource allocation.

2. **FWP effect on cumulative disk space with node expansion.**

   This experiment investigates the effects of FWP on cumulative disk space usage across an expanding node federation. Figure 9 provides a visual representation of cumulative disk space usage across ten nodes, both before and after the implementation of FWP. The cumulative disk space usage is presented in gigabytes (GB) and is measured as the sum of disk space occupied by each node in the federation. The blue line represents the cumulative disk space consumption before FWP, showcasing the gradual increase in storage usage as more nodes are incorporated into the network. Each point on the line corresponds to a specific node, illustrating the progressive accumulation of data storage requirements across the network. In contrast, the orange line represents cumulative disk space usage after FWP activation. There is a similar growth trend as more nodes join the network. The graph utilizes a dual-axis approach, with separate y-axes on the left and right sides, to
present cumulative disk space usage before and after the implementation of FWP. This separation prevents data from overlapping with the other, ensuring that both trends can be easily examined and compared.

Figure 8. Cumulative memory usage before and after FWP.

The graph suggests that the difference in cumulative disk space between before and after the implementation of FWP is generally negligible. In practical terms, this indicates that FWP has not significantly impacted the cumulative disk space usage across the ten nodes. Overall, the negligible difference in cumulative disk space suggests that FWP has moderately reduced but not substantially increased or decreased the total storage requirements of the system, at least within the observed scope of these ten nodes.

Figure 9. Cumulative disk space consumption before and after FWP.
3. **FWP's influence on network I/O with increasing federation nodes.**

In this experiment, we investigate the effects of FWP on network I/O statistics as the number of nodes within the federation expands.

Figure 10 offers a comprehensive perspective on network I/O (Input/Output) statistics for ten nodes, featuring four key components. The blue bars denote cumulative bytes sent before the introduction of Federated Workload Profiling (FWP), while the green bars represent cumulative bytes received before FWP. Similarly, the orange bars depict cumulative bytes sent after FWP, and the red bars showcase cumulative bytes received after FWP. Additionally, the graph incorporates purple bars, which represent the cumulative difference in bytes sent, and cyan bars that represent the cumulative difference in bytes received. The two y-axes in the graph serve distinct purposes: the primary y-axis represents cumulative network traffic volumes, while the secondary y-axis illustrates the cumulative differences in network activity before and after the implementation of Federated Workload Profiling (FWP). The cumulative difference bars on the secondary y-axis appear relatively higher, mainly because they are scaled using a factor to emphasize even subtle variations in network behavior. This comprehensive visualization aids in understanding the impact of FWP on network data flow across multiple nodes.

![Figure 10. Cumulative network I/O Stats before and after FWP.](image)

The graph demonstrates a notable network traffic pattern. Initially, there is a reduction in data bytes sent and received at Node 1 after implementing FWP. However, as more nodes join the federation, there is a consistent increase in network activity. Notably, the cumulative difference bars (purple and cyan) remain positive, highlighting FWP’s ability to effectively manage and optimize data traffic as the network scales. This trend suggests significant network traffic savings when FWP is employed with a larger number of nodes, enhancing network efficiency and resource utilization.

6.3.4. Scenario 4: Federated Resource Profiling on Tree Depth

1. **Faster convergence of ML algorithms.**

When compared with the conventional approaches, the execution time of each of the five different machine learning algorithms evaluated with the proposed FWP was found to be significantly reduced, as shown in Figure 11. According to the results, Decision Tree and Random Forest performed better than other ML algorithms and converged more quickly. Even though the worst-performing model with the longest execution time was GBDT, our proposed model, FWP, cut the execution time to almost half.
2. Memory utilization

Algorithms for machine learning, such as Random Forest, have memory requirements that vary with the number and size of the trees in the algorithm. When dealing with big or complex datasets, the tree might be quite deep and include thousands of nodes. In such circumstances, the memory usage will increase exponentially. A decrease in the depth of the tree will lead to a reduction in the amount of memory that is used.

In Figure 12, the graph provides us with a visual representation of how the Random Forest Tree Depth affected execution time when FWP is utilized. It is evident that this was the reason for the decreased utilization of resources following FDP implementation.

6.3.5. Scenario 5: Federated Workload Profiling Aiding in Edge Node Selection and FPSN

Depending on the application, the edge nodes can be mobile or static in edge computing. Mobility reduces the efficiency with which edges can exploit their resources, which can result in the FWP deciding on which edge nodes to be removed. To keep resource utilization and performance at their peak, the algorithm also takes into account the reputation of each node, keeping only the most trustworthy edge nodes even if they are mobile. The
response time; metric scores related to CPU, memory, network, sensors, processes, and storage; reputation score; mobility; etc., all factor into the weighted average score calculated at the server and used as a selection criterion for the EWP nodes. If two nodes have the same weighted average score, the server prioritizes the node with the highest reputation score, followed by the node with the lowest response time, the lowest mobility, and the best CPU and memory availability and selects the best n edges. Ultimately, the weight given to each of the factors depended on the specific requirements of the system and the trade-offs between aspects such as performance, scalability, and cost. It is important to continuously evaluate and refine the approach based on real-world performance to ensure optimal results.

Because of its poor response time, limited mobility, excessive congestion, and low reputation, Edge Node 6 is removed after FWP is applied. Since time is of the essence in FPSN calculation, only the most resource-rich nodes were selected. To reduce the response time and to get a precise performance measurement while maintaining high accuracy, only trustworthy nodes were selected to use in our FPSN calculation by factoring in their reputation. It is evident that the performance measure accuracy has slightly improved after resource optimizations following FWP, which was applied at the nodes and used in FPSN calculations as shown in Figure 13.

![FWP Optimization effect on FPSN Accuracy](image)

**Figure 13.** FWP enabled FPSN performance.

6.4. Results and Discussion

In this study, we presented four different scenarios that utilized Federated Workload Profiling (FWP) to optimize the resource utilization of edge computing nodes.

In Scenario 1, we evaluated the memory usage of five different machine learning models before and after applying FWP optimization techniques. On an average, applying FWP led to a 47.8% reduction in memory usage across all machine learning algorithms. The algorithms that benefited the most from FWP in terms of memory usage reduction were Random Forest Tree and Decision Tree, with reductions of 73.9% and 69.3%, respectively. XGBoost and LightGBM had the smallest reductions in memory usage, at 53% and 43.5%, respectively. We found that Random Forest Tree was the best-performing model in terms of memory consumption and used it for the rest of the experiments.

In Scenario 2, we analyzed the impact of FWP on disk space usage and network I/O statistics. Our proposed optimization strategies resulted in slightly lower disk space usage and decreased network I/O after implementing FWP. Specifically, the total disk space used decreased by 336,304 bytes. With the network I/O statistics analysis, we can see that the on an average the number of bytes sent by a node decreased by 2,623,071 bytes (or about 1.4%), while the number of bytes received by a node decreased by 40,266 bytes (or about 0.04%).

In Scenario 3, the cumulative impact of FWP across expanding node networks was examined. As the number of nodes in the federation increased, FWP demonstrated its effectiveness in optimizing memory resources, resulting in a notable reduction in memory
utilization of approximately 9%. Additionally, FWP showcased its ability to efficiently manage network traffic, leading to significant network traffic savings of around 27%. These findings underscore the scalability and positive impact of FWP in enhancing system performance and resource allocation as federated network nodes increase.

In Scenario 4, we focused on evaluating the impact of FWP on the convergence time and memory utilization of machine learning algorithms. Our results showed that the execution time of the evaluated algorithms was significantly reduced, with Decision Tree and Random Forest performing the best in terms of faster convergence. The execution time decreased after FWP was applied, with percentage decreases ranging from 19.2% to 62.3%. Specifically, the execution times for Random Forest, Decision Tree, GBDT, XGBoost, and LightGBM decreased by 19.2%, 33.3%, 53.9%, 62.3%, and 57.9%, respectively. For the Random Forest tree depth and training time before and after FWP, we found that the average tree depth decreased by 3.34, or about 27.5%, after FWP was applied which will in turn lead to a reduction in the amount of memory. Similarly, the average execution time decreased by 24.68 ms, or about 20.4%, after FWP was applied. These results suggest that FWP had a significant positive impact on both tree depth and training time for Random Forest across all nodes.

Finally, in Scenario 5, we utilized FWP to aid in edge node selection and FPSN accuracy calculation. We removed an underperforming edge node and only selected trustworthy nodes to use in our FPSN calculation. On an average, applying FWP led to a 6.8% increase in accuracy across all nodes. However, it is worth noting that the degree of improvement varied across different nodes, with Node 1 having the largest increase in accuracy (8%), and Node 4 having the smallest increase (4%).

Overall, the results of our experiments demonstrate the effectiveness of FWP in optimizing the resource utilization of edge computing nodes. The implementation of our proposed optimization strategies yielded substantial reductions in memory usage and modest improvements in disk space efficiency. Concurrently, these strategies enhanced network I/O performance and led to more efficient convergence times for machine learning algorithms. Furthermore, the FWP approach can be used to aid in edge node selection and improve the accuracy of FPSN.

7. Future Directions

As part of our ongoing efforts to enhance FL systems, we will focus on several key areas of development that align with the evolving challenges in the field. Our future research will concentrate on the following advancements:

**Integrating Statistical Heterogeneity and Optimizing Edge Reputation in FL:**

In our future work, we aim to incorporate statistical heterogeneity into our resource-aware federated profiling framework, drawing inspiration from methodologies in studies such as Oort [47] and PyramidFL [48]. This will enhance our model’s adaptability in environments with diverse data distributions across nodes. Simultaneously, we plan to optimize the edge reputation factor, which assesses the historical performance and contribution of nodes. This optimization will include integrating advanced utility profiling mechanisms that adapt to the changing performance and contributions of nodes over time, ensuring more effective and reliable node selection.

**Addressing Computational Stragglers in Federated Learning:**

We intend to tackle the challenge of computational stragglers, employing strategies such as AD-PSGD algorithm [49] and Prague [50], for efficient information exchange and robust asynchronous operations. AD-PSGD algorithm [49], aims to enhance the overall efficiency, particularly in dealing with nodes that exhibit slower or irregular computational performance. The Prague algorithm [50], a high-performance, heterogeneity-aware training method that enhances synchronization efficiency and reduces conflicts, using advanced techniques like Partial All-Reduce and static group scheduling make it an ideal model for optimizing federated learning systems and managing computational stragglers.

**Expanding Experimentation with Diverse Datasets:**
Our experiments are currently limited to a specific, homogeneous dataset, highlighting the need for future research to include more varied and heterogeneous data to comprehensively assess the proposed design’s effectiveness and scalability in diverse real-world scenarios. To validate and refine our approach, in the future we will employ a range of datasets in our experiments. This will enable us to assess the robustness and scalability of our design under different conditions and conduct comprehensive comparisons with state-of-the-art resource optimization strategies and systems.

Enhancing Fine-grained Client Selection:

We will also explore fine-grained client selection strategies to improve time-to-accuracy performance in FL systems. A key technique we plan to incorporate is client clustering based on data characteristics. This method involves grouping clients with similar data distributions or computational capabilities, allowing for more targeted and efficient model updates. This approach will focus on exploiting data and system heterogeneity within selected clients for more efficient utility profiling, leading to improved model training efficiency and effectiveness.

In addition to the areas mentioned, further research is critical to fully evaluate the efficacy and scalability of our system in real-world scenarios, particularly in addressing privacy and security concerns. As improving node collaboration and resource utilization remains a top priority, future developments will focus on enhancing the FWP to allow for resource offloading to resource-rich nodes. By doing so, we aim to maximize the efficiency of resource allocation across the network, ensuring optimal utilization of available resources while maintaining system integrity and security.

8. Conclusions

As a result of digital transformation, healthcare is becoming decentralized, and reaching out to the network’s edge will allow us to better capture data, meet the demands of patients, and foster innovation. The research highlights the importance of optimizing node collaboration and resource utilization, with a specific focus on resource profile federation and resource optimization evaluation. FWP combined with node reputation are two effective measures to ensure optimum edge node selection. In order to analyze and optimize the resources, and select appropriate edge nodes, edge workload profiling and Federated Workload Profiling are proposed to support the deployment of FPSN. Time is often a deciding factor in FPSN and PSN deployments; consequently, our proposed model was successful in selecting the nodes with the quickest response times by facilitating faster convergence of the ML algorithms through resource optimizations. The experimentation conducted on various scenarios validates the effectiveness and performance of the proposed FWP model, showcasing its potential for improving resource utilization in a distributed edge computing environment through reputation-based node selection and a resource-aware federated hybrid profiling strategy. Overall, the proposed model successfully selected edge nodes with quick response times, facilitating faster convergence of machine learning algorithms, and enabling more effective, time-critical, and decision-intensive health applications. Although the computational overhead may increase with an increasing number of nodes, our approach prioritizes a lightweight resource profile federation, ensuring that this potential increase does not pose a significant concern given our focus on efficient resource utilization.

Author Contributions: A.N.N. conceived the main conceptual ideas related to the resource aware FPSN, the architecture, the literature review, and the overall implementation and execution of the experiments. H.T.E.K. contributed to the formal modeling, the literature review, and the analysis of the results. M.A.S. contributed to the architecture of the model and the edge resources classification, and he ensured that the study was carried out with utmost care and attention to detail, while overseeing the overall direction and planning. E.S.B. was involved in the general evaluation of the proposed model. All authors contributed to the writing of the manuscript and the revision and proofreading of the final version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Zayed Health Science Center under fund # 12R005.
Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable.


Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations
The following abbreviations are used in this manuscript:

- CPU: Central Processing Unit
- EWP: Edge Workload Profiling
- EWPM: Edge Workload Profiling Module
- FL: Federated Learning
- FWP: Federated Workload Profiling
- FPSN: Federated Patient Similarity Network
- GBDT: Gradient Boosted Decision Trees
- GPU: Graphics Processing Unit
- HFL: Hierarchical Federated Learning
- I/O: Input/Output
- LightGBM: Light Gradient Boosting Machine
- ML: Machine Learning
- MCDM: Multiple-Criteria Decision Making Model
- NSM: Node Selection Module
- NRMA: Node Resource Monitoring Agent
- PSN: Patient Similarity Network
- QoS: Quality of Service
- RPM: Revolutions Per Minute

References
16. Huang, W.; Han, Z.; Zhao, L.; Xu, H.; Li, Z.; Wang, Z. Resource allocation for intelligent reflecting surfaces assisted federated learning system with imperfect CSI. Algorithms 2021, 14, 363. [CrossRef]
32. Ashouri, M.; Lorig, F.; Davidsson, P.; Spalazzese, R. Edge computing simulators for IoT system design: An analysis of qualities and metrics. Future Internet 2019, 11, 235. [CrossRef]
33. Aslanpour, M.S.; Gill, S.S.; Toosi, A.N. Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. Internet Things 2020, 12, 100273. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.