applied sciences

MDPI

Article

Two-Stage Optimal Task Scheduling for Smart Home Environment Using Fog Computing Infrastructures

Oshin Sharma [1], Geetanjali Rathee [2], Chaker Abdelaziz Kerrache [3,*] and Jorge Herrera-Tapia [4]

[1] Department of Computer Science and Engineering, SRMIST-NCR Campus, Modinagar 201204, India
[2] Department of Computer Science and Engineering, Netaji Subhas University of Technology, New Delhi 110078, India
[3] Laboratoire d'Informatique et de Mathématiques, Université Amar Telidji de Laghouat, Laghouat 03000, Algeria
[4] Faculty of Computer Science (FACCI), Universidad Laica Eloy Alfaro de Manabí, Manta 130212, Ecuador
* Correspondence: ch.kerrache@lagh-univ.dz

Abstract: The connection of many devices has brought new challenges with respect to the centralized architecture of cloud computing. The fog environment is suitable for many services and applications for which cloud computing does not support these well, such as: traffic light monitoring systems, healthcare monitoring systems, connected vehicles, smart cities, homes, and many others. Sending high-velocity data to the cloud leads to the congestion of the cloud infrastructure, which further leads to high latency and violations of the Quality-of-Service (QoS). Thus, delay-sensitive applications need to be processed at the edge of the network or near the end devices, rather than the cloud, in order to provide the guaranteed QoS related to the reduced latency, increased throughput, and high bandwidth. The aim of this paper was to propose a two-stage optimal task scheduling (2-ST) approach for the distribution of tasks executed within smart homes among several fog nodes. To effectively solve the task scheduling, this proposed approach uses a naïve-Bayes-based machine learning model for training in the first stage and optimization in the second stage using a hyperheuristic approach, which is a combination of both Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). In addition, the proposed mechanism was validated against various metrics such as energy consumption, latency time, and network usage.

Keywords: quality-of-service; optimization techniques; task scheduling mechanism; delay sensitivity; cloud computing

Citation: Sharma, O.; Rathee, G.; Kerrache, C.A.; Herrera-Tapia, J. Two-Stage Optimal Task Scheduling for Smart Home Environment Using Fog Computing Infrastructures. *Appl. Sci.* **2023**, *13*, 2939. https://doi.org/10.3390/app13052939

Academic Editor: Dimitris Mourtzis

Received: 1 February 2023
Revised: 22 February 2023
Accepted: 22 February 2023
Published: 24 February 2023

1. Introduction

The modern era of the Internet of Things (IoT) has made our world connected in a smarter way. Here, a thing can be a person with a wearable device, an animal with a biochip, or any device with a built-in sensor that can be associated with an IP address to transfer the data over a network. The IoT has become one of the most-important technologies of the 21st Century by which we can connect our home appliances, workstations, cars, baby monitors, and many other devices to the Internet and communicate with them. The IoT connects billions of devices, human beings, and computing elements in order to make decisions and act accordingly. The IoT has started a new industrial revolution worldwide known as Industry 4.0, which has changed the entire manufacturing industry. The emerging 5G technology is the key to Industry 4.0. The high speed and low latency of 5G is very effective for automation, healthcare, media, and entertainment. 5G will be responsible for critical communication in real-time, which means that data can be accessed reliably and securely in a wireless manner. The concept of the IoT has existed for a long time, but the advancement of technologies such as low-power sensor technology, cloud computing platforms, artificial intelligence, and machine learning has resulted in hands-on experience for the IoT. Cloud computing platforms for storing, processing, and analyzing the data

generated from a large number of connected devices became a great innovation for the IoT, but this has brought many challenges with respect to their centralized architecture [1,2]. Due to the huge demand of IoT devices and location-aware services, applications are generating amounts of data that are almost inconceivable [3]. It has been estimated that the cloud alone may not be sufficient to handle the big data generated by IoT devices with respect to time-critical or emergency situations. Thus, the concept of fog computing was proposed by Cisco in 2012 to cope up with these situations [4]. This emerging paradigm of fog computing lies between the cloud and end users (IoT devices) and aims to bring storage, networking, and computation close to the end users. Fog computing extends the cloud environment to the edge of the network. Although the cloud and fog use the same types of mechanisms (virtualization, resource provisioning, task scheduling, and many others) on the same kinds of resources (storage, computations, networking, etc.), still, there are few differences between them. The fog environment is suitable for many services and applications for which cloud computing does not support them well [4,5] such as: traffic light monitoring systems, healthcare monitoring systems, connected vehicles, smart cities, homes, and many others. Sending high-velocity data to the cloud leads to the congestion of the cloud infrastructure, which further leads to high latency and violations of the QoS.

Thus, these delay-sensitive applications need to be processed at the edge of the network or near the end devices, rather than the cloud, in order to provide the guaranteed QoS related to the reduced latency, increased throughput, and high bandwidth. Fog computing plays a very important role in supporting rapid decision-making during these time-critical applications [6]. This study proposes a machine-learning-based resource scheduling in a fog–cloud environment considering smart homes as a case study. Figure 1 shows the fog–cloud architecture for smart homes. The fog computing architecture is divided into three layers [7]: The end device (IoT) layer contains the end devices such as smart phones, computers, vehicles, smart equipment, wearables, and smart home appliances. The second layer is the fog layer (middle layer), which includes fog devices such as gateways, routers, switches, and workstations. Finally, the third layer, i.e., the cloud layer, which is also known as the upper layer, contains high-speed computers, servers, and data centers.

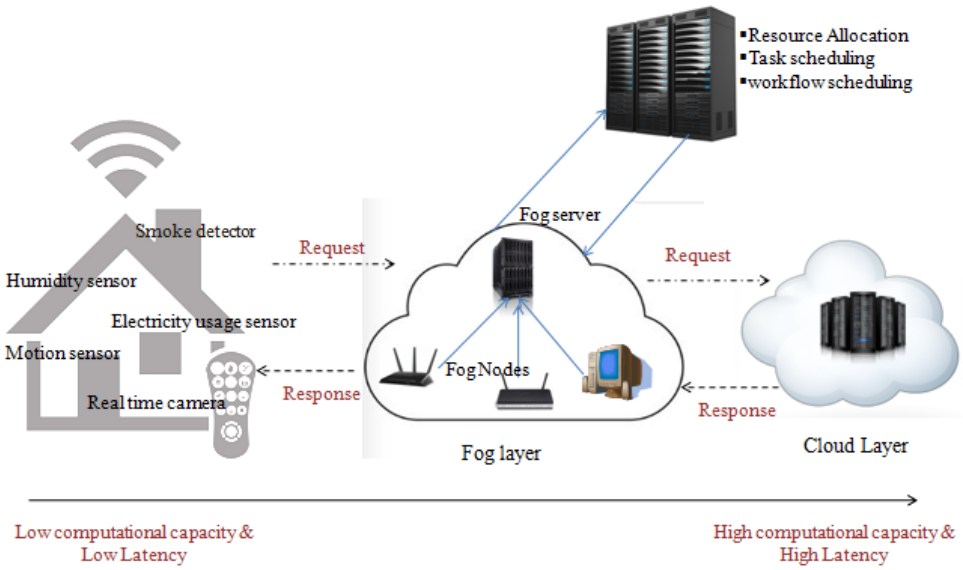

Figure 1. Fog–cloud architecture for smart homes.

1.1. Motivation

To improve the service and operation layers in fog computing, practical scheduling algorithms are required, which comprises one of the main challenges in the fog environment. Three major scheduling issues in fog computing are resource allocation, task scheduling, and workflow scheduling [8,9]. Firstly, resource allocation deals with the optimal allocation

of tasks $(T1, T2, \ldots, Tn)$ over fog nodes $(FN1, FN2, \ldots, FNn)$ to improve resource utilization, provide a faster response time, and reduce the energy consumption and overall cost. Secondly, task scheduling helps with the optimal resource allocation by assigning tasks to different resources considering the task's input time, wait time, deadlines, and the cost of performing the task as important parameters. Finally, workflow scheduling, which is also known as task-dependent scheduling, is based on the distribution of tasks on heterogeneous fog nodes. In order to identify an efficient technique for resource management and task scheduling, many researchers have built new algorithms. The quest to improve the efficiency of task scheduling algorithms to minimize energy consumption has led to a novel technique named the two-stage optimal task scheduling technique (2-ST), which could be one of the most-promising techniques for the IoT industry and its related applications. The IoT environment supports various real-time applications for time-sensitive data such as healthcare, smart cities, smart homes, smart traffic management, and many others, and it further deals with a large number of computing nodes or devices. However, due to the long response time taken by computing devices with their integration with the cloud environment, this technique provides the integration of fog computing and smart homes in the IoT environment. The main aim of 2-ST in smart homes is to understand the topology of the smart homes, provide optimal task allocation with less response time, make decisions, and act accordingly.

1.2. Contributions

The main contributions of this study are as follows:

- The objective of the proposed machine-learning-based task allocation is to provide the optimal allocation of fog nodes, improve network utilization, reduce the latency time, and improve energy consumption.
- Delay-sensitive tasks such as smoke detectors and water leaks need to be prioritized in the service queue, which would then be allocated first in the fog nodes.
- The naïve Bayes algorithm for task allocation and a hybrid algorithm for task optimization were designed.

The remainder of this paper is organized as follows. Section 2 provides the recent research and background of resource allocation and task and workflow scheduling. The system architecture and methodology are discussed in Section 3. Section 4 describes the evaluation metrics and performance parameters used in this study then discusses the results and analyzes the proposed methodology, and finally, section 5 concludes the paper along with the future scope and research directions.

2. Related Work

This section discusses a few related survey papers with respect to the scheduling approaches in fog computing. The main aim of the scheduling approaches is to find the optimal set of solutions with respect to the tasks, resources, and workflows on a set of machines without compromising the desired QoS. Three main scheduling approaches are: resource allocation, task scheduling, and workflow scheduling. The background literature with respect to these approaches is discussed below [10,11]. In order to execute the tasks and their corresponding jobs, optimal node allocation needs to be performed to satisfy the resource requirements. Furthermore, an optimal and efficient execution of the tasks can increase the energy efficiency, reduce the cost, and increase the accuracy, as well as the resource utilization such as the memory, bandwidth, processor use, and many others. Although there are many challenges in these scheduling approaches, different studies have proposed efficient scheduling algorithms to increase the resource utilization and optimize the performance metrics.

The most-recent survey in task scheduling by Islam et al. [12] presented a detailed survey considering different simulation tools, case studies, and optimization metrics based on context-aware scheduling approaches. Judy C. et al. [13] discussed the static approach of workflow algorithms in order to reduce the task's execution time and the total makespan

cost of the fog–cloud system. The task scheduler was designed in order to decide the location for the applications to run, i.e., either the fog or the cloud. The project was implemented using Java by making use of integer linear programming and round robin classification. Within the context of resource scheduling, Hoseiny et al. [14] presented a dynamic programming approach for energy-efficient fog node selection in the fog environment. Energy savings were achieved by implementing and validating the model in the iFogSim simulator. Moreover, the authors used Software-Defined Network (SDN) policies to control the entire network. The researchers used MATLAB as a simulation environment in [14–16] because of the graphics user interface, the large databases for the built-in algorithms, and the easy implementation, whereas iFogSim was used in [8,13,17,18] because of its ability to be used in IoT-enabled smart homes, smart parking, healthcare, smart cities, and many more.

Shaaban et al. [18] and Gupta et al. [19] proposed a real-time architecture for the dynamic distribution of health care tasks among computational nodes. A mobility-aware heuristic model was proposed and validated using the iFogSim simulator to reduce the makespan cost and increase the energy efficiency. Along with this, the authors also presented the idea of a realistic simulation model inside a hospital in Chicago. Furthermore, a bio-inspired hybrid algorithm was proposed by Li et al. [20] and Baccarelli et al. [21] to manage the resources at the fog layer. A modified particle swarm optimization was used to achieve a good average response time and the optimal resource utilization and execution time. The simulations were performed using the iFogSim simulator.

There are many works in the literature about task scheduling, resource allocation, and workflow scheduling; most of them discuss the total cost, makespan, network delay, and wait time, but very few articles were found regarding the energy consumption, network utilization, and latency within the fog environment. Moreover, the recent literature on task scheduling algorithms in fog computing is sparse; therefore, this article discusses the partially explored algorithms used in the fog environment. Thus, the objective of this work was to perform task scheduling using a modified particle swarm optimization in order to achieve the optimal resource utilization while improving the three above-mentioned parameters. The simulations were performed using the Java-based open-source simulation tool iFogSim, which is one of the most-capable tools for the evaluation of resource management policies with respect to the latency time, energy consumption, network utilization, and overall operational cost.

Ali et al. in [22] proposed a fuzzy-based task scheduling technique to divide the task between the fog and cloud layers. They distributed the task between the cloud and fog layers by considering different processing units such as the storage, bandwidth, data size, and computations. The objective of this study was to improve the efficiency of task execution, but they did not mention which real-time application they were intending to addresswith their proposal. Swarup et al. [23] proposed a deep-reinforcement-learning-based algorithm to reduce the wait time of tasks in a virtual machine queue. A parallel queuing approach was followed to ensure faster and optimal resource allocation. The main challenge of rescheduling tasks with long wait times was not addressed in this article; thus, we used the machine learning naïve Bayes classifier for task scheduling, and in the future, we will address the issue of rescheduling tasks with long wait times [24–27]. Table 1 provides the findings from the survey of the different static, dynamic, and hybrid scheduling approaches and simulation environments used.

Table 1. Comparison of the existing state-of-the-art solutions.

Author's Name	Type of Algorithm	Workflow	Findings	Simulation Environment
Nguyen et al. [28]	Static	Task scheduling	Tradeoff between total cost and makespan	iFogSim
Jamil et al. [8]	Static	Task scheduling	Reducing network delay, wait time, and network usage	iFogSim
Li et al. [17]	Static	Resource allocation	Providing a fast convergence speed	MATLAB
Sun et al. [15]	Dynamic	Task scheduling	Optimal resource utilization and reducing execution time	Java
Abdelmoneem [18]	Dynamic	Resource allocation	Reducing energy consumption in the fog environment	iFogSim
Wang [16]	Hybrid	Task scheduling	Reducing compilation time and energy consumption and making the fog environment more reliable	MATLAB
Rafique et al. [29]	Hybrid	Resource allocation	Optimizing resource utilization and minimizing processing cost and response time	iFogSim
Guevara et al. [13]	Static	Workflow scheduling	Reducing the task's execution time and makespan	Java
Xu et al. [30]	Hybrid	Workflow scheduling	Reducing the task completion time	MATLAB

3. Proposed Model

3.1. System Architecture

In this section, we set up the system architecture for the optimal task allocation with respect to smart homes. We considered a fog–cloud system, which consists of three layers, the IoT sensor layer, the fog layer, and the cloud layer, as shown in Figure 2. The sensor layer comprises intelligent sensor nodes located in different areas of the smart home sending data to the fog layer for further processing. The fog layer contains special network devices such as routers, switches, and virtual network functions, which have better storage and computing capabilities. The last layer, i.e., the cloud layer, also known as the high-performance layer, comprises a high-performance server with a large storage and high processing power. The IoT layer forwards the data to the fog layer in terms of the task requests. The fog layer further processes those tasks using the machine learning model shown in Figure 2 and returns the results to the IoT layer. The set of tasks in the smart home are in the form of sensors generating datasets such as a real-time camera, smoke detector, humidity sensor, water leak sensor, motion sensor, electricity usage sensor, and light sensor. The objective of the proposed machine-learning-based task allocation is to provide the optimal allocation of the fog nodes, to improve the resource utilization, to make the response time faster, and to improve the overall cost. Delay-sensitive tasks such as the smoke detector and water leaks need be given priority in the service queue, which would be allocated first on the fog nodes. To add these tasks into the priority queue, we propose an intensity-based task allocation model, which works as follows:

- A virtual node is assigned to the fog layer for a smart home application in order to check the intensity of every incoming task.
- This virtual node monitors every generated task from the intelligent devices of the smart home.
- The virtual node will rank every task according to its intensity level (example: T1–>R1, i.e., Rank 1 is given to Task 1).
- Here, intensity means how much computing resource (CR) a task will use, how much delay (DL) a particular task can tolerate while performing the task, and how much memory (M) space is required by every task. Rank 1 will be given to the task that requires less computing resource, that requires less memory, and that is not delay-tolerant.

- Tasks will be added to the priority queue according to their ranks and assigned accordingly over the fog nodes; thus, this allocation policy results in the optimal allocation of the tasks.

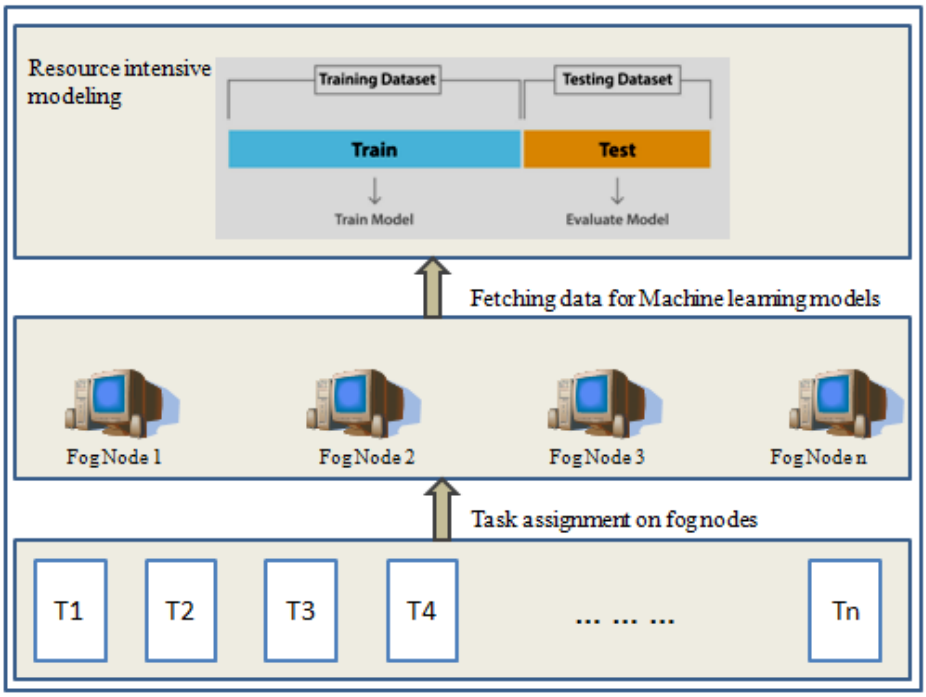

Figure 2. Proposed fog architecture for task allocation.

Table 2 provides the list of abbreviations used in this article. The proposed 2-stage task scheduling algorithm (2-ST) will be run by a task scheduler, which is the heart of the fog broker and sits in the fog layer.

Table 2. List of abbreviations.

Symbols	Definition
$T1, T2, \ldots, T_n$	Number of tasks
R1	Rank 1 assigned to a particular task
CR	Computing resource
DL	Delay during execution
2-ST	Two-stage task allocation
$N1, N2, \ldots, N_n$	Number of computing nodes
RT	Response time
EC	Energy consumption
TE and TS start time of execution	End time and
NU	Network utilization
HP_r	Host power utilization
NS	Network size
L_i	Latency of ith task

3.2. Problem Formulation

The mathematical formulation of the task scheduling model is presented in this section. The proposed architecture contains n independent tasks $T = T1, T2, \ldots, Tn$, as depicted in Figure 2.

$$EC = E \times (T_c - T_r) + HP_r \tag{1}$$

Here, T denotes the number of tasks, and T_s and TE denote the start and end of the execution. There are two states for the fog nodes: idle and busy. Whenever the fog node is not processing any task, it is said to be in idle mode, and when it is executing any task, it is said to be in busy mode. The fog device's energy consumption is calculated in iFogSim using Equation (3). The energy consumption of the fog nodes (EC) is calculated using the current energy consumption E, the current time T_c, the updated time during the last utilization T_r, and the host power utilization HP_r. HPR is the host power in the last utilization. Energy may be calculated using the total of all the hosts' power during the set length of time for any of the fog devices. The smart home case study deals with a large number of sensor devices; therefore, network congestion caused by these devices is very common, which can degrade the overall performance of the system. Thus, by allocating the loads to several fog nodes, we minimized the network congestion in this system. The network usage is calculated using Equation (4), where NU is the network utilization, n is the total number of tasks, L_i is the latency, and NS is the network size of the ith task. Furthermore, the fitness function is calculated to check whether the model needs to be trained again or should enter the optimization phase. The fitness function is provided in Equation (5), and it depends on the latency and network usage of the network. High values of the latency and network usage reduce the fitness function, and correspondingly, we have to satisfy the fitness function. The grater the value of the fitness function, the better solution is.

$$NU = \sum_{i=1}^{n} L_i \times NS_i \tag{2}$$

$$FitnessFunction = \frac{1}{latency + NU} \tag{3}$$

Figure 3 provides the entire data flow of the machine-learning-based optimal model for task allocation. The set of tasks is generated by various IoT sensors used in the smart home environment. Furthermore, Table 2 gives the proposed ML model for optimal task allocation in smart homes and provides the number of tasks that we considered for training the ML model, where the naïve Bayes multi-classifier was used for the training and testing for the allocation of the tasks on the fog nodes. Before applying the naïve Bayes model, tasks were ranked according to the ranking table given in Table 3. According to Table 3, a task is ranked first if it does not tolerate any delay with respect to the services and it occupies less memory space and consumes less computing resource. As naïve Bayes gives the best results for classification, here, while training the machine learning model, we used the constraints mentioned in the ranking table to correctly classify the task with different ranks, and similarly, we again used naïve Bayes for the allocation of the VMs. Being a probabilistic classifier, it predicts the allocation on the basis of the probability given in Equation (6).

$$P(VM_i|tk = compatible = \frac{p(tk = compatible|VM_i)(VM_i)}{p(tk = compatibility)}) \tag{4}$$

For all VMs, $P(tk = compatible)$ is constant and obtained according to Equation (7).
If (resource capacity of VMs > resource capacity of task):
$P(tk = compatible) = 1$.
Else:

$$P(tk = compatible = \frac{Res.cap.ofallVM - Res.cap.oftask}{Res.cap.oftask}) \tag{5}$$

Table 3. Fog node parameters.

Fog Node Parameters	Values
Number of fog nodes	4
Processing capacity	11 MIPS
RAM	8 KB
Bandwidth	1024 Mbps
Storage	92 KB
Idle power consumption	0.36 Watt
Execution power consumption	0.44 watt

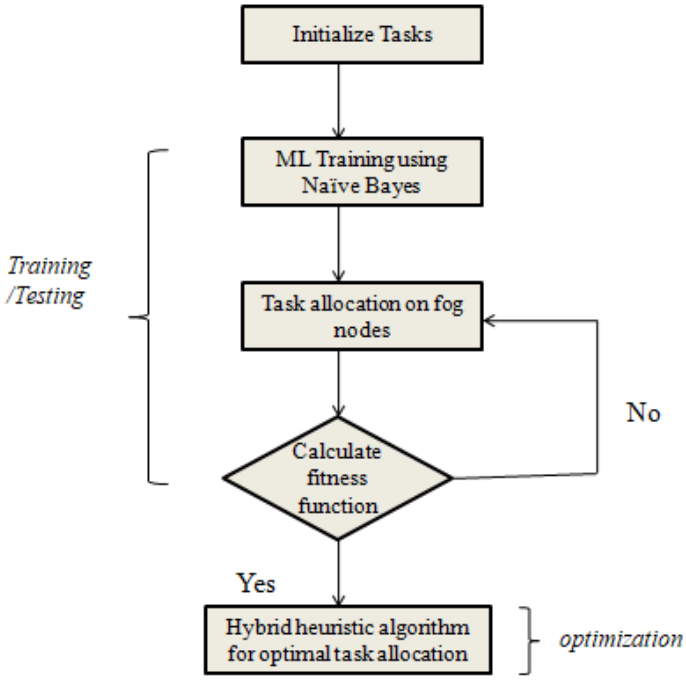

Figure 3. Data flow of the 2-stage task scheduling algorithm for optimal task allocation.

The model is trained until it satisfies the fitness function given in Equation (5), and once it satisfies the fitness function, the optimal task allocation is performed using the hybrid heuristic algorithm [16]. The hybrid heuristic algorithm is a combination of ACO and PSO, and the idea behind their combination is that PSO has a fast search speed and ACO provides high accuracy; thus, their combination contributes to the optimal task allocation in the fog environment. The working of the proposed 2-ST hybrid algorithm can be easily understood by looking at Algorithms 1 and 2. In particle swarm optimization, the position of each particle represents a task's possible scheduling, and according to the task allocation decisions $s_ij(t)$, the position of particle i can be simplified as s_ij, $s_ij \epsilon 0, 1$, $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$. Similarly, the speed of particle i can be simplified as v_ij, $v_ij \epsilon$ [vmax, vmax], $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.

The inertia weight w determines the search ability of the particles in the global and local search, which follows the premise that the greater w is, the stronger the global searching ability of the algorithm. Otherwise, the local search ability of the algorithm is stronger. To improve the intelligence of the particle swarm optimization algorithm, many methods have been proposed regarding inertia weighting, such as a linearly decreasing weight, the weight decrease of the linear differential equation, and so on. This paper combined the

inertia weight with the number of iterations, and the solution formula of the inertia weight is shown below.

$$\begin{aligned} w &= w_{max} - \frac{(w_{max})-w_{min}}{k_{max}} k < 0.7 \times k_{max} \\ &= w_{min} + (w_{max} - w_{min} \times rand) k \geq 0.7 \times k_{max} \end{aligned} \quad (6)$$

where w_{max} is the maximum inertia weight, w_{min} is the minimum inertia weight, k is the number of iterations, and k_{max} is the largest number of iterations. The inertia weight changes dynamically with the increase of the iterations, which guarantees that the algorithm has the opportunity to gain a larger inertia weight value later in the search and is prevented from falling into the local optimum. This approach is used to maintain a balance between exploration–exploitation for this hybrid approach of optimization.

The entire proposed mechanism can be easily understood by looking at Algorithm 1.

Algorithm 1 Optimal task allocation algorithm.

Input value: Task ($T1, T2, \ldots, T_n$)
Output: *Optimal task allocation*
Step 1: DOT_i = *new received task*
Step 2: *All nodes* $I_n = I = 1, 2, \ldots, N$
Assign rank for T_i
Step 3: *Priority queue* = T_i
Step 4: *Sort priority queue according to ranks*
Step 5: *Allocation of VM <– allocation*$_u$*singNaiveBayes()*
Step 6: *Calculate fitness function*
Step 7: *Fitness optimization using hybrid heuristic algorithm*

The smart home environment as an example was considered in this study, and Figure 4 shows a few tasks that could be considered in this scenario, for example: alarm generation, processing of the humidifier, lights on/off, surveillance video, and many others. A few tasks would be delay-tolerant, and a few would be non-delay-tolerant. These tasks will be allocated to the fog nodes. Suppose in the above scenario that the smoke detector is a non-delay-tolerant task and considered as a Rank 1 task. Similarly, the water pump turning on/off is also a non-delay-tolerant task, so the ranking table plays an important role here. Such tasks will be allocated to the fog nodes with specified resources. The proposed 2-ST hybrid algorithm is used for optimal task allocation using Algorithms 1 and 2. Algorithm 1 is used for training the machines using the naïve Bayes algorithm for the rank determination and allocation of the VMs. If a particular task allocation meets the criteria of the fitness function, then this allocation will proceed to the hybrid algorithm using ACO and PSO.

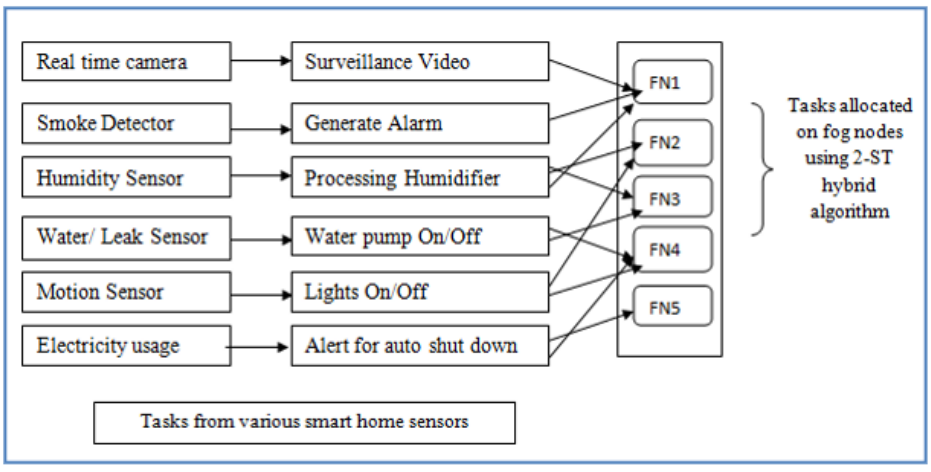

Figure 4. Example of task allocation using the 2-ST hybrid algorithm.

Algorithm 2 The 2-ST hybrid algorithm.

Input value: Task ($T1$, $T2$, …, T_n)
Output: *Optimal task allocation*
Step 1: *Calculate the fitness value of particles and looking for individual optimal solution*
Step 2: *Look for global optimal solution, and update particles' velocity and positions*
Step 3: *Check for termination condition; if no, then* (1), *else* (5)
Step 4: *Initialize population, and check for ant number set*
Step 5: *Increment ant number*
Step 6: *Transition probability used by nodes to choose net node*
Step 7: *Local update of pheromone*
Step 8: *Traverse all ants; if no, go to* (6), *else update global pheromone*
Step 9: *Check for termination condition; if no*, (5), *else allocate tasks.*

4. Performance Analysis and Evaluation Matrix

This section discusses the simulation setup, performance parameters, and the analysis of the results of our proposed ML model for the task allocation in a smart home architecture. We conducted the simulations using iFogSim [19], which provides a complete cloud–fog environment for simulating cloud and fog applications. The environment and simulation setup for a smart home are described as follows: the smart home area was set to 200 × 200 m along with 4 fog devices, and the number of IoT sensors was 7. These fog devices communicate with the fog gateways using WiFi technology. We assumed that every fog node has its own processing (in MIPS), memory, and transmission capacity. For the evaluation of the smart home architecture in iFogSim, we needed to define the fog devices and set the values of a few parameters. There were 2 types of computational devices in the smart home fog environment. The first was the fog nodes for the processing of the data generated from the IoT sensors, and the second was the IoT sensors for the data generation. Table 3 provides the details of the fog nodes. The fog nodes' data were taken from [20,21]. Table 3 provides the parametric details of the IoT sensors. The simulations were carried out on a Dell computer (Intel core i5-12400, 256 GB hard drive) with the Windows 10 professional operating system. Table 4 provides the details of the sensors we used in the smart home environment.

Table 4. Smart home IoT sensors' features.

IoT Sensors' Parameters	Values
Number of IoT sensors	7
RAM	1 KB
Processing capacity	4 MIPS

4.1. Simulation Setup

The numerical simulation was performed over both adversarial and legitimate models by publishing both ideal and intruder devices on the network. The intruders may steal or hack the consumer device for their own purposes or jam the network to perform a distributed denial of service attack. The trust values and communication track of both legitimate and fake devices were recorded after every specific interval of time as S(t). Let N(I) represent the number of ideal consumer electronic devices communicating in the network and N(M) denote the number of devices hacked by the intruders in the network. Table 5 illustrates the types of sensors and the units of measurement.

Three performance parameters, latency, energy consumption, and network usage, were considered for the comparison of the two-stage task scheduling (2-ST) algorithm with the hybrid heuristic algorithm [12] in both the fog and cloud environment. Fog nodes are intermediate nodes and helpful for providing the minimum latency period for time-sensitive data, rather than processing the data in the cloud. Fog computing supports latency-sensitive real-time applications, and thus, the latency is considered one of the

important performance parameters. Secondly, the energy consumption reduces the power utilization of the fog nodes, which further reduces the overall electricity bill for the smart home environment. Energy consumption was also compared for both the cloud and the non-cloud environment and is discussed in Section 5. The last performance parameter considered in this study was the network usage. An increase in the number of smart devices in the smart home leads to network congestion, but the concepts of resource management and task management play very important roles in the allocation of tasks on the different fog nodes in order to reduce the network congestion.

Table 5. IoT sensors' parameters.

Sensor Data	Unit of Measurement
Humidity	% age
Motion sensor	On/off
Smoke	Obscuration in dBA
Water leak sensor	Buzzer on/off
Electricity usage sensor	KWH
Light sensor	Lumens; on/off

4.2. Results and Discussion

This study discusses the proposed two-stage optimal model for task scheduling within the fog environment with an application to smart homes. The smart home case study deals with a time series solution, which further reduces the latency time, minimizes the usage of the network bandwidth, has a quick response time, and has many other benefits.

Fog computing is one the most-suitable environments for providing all these benefits to smart homes. The concepts of resource management and task scheduling help to distribute the tasks and resources among other fog nodes in order to improve the performance of the network and services. Thus, the proposed 2-ST approach is very efficient in the smart home scenario as compared to the hybrid approach proposed by Wang et al. [12] for both fog and non-fog environments. The efficiency of the smart home scenario within the fog environment is calculated in terms of energy consumption, latency, and network usage. Figure 5 shows the energy consumption in Joules for the proposed 2-ST model and its comparison with the hybrid approach for both the fog and the cloud environment. The corresponding figure shows the reduced energy consumption for an increased number of tasks within the smart home fog environment. Due to the decentralized nature of cloud computing, it provides the best performance with respect to the data storage and processing, but brings an increase in the latency, energy consumption, and network utilization. Thus, cloud computing is not the best solution for time-critical applications such as smart homes, smart cities, healthcare, and many others.

In contrast to the aforementioned benefits and drawbacks of cloud computing, a new layer has been added into the existing cloud architecture, named the fog layer, and thus, a new environment came into existence with its own benefits, known as fog computing. One of the main advantages of the fog environment is the latency time, and Figures 6 and 7 show the same along with a comparison of the proposed and hybrid approaches for the fog and cloud environment. The fog layer is responsible for bringing resources near the IoT sensors for the actions or tasks to perform, which further reduces the load on the cloud server and reduces the latency time during task execution.

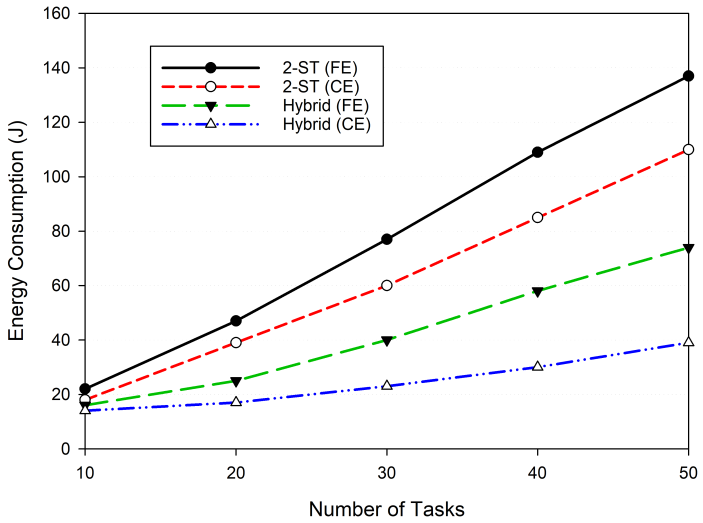

Figure 5. Energy consumption in J for different tasks within the smart home environment.

Our proposed 2-ST, i.e., the two-stage optimal task scheduling approach, reduced the energy consumption, latency time, and network utilization significantly compared to the hybrid approach for both the fog and cloud environment. The results clearly showed that the energy consumption, network utilization, and latency time increased with the increase in the number of tasks and sensors within the smart home. The comparison between the hybrid and 2-ST approach showed small differences with respect to the energy consumption, but a great difference with respect to the latency time. Similarly, the network usage also showed a significantly large difference between the 2-ST and hybrid approach for the cloud environment, as well as the fog environment. Thus, overall, the results showed that the proposed 2-ST approach performed the best for the smart home scenario.

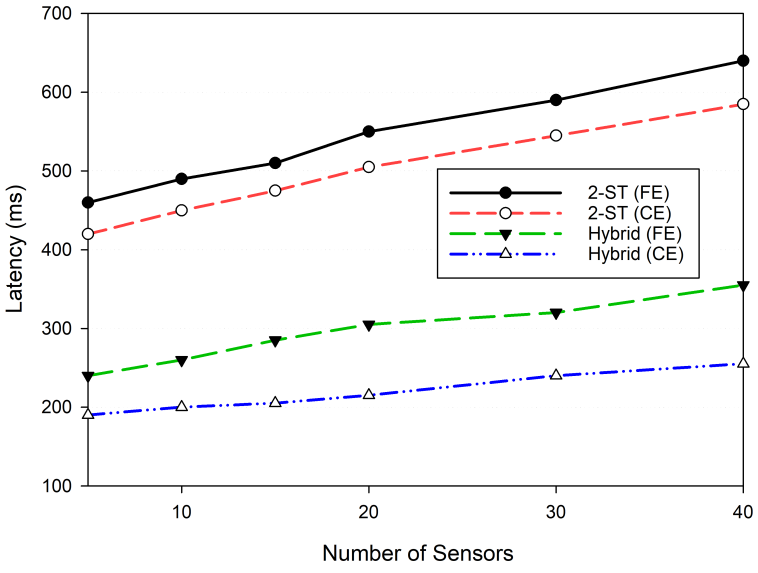

Figure 6. Latency comparison of the 2-ST and hybrid approaches for both the cloud and fog environment.

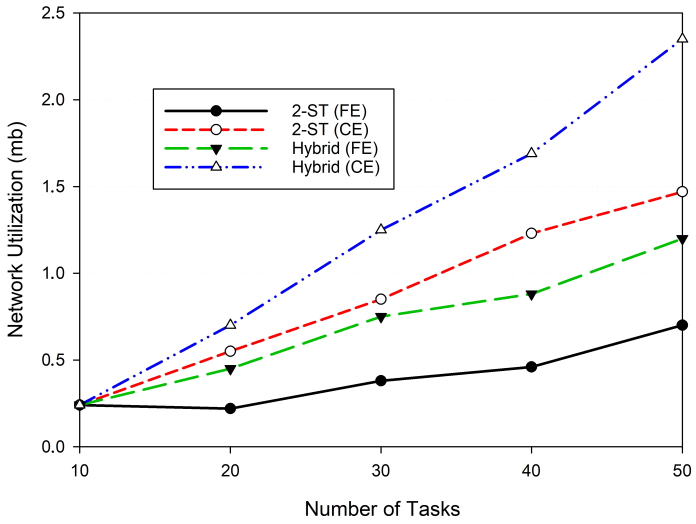

Figure 7. Network utilization of both the cloud and fog environment using the 2-ST and hybrid approaches for task scheduling.

5. Conclusions

Cloud computing does not provide efficient results in terms of bandwidth, energy consumption, network usage, latency, and many other performance parameters for time-critical applications. Therefore, fog computing assists the cloud environment to overcome the performance drawbacks of cloud computing by bringing resources near the edge of the network and distributing the load among different fog nodes. The distribution of the load in the fog environment is a significant challenge, which involves several task scheduling, resource scheduling, and job scheduling algorithms. This study proposed a two-stage optimal task scheduling (2-ST) approach for the distribution of tasks executed within a smart home among several fog nodes. To solve the task scheduling effectively, this proposed approach uses a naïve-Bayes-based machine learning model for training in the first stage and optimization in the second stage using a hyperheuristic approach, which is a combination of both ACO and PSO. The proposed approach was validated by performing simulations in iFogSim, and the corresponding results showed better performance in terms of energy consumption, latency time, and network usage. In this study, we tried to use machine learning and optimization algorithms together to perform the task scheduling algorithms for a small-scale smart home environment. The reason for combining ACO and PSO was that PSO provides computational efficiency and an easy implementation, while ACO is used to solve different problems like ours, in which we had to compute intensive tasks and delay-sensitive tasks. The results showed a significant improvement, and therefore, we will use this combination for large databases in our future work. Along with this, a future work in the context of of this study is also to extend it to the prediction of future network usage, bandwidth consumption, and energy consumption by predicting the user's behavior on the basis of his/her daily routine within the smart home environment. This study was tested on small and static datasets described in Tables 4 and 5. Therefore, the proposed methodology can be tested on large and dynamic datasets.

Author Contributions: The authors equally contributed to this work in all aspects, with more implementation-related efforts by O.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All implementation details, sources, and data will be delivered upon request by the corresponding author Chaker Abdelaziz Kerrache.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Seth, B.; Dalal, S.; Jaglan, V.; Le, D.N.; Mohan, S.; Srivastava, G. Integrating encryption techniques for secure data storage in the cloud. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4108. [CrossRef]
2. Jamil, B.; Ijaz, H.; Shojafar, M.; Munir, K.; Buyya, R. Resource Allocation and Task Scheduling in Fog Computing and Internet of Everything Environments: A Taxonomy, Review, and Future Directions. *Acm Comput. Surv. CSUR* **2022**, *54*, 233. [CrossRef]
3. Luo, Y.; Chen, Y.; Wu, J. Energy Efficient Fog Computing with Architecture of Smart Traffic Lights System. In Proceedings of the 2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT), Surabaya, Indonesia, 9–11 April 2021; pp. 248–254.
4. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the Internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
5. Djeradi, S.; Bendouma, T.; Lagraa, N.; Kerrache, C.A.; Lakas, A.; Brik, B. Toward the Integration of UAVs' Services into the Cloud. *IEEE Commun. Stand. Mag.* **2021**, *5*, 25–32. [CrossRef]
6. Kabirzadeh, S.; Rahbari, D.; Nickray, M. A hyper heuristic algorithm for scheduling of fog networks. In Proceedings of the 2017 21st Conference of Open Innovations Association (FRUCT), Helsinki, Finland, 6–10 November 2017; pp. 148–155.
7. Sharma, O.; Anusha, S. Large-scale data streaming in fog computing and its applications. In *Large-Scale Data Streaming, Processing, and Blockchain Security*; IGI Global: Hershey, PA, USA, 2021; pp. 50–65.
8. Jamil, B.; Shojafar, M.; Ahmed, I.; Ullah, A.; Munir, K.; Ijaz, H. A job scheduling algorithm for delay and performance optimization in fog computing. *Concurr. Comput. Pract. Exp.* **2020**, *32*, e5581. [CrossRef]
9. Yang, X.; Rahmani, N. Task scheduling mechanisms in fog computing: Review, trends, and perspectives. *Kybernetes* **2020**, *50*, 22–38. [CrossRef]
10. Forestiero, A. Heuristic recommendation technique in Internet of Things featuring swarm intelligence approach. *Expert Syst. Appl.* **2022**, *187*, 115904. [CrossRef]
11. Abualigah, L.; Elaziz, M.A.; Khodadadi, N.; Forestiero, A.; Jia, H.; Gandomi, A.H. Aquila Optimizer Based PSO Swarm Intelligence for IoT Task Scheduling Application in Cloud Computing. In *Integrating Meta-Heuristics and Machine Learning for Real-World Optimization Problems*; Springer: Cham, Switzerland, 2022; pp. 481–497.
12. Islam, M.S.U.; Kumar, A.; Hu, Y.C. Context-aware scheduling in Fog computing: A survey, taxonomy, challenges and future directions. *J. Netw. Comput. Appl.* **2021**, *180*, 103008. [CrossRef]
13. Guevara, J.C.; da Fonseca, N.L. Task scheduling in cloud-fog computing systems. *Peer-Peer Netw. Appl.* **2021**, *14*, 962–977. [CrossRef]
14. Hoseiny, F.; Azizi, S.; Shojafar, M.; Tafazolli, R. Joint QoS-aware and cost-efficient task scheduling for fog–cloud resources in a volunteer computing system. *ACM Trans. Internet Technol. (TOIT)* **2021**, *21*, 86 . [CrossRef]
15. Sun, Z.; Li, C.; Wei, L.; Li, Z.; Min, Z.; Zhao, G. Intelligent sensor-cloud in fog computer: A novel hierarchical data job scheduling strategy. *Sensors* **2019**, *19*, 5083. [CrossRef]
16. Wang, J.; Li, D. Task scheduling based on a hybrid heuristic algorithm for smart production line with fog computing. *Sensors* **2019**, *19*, 1023. [CrossRef] [PubMed]
17. Li, G.; Liu, Y.; Wu, J.; Lin, D.; Zhao, S. Methods of resource scheduling based on optimized fuzzy clustering in fog computing. *Sensors* **2019**, *19*, 2122. [CrossRef] [PubMed]
18. Abdelmoneem, R.M.; Benslimane, A.; Shaaban, E. Mobility-aware task scheduling in cloud-Fog IoT-based healthcare architectures. *Comput. Netw.* **2020**, *179*, 107348. [CrossRef]
19. Gupta, H.; Vahid Dastjerdi, A.; Ghosh, S.K.; Buyya, R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* **2017**, *47*, 1275–1296. [CrossRef]
20. Baccarelli, E.; Scarpiniti, M.; Momenzadeh, A. Ecomobifog–design and dynamic optimization of a 5g mobile-fog–cloud multi-tier ecosystem for the real-time distributed execution of stream applications. *IEEE Access* **2019**, *7*, 55565–55608. [CrossRef]
21. Li, Y.; Orgerie, A.C.; Rodero, I.; Amersho, B.L.; Parashar, M.; Menaud, J.M. End-to-end energy models for Edge Cloud-based IoT platforms: Application to data stream analysis in IoT. *Future Gener. Comput. Syst.* **2018**, *87*, 667–678. [CrossRef]
22. Ali, H.S.; Rout, R.R.; Parimi, P.; Das, S.K. Real-time task scheduling in fog–cloud computing framework for iot applications: A fuzzy logic based approach. In Proceedings of the 2021 International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 5–9 January 2021; pp. 556–564.
23. Swarup, S.; Shakshuki, E.M.; Yasar, A. Energy Efficient Task Scheduling in Fog Environment using Deep Reinforcement Learning Approach. *Procedia Comput. Sci.* **2021**, *191*, 65–75. [CrossRef]
24. Majumder, S. Some network optimization models under diverse uncertain environments. *arXiv* **2021**, arXiv:2103.08327.
25. Nagarajan, S.M.; Deverajan, G.G.; Chatterjee, P.; Alnumay, W.; Ghosh, U. Effective task scheduling algorithm with deep learning for Internet of Health Things (IoHT) in sustainable smart cities. *Sustain. Cities Soc.* **2021**, *71*, 102945. [CrossRef]

26. Selvaraj, A.; Patan, R.; Gandomi, A.H.; Deverajan, G.G.; Pushparaj, M. Optimal virtual machine selection for anomaly detection using a swarm intelligence approach. *Appl. Soft Comput.* **2019**, *84*, 105686. [CrossRef]
27. Nagarajan, S.M.; Devarajan, G.G.; Mohammed, A.S.; Ramana, T.; Ghosh, U. Intelligent Task Scheduling Approach for IoT Integrated Healthcare Cyber Physical Systems. *IEEE Trans. Netw. Sci. Eng.* **2022**, 1–11. [CrossRef]
28. Nguyen, B.M.; Thi Thanh Binh, H.; The Anh, T.; Bao Son, D. Evolutionary algorithms to optimize task scheduling problem for the IoT based bag-of-tasks application in cloud–fog computing environment. *Appl. Sci.* **2019**, *9*, 1730. [CrossRef]
29. Rafique, H.; Shah, M.A.; Islam, S.U.; Maqsood, T.; Khan, S.; Maple, C. A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing. *IEEE Access* **2019**, *7*, 115760–115773. [CrossRef]
30. Xu, R.; Wang, Y.; Cheng, Y.; Zhu, Y.; Xie, Y.; Sani, A.S.; Yuan, D. Improved particle swarm optimization based workflow scheduling in cloud-fog environment. In Proceedings of the International Conference on Business Process Management, Sydney, Australia, 9–14 September 2018; Springer: Cham, Switzerland, 2018; pp. 337–347.