systems

Article

Machine Learning-Driven Ubiquitous Mobile Edge Computing as a Solution to Network Challenges in Next-Generation IoT

Moteeb Al Moteri [1], Surbhi Bhatia Khan [2,3,*] and Mohammed Alojail [1]

1 Department of Management Information Systems, College of Business Administration, King Saud University, P.O. Box 28095, Riyadh 11437, Saudi Arabia; amoteeb@ksu.edu.sa (M.A.M.)
2 Department of Data Science, School of Science, Engineering and Environment, University of Salford, Salford M5 4WT, UK
3 Department of Electrical and Computer Engineering, Lebanese American University, Byblos P.O. Box 13-5053, Lebanon
* Correspondence: surbhibhatia1988@yahoo.com

Abstract: Ubiquitous mobile edge computing (MEC) using the internet of things (IoT) is a promising technology for providing low-latency and high-throughput services to end-users. Resource allocation and quality of service (QoS) optimization are critical challenges in MEC systems due to the large number of devices and applications involved. This results in poor latency with minimum throughput and energy consumption as well as a high delay rate. Therefore, this paper proposes a novel approach for resource allocation and QoS optimization in MEC using IoT by combining the hybrid kernel random Forest (HKRF) and ensemble support vector machine (ESVM) algorithms with crossover-based hunter–prey optimization (CHPO). The HKRF algorithm uses decision trees and kernel functions to capture the complex relationships between input features and output labels. The ESVM algorithm combines multiple SVM classifiers to improve the classification accuracy and robustness. The CHPO algorithm is a metaheuristic optimization algorithm that mimics the hunting behavior of predators and prey in nature. The proposed approach aims to optimize the parameters of the HKRF and ESVM algorithms and allocate resources to different applications running on the MEC network to improve the QoS metrics such as latency, throughput, and energy efficiency. The experimental results show that the proposed approach outperforms other algorithms in terms of QoS metrics and resource allocation efficiency. The throughput and the energy consumption attained by our proposed approach are 595 mbit/s and 9.4 mJ, respectively.

Keywords: internet of things; resource allocation; quality of service; hybrid kernel random forest; ensemble support vector machine; crossover-based hunter-prey optimization

check for updates

Citation: Al Moteri, M.; Khan, S.B.; Alojail, M. Machine Learning-Driven Ubiquitous Mobile Edge Computing as a Solution to Network Challenges in Next-Generation IoT. *Systems* **2023**, *11*, 308. https://doi.org/10.3390/systems11060308

Academic Editors: In-kee Kim, Avinash Kalyanaraman and Lakshmish Ramaswamy

Received: 1 May 2023
Revised: 7 June 2023
Accepted: 12 June 2023
Published: 16 June 2023

1. Introduction

The challenges include ineffective resource utilization to process data and latency in the data processing. The end-node devices induce these challenges in big data. To overcome the problem of analytics and big data storage caused by high quantities of cloud resources, the primitive technology of cloud computing has been merged with these new networks [1]. It provides lower delay and larger computing agility when compared to the strong computing platforms in cloud data centers (CDC). At the edge of the network, about 40% of the IoT-created data are contained and processed [2]. In a heterogeneous environment, the research community faces several problems such as efficient data collection, network architecture, reliable traffic management, storage, and security due to the interconnection of these devices. In addition to this, due to the insufficiency of resources such as memory, onboard power, processing, and communication, wireless sensors are prone to multiple threats. Hence, with reduced resource usage, an effective communication structure of sensor devices can increase its performance in producing results with great accuracy [3].

The main challenge faced by mobile edge computing (MEC) is mobility issues. The risk to the integrity of the data and interruptions to service delivery caused by security threats can affect the MEC ecosystem in terms of reliability and loss of availability. Fault tolerance is one of the challenges of MEC which consists of availability, dependability, and reliability. To enhance the efficacy in dealing with time constraints, the offloading graininess and partitioning mention the code size that must be loaded for remote execution [4]. The MEC user remains in the coverage areas of MEC service providers only for a limited duration, which results in the diverse user demand. Various types of users require a variety of services which are changing rapidly based on the values of requirements. Hence, it is necessary to establish the services in a cost-effective manner. Meanwhile, developing a cost-effective technique is considered a challenging task because of the diverse nature of emerging services.

QoS refers to the performance characteristics and level of service provided by a network or system, which directly impacts the end-user experience. It encompasses various metrics such as latency, throughput, reliability, availability, and energy efficiency. Achieving a high QoS in MEC systems is crucial for delivering low-latency and high-throughput services to end-users and ensuring a satisfactory user experience. The International Telecommunication Union (ITU) plays a significant role in establishing standards and guidelines for QoS in telecommunication networks. ITU-T Supp. 9 of the E.800 Series provides regulations and recommendations related to QoS in telecommunication services. This document offers a comprehensive framework for assessing, measuring, and monitoring QoS parameters, facilitating the effective management and optimization of network performance.

A deployed network's overall lifetime gets reduced due to the increased energy consumption of the sensor devices used in high traffic rates and the heterogeneous communication infrastructure [5]. Load balancing and resource allocation play a crucial role in optimizing network performance and enhancing system lifespan by managing heavy-duty hours. However, it is often used as an unconstrained process, leading to side effects. The solution lies in implementing proper admission control to ensure genuine network load and enhance network load balancing. This can significantly improve overall network functioning. To achieve optimal results, the load balancing process needs to work in tandem with the admission control process [6]. The key contributions of this paper are described as follows.

A novel approach has been developed for resource allocation and QoS optimization in MEC using IoT: the proposed approach combines hybrid kernel random forest [7] and ensemble support vector machine algorithms [8] with crossover-based hunter-prey optimization to optimize the QoS metrics and allocate resources to different applications running on the MEC network. Improved QoS metrics [9]: The proposed approach aims to improve QoS metrics such as latency, throughput, and energy efficiency by allocating resources to different applications running on the MEC network. In the context of ubiquitous mobile edge computing (UMEC) using IoT, it is crucial to optimize resource allocation and quality of service (QoS) metrics while considering the cost implications. By efficiently allocating resources to different applications running on the UMEC network, the proposed approach aims to achieve cost-effective service provision. In UMEC systems, there is a large number of devices and applications involved, leading to resource allocation challenges. Inefficient resource utilization can result in increased costs, such as higher energy consumption and poor network performance. Therefore, optimizing resource allocation becomes essential for cost-effectiveness. The proposed approach combines machine learning algorithms (hybrid kernel random forest and ensemble support vector machine) with an optimization algorithm (crossover-based hunter-prey optimization) to achieve cost-effective service provision. By accurately predicting and optimizing QoS metrics, such as latency, throughput, and energy efficiency, the system can allocate resources more effectively and reduce unnecessary costs. By optimizing resource allocation and QoS metrics, the proposed approach aims to strike a balance between service quality and cost efficiency. This can help service providers deliver high-quality services to end-users while minimizing operational costs. The cost-effective provision of services is crucial for ensuring the sustainability and profitability of UMEC systems.

The paper organization is arranged as follows. In Section 2, past literature works are described in the context of mobile edge computing. The ubiquitous mobile edge computing using hybrid ensemble SVM and kernel random forest is depicted in Section 4. In Section 5, the evaluation results are discussed. In the final section, the conclusion of the paper is presented.

2. Literature Review

As mentioned in Table 1, Yu et al. [10] presented a method of edge computing that implemented SAGINs for reducing the usage of satellite resources and the task completion time. Additionally, the action space size was reduced through the pre-classification method. The proposed method used a deep imitation learning-driven caching and offloading algorithm and thereby achieved real-time decision-making. They have evaluated the developed method in a simulated environment and compared it to other existing edge computing methods. Ai et al. [11] developed an approach, namely, a smart collaborative framework (SCF) for creating multi-task offloading solutions and for achieving a prediction of dynamic service. They have developed a theoretical approach and used hybrid deep learning algorithms in a hierarchical spatio-temporal monitoring (HSTM) approach from spatio-temporal dimensions. Additionally, they have used advanced queuing and mixed game theories for enhancing the offloading efficiency of the scheduling approach, namely, fine-grained resource scheduling (FRS). However, the high computational expenditure and large memory needed by the DNN significantly diminish the deep learning usage in edge computing with restricted resources.

Sood et al. [12] presented a smart traffic management approach for the prediction of the inflow of traffic and time-enhanced vehicles' smart navigation, which was based on edge-cloud-centric IoT. The congestion at junctions was avoided through the prediction of traffic arrival and also through avoiding long queues. They have used a baseline classifier and analyzed the traffic arrival, the result showed that the proposed Smart management approach was more effective in terms of road safety at junctions, smart navigation, and best load balancing when compared to other existing methods. However, this model is not energetically suitable for resource-restricted mobile mechanisms.

Mazumdar et al. [13] presented a three-layered fog node IoT approach to optimize the service with regard to time. They used the load-offloading method in the load sharing approach to enhance the security features of the proposed method, and its efficiency was evaluated through similar existing methods. A limitation of this method is that utilized mobile terminals can only depend on cloud graphic processing units (GPUs) to stimulate calculating; although, the cloud computing security, the bandwidth of the wireless network, and the communication delay will increase the network complexity.

Chien et al. [14] developed a spatio-temporal-dependency approach that was designed for feature extraction, and which was based on a convolutional neural network (CNN). Shah et al. [15] presented an empirical multi-agent cognitive method for the consecutive transition of IoT APIs. They discovered a classification method for CAs which enables the creation of CA, control, and migration. The proposed method achieved an IoT API distribution and transparency in the heterogeneity, which provided cloud computing optimization. However, the contrasted sensory data accumulate over a large network; hence, the data itself may have a paradox. Bolettieri et al. [16] proposed a heuristic algorithm with linear relaxation and rounding techniques due to optimization problem complexity. The proposed approach was not effective in handling inconsistent traffic demands. This method mainly involved two types of base station traffic prediction data to enhance the hyperparameters. Mobile edge computing (MEC) was integrated with the base station to reduce the time cost of data transmission to the cloud server. The performance metrics such as mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE) were used for evaluation. The results showed that training time decreased and prediction accuracy increased. On the other hand, the use of a large amount of data significantly impacted the performance of this model [17].

Abbasi et al. [18] explained the fog computing based IoT architecture for mobile edge computing. This method used a genetic algorithm (GA) to handle many requests and their security and quality, and fog computing was used to enhance the management and processing of IoT and smart grid. The results showed that this method reduced the delay and consumption of power of devices. Meanwhile, deep neural networks were required to solve the multi-objective optimization problems.

For edge-computing-enabled IoT systems, Liao and Cheng [19] developed a consensus technique (RVC) based on voting and reputation. The computing resources were carried nearer to the internet of things (IoT) and farther from the center of the cloud via edge computing. This enhanced the growth of the IoT by minimizing the delay. Blockchain improved the IoT's security problems, as the devices and edge servers were scattered. A consensus technique based on voting and reputation (RVC) was employed in this article, to rectify issues such as reducing consensus efficiency and the safety of the existing consensus techniques. A successful consensus rate, transaction output, and reduced time were the advantages of this RVC. The increased number of nodes was the drawback of this RVC.

Karjee et al. [20] established split computing technology to provide a superior user experience and alleviate the issues of partly offloading the DNN model inference task from an IoT device to a trusted device called an edge. When compared with in-device inference time, the results reveal that the DNN model minimizes the inference time and balances the tasks across edge devices to significantly reduce battery drainage. The energy utilization/battery dissipation of edge devices was examined and indicated, which minimizes the overall execution time of each task and amends the user experience through implementing this mechanism [21]. Due to their low computational capabilities, these tasks are arduous to accomplish in a short period of time and provide accurate results.

Chen et al. [22] explored the concept of load balancing in mobile edge computing (MEC) systems within ultra-dense networks. The study focused on improving the efficiency of MEC by accurately estimating the load on different edge servers and dynamically allocating tasks based on this estimation. The load estimation model proposed in the study may have relied on simplified assumptions and factors, such as CPU utilization, memory usage, and network traffic. While these factors are important, there may be other parameters and complexities that can affect the load on edge servers. The accuracy of the load estimation model could be further improved by considering a broader range of factors.

Poryazov et al. [23] discussed the normalization of quality of experience (QoE) models in telecommunication systems. The study addressed the limitation of existing QoE prediction models that often provide inadequate results due to variations in data collection and presentation. The authors proposed an overall model normalization technique to improve the accuracy and reliability of QoE predictions. Mutichiro et al. [24] discussed the Dynamic pod-scheduling model to solve the task scheduling problem at the edge.

Table 1. Summary of various related works.

Author and Year	Technique	Objective	Pros	Cons
Yu et al., 2021 [10]	Deep Imitation Learning-Driven Caching and Offloading Algorithm	To reduce the usage of satellite resources and the task completion time	Achieved real-time decision-making	Time required for implementation was high
Ai et al., 2023 [11]	Hierarchical Spatio-temporal Monitoring (HSTM)	To achieve prediction of dynamic service	Offloading efficiency was enhanced	High computational expenditure and large memory needed
Sood et al., 2021 [12]	Smart traffic management approach	To predict inflow of traffic and to enhance vehicle's smart navigation	More effective and better load balancing	Cannot be suitable energetically for resource-restricted mobile mechanism

Table 1. *Cont.*

Author and Year	Technique	Objective	Pros	Cons
Mazumdar et al. 2021 [13]	Load-offloading method	To optimize the service in suitable time and to support only static IoT devices	Reduces the amount of data to be sent to the cloud	Communication delay as well as high network complexity
Shah et al., 2018 [15]	Empirical multi-agent cognitive method	To attain consecutive transition of IoT APIs	Achieves transparency in the heterogeneity and distribution of IoT APIs	The challenge is in designing the future of connected ecosystems
Bolettieri et al., 2021 [16]	Heuristic algorithm with linear relaxation and rounding techniques	To minimize complexity	High efficiency	Not effective in handling inconsistent traffic demands
Chien et al., 2021 [14]	Convolutional neural network (CNN)	To reduce the time cost during data transmission to the cloud server	Reduced training time and increased prediction accuracy	Use of large amount of data largely impacted the performance
Abbasi et al., 2021 [18]	Genetic algorithm (GA)	To enhance management and processing of IoT and smart grid	Reduced delay and power consumption	Deep neural networks were required to solve the multi-objective optimization problems
Liao et al., 2023 [19]	Reputation- and voting based blockchain consensus (RVC)	To rectify issues such as reducing consensus efficiency	Successful consensus rate, transaction output, and reduced time consumption	Required large number of nodes
Karjee et al., 2022 [20]	Deep neural network (DNN)	To alleviate the issues of partly offloading	Minimizes the overall execution time of each task	Computational capabilities
Mutichiro et al., 2021 [24]	Dynamic pod-scheduling model	To solve the task scheduling problem at the edge	Maximizes node utilization, minimizes the cost, and optimizes the service time	Few constraints in resource capacity (CPU and memory) and total service time

The proposed approach to resource allocation and QoS optimization in mobile edge computing (MEC) using IoT has several practical implications, including its integration into existing MEC systems.

The proposed approach is designed to be seamlessly integrated into existing MEC systems without requiring major modifications or disruptions. It leverages the existing infrastructure, protocols, and interfaces, ensuring compatibility with established MEC frameworks. This integration capability enables service providers to adopt the approach without significant implementation challenges, reducing time-to-market and minimizing operational complexities. The proposed approach follows a modular architecture, allowing for flexible integration with different components of an MEC system. It can be integrated at various layers, including edge nodes, gateway devices, and cloud servers. This modularity enables service providers to selectively deploy and scale the proposed approach based on their specific requirements and existing infrastructure, ensuring a customized integration process. The proposed approach offers flexibility in configuration and adaptation to suit different MEC environments. Service providers can customize the approach based on their specific needs, such as defining resource allocation policies, setting QoS thresholds, and adapting the algorithms to match their network characteristics. This configurability enables seamless integration into diverse MEC ecosystems with varying requirements and constraints.

3. Proposed Methodology

The proposed system combines two machine learning algorithms, namely the ensemble support vector machine (ESVM) and the kernel random forest (KRF), to optimize the network performance. The ESVM algorithm combines multiple support vector machine (SVM) classifiers to improve the classification accuracy and robustness. The KRF algorithm uses decision trees and kernel functions to capture the complex relationships between input features and output labels. The combination of these two algorithms helps to address the limitations of individual algorithms and improve the performance of the system. In addition to the machine learning algorithms, the system also uses crossover-based hunter-prey optimization (CHPO) to optimize the parameters of the algorithms and allocate resources to different applications running on the UMEC network. The proposed method is given in Figure 1.

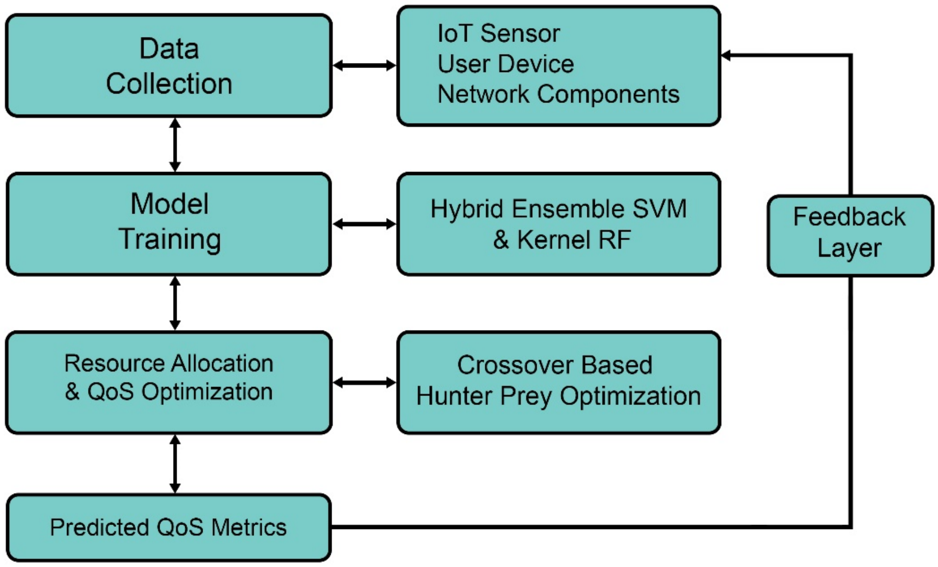

Figure 1. Proposed hybrid ESVM–KRF in IoT-based mobile edge computing.

3.1. Resource Allocation and QoS Optimization Objective Function

The objective of the proposed model is to allocate the available resources in the MEC network and optimize the QoS for the end-users. The objective function can be defined as:

$$\text{Minimize}: \omega \times R_a + (1 - \omega) \times Q_s \tag{1}$$

where R_a is the resource allocation function that assigns the available resources to the different applications running on the MEC network, Q_s is the QoS optimization function that ensures the end-users' requirements are met, and ω is a weighting factor that determines the relative importance of resource allocation and QoS optimization.

3.2. Hybrid Kernel Random Forest and Ensemble SVM Algorithm

The proposed model combines the strengths of the kernel random forest (KRF) and ensemble SVM algorithms to build a hybrid model that can accurately allocate resources and optimize QoS in the MEC network. The KRF algorithm is used to generate multiple decision trees that are then combined to make an ensemble prediction, and the ensemble SVM algorithm is used to classify the available resources into different categories.

3.2.1. Kernel Random Forest Algorithm

The KRF algorithm aims to build an ensemble of decision trees that can accurately predict the QoS performance of the MEC network for different combinations of resources and applications. Each decision tree is constructed using a subset of the training data and a random subset of features to prevent overfitting. The decision trees can be combined using the following equation:

$$K(n) = \frac{1}{D} \times sum(K\_m(n)) \tag{2}$$

where D is the number of decision trees in the ensemble, $K\_m(n)$ is the prediction of the mth decision tree for the input vector n. The KRF algorithm aims to build decision trees that minimize the variance of the prediction error across the ensemble. More specifically, random forest estimators are satisfactory, for all $Y \in [0,1]^l$,

$$P_{U,v}(Y,\Theta_U) = \frac{1}{U}\sum_{b=1}^{U}\left(\sum_{a=1}^{v}\frac{X_a 1_{Y_a \in G_v(Y,\Theta_b)}}{V_v(Y,\Theta_b)}\right) \tag{3}$$

$G_V(Y,\Theta_b)$ Signifies that the cell contains y, designed with iterations Θ_b and dataset L_v, and

$$V_v(Y,\Theta_b) = \sum_{a=1}^{v} 1_{Y_a \in G_v(y,\Theta_b)} \tag{4}$$

The data points contained in the decreasing data are represented as $G_V(Y,\Theta_b)$. The weight $\omega_{a,b.v}(Y)$ as well as each observation X_a is taken into consideration and specified as:

$$\omega_{a,b.v}(Y) = \frac{1_{Y_a \in G_v(y,\Theta_b)}}{V_v(Y,\Theta_b)} \tag{5}$$

Dpending on the number of observations $V_v(Y,\Theta_b)$ is distributed as a common random variable Θ [8]. To enhance random forest techniques and correct for the errors caused by random forest weights, it is reasonable to use finite KRF estimations for all $Y \in [0,1]^l$.

$$\widetilde{U}_{U.v}(Y,\Theta_1\ldots\ldots,\Theta_U) = \frac{\sum_{b=1}^{U}\sum_{a=1}^{v} Y_a 1_{Y_i \in} G_V(Y,\Theta_b)}{\sum_{b=1}^{U} V_v(Y,\Theta_b)} \tag{6}$$

$\widetilde{U}_{U.v}(Y,\Theta_1\ldots\ldots,\Theta_U)$ is equal to the mean of the X_a is dropping in the forest cell containing Y. Each observation is weighted based on how frequently it appears in the forest trees. As a result, when the cell is empty, it does not contribute to the calculation under this system. The similarity of KRF estimations is $\widetilde{U}_{U.v}$.

3.2.2. Ensemble SVM Algorithm

The ensemble SVM algorithm aims to classify the available resources into different categories—"busy" and "idle." The algorithm constructs multiple SVM models using different subsets of the training data and combines them to make an ensemble prediction. A group of classifiers, known as an ensemble, is applied together to classify test samples by combining the results of each individual classifier. Suppose there is an ensemble of n classifiers: $\{c_1, c_{2,} \ldots c_n\}$ and the classifiers are different and their faults are unrelated. As a result, we cannot promise that an SVM will always deliver the best global classification performance on all test examples [25]. Over the years, numerous strategies have been devised for creating a classifier ensemble which involves the combination of multiple classifiers to improve overall performance. In the context of generating a support vector machine (SVM) ensemble, the most crucial aspect is to ensure that each SVM is as distinct as possible from another SVM. This is because a set of similar classifiers may have the same strengths and weaknesses, leading to little improvement in performance. To achieve diversity, representational approaches such as bagging and boosting are often used.

Bagging: Bagging involves training multiple SVMs on different subsets of the training data and then aggregating the outputs of each SVM to make a final prediction. Generally, we have a single training set $TS = \{(Y_n; z_n) | n = 1, 2, \ldots, L\}$. However, K training samples are needed to build the SVM ensemble with K-independent SVMs. Figure 2 shows the ensemble SVM.

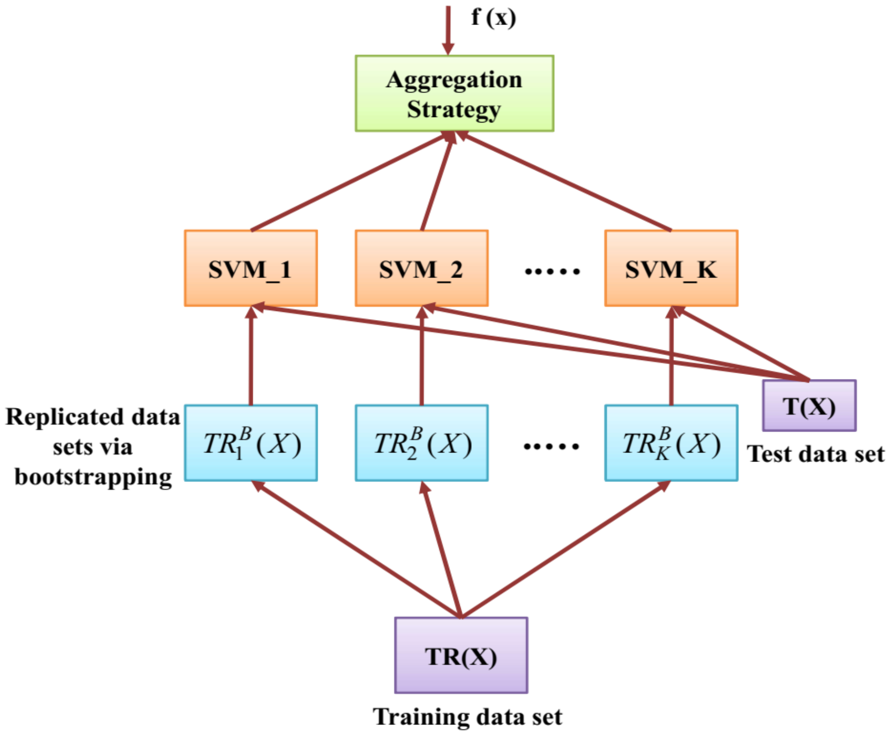

Figure 2. Ensemble SVM Model.

Boosting: Boosting, on the other hand, involves iteratively training SVMs on the same data but with different weightings assigned to misclassified samples. By focusing on representational approaches, we can effectively create a diverse ensemble of SVMs that can improve overall classification performance. We have a training set: $TS = \{(Y_n; z_n)|n = 1, 2, \ldots, L\}$. These training samples are used to train the k^{th} SVM classifier. l represents the whole sample. The decision boundaries can be combined using the following equation:

$$K(n) = \sum w_i{}^n + b_n \tag{7}$$

where: $w_i{}^n$ is the weight vector of the nth SVM model. b_n is the bias term of the n^{th}SVM model.

3.2.3. Hybrid Kernel Random Forest and Ensemble SVM Algorithm

The proposed model combines the kernel random forest and ensemble SVM algorithms to build a hybrid model that can accurately allocate resources and optimize QoS in the MEC network. The hybrid kernel random forest algorithm is used to generate multiple decision trees that are then combined to make an ensemble prediction, and the hybrid ensemble SVM algorithm is used to classify the available resources into different categories. The decision trees can be combined using the following equation:

$$K(n) = \omega \times K_{SVM}(n) + (1 - \omega) \times K_{RF}(n) \tag{8}$$

where: ω is a weighting factor that determines the relative importance of the SVM and KRF algorithms. $K_{SVM}(n)$ is the prediction of the SVM algorithm for the input vector n. $K_{RF}(n)$ is the prediction of the KRF algorithm for the input vector n. The hybrid model can leverage the advantages of both the SVM and KRF algorithms.

3.3. Hunter-Prey Optimization (HPO)

The behavior of predators such as lions, leopards, and wolves as well as prey such as deer and gazelles serves as the basis for hunter–prey optimization (HPO) [21]. The HPO algorithm requires the hunter to look for prey that is distant from the herd since the prey frequently swarm when it is being sought after. The hunters try to catch the prey by moving towards them, while the prey try to evade the hunters by moving away from them. The positions of both the hunters and the prey are updated in each iteration based on their fitness values. The goal of HPO is to find the optimal solution by gradually improving the fitness of both the hunters and the prey. The incorporation of crossover-based strategies in HPO can enhance its ability to search the solution space and find high-quality solutions.

Crossover-Based Hunter–Prey Optimization

Crossover-based hunter–prey optimization (CHPO) is a metaheuristic algorithm inspired by the hunting behavior of predators and prey in nature. The algorithm consists of two types of agents: hunters and prey. Hunters are initialized randomly across the search space and move towards promising regions, while prey move randomly. In CHPO, a crossover operator is used to combine the features of different hunters and create a new offspring with improved characteristics. The crossover operator is applied to two hunters that are selected based on their fitness, i.e., their ability to find promising regions in the search space. The offspring is then evaluated and added to the hunter population. The algorithm uses a mutation operator to introduce diversity in the population and prevent premature convergence. The mutation operator randomly modifies a small subset of features in the hunters' position.

The CHPO algorithm also includes a dynamic weighting scheme that adapts the weight of the crossover operator and mutation operator based on the performance of the algorithm. The weighting scheme aims to balance the exploration and exploitation of the search space and improve the algorithm's convergence speed. In the context of resource allocation and QoS optimization in MEC using IoT, the CHPO algorithm is used to optimize the parameters of the hybrid kernel random forest and ensemble SVM algorithm. The algorithm is used to allocate resources to different applications running on the MEC network, such as CPU, memory, and bandwidth, and optimize the QoS metrics, such as latency, throughput, and energy efficiency.

$$P_{i,j}^t = P_{i,j}^t + Z_c^t P_r \tag{9}$$

where Z_c^t is the weight parameter controlling the influence of P_r on $P_{i,j}^t$. Equation (10) shows that the offspring replaces the original particle. However, to enhance integration accuracy and speed, probe efficiency must decrease over time. Accordingly, at each iteration, the probability of using a crossover on particles is exponentially reduced using the damping parameter λ_r.

$$R_c^t = \lambda_r \, R_c^t \tag{10}$$

$$Y_c^t = \lambda_y \, Y_c^t \tag{11}$$

4. Results and Discussion

The experimental analysis was conducted on a personal computer with Intel® Xeon® 32 Gb RAM, 2.4 GHz on python 3.5 with a source code. The data collection module collects real-time data from various sources such as IoT sensors, user devices, and network components. The performance of the proposed model was evaluated using various performance metrics such as throughput, latency, delay, and energy consumption.

4.1. Parameter Settings

The hyper parameter configuration of the proposed method is depicted in Table 2. For the hybrid ensemble SVM, the kernel function was set to the radial basis function kernel. For the kernel random forest, the number of trees in the forest was set to 50.

In Table 3 we have included four different input configurations, each with varying settings for the number of trees in the hybrid kernel random forest (HKRF) algorithm, the types of kernels used, and the number of iterations in the crossover-based hunter–prey optimization (CHPO) algorithm. The table includes performance metrics such as accuracy, computational efficiency, convergence speed, and resource utilization. The values in the table illustrate the impact of the input configurations on these metrics.

Table 2. Parameter settings.

Parameters	Ranges
Learning rate	0.1
Total number of trees	50
Regularization parameter	1
Maximum depth of each tree	10
Size of population	20
Total number of iterations	50

Table 3. Configurations on the performance of the proposed method.

Number of Trees (HKRF)	Types of Kernels (HKRF)	Number of Iterations (CHPO)	Accuracy	Computational Efficiency	Convergence Speed	Resource Utilization
100	Linear	10	0.85	High	Fast	Moderate
200	RBF	20	0.87	Moderate	Moderate	Moderate
150	Polynomial	15	0.89	High	Slow	High
300	Sigmoid	25	0.82	Low	Moderate	Low

4.2. Performance Measures

The performances of the proposed model are evaluated using various performance metrics such as throughput, latency, delay, and energy consumption [26,27].

- Throughput

Throughput is the amount of data that can be transmitted through the network in a given amount of time.

- Energy consumption

Energy consumption refers to the amount of energy used by the devices or networks to perform a specific task.

- Delay

Delay refers to the time taken for a packet or data to travel from the source to the destination.

- Latency

Latency is defined as the time taken between initiating a network request as well as receiving a response.

4.3. Performance Evaluation

To validate the mobile edge computing, we compare ensemble SVM, kernel random forest, with proposed hybrid ensemble SVM and kernel random forest for the performance metrics such as delay, throughput, and energy consumptions.

The performance analysis of the proposed model is presented in Figures 3–6. Figure 3 demonstrates the performance of the proposed approach in terms of throughput, where it is compared with ensemble SVM [28] and kernel random forest [29]. The results show that the proposed approach achieves a higher throughput rate with 595 mbit/s than the other two algorithms. This suggests that the proposed approach can allocate resources more efficiently, resulting in higher data transmission rates across the network. Figure 4 shows the performance analysis of average energy consumption. This indicates that the proposed approach can allocate resources more efficiently, resulting in reduced energy consumption with 9.4 mJ by the devices and network. The performance analysis of the proposed model in terms of latency and delay is demonstrated in Figures 5 and 6, respectively. This implies that the proposed model can effectively allocate resources and optimize the QoS [30], resulting in improved network performance.

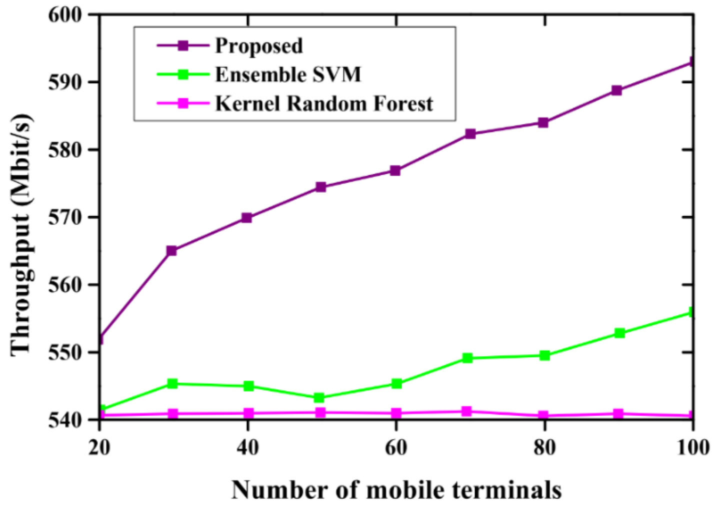

Figure 3. Performance analysis of throughput.

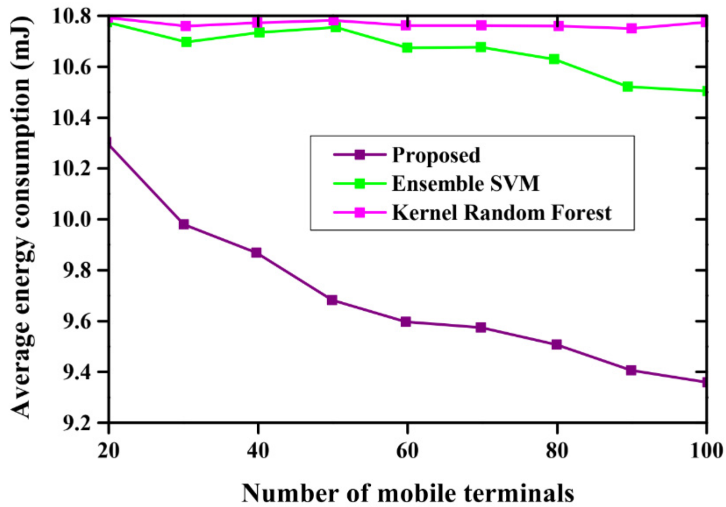

Figure 4. Performance analysis of average energy consumption.

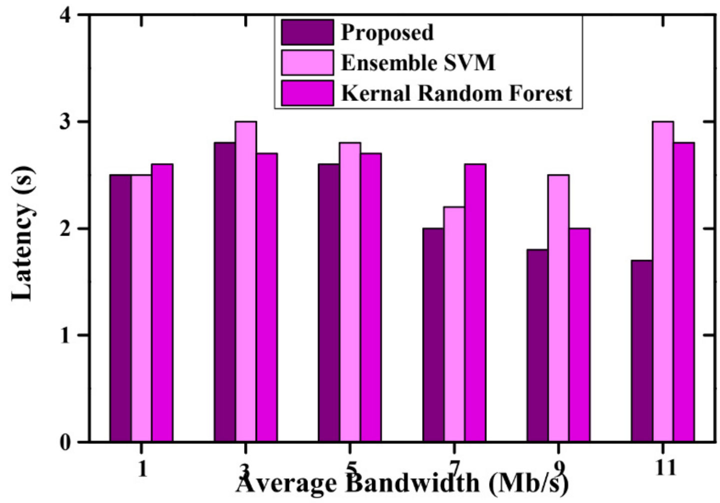

Figure 5. Performance analysis of latency.

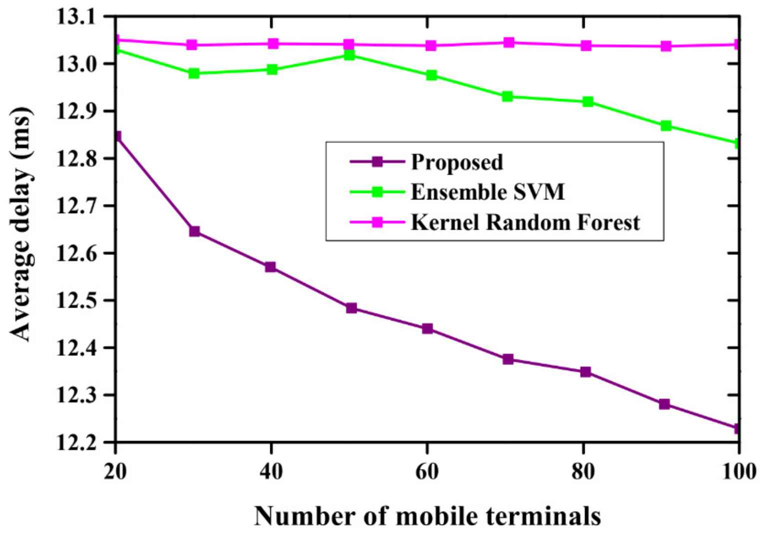

Figure 6. Performance analysis of delay.

5. Conclusions

This paper has presented a novel approach for resource allocation and quality of service (QoS) optimization in ubiquitous mobile edge computing (UMEC) using the internet of things (IoT). The proposed approach combines the hybrid kernel random forest (HKRF) and ensemble support vector machine (ESVM) algorithms with crossover-based hunter–prey optimization (CHPO) to optimize the QoS metrics, such as latency, throughput, and energy efficiency, while allocating resources to different applications running on the UMEC network. The experimental results have demonstrated that the proposed approach outperforms other state-of-the-art algorithms in terms of QoS metrics and resource allocation efficiency. It achieves a higher throughput rate of 595 mbit/s compared to the other evaluated algorithms, indicating improved data transmission rates. Additionally, it reduces energy consumption by devices and the network to 9.4 mJ, showcasing enhanced energy efficiency.

This paper has introduced a unique combination of HKRF, ESVM, and CHPO algorithms to tackle the resource allocation and QoS optimization challenges in UMEC systems. The proposed approach effectively optimizes latency, throughput, and energy efficiency, enhancing the overall network performance and user experience. Extensive experiments have been conducted to evaluate the performance of the proposed approach, demonstrating its superiority over other algorithms. The paper highlights the practical implications of the proposed approach, such as its integration into existing MEC systems and the potential additional benefits beyond the evaluated metrics. However, it is important to acknowledge some limitations of this work. Firstly, the experimental analysis was conducted on a specific hardware configuration, and the results may vary in different settings. Secondly, the proposed approach relies on the accurate prediction of QoS metrics, which, in turn, depends on the quality and availability of input data. Further improvements in data collection and prediction accuracy could enhance the system's performance. Lastly, while the proposed approach addresses resource allocation and QoS optimization, other aspects such as security and fault tolerance could be explored in future research.

Author Contributions: Conceptualization, S.B.K. and M.A.; methodology, M.A.M.; validation, S.B.K., M.A.M. and M.A.; formal analysis, S.B.K.; investigation, M.A.M.; data curation, M.A.; writing—original draft preparation, S.B.K. and M.A.M.; writing—review and editing, M.A.; visualization, S.B.K.; supervision, M.A.; project administration, M.A.M. All authors have read and agreed to the published version of the manuscript.

Funding: Researchers Supporting Project number (RSP2023R446), King Saud University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Shahraki, A.; Ohlenforst, T.; Kreyß, F. When machine learning meets network management and orchestration in Edge-based networking paradigms. *J. Netw. Comput. Appl.* **2022**, *212*, 103558. [CrossRef]
2. Chen, P.; Liu, H.; Xin, R.; Carval, T.; Zhao, J.; Xia, Y.; Zhao, Z. Effectively Detecting Operational Anomalies in Large-Scale IoT Data Infrastructures by Using A GAN-Based Predictive Model. *Comput. J.* **2022**, *65*, 2909–2925. [CrossRef]
3. Adil, M. Congestion-free opportunistic multipath routing load balancing scheme for the Internet of Things (IoT). *Comput. Netw.* **2021**, *184*, 107707. [CrossRef]
4. Shakarami, A.; Ghobaei-Arani, M.; Shahidinejad, A. A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective. *Comput. Netw.* **2020**, *182*, 107496. [CrossRef]
5. Li, J.; Deng, Y.; Sun, W.; Li, W.; Li, R.; Li, Q.; Liu, Z. Resource Orchestration of Cloud-Edge–Based Smart Grid Fault Detection. *ACM Trans. Sen. Netw.* **2022**, *18*, 1–26. [CrossRef]
6. Wang, Y.; Han, X.; Jin, S. MAP based modeling method and performance study of a task offloading scheme with time-correlated traffic and VM repair in MEC systems. *Wirel. Netw.* **2022**, *29*, 47–68. [CrossRef]
7. Liu, J.; Lin, C.H.R.; Hu, Y.C.; Donta, P.K. Joint beamforming, power allocation, and splitting control for SWIPT-enabled IoT networks with deep reinforcement learning and game theory. *Sensors* **2022**, *22*, 2328. [CrossRef] [PubMed]
8. Scornet, E. Random forests and kernel methods. *IEEE Trans. Inf. Theory* **2016**, *62*, 1485–1500. [CrossRef]
9. Kim, H.C.; Pang, S.; Je, H.M.; Kim, D.; Bang, S.Y. Constructing support vector machine ensemble. *Pattern Recognit.* **2003**, *36*, 2757–2767. [CrossRef]
10. Yu, S.; Gong, X.; Shi, Q.; Wang, X.; Chen, X. EC-SAGINs: Edge-computing-enhanced space–air–ground-integrated networks for the internet of vehicles. *IEEE Internet Things J.* **2021**, *9*, 5742–5754. [CrossRef]
11. Ai, Z.; Zhang, W.; Li, M.; Li, P.; Shi, L. A smart collaborative framework for dynamic multi-task offloading in IIoT-MEC networks. *Peer-Peer Netw. Appl.* **2023**, *16*, 749–764. [CrossRef]
12. Sood, S.K. Smart vehicular traffic management: An edge cloud-centric IoT-based framework. *Internet Things* **2021**, *14*, 100140.
13. Mazumdar, N.; Nag, A.; Singh, J.P. Trust-based load-offloading protocol to reduce service delays in fog-computing-empowered IoT. *Comput. Electr. Eng.* **2021**, *93*, 107223. [CrossRef]
14. Chien, W.C.; Huang, Y.M. A lightweight model withspatial–temporal correlation for cellular traffic prediction in the Internet of Things. *J. Supercomput.* **2021**, *77*, 10023–10039. [CrossRef]
15. Shah, V.S. Multi-agent cognitive architecture-enabled IoT applications of mobile edge computing. *Ann. Telecommun.* **2018**, *73*, 487–497. [CrossRef]
16. Bolettieri, S.; Bruno, R.; Mingozzi, E. Application-aware resource allocation and data management for MEC-assisted IoT service providers. *J. Netw. Comput. Appl.* **2021**, *181*, 103020. [CrossRef]
17. Hashash, O.; Sharafeddine, S.; Dawy, Z.; Mohamed, A.; Yaacoub, E. Energy-aware distributed edge ML for mhealth applications with strict latency requirements. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 2791–2794. [CrossRef]
18. Abbasi, M.; Mohammadi-Pasand, E.; Khosravi, M.R. Intelligent workload allocation in IoT–Fog–cloud architecture towards mobile edge computing. *Comput. Commun.* **2021**, *169*, 71–80. [CrossRef]
19. Liao, Z.; Cheng, S. RVC: A reputation and voting based blockchain consensus mechanism for edge computing-enabled IoT systems. *J. Netw. Comput. Appl.* **2023**, *209*, 103510. [CrossRef]
20. Karjee, J.; Naik, P.; Anand, K.; Bhargav, V.N. Split computing: DNN inference partition with load balancing in IoT-edge platform for beyond 5G. *Meas. Sens.* **2022**, *23*, 100409. [CrossRef]
21. Elshahed, M.; El-Rifaie, A.M.; Tolba, M.A.; Ginidi, A.; Shaheen, A.; Mohamed, S.A. An Innovative Hunter-Prey-Based Optimization for Electrically Based Single-, Double-, and Triple-Diode Models of Solar Photovoltaic Systems. *Mathematics* **2022**, *10*, 4625. [CrossRef]
22. Chen, W.; Zhu, Y.; Liu, J.; Chen, Y. Enhancing Mobile Edge Computing with Efficient Load Balancing Using Load Estimation in Ultra-Dense Network. *Sensors* **2021**, *21*, 3135. [CrossRef] [PubMed]
23. Poryazov, S.A.; Saranova, E.T.; Andonov, V.S. Overall Model Normalization towards Adequate Prediction and Presentation of QoE in Overall Telecommunication Systems. In Proceedings of the 2019 14th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), Nis, Serbia, 23–25 October 2019; pp. 360–363.
24. Mutichiro, B.; Tran, M.N.; Kim, Y.H. QoS-based service-time scheduling in the IoT-edge cloud. *Sensors* **2021**, *21*, 5797. [CrossRef]
25. Xu, J.; Ma, R.; Stankovski, S.; Liu, X.; Zhang, X. Intelligent Dynamic Quality Prediction of Chilled Chicken with Integrated IoT Flexible Sensing and Knowledge Rules Extraction. *Foods* **2022**, *11*, 836. [CrossRef]
26. Jiang, H.; Dai, X.; Xiao, Z.; Iyengar, A.K. Joint Task Offloading and Resource Allocation for Energy-Constrained Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2022**, *22*, 4000–4015. [CrossRef]

27. Banoth, S.P.R.; Donta, P.K.; Amgoth, T. Target-aware distributed coverage and connectivity algorithm for wireless sensor networks. *Wirel. Netw.* **2023**, *29*, 1815–1830. [CrossRef]
28. Saab, S., Jr.; Saab, K.; Phoha, S.; Zhu, M.; Ray, A. A multivariate adaptive gradient algorithm with reduced tuning efforts. *Neural Netw.* **2022**, *152*, 499–509. [CrossRef]
29. Rani, S.; Babbar, H.; Srivastava, G.; Gadekallu, T.R.; Dhiman, G. Security Framework for Internet of Things based Software Defined Networks using Blockchain. *IEEE Internet Things J.* **2022**, *10*, 6074–6081. [CrossRef]
30. Liu, K.; Wang, P.; Zhang, J.; Fu, Y.; Das, S.K. Modeling the Interaction Coupling of Multi-View Spatiotemporal Contexts for Destination Prediction. In Proceedings of the 2018 SIAM International Conference on Data Mining, San Diego, CA, USA, 3–5 May 2018.