Minimax Theory and its Applications
Volume 11 (2026), No. 3s, 734 to 754

CCO-TA: Collaborative Computation Offloading and Task Allocation Model for Vehicular Cloud Environment using 6G Assisted Volunteer Edge Computing and mmWave Technologies

K.Rajeswari[1], Dr.B.Arun Kumar[2]

[1]Research scholar Department of Computer Science and Engineering Faculty of Engineering Karpagam Academy of Higher Education Coimbatore, Tamil Nadu, India, Email: rajeswarikamalk@gmail.com
[2]Professor and Head Department of Artificial Intelligence and Data Science Faculty of Engineering Karpagam Academy of Higher Education Coimbatore, Tamil Nadu, India, Email: arunkumar.oct06@gmail.com

Abstract
Computation offloading and task allocation in IoV (Internet of Vehicles) are vital for optimizing resource utilization and enhancing computational efficiency by distributing computing tasks among vehicles and edge/cloud infrastructure. This approach is essential for supporting resource-intensive applications, such as real-time traffic management,autonomous driving, and collision avoidancewhile minimizing latency and maximizing system performance in dynamic vehicular environments.However, existing works are majorly limited with resource constraints, energy efficiency and reliability of IoV network. To alleviate these issues, we enhance the computational offloading challenge in vehicular cloud environments through the adoption of Volunteer Edge Computing (VEC), millimeter Wave (mmWave), and 6G technologies. VEC resolves computation issues, mmWave improves communication efficiency, and 6G technology reduces communication overhead. Our framework performs Collaborative Computation Offloading and Task Allocation addressed as CCO-TA. Initially, AVs are clustered using the Density Peak-based K-Nearest Neighbor (DP-KNN) algorithm based on task emergency, vehicle mobility, and density. Each cluster selects a Cluster Head (CH) with higher computation capability for VEC execution, while less critical tasks are offloaded to RSUs. RSUs perform collaborative offloading among themselves within their range using the Binary Corona Virus Optimization (BCVO) algorithm based on vehicle mobility. Additionally, load prediction in RSUs is accomplished using the Deep Learning (DL) algorithm Load Prediction Network (LPN), comprising LSTM encoder, convolution layers, and LSTM decoder for task allocation. Communication among vehicles (V2V) is managed by mmWave technology, while 6G communication oversees communication across the entire environment.The simulation results showcase that the proposed CCO-TA approach outperforms other methods in terms of system utility, resource constraints and efficiency, particularly under heavy system workloads.

Keywords: Collaborative Computation Offloading, Task Allocation, Volunteer Edge Computing, AV Clustering, mmWave communication, IoV.

Introduction

By equipping cars with computational power and communication capabilities, the Internet of Vehicles (IoV) has completely changed transportation networks. cars in the Internet of Vehicles (IoV) can communicate with infrastructure, the cloud, and other cars by means of sensors, actuators, and computing devices installed in them [1][2]. Numerous creative applications are made possible by this integrated environment, such as autonomous driving, real-time traffic management, collision avoidance, and remote diagnostics. However,effective management of computing resources is necessary to fully realise the promise of IoV. In IoV contexts, job allocation and computation offloading are critical to maximising resource use and improving computing performance [3][4]. Computation offloading is the process of moving computationally demanding jobs from devices with limited resources, such cars, to servers in the cloud or on the edge. Vehicles can save energy, speed up processing, and enhance system performance by assigning work to outside resources [4]-[7]. On the other hand, task allocation chooses how to divide up work among available resources tonai ah maximise system performance. Allocating tasks effectively guarantees that computing resources are used as efficiently as possible, reducing latency and increasing throughput [8]-[10].

Computation offloading and job allocation in IoV encounter several difficulties and constraints. Resource limitations are one of the main issues [11][12]. IoV vehicles usually have constrained memory, battery life, and processing power. This limitation limits the kinds of tasks that may be performed locally and forces demanding processes to be offloaded to external servers [13]. Furthermore, because IoV vehicles run on batteries, energy efficiency is a major challenge [14]. The workload on onboard processors can be decreased by offloading processes to distant servers, which can aid in energy conservation.Moreover, a major problem in IoV contexts is reliability. Because they are always in motion, vehicles may experience unstable network connections, which can cause delays or breakdowns in communication [15]. This instability can interfere with job allocation and compute offloading, which will impact user experience and system performance. Furthermore, IoV's dynamic nature adds even more complication. Changes in vehicle participation can lead to variations in network topology and resource availability. It could be difficult for conventional task allocation algorithms to effectively adjust to these dynamic shifts [16].

One more drawback of current methods is their inability to scale. The computational and communication overhead related to job allocation and computation offloading rises as the number of vehicles in IoV networks does [17]. Performance deterioration and scalability difficulties may result from traditional centralised systems being overloaded with data and computing workloads. Collaborative computation offloading shows promise as a way around these restrictions [18] Using the combined power of edge servers, cloud computing, and automobiles, collaborative offloading maximises resource allocation and work distribution. Collaborative offloading can alleviate resource limitations, boost energy efficiency, increase dependability, and efficiently scale to meet expanding network capacities by dividing up work across several organisations [19][20]. We will examine collaborative computation offloading in more detail and consider how it might be used to overcome the aforementioned constraints in Internet of Vehicles scenarios in the sections that follow. To assess if collaborative offloading improves system performance and efficiency in Internet of Vehicles (IoV) networks, we will look at current ways, suggest new ones, and carry out experimental evaluations.

Research Contribution

The proposed work's novel contributions can be summed up as follows:

Introduction of Advanced Technologies: The adoption of VEC, mmWave, and 6G technologies represents a novel approach to addressing computational offloading challenges in vehicular cloud environments. VEC resolves computation issues by leveraging edge resources, mmWave improves communication efficiency with high bandwidth and low latency capabilities, and 6G technology reduces communication overhead, collectively enhancing system performance and efficiency.

Framework for Collaborative Computation Offloading and Task Allocation (CCO-TA): The development of the CCO-TA framework introduces a comprehensive solution for collaborative task allocation among vehicles and edge/cloud infrastructure. By clustering AVs based on task emergency, vehicle mobility, and density using the DP-KNN algorithm, selecting CHs with superior computation capabilities, and performing collaborative offloading among RSUs using the BCVO algorithm, the framework optimizes task allocation and resource utilization.

Integration of Machine Learning for Load Prediction: The incorporation of machine learning techniques, such as the DL algorithm LPN, facilitates accurate load prediction in RSUs. By utilizing LSTM encoder, convolution layers, and LSTM decoder, the LPN enables efficient task allocation by predicting RSU load, thereby improving system efficiency and performance.

These contributions collectively represent a significant advancement in computational offloading and task allocation within vehicular cloud environments, offering novel solutions to address existing challenges and enhance system capabilities.

Paper Organization

The paper remainder is structured as results: Fragment II presents a review of existing literature. In Fragment III, we elaborate on the suggested work, providing insights into the methodology through the presentation of algorithms, pseudocodes, and diagrams. Fragment IV focuses on experimental analysis, covering aspects such as simulation setup, and comparative analysis. Finally, Section V concludes by summarizing the proposed CCO-TA framework.

Literature Survey

In recent times, edge computing has appeared as a viable solution, which address the escalating demands of computation-intensive and latency-sensitive applications. Among the pivotal strategies within edge computing, computation offloading stands out as a crucial technique, essential for enhancing system performance and resource utilization.This review provides a comprehensive analysis of five significant publications that have contributed to advancing computation offloading methods across diverse edge computing scenarios.

Yang et al. [21] introduced a novel multi-task learning technique tailored specifically for compute offloading, in the field of multi-access edge computing (MEC). By harnessing the inherent synergies among various related tasks, the proposed approach significantly enhances prediction accuracy and resource allocation efficiency. This innovative fusion of machine learning methodologies with edge computing holds promise for optimizing compute offloading decisions within the dynamic and heterogeneous landscape of MEC systems.Zhang et al. [22] addresses the challenge of effective task allocation for compute offloading within automotive environments, this study focuses on vehicular edge computing (VEC). Offering a tailored solution, the study proposes dynamic task allocation algorithms aimed at optimizing resource utilization and reducing latency for automotive applications. By considering variables such as vehicle movement patterns and network conditions, the suggested approach stands to make substantial strides in enhancing compute offloading optimization within mobile and highly dynamic edge computing scenarios.Jiang et al. [23] presented a comprehensive methodology for collaborative offloading of task and allocation of resource, particularly focusing on constrained of energy scenarios within MEC environments. The proposed approach enhances task placement and allocation of resource strategies to decrease consumption of energy while meeting application requirements, accounting for the energy limitations of mobile devices. By addressing the critical challenge of energy efficiency in edge computing contexts with limited resources, this paper offers valuable insights into optimizing resource utilization.Tan et al. [24] introduced an energy-efficient tactic for cooperative resource allocation and job unloading in collaborative edge computing environments utilizing orthogonal frequency-division multiple access (OFDMA). Leveraging the collaborative nature of edge resources, the proposed method maximizes task offloading decisions to minimize energy consumption and improve overall system performance. This research significantly contributes to enhancing energy efficiency within collaborative edge computing scenarios. Tang et al. [25] introduced a hybrid paradigm for offloading of computation and allocation of resource, aimed at addressing task overflow scenarios in mobile-edge computing (MEC). By dynamically adapting offloading decisions to network conditions and edge resource availability, the proposed technique effectively mitigates task overflow while optimizing resource utilization. This study provides valuable insights into managing resource constraints in dynamic MEC environments.

Alfakih et al. [26] presented for task offloading and resource allocation in MEC, a novel deep reinforcement learning approach based on SARSA (State-Action-Reward-State-Action). By framing the problem as a sequential decision-making process, the proposed method leverages deep reinforcement learning to learn optimal offloading decisions, considering resource constraints and network dynamics. In dynamic MEC contexts, the integration of deep reinforcement learning with SARSA offers a promising avenue for flexible and efficient offloading strategies.Zhou et al. [27] introduced a framework for fairness-aware task offloading in cooperative mobile-edge computing (MEC) systems, focusing on equitable resource allocation. The proposed method optimizes system performance while maintaining fairness by integrating fairness criteria into the offloading decision process. Ensuring fairness and inclusiveness in MEC systems is paramount for delivering a superior user experience, and our research contributes to nurturing these essential aspects.Yan et al. [28] employed to tackle the task offloading challenge within MEC environments featuring a diverse task graph topology, a deep reinforcement learning strategy. Leveraging deep reinforcement learning methods and graph-based representation of tasks, the proposed approach dynamically schedules tasks and allocates resources to minimize latency and enhance throughput. This research extends the applicability of reinforcement learning techniques to complex task structures within MEC contexts, thereby advancing the field's understanding and capabilities in optimizing task offloading.Wang et al. [29] introduced an allocation of optimal energy and offloading of task policy tailored for wireless-powered MEC devices, focusing on energy conservation. The proposed approach enhances system throughput while adhering to energy constraints by concurrently optimizing energy allocation from wireless power transfer and task offloading decisions. By addressing the critical challenge of energy management within resource-constrained MEC systems, our research paves the way for the progress of self-sustaining and environmentally approachable edge computing solutions.Zhao et al. [30] proposed for resource allocation and task offloading targeting time-sensitive services in MEC systems, an energy-aware methodology. The suggested strategy efficiently offloads tasks and dynamically allocates resources to minimize energy consumption while meeting service deadlines, considering both energy usage and service latency requirements. In MEC scenarios, our study contributes to enhancing energy efficiency without compromising service quality for time-sensitive applications.

Zhang et al. [31] introduced a dynamic approach to resource allocation and task offloading tailored specifically for dense cloud radio access network (RAN) scenarios. By dynamically adjusting offloading decisions based on resource availability and network conditions, the proposed method effectively minimizes latency and optimizes system throughput. This study underscores the importance of flexibility within MEC systems and provides insights into addressing challenges associated with dense RAN deployments. Dai et al. [32] showed an edge intelligence framework designed for allocation of resource and offloading of computation within future 5G and beyond networks, beside a primary focus on energy efficiency. By harnessing edge intelligence capabilities such as machine learning and predictive analytics, the proposed method maximizes offloading decisions to reduce energy consumption while maintaining service quality. The research showcases the potential of edge intelligence in developing energy-efficient MEC solutions for the forthcoming wireless networks. Yu et al. [33] introduced a collaborative paradigm for offloading of task and allocation of resource, aiming to explore the advantages of integrating UAVs with MEC. By leveraging UAVs as mobile edge servers, the proposed solution enhances offloading decisions, considering UAV mobility and resource constraints. The research sheds light on the potential of UAV-enabled MEC to enhance coverage and connectivity in dynamic environments. Tong et al. [34] addressed the dynamic nature of MEC environments, an adaptive technique for resource allocation and computation offloading. By continuously monitoring system parameters and user requirements, the proposed method dynamically adjusts offloading decisions to optimize resource efficiency and meet application demands. This study underscores the importance of flexibility within MEC systems to accommodate changing workloads and network conditions effectively. Li et al. [35] introduced a comprehensive methodology for service allocation and computation offloading within MEC environments. By optimizing offloading decisions to enhance system performance and user satisfaction, the proposed technique considers network conditions, resource availability, and service requirements. This study contributes to the design of efficient mechanisms for service allocation in MEC systems, aiming to deliver a seamless user experience.

Chu et al. [36] presented a utility-based approach for selection of online optimal service, allocation of resource, and offloading of task, a MEC. By dynamically evaluating the value of available services and resources, the proposed method maximizes user satisfaction and efficiently utilizes edge resources. This research advances the agility and responsiveness of MEC systems, ensuring the performance in dynamically changing environments. Yuan et al. [37] presented a profit-maximized strategy for resource allocation in distributed cloud and edge computing systems, specifically targeting collaborative computation offloading. By optimizing offloading decisions to maximize overall system profitability while preserving service quality, the proposed technique considers the profit objectives of both service providers and customers. The research underscores the potential of cooperation among multiple stakeholders to enhance financial gains within edge computing ecosystems. Chang et al. [38] introduced a dynamic approach to resource allocation and computation offloading, tailored to address the unique requirements of Internet of Things (IoT) fog computing systems. The proposed method adapts resource allocation and offloading decisions to maximize system performance and energy efficiency, accounting for factors such as device heterogeneity and fluctuating workload intensities. By enhancing the scalability and responsiveness of IoT fog computing environments, this research contributes to advancing the capabilities of such computing environments. Chai et al. [39] introduced a collaborative approach to convolutional neural network (CNN) layer scheduling, resource allocation, and task offloading in computer systems. By simultaneously optimizing CNN layer scheduling and offloading decisions, the proposed method efficiently utilizes available resources to minimize latency and enhance overall system performance. The research underscores the importance of collaborative efforts among computing entities to collectively improve system performance. Zhang et al. [40] extended beyond resource allocation and computation offloading to address service caching, offering a comprehensive methodology for enhancing mobile edge computing (MEC) systems. By integrating offloading decisions, resource allocation, and service caching strategies, the proposed solution enhances service availability and reduces latency for mobile users. This research illustrates how caching mechanisms and offloading techniques can synergistically improve user experience within MEC environments.

Overall, these research articles represent significant progress in task offloading, resource allocation, and service selection strategies within edge computing environments. By addressing critical challenges such as profit maximization, dynamic resource management, online optimization, and collaborative computing, these studies pave the way for more efficient and scalable edge computing solutions. Their findings contribute to meeting the diverse requirements of modern users and applications in the rapidly evolving landscape of edge computing.

CCO-TA Framework

With the continuation of previous work, we improve the computational offloading problem in the vehicular cloud environment by adopting Volunteer Edge Computing (VEC), millimeter Wave (mmWave), and 6G technology. The VEC was utilized to resolve the computation issues, mmWave to improve the communication

efficiency, and 6G technology to reduce the communication overhead. The entities involved in this work are Autonomous Vehicles (AVs), Road Side Units (RSUs), 6G Base Station (BS), RSU Manager, and Cloud Server (CS). Figure 1 demonstrates the overall architecture of proposed CCO-TA framework.

AV's Clustering

At first, the AVs in the environments are clustered using **Density Peak K-Nearest Neighbors (DP-KNN)** algorithm based on task emergency, vehicle mobility, and density respectively. For every cluster, Cluster Head (CH) is selected with higher computation capability and the CH is considered for VEC. To be more clear, the highly emergency tasks are executed by the CH and the other tasks are offloaded to the RSU.

Density Peak Metric as per KNN and Shannon Entropy: The current DPC method still exhibits imperfections. A notable drawback lies in the necessity to manually specify the cut-off distance $\mathbb{d}_{\mathbb{c}}$ value in advance, which significantly influences the resulting clustering outcomes. To address this limitation, a novel approach is proposed, leveraging the computation of density metrics using Shannon entropy and K-nearest neighbours. In various domains such as classification, clustering, and local outlier identification, among others, KNN are commonly employed to measure an instance's local neighbourhood. The objective of this method is to identify a sample's KNN among $\mathfrak{n}$ samples. Typically, Euclidean distance is utilized to determine the distances between points. Enable $\mathrm{KNN(i)}$ denote the set of closest neighbours of a point i. This can be circulated as:

$$\mathrm{KNN(i)} = \left\{ \mathrm{j} \middle| \mathbb{d}_{\mathrm{ij}} \leq \mathbb{d}_{\mathrm{i,kth}} \right\}, \qquad (1)$$

Here, $\mathrm{kth(i)}$ is the k-th closest neighbour of i, and $\mathbb{d}(\mathrm{x_i}, \mathrm{x_j})$ represents the Euclidean distance between $\mathrm{x_i}$ and $\mathrm{x_j}$. The term KNN pertains to local regions evaluated by KNN, yet in practice, it denotes a circular or spherical region with a radius $\Re = \mathbb{d}_{\mathrm{i,kth(i)}}$. Consequently, datasets featuring clusters with non-spherical distributions are not well-suited for the KNN-based technique. Hence, when dealing with datasets containing clusters of diverse geometries, these methods typically yield suboptimal clustering results.

Shannon entropy quantifies the degree of molecular activity, where higher values indicate increased system instability and lower values correspond to greater stability. Denoted by $\mathbb{S}(\mathbb{x})$, Shannon entropy can be computed using:

$$\mathbb{S}(\mathbb{x}) = -\sum_{\mathrm{i}=0}^{\mathfrak{n}} \wp_{\mathrm{i}} \log_2(\wp_{\mathrm{i}}) \qquad (2)$$

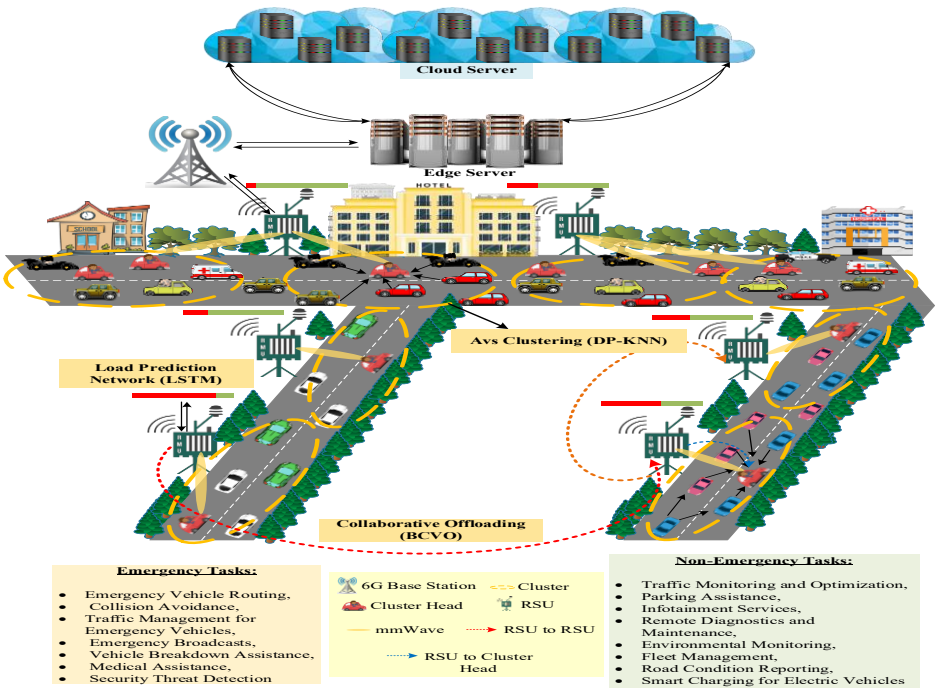


Fig.1 Overall Architecture of Proposed CCO-TA Framework

Here, i denotes the set of items, and $\wp_i$ represents the probability that object i will appear in $\wp_i$. When evaluating the distance among clusters, a lesser value of $\mathbb{S}(\mathbb{x})$ indicates a better clustering outcome. Therefore, to calculate the data point density using the KNN technique, Shannon entropy is integrated. This integration accounts for both the position of the data point and the distance metric, enhancing the density computation. Conversely, the decision graph is determined by multiplication with $\wp_i$. Selecting the optimal clustering centers is facilitated by a higher value of $\wp_i$. Hence, the mutual form of Shannon entropy is utilized. Discrepancies in the metrics for $\wp_i$ and λ_i could lead to them playing diverse roles in the conclusion graph computation. Hence, normalizing $\wp_i$ and λ_i is crucial. The precise computation procedure is as follows: Firstly, the local density of data point x_i is computed.

$$\Pi_i^{'} = \sum_{j=1,j\neq i}^{n} \frac{1}{\left\|x_i - x_j\right\|}, \qquad (3)$$

Where the data points are denoted as x_i and x_j, and the thickness of data point x_i is represented by $\Pi_i^{'}$. Subsequently, the thickness of data point x_i is normalized, and the resulting regularized density is denoted by Π_i.

$$\Pi_i = \frac{\Pi_i^{'}}{\sum_{i=1}^{n} \Pi_i^{'}} \qquad (4)$$

Ultimately, the density measure employing the KNN approach is given by,

$$\wp_i = -\frac{1}{(1/K)\sum_{i\in KNN} \Pi_i \log(\Pi_i)} \qquad (5)$$

Normalizing λ_i is also necessary to ensure consistency in the measurements of $\wp_i$ and λ_i.

Tissue-Like P System with Lively Membranes for AmendedDPC: KST-DPC system, a tissue-like P scheme with lively membranes for DPC, is proposed as follows. As declared earlier, let us consider $\mathfrak{X} = \{\mathfrak{X}_1, \mathfrak{X}_2, \ldots, \mathfrak{X}_\cap\}$ representing the dataset containing $\cap$ data points. The Euclidean distance amongrespectivelydata points pair in the dataset is computed and kept as a matrix before any specific DPC algorithm calculations are performed. Figure 3 illustrates the basic configuration of this P system.

During system initialization, object ϖ is in membrane $\cap+1$, indicating the absence of any object, while objects $x_i, z_1, z_2, .., z_\cap$ are in membrane i for $1 \leq i \leq \cap$. Initially, using the formula $\mathfrak{r}_1 = \left\{[x_i, z_1, z_2, .., z_\cap]_i \rightarrow \left[\mathbb{d}_{i,1}^{\Pi_i,1}\mathbb{d}_{i,2}^{\Pi_i,2} \ldots..\mathbb{d}_{i,\cap}^{\Pi_i,\cap}\right]_i | 1 \leq i \leq \cap\right\}$, where x_j for $1 \leq j \leq \cap$ is represented as $z_1, z_2, .., z_\cap$. The distance matrix, or similarity matrix, $\mathbb{D}$, is where the findings are stored.

$$\mathbb{D}_{\cap\cap} = \begin{pmatrix} \Pi_{11} & \Pi_{12} & \cdots & \Pi_{1\cap} \\ \Pi_{21} & \Pi_{22} & \cdots & \Pi_{2\cap} \\ \Pi_{\cap 1} & \Pi_{\cap 2} & \cdots & \Pi_{\cap\cap} \end{pmatrix} \qquad (6)$$

Initially, the $\wp$ system contains $\cap+1$ membranes. Once distances are determined, the elements $x_i, \mathbb{d}_{i,1}^{\Pi_i,1}\mathbb{d}_{i,2}^{\Pi_i,2} \ldots..\mathbb{d}_{i,\cap}^{\Pi_i,\cap}$, are arranged within each membrane for $1 \leq i \leq \cap$. Following this, the thicknesses of the data points are designed using the formula $\mathfrak{r}_2 = \left\{\left[\mathbb{d}_{i,1}^{\Pi_i,1}\mathbb{d}_{i,2}^{\Pi_i,2} \ldots..\mathbb{d}_{i,\cap}^{\Pi_i,\cap}\right]_i \rightarrow \left[\Pi_i^{'}\right]_i | 1 \leq i \leq \cap\right\}$. Subsequently, values of $\wp_i$, λ_i, and Ψ_i are computed and inserted into membrane i for $1 \leq i \leq \cap$ using the send-in and send-out communication rules. The number of clusters can then be determined based on the sorted results for $1 \leq i \leq \cap$. Membrane $\cap+1$ is subdivided into κ membranes using the active membrane rule, as illustrated in Figure 4. Membranes $\cap+1$ to $\cap+\kappa$ represent the locations of the κ cluster centers. In the final step, the outstanding data points are partitioned, with respectively one placed into a membrane containing the nearest cluster center. This completes the clustering process.

Time Complexity Examination of KST-DPC: In the tissue-like $\wp$ system's cells, parallel computation of tasks is standard practice. The production of the dissimilarity matrix involves $\cap$ compute steps when implemented in parallel. Specifically, one computing step suffices for generating data point densities. Determining the final density involves $\wp_i$ computing processes, which in turn encompasses κ stages for calculating λ_i and $\cap$. The

computation of Ψ_i occurs in a single step. Sorting $\Psi_i$employs log steps for $1 \leq i \leq \cap.$, and the final clustering necessitates an additional step. Considering these steps, the overall temporal complexity of KST-DP amounts to $\cap +1 + \kappa + \cap +1 + \cap \log \cap +$. In contrast, DP-KNN exhibits a temporal complexity of $G(\cap^2)$. Notably, KST-DP shifts time complexity to space complexity, resulting in a lower time complexity compared to DP-KNN.

Collaborative Offloading

Based on the vehicle mobility, the RSU then performs collaborative offloading among themselves within its range using **Binary Corona Virus Optimization (BCVO)** algorithm. A binary vector consisting solely of zeros and ones is employed to represent a binary scenario, such as feature selection, wherein a value of one denotes the collection of a feature, while zero designates its segregation. The length of this vector corresponds to the number of features within the dataset. In Figure 4, the binary depiction of a COVID solution is illustrated for a dataset comprising $\mathfrak{D}$ features.

Initial stage: The preliminary stage of the COVID approach entails generating a population, denoted as "pop," comprising arbitrarilyengendered individuals. To each individual within this population characterizes a prospective solution to the given problematic. The initial population is created utilizing the following equation.

$$\mathcal{Y}_i = \text{minVal}_i + \sigma_i \times (\text{maxVal}_i - \text{minVal}_i), i = 1,2,..,\mathfrak{D}$$

Where maxVal_i and minVal_i represent the upper and lower limits of the problem correspondingly; σ_i is a randomly generated number between 0 and 1; and $\mathcal{Y}_i$ denotes the solution at the ith position in the population "pop". However, in the binary version, $\text{maxVal}_i = 1$ and $\text{minVal}_i = 0$. Each outcome in the proposed binary method undergoes binarization, meaning it is transformed into its binary representation. One of the commonly employed transformation functions from the δ-shaped family is the sigmoid function. This function achieves the mapping of every real-valued answer to either 0 or 1:

$$\delta(\mathcal{Y}_i) = \frac{1}{1 + e^{\mathcal{Y}_i}}, \mathcal{Y}_{\text{binary}} = \begin{cases} 1 & \text{if randm} \geq \delta(\mathcal{Y}_i) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

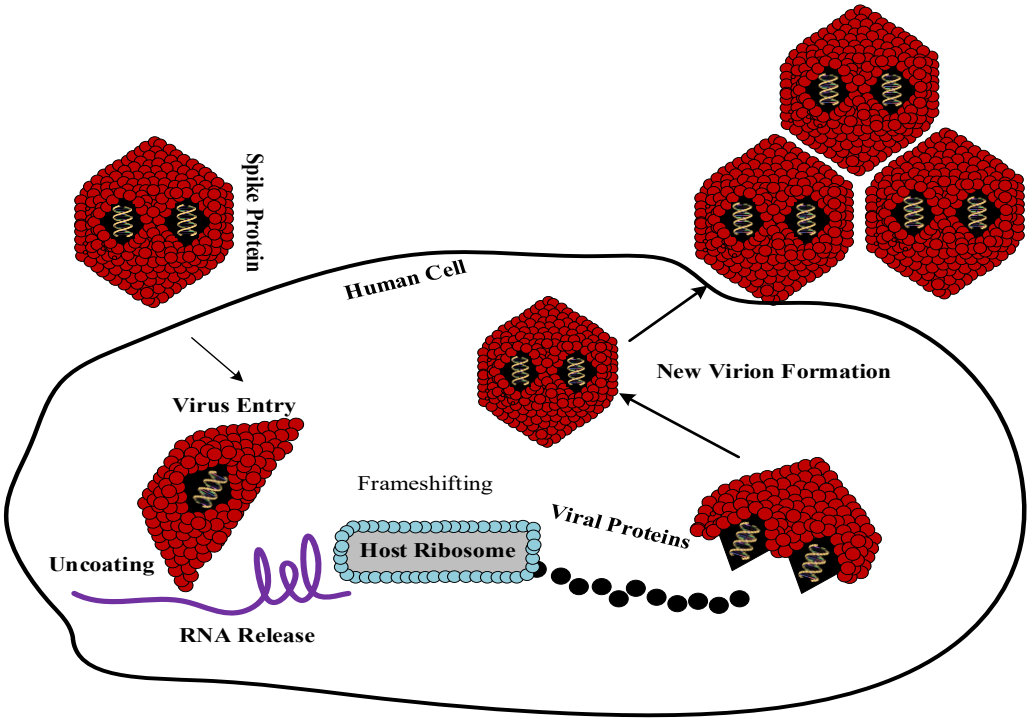


Fig.2 BCVO Lifecycle

KNN classifier: The KNN classifier (with $\mathfrak{k} = 5$) enhances classification accuracy for each solution due to the following reasons:

It boasts ease of implementation. Implementing KNN necessitates only two parameters: the number of neighbours (𝔨) and the distance function (e.g., Euclidean or Manhattan).

Its memory-based technique facilitates rapid response to changes in the input.

It efficiently identifies the optimal subset of characteristics with excellent resilience to noisy data.

KNN technique allows users to select distances (e.g., Euclidean, Hamming, Manhattan, etc.) while creating the model.

KNN classifier categorizes each data point into a class containing the majority of its nearest ŧ neighbours. Datasets are divided into training, validation, and testing sets using cross-validation, as detailed in [41]. Cross-validation involves using $ŧ - 1$ folds for training and validation, while the remaining fold is allocated for testing.

The K-NN classifier functions as follows: To determine the ŧ nearest neighbours for each sample in the test dataset, the KNN method computes the Euclidean distance according to the formula below:

$$\mathfrak{D}(\mathcal{Y}, \mathcal{X}) = \sqrt{\sum_{i=1}^{d} (\mathcal{Y}_i - \mathcal{X}_i)^2} \qquad (8)$$

Where $d$indicates how many features there are in a particular dataset.In Figure 6, when $ŧ = 3$, the classifier allocates the unknown sample to class B based on its proximity to two points from that class. By dividing the total number of examples in the dataset by the number of accurate instances, one can calculate the classifier's classification accuracy. Divide the total number of occurrences in the dataset by the number of wrong occurrences to find the classification error rate. Utilizing the KNN classifier, determine the classification accuracy rate for respectively solution. The best answer is the one that has the highest accuracy rate.

Fitness function: It is important to consider both the number of designated features and the classification accuracy rate while analysing solutions. When two solutions show the same classification accuracy, the solution with fewer characteristics chosen is considered better. The fitness function, which is formulated as follows, is intended to decrease the number of selected attributes while concurrently optimizing classification accuracy.

$$\text{Fitness} = \sigma\Phi_c + (1 - \sigma)\frac{\delta}{\eta} (9)$$

In this equation, σ varies between 0 and 1, where Φ_c represents the classifier's error rate, δ denotes the number of chosen features, and η represents the entire number of features.

Position update: Virus particles utilize frameshifting to generate diverse protein sequences, amalgamating them to form a new-fangled particle (solution). In the innovative COVIDOA version, the +1-frameshifting method involves shifting variables to the right by one and assigning a random value within the range [maxVal, minVal] as outlined in Equations (1) and (2). The binary version employs the frameshifting approach in the following manner:

$$\delta_k(1) = \begin{cases} 1 & \text{if randm}(0,1) < 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad (10)$$

$$\delta_k(2{:}ɪ) = \rho(1{:}ɪ - 1), \qquad (11)$$

δ_k represents the k-th produced protein, ρ denotes the parent solution, ɪ signifies the problem dimension, and random() generates a random number between 0 and 1. Following replication, the binary version can be adjusted as follows:

$$\mathcal{Z}_i = \begin{cases} \begin{cases} 1 & \text{if randm}(0,1) < 0.5 \\ 0 & \text{otherwise} \end{cases} & \text{if randm}(0,1) < \text{MR} \\ \mathcal{Y}_i & \text{otherwise} \end{cases} \qquad (12)$$

$\mathcal{Y}$ denotes the original solution, and $\mathcal{Y}_i$and $\mathcal{Z}_i$ represent the ith element in both the ancient and new solutions, respectively. MR represents the mutation rate.

Load Prediction

Furthermore, the load in RSUs can be predicted using Deep Learning (DL) algorithm named **Load Prediction Network (LPN).** The LPN is composed of LSTM encoder, convolution layers, and LSTM decoder for predicting and allocating the RSUs load to process the computation tasks. In addition to that, the communication among the vehicles (V2V) managed by mmWave technology and communication in whole environment can be managed by 6G communication. LSTM

Module: LSTM represents a RNN architecture designed for temporal data processing, incorporating gating mechanisms (input, forget, and output gates) within its structure. Unlike traditional RNNs, LSTM efficiently manages information retention and discard over time, thereby mitigating issues such as gradient dispersion and enhancing spatial-temporal correlations within models. Illustrated in Figure 2, the LSTM prediction model adopts a spatial-temporal attention encoder-decoder framework. The LSTM encoding network, LSTM decoding network and spatial-temporal attention layer are the three main parts of this architecture. The LSTM encoder utilizes a graph convolutional layer to remove spatial-temporal characteristics from the input sequence during the encoding phase. The spatial-temporal attention vector is then used to create a latent representation of the historical data that is comprehensive. This encoded spatial-temporal attention vector is used by the LSTM decoder to forecast the destination sequence throughout the decoding process.

When it comes to managing time-series data with prolonged time-dependent properties, the LSTM model has shown itself to be very effective. Long-term spatial-temporal sequences can be learned from the input in this model because the LSTM encoder efficiently captures the periodic time-dependence and spatial-temporal correlations. The formed vector is then integrated with the spatial-temporal attention vector to create a complete picture of deep spatial-temporal characteristics obtained from past terminus sequence data. The encoded spatial-temporal attention vector is employed by the LSTM decoder, which functions as the target predictive model, to facilitate decoding, as illustrated in Figure 2. Equations encapsulate the operational methodology of this process (Li et al., 2020).

$$\mathcal{f}_t = \pi\left(\omega_{x\mathcal{f}} x_t + \omega_{\hbar\mathcal{f}} \hbar_{t-1} + \mathcal{B}_{\mathcal{f}}\right) \quad (13)$$

$$\mathcal{U}_t = \pi\left(\omega_{x\mathcal{U}} x_t + \omega_{\hbar\mathcal{U}} \hbar_{t-1} + \mathcal{B}_{\mathcal{U}}\right) \quad (14)$$

$$\mathcal{O}_t = \pi\left(\omega_{x\mathcal{O}} x_t + \omega_{\hbar\mathcal{O}} \hbar_{t-1} + \mathcal{B}_{\mathcal{O}}\right) \quad (15)$$

$$\mathcal{C}_t = \mathcal{f}_t \odot \mathcal{C}_{t-1} + \mathcal{U}_t \odot \tan\hbar\left(\omega_{x\mathcal{C}} x_t + \omega_{\hbar\mathcal{C}} \hbar_{t-1} + \mathcal{B}_{\mathcal{C}}\right) \quad (16)$$

$$\hbar_t = \mathcal{O}_t \odot \tan\hbar(\mathcal{C}_t) \quad (17)$$

The forgetting gate, update gate, and output gate are represented as $\mathcal{f}_t$, $\mathcal{U}_t$, and $\mathcal{O}_t$ respectively. The cell memory state vectors are denoted by $\mathcal{C}_t$, and the hidden state vector by $\hbar_t$. The input vector is x_t, the element-wise product is denoted by $\odot$, and the sigmoid function is represented by π in these equations. In the training process, the weights and biases are denoted by ω and $\mathcal{B}$. As shown in Equation (19), we simplify the LSTM representations presented in Equation (18).

$$\hbar_t, \mathcal{C}_t = \mathrm{LSTM}(x_t, \hbar_{t-1}, \mathcal{C}_{t-1}) \quad (18)$$

Two models make up the architecture: one encodes the input sequence into a fixed-length vector by processing it, and the other decodes this vector to produce the expected sequence. The preceding time step's concealed state as well as the input time series data are both accepted by the encoder. Equation 19 emphasizes that the encoder latent vector $\mathcal{C}_t$ and the hidden state $\hbar_t$ are the outputs of the t-th time step. The decoder iteratively analyzes state vectors and integrates prediction outputs back into the network to improve multi-step prediction accuracy and reduce the quick accumulation of mistakes

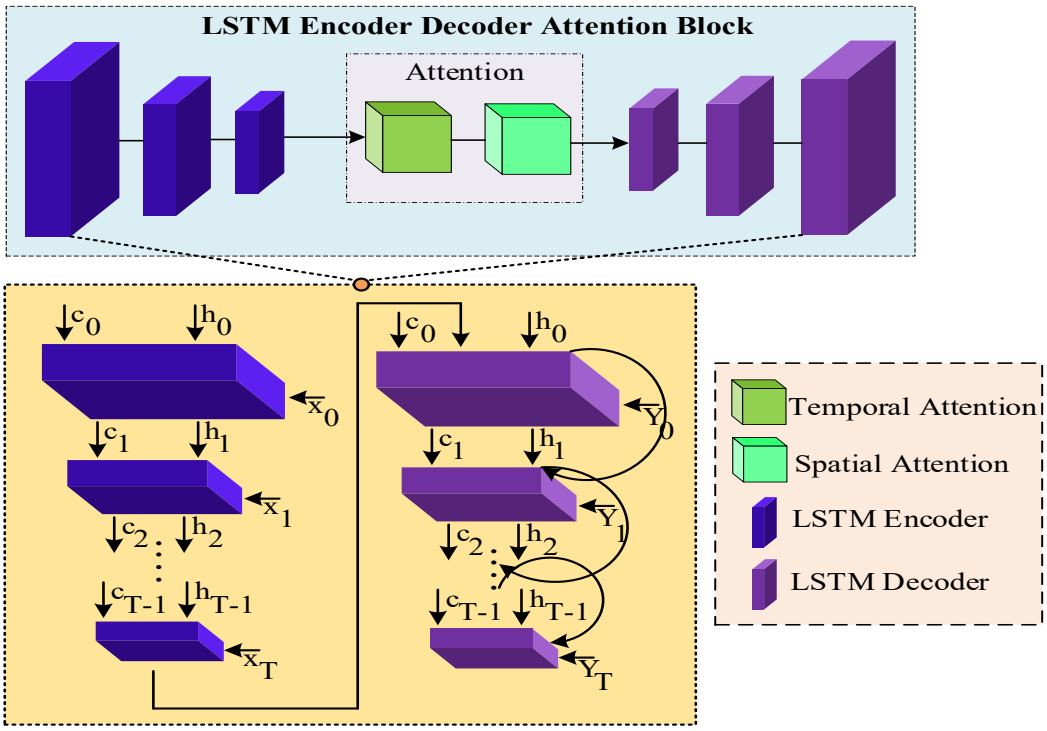


Fig 3. Workflow of LSTM

provided by the encoder. The encoder transfers $\mathcal{C}_t$ and $\hbar_t$ to the decoder, which accepts them. Initially, the decoder receives γ_0 and computes $\mathcal{C}_{1'}$ and $\gamma_{1'}$ using initial vectors $\gamma_{0'}$ and $\mathcal{C}_{0'}$ (as shown in Equations 19 and 20). The position of the online car-hailing service is denoted by $\gamma_{1'}$. Equation 20 calculates the sequence of prediction outcomes, $\gamma_{1'}$ through $\gamma_{H'}$.

$$\gamma_{t'} = \omega_l \hbar_{1'} \tag{19}$$

$$\gamma_{t'} = \frac{e^{\gamma_{t'}}}{\sum_{j=1}^{t} e^{\gamma_{t'}}} \tag{20}$$

$$\gamma_{t'} = \max(\gamma_{t'}) \tag{21}$$

Spatial-Temporal Attention Mechanism: To capture the dynamic spatial-temporal correlation within the load prediction network, we employ the spatial-temporal attention mechanism (see Figure 3). The spatial attention mechanism determines how important the position sequence is on a daily basis in relation to the goal location that requires to be predicted. In the meantime, through weight distribution, the temporal attention mechanism dynamically modifies the relative value of each place in the day's time series.

Spatial Attention Mechanism: Here, we dynamically capture correlations between nodes in spatial dimensions by utilizing the attentional process. Equations (21, 22) delineate the computational approach of the spatial attention mechanism.

$$\mathcal{S} = \mathcal{P}_{\mathcal{S}} . \pi \left(\left(x_i^{(r-1)} \omega_1 \right) \omega_2 \left(\omega_3 x_i^{(r-1)} \right)^T + \mathcal{B}_{\mathcal{S}} \right) \tag{22}$$

$$\mathcal{S}'_{i,j} = \frac{\exp(\mathcal{S}_{i,j})}{\sum_{j=1}^{N} \exp(\mathcal{S}_{i,j})} (23)$$

The input of the r-th spatial-temporal block is denoted by $x_i^{(r-1)}$. The number of input data channels in the r - th layer is represented by c_{r-1}. Learnable parameters include $\mathcal{P}_{\mathcal{S}}, \mathcal{B}_{\mathcal{S}} \in \mathcal{R}^{n \times n}, \omega_1 \in \mathcal{R}^{T_{r-1}}, \omega_2 \in \mathcal{R}^{c_{r-1} \times T_{r-1}}, \omega_3 \in \mathcal{R}^{c_{r-1}}$. The activation function is the sigmoid function π. The current input of this layer

determines how dynamically the spatial attention matrix $\mathcal{S}$ is calculated. The correlation strength between nodes i and j is semantically represented by the value of an element $\mathcal{S}_{i,j}$ in $\mathcal{S}$.

Temporal Attention Mechanism: By retaining the encoder's temporal hidden state throughout successive temporal vector generations, the temporal attention mechanism scrutinizes the temporal behavioral characteristics vector. Equations (23, 24) elucidate how the weighted sum of the sequence $\{x_{t'}, 1 \le t' \le T\}$ constitutes the temporal attention mechanism. Utilizing an LSTM-decoder, the sequence of $x_{t'}$ is leveraged to forecast the destination $Y_{t+1'}$ at time $t + 1'$. We observed strong correlations between certain locations at specific times during the prediction process. For instance, during the morning rush hour (7:00–9:00), a majority of online vehicle hailing destinations are businesses, educational institutions, and medical facilities. Conversely, there are more residential clusters during the evening peak, which spans from 16:00 to 19:00. Over a specific timeframe, the temporal attention mechanism assigns higher weights to destinations deemed more significant after learning the long-term time-dependent properties of past data. As a result, by scrutinizing destination weights and streamlining the attention model, we can grasp the underlying movement patterns.

$$\Im = \mathcal{P}_u . \pi \left(\left(\left(x_i^{(r-1)} \right)^T \omega_1^{'} \right) \omega_2^{'} \left(\omega_3^{'} x_i^{(r-1)} \right) + \mathcal{B}_u \right) \quad (24)$$

$$\Im_{i,j}^{'} = \frac{\exp\left(\Im_{i,j}\right)}{\sum_{j=1}^{T_{r-1}} \exp\left(\Im_{i,j}\right)} \quad (25)$$

The learnable parameters include $\mathcal{P}_u, \mathcal{B}_u \in \mathcal{R}^{T_{r-1} \times T_{r-1}}, \omega_1^{'} \in \mathcal{R}^{\mathcal{N}}, \omega_2^{'} \in \mathcal{R}^{c_{r-1} \times \mathcal{N}}, \omega_3^{'} \in \mathcal{R}^{c_{r-1}}$. Various inputs influence the temporal attention matrix $\Im$. The correlation strength between time i and time j is semantically represented by the value of an element.

Experimental Results

Simulation Setup

In this sub-division, the 'proposed CCO-TA framework is determined by accomplishing simulations in MATLAB(R) 9.9 (R2020b) in i7 processor of 2.70GHz Intel Core. Here, the road scenario is computed through Automated Driving Toolbox 3.2, in which 10 RSU is deployed on 10 km 6-lane 100 vehicles are arbitrary placed on road primarily, where 30 are VE from each clustering. Furthermore, vehicles are travelling at the [2, 30] m/s range speed in whichever direction. Table I defines the simulation parameter configuration.

Comparative analysis

In this work, the proposed CCO-TA framework is evaluated in order to perform comparative analysis in two schemes which are detailed as follows,

Comparative among Benchmark Schemes

Initially, the sub-section estimates the proposed CCO-TA in comparison with several benchmark schemes, i.e., ExO, ECO, EL, and nearest VECO. Moreover, OPORA and NCO are personalized to be appropriate to the model in this paper meanwhile there is no viable solution which can be unswervingly adapted to this difficulty. The models are articulated as follows,

ExO: the challenges of individual vehicle are comprehensively offloaded by optimal offloading scheme.

EL: overall vehicles perform the task in local manner.

VECO: the task of individual vehicle

are offloaded to near VEC server along with vehicle which are involved presently.

ECO: here, entire challenges are overloaded to the cloud server.

NCO: Each vehicle plays a scattered, non-cooperative game to solve the likelihood of optimal offloading in a competitive manner. NCO is now offered for the server situation of single-time decision and one-VEC.

OPORA: Vehicle tasks are given to VEC servers based on a one-to-one matching mechanism, and price-rising mechanisms are used to incentivize individual VEC servers to assign resources.

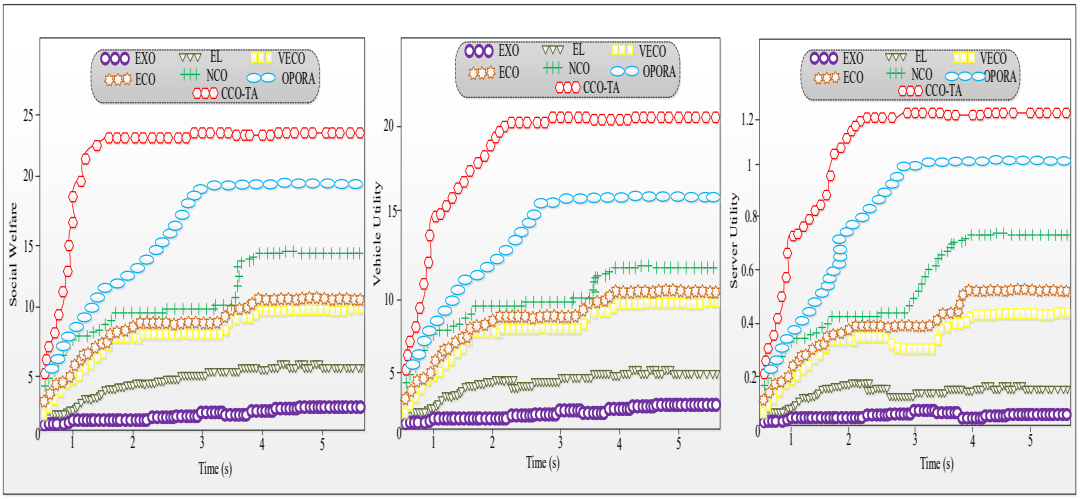


Fig 4. Benchmark and ProposedModel Performance in terms of Time with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

(ii) Comparative among Existing Approaches

Likewise, our proposed CCO-TA framework is also evaluated and compared with following prior works includes EETO-RA [24], FATO-RA [27], EATO-RA [30], PM-CCO-TA [37], and DRA-CO [38] which are defined as follows,

EETO-RA: By utilizing OFDMA approach, offloading of joint task and allocation of resource is performed on the basis of energy-efficient manner.

FATO-RA: In this work, cooperative MEC is adapted for performing effective task offloading and resource allocation based on fairness-aware.

EATO-RA: This work majorly focused on allocating the resource for services of time-sensitive in MEC. Besides, task offloading is adapted in energy-aware manner.

PM-CCO-TA: This work adopts edge computing with distributed cloud computing system for preforming computation offloading and resource allocation. Additionally, offloading and allocation are collaboratively accomplished for maximized profit.

DRA-CO: Here, the study mainly concentrates on implementing offloading of computation and allocation of resource dynamically.

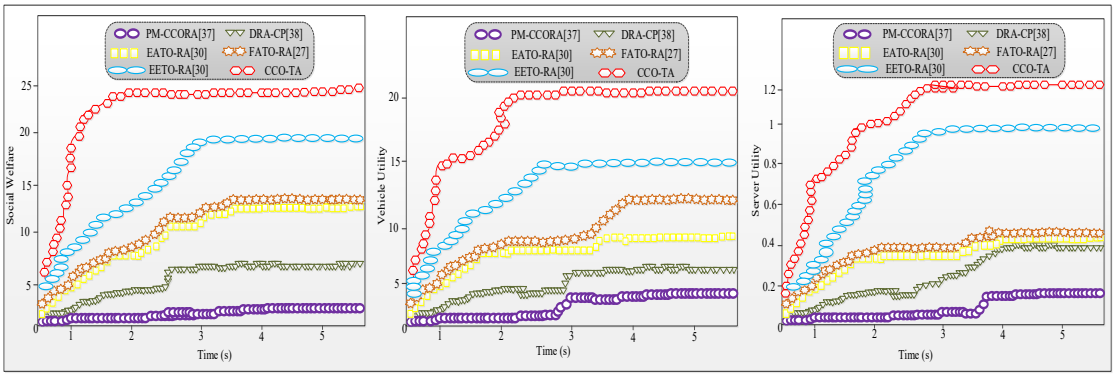


Fig 5. Existing and Proposed Model Performance in terms of Time with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

Performance of System

This division assesses the impact of various system characteristics on the efficiency of social welfare, utilityof vehicle, and utility of server all of which are detailed in the subsequent subsection.,

Time Effect

The temporal comparison of the vehicle total utility, VEC server total utility, and societal welfare for seven algorithms is shown in Figures 4(a), 4(b), and 4(c). The overall utility of vehicles, social welfare, and servers all increase over time for all algorithms, as seen in Figure 4. This growing trend can be attributed to automobiles' increased ability to successfully complete more tasks over time. Moreover, it is evident that CCO-TA outperforms ExO, ECO, EL, VECO, OPORA, and NCO in terms of social welfare, vehicle utility, and server utility. This superiority has a number of reasons. First off, EL, ExO, VECO, and NCO prioritize optimizing the work offloading mechanism for vehicles above considering the resource allocation approach for VEC servers. The offloading strategies employed by these schemes—local offloading, exhaustive offloading, nearest VEC server offloading, full cloud offloading, and competitive offloading—may cause congestion and resource misuse in some vehicles or VEC servers. Furthermore, OPORA outperforms EL, ExO, VECO, and NCO in terms of VEC utility because to its price incentive approach; but, it is not as successful as CCO-TA since it employs a less successful cooperative strategy.

Likewise, in comparative analysis of existing approaches with our proposed CCO-TA model, the proposed attains betterment performance. However, In EETO-RA, as the vehicle count rises, the joint task offloading and resource allocation using OFDMA may become more intricate, potentially introducing scalability concerns and heightened computational demands. In FATO-RA, with an expanding fleet of vehicles, the cooperative MEC approach for task offloading and resource allocation may struggle to maintain fairness among a larger user base, potentially leading to extended wait times or conflicts over resources. In EATO-RA, a growing number of vehicles could present challenges in efficiently allocating resources for time-sensitive services in MEC, potentially resulting in heightened latency or diminished service quality as the system accommodates more simultaneous users. In PM-CCO-TA, the scalability of edge and distributed cloud computing systems for computation offloading and resource allocation may face scrutiny as the number of vehicles accessing the system

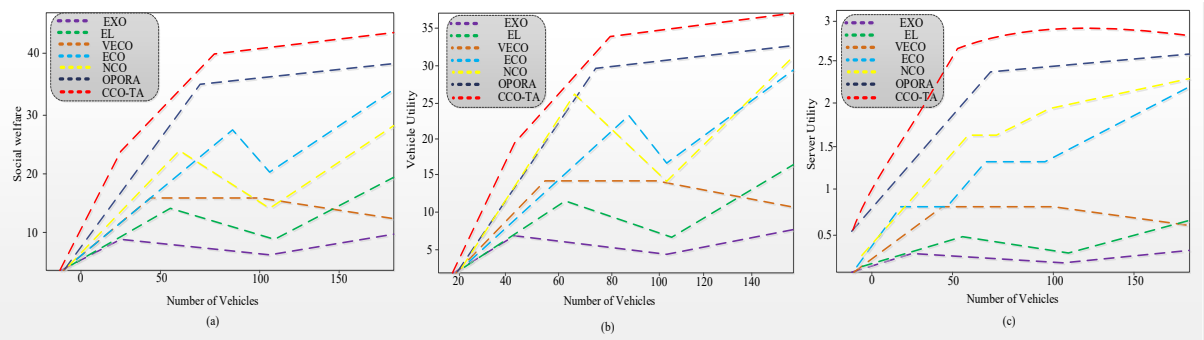


Fig 6. Benchmark and Proposed Model Performance in terms of number of vehicles with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

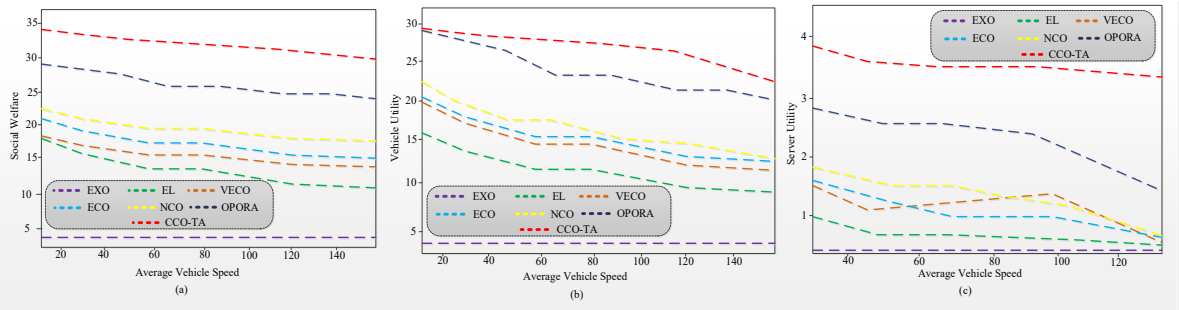


Fig 7. Benchmark and Proposed Model Performance in terms of average vehicle speed with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

increases, potentially leading to performance dips or longer response times. In DRA-CO, with an increasing number of vehicles dynamically offloading computations, the system's ability to adapt and allocate resources in real-time may encounter greater challenges, potentially resulting in delays or suboptimal resource utilization in dynamic IoV environments.

Conversely, CCO-TA employed BCV optimization approach, facilitating improved task offloading and computation resource utilization by horizontally or vertically tasks offloading to VEC servers or cloud servers. Additionally,CCO-TA's collaborative task offloading in cloud IoV enhances system performance, reduces latency, optimizes resource usage, improves scalability, conserves energy, ensures fault tolerance, enables dynamic adaptation, and offers cost-effectiveness. Collaborative computation offloading utilizing the BCVO algorithm proves effective in IoV networks by dynamically adapting task allocation based on varying time constraints, ensuring timely processing of computational tasks and minimizing latency. This approach optimizes resource utilization and task scheduling, enhancing the overall efficiency and responsiveness of the IoV network across different time periods. Consequently, the simulation results indicate that CCO-TA exhibits superior performance in terms of social welfare, utility of vehicle, and utility of VEC server among the seven algorithms considered.

Vehicles Number Effect

The effects of changing car counts on the social welfare performance metrics, utility of total vehicle, and utility of total server across schemes are depicted in Figures 3(a), 3(b), and 3(c). Notably, because all functions are carried out locally on vehicles, EL performs the worst in terms of social welfare, utility of vehicle, andutility of server. Furthermore, when the number of vehicles rises, initial high trends are followed by declining tendencies in the social welfare,utility of vehicle, and utility of server of NVO and ECO. The phenomena is ascribed to the cumulative quantity of tasks associated with a growing number of cars, which may result in overload at the closest VECO servers when implementing NVO and a growing number of incomplete tasks when implementing ECO. Furthermore, OPORA indicates that whereas server utility increases at a decreasing pace, social welfare and vehicle utility first improve as the number of vehicles rises before declining. OPORA's one-to-one dumping mechanism and randomly growing price incentive are the main causes of this pattern. Social welfare andutility of vehicle show erratic swings in NCO, but utility of server slightly rises with the number of cars. Because of its probabilistic offloading technique and increased vehicle competition, NCO performs worse than most other schemes, even when there is no statistically significant variation. With a rising number of vehicles, ExO shows early increasing trends in social welfare, utility of vehicle, and utility of server; however, these trends reflect a progressive slowing down or even a falling tendency. As the number of vehicles keeps increasing, ExO's thorough offloading approach could cause congestion or resource problems for some VECO servers.

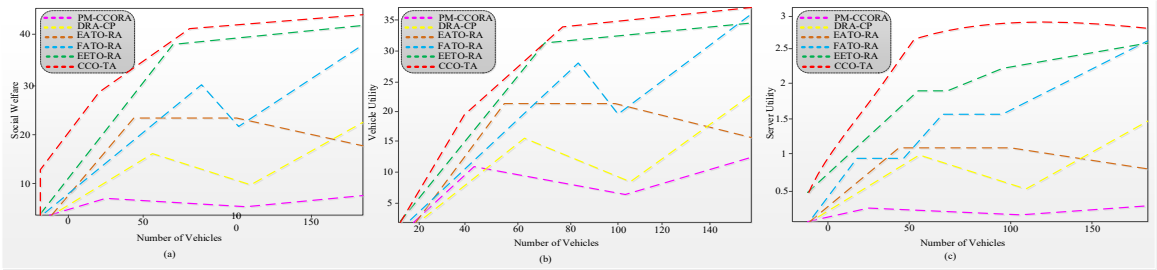


Fig 8. Existing and Proposed Model Performance in terms of number of vehicles with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

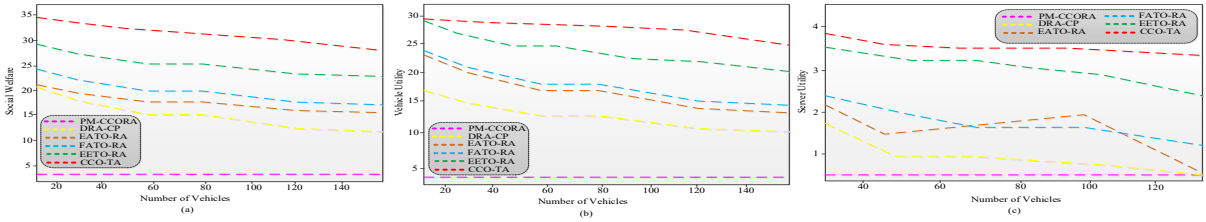


Fig 9. Benchmark and Proposed Model Performance in terms of average vehicle speed with (a) Social Welfare (b) Vehicle Utility & (c) Server Utility

Additional, individual prior works mentioned may face drawbacks related to the effects of vehicle numbers are: As the vehicle count rises in EETO-RA, the complexity of offloading of joint task and allocation of resource using OFDMA may also increase, potentially resulting in scalability issues and higher computational overhead With an expanding number of vehicles, the cooperative MEC approach for offloading of task and allocation of resource in FATO-RA may face challenges in ensuring fairness among users, potentially leading to longer queue times or resource contention issues. In EATO-RA, the growing number of vehicles could present challenges in efficiently allocating resources for time-sensitive services in MEC, potentially resulting in higher latency or reduced quality of service as the system accommodates more concurrent users.The scalability of edge and distributed cloud computing systems for offloading of computation and allocation of resource in PM-CCO-TA may be scrutinized as the number of vehicles accessing the system expanded, potentially resulting in performance degradation or increased response times.As the number of vehicles dynamically offloading computations increases, the system's ability to adapt and allocate resources in real-time may become more challenging in DRA-CO, potentially leading to delays or suboptimal resource utilization in dynamic IoV environments. In contrast, CCO-TA displays progressively increasing trends in social welfare, vehicle utility, and server utility, maintaining a relatively superior level among the seven schemes and baseline existing approaches. Collaborative computation offloading employing the BCVO algorithm demonstrates effectiveness in IoV networks by dynamically adjusting resource allocation based on fluctuating vehicle numbers, thereby maintaining optimal performance levels and mitigating resource contention issues as the network scales. This approach optimizes task distribution among vehicles, ensuring efficient utilization of computational resources regardless of variations in the number of vehicles within the IoV network. These duplicate effects indicate the superior scalability of the suggested CCO-TA approach with an expanding number of vehicles.

Vehicle Speed Effect

Social welfare Comparisons, utility of total vehicle, and utility of total server among the algorithms under consideration is shown in Fig... 4(a), 4(b), and 4(c), respectively. Remarkably, when it comes to vehicle average speed, EL continuously performs the terms in worst of societal welfare, car utility, and server utility indicators (see Figure 4). Given that functions are carried out locally on cars without requiring connectivity with VEC servers, this is not surprising. Additionally, the curves representing societal welfare, utility of vehicle, and utility of server for algorithmsExO, ECO, OPORA, NVO, NCO, and CCO-TA show general declining tendencies as average vehicle speed rises. The primary cause of this tendency is the high mobility of cars, which causes them to move outside of the VEC servers' service range more frequently when downloading or computing tasks. As a result, there are more instances of repetitive handovers or service interruptions, which raise the risk of job failures and service delays. In the end, this lowers vehicle satisfaction and lowers server income. Notably, when the average speed above 100 km/h, the performance decrease for ExO, ECO, OPORA, NVO, and NCO becomes severe.

Moreover, the comparative analysis of existing works with our proposed CCO-TA model in terms of vehicle speed effect results in attaining effective performance of CCO-TA where the limitations are follows. In EETO-RA, the utilization of OFDMA for offloading of joint task and allocation of resource may encounter difficulties in accommodating varying vehicle speeds, potentially resulting in inefficiencies in resource usage and task scheduling as vehicles exhibit different velocities. In FATO-RA, the fairness of task offloading and resource allocation within Cooperative MEC may be compromised by differences in vehicle speeds, potentially leading to uneven distribution of resources and prolonged processing times for slower-moving vehicles. EATO-RA may face challenges in ensuring timely delivery of tasks for time-sensitive services in MEC while considering vehicle speed, particularly for fast-moving vehicles, which could result in increased latency and diminished service quality. PM-CCO-TA may encounter hurdles in dynamically adjusting offloading of computation and allocation of resource placed on fluctuations in vehicle speed, potentially leading to suboptimal profit maximization and resource utilization within edge computing and distributed cloud systems.In DRA-CO, the adaptability of dynamic computation offloading and resource allocation may be hindered by changes in vehicle speeds, resulting in delays in task processing and inefficient resource allocation, especially in scenarios where vehicles frequently change speeds or trajectories.

However, the performance of the proposed CCO-TA shows comparatively constant declining tendencies as vehicle average speed rises, keeping top rankings among the seven schemes. This can be attributed to CCO-TA’scollaborative decision-making process for task offloading, which considers emergency and non-emergency tasks.Collaborative computation offloading employing the BCVO algorithm proves effective in IoV networks by dynamically adjusting task allocation based on varying vehicle speeds, ensuring efficient utilization of resources and minimizing latency. This approach optimizes computational tasks distribution among vehicles,

thereby enhancing the overall responsiveness and performance of the IoV network despite fluctuations in vehicle speeds.

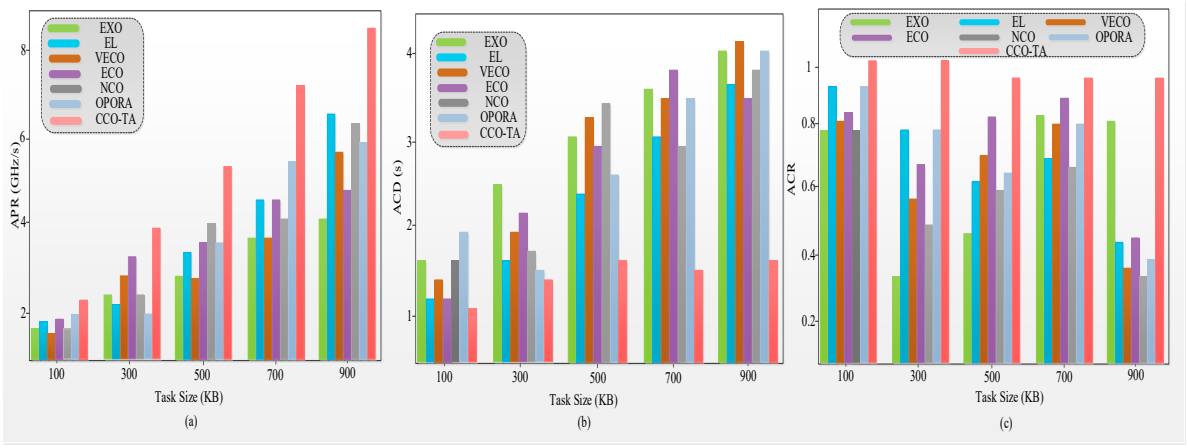


Fig 10. Benchmarks and Proposed System Efficiency in terms of (a)Avg_{RP} (b) Avg_{DC}& (c) Avg_{RC}

Evaluation of System Efficiency

The following data is gathered for performance assessment:the duration of each task's generation$t_i^{DT}(t), \Delta i \in \partial, t \in \mathcal{T}$; the time it takes for each job to be done successfully $t_i^{ST}(t), \Delta i \in \partial, t \in \mathcal{T}$; the total number of tasks that are successfully finished within the examined timeline, indicated asN_T^S. These statistics are used to define the performance measures that follow.

Average rate of processing (Avg_{RP}) is the average number of tasks (measured in bits) processed in a unit of time. It may be found as follows:

$$Avg_{RP} = \frac{\sum_{t\in\mathcal{T}} \sum_{i\in\partial} \rho_i^{gn}(t).C_i^{rq}(t)}{\sum_{t\in\mathcal{T}} \sum_{i\in\partial} \rho_i^{gn}(t).\left(t_i^{cp}(t) - t_i^{rq}(t)\right)} \quad (26)$$

Average delay of completion (Avg_{DC}) is described as the average completion delay of task successfully which is calculated as,

$$Avg_{DC} = \frac{\sum_{t\in\mathcal{T}} \sum_{i\in\partial} \rho_i^{gn}(t).\left(t_i^{cp}(t) - t_i^{rq}(t)\right)}{\sum_{t\in\mathcal{T}} \sum_{i\in\partial} \rho_i^{gn}(t)} \quad (27)$$

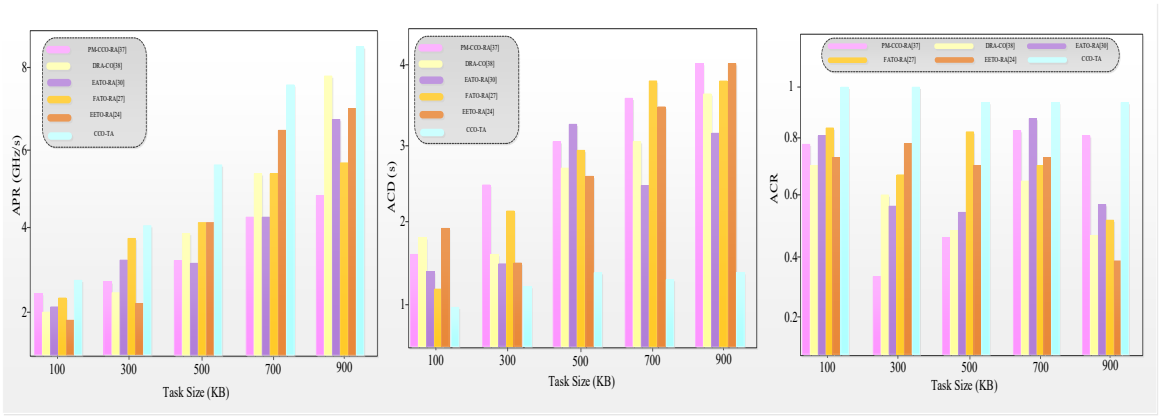


Fig 11. Existing and Proposed System Efficiency in terms of (a)Avg_{RP} (b) Avg_{DC}& (c) Avg_{RC}

Average ratio of completion (Avg_{RC}) is well-defined as the tasks ratio which are completed successfully to the entire number of tasks computed while contemplating the duration as,

$$Avg_{RC} = \frac{N_T^S}{\sum_{t\in\mathcal{T}} \sum_{i\in\partial} \rho_i^{gn}(t)} \qquad (28)$$

Task Size Effect: Avg_{RP}, Avg_{DC}, and Avg_{RC} metrics for seven algorithms across different task sizes are compared in Figures 5(a), 5(b), and 5(c) respectively. Initially, when analyzing NVO, ECO, EL, and ExO along with NCO, it's observed that their APR and ACD generally increase as task sizes grow, while their Avg_{RC} exhibits the opposite trend. This is primarily attributed to the heavier workloads on cars or servers as task sizes escalate, leading to higher processing costs per unit of time, prolonged job completion delays, and a decrease in task completion ratio.Furthermore, as task sizes increase, CCO-TA displays a significant increase in Avg_{RP}, a slight upward trend in Avg_{DC}, and a minor decrease in Avg_{RC}. This indicates that with growing workloads, CCO-TA demonstrates a noticeable rise in processing rate with comparatively modest delays and failure rates. Additionally, it surpasses other methods in terms of Avg_{RP}, Avg_{DC}, and Avg_{RC} as task sizes expand. CCO-TA aims to facilitate intra-server resource allocation by promoting cooperation between servers for inter-server task offloading as well as collaboration between servers and vehicles. Ultimately, the results depicted in Figure 5 illustrate the effectiveness of CCO-TA across various task sizes in terms of Avg_{RP}, Avg_{DC}, and Avg_{RC}.

2The drawbacks of task size on the Avg_{RP}, Avg_{DC}, and Avg_{RC} in the mentioned prior works are as follows. In EETO-RA,different task sizes may affect the $Avg_{RP}$when using OFDMA for joint task offloading and resource allocation. This could result in variations in throughput efficiency and longer task completion times, especially for bigger workloads. Furthermore, because smaller activities may unfairly delay larger jobs due to resource contention, larger tasks may upset the fairness of resource allocation, affecting the Avg_{RC}.Cooperative MEC may find it difficult to control task sizes, which can have an impact on both the $Avg_{RP}$and the Avg_{DC}. Larger tasks could take longer to process, which could cause delays in finishing and jeopardise user fairness. This discrepancy in task completion rates may compromise the system's overall effectiveness of FATO-RA. In EATO-RA, task size may have varying effects on Avg_{RP} and Avg_{DC}, even with MEC's focus on time-sensitive services. Bigger tasks could take longer and more resources to complete, which could cause delays and have an impact on the completion rate.

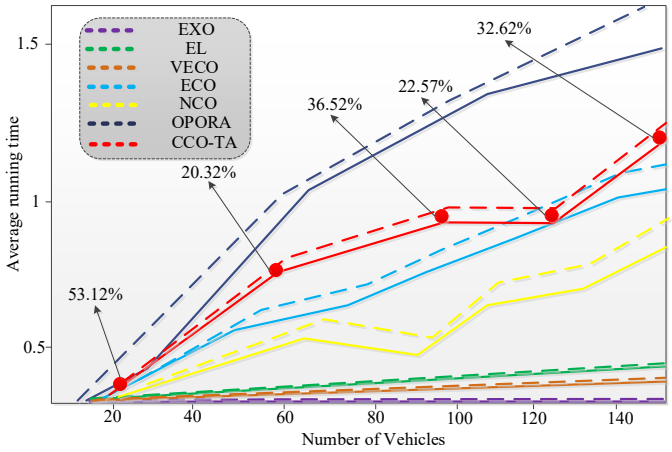


Fig 12. Benchmark Schemes and Proposed Model Average Running Time in terms of Number of Vehicles

Furthermore, smaller jobs may be given priority for energy efficiency through energy-aware task offloading, which would further affect the completion percentage.The implementation of edge computing and distributed cloud systems in PM-CCO-TA may encounter difficulties in effectively managing diverse task sizes, hence affecting the $Avg_{RP}$and Avg_{DC}. Bigger jobs could put a load on the system, causing delays and lengthier processing times. Certain activities may be prioritised above others in collaborative offloading for profit maximisation, which could have an impact on fairness and the Avg_{RC}.Resource allocation and dynamic computation offloading may find it difficult to adjust effectively to changing task sizes, which could affect both $Avg_{RP}$and the Avg_{DC}in DRA-CO. Larger activities could take longer to complete and demand more resources, which could have an impact on the Avg_{RC}. Furthermore, dynamic allocation might give some tasks more priority than others, which would dynamically alter completion ratios and fairness.Collaborative computation offloading employing the BCVO algorithm proves effective in IoV networks by dynamically allocating computational tasks based on their sizes, ensuring efficient utilization of resources and minimizing processing delays. This approach optimizes task distribution among vehicles, prioritizing smaller tasks to expedite completion and minimize the impact of larger tasks on overall system performance. By adaptively managing

task sizes, BCVO facilitates smoother task execution, enhancing the responsiveness and efficiency of the IoV network across diverse computational workloads.

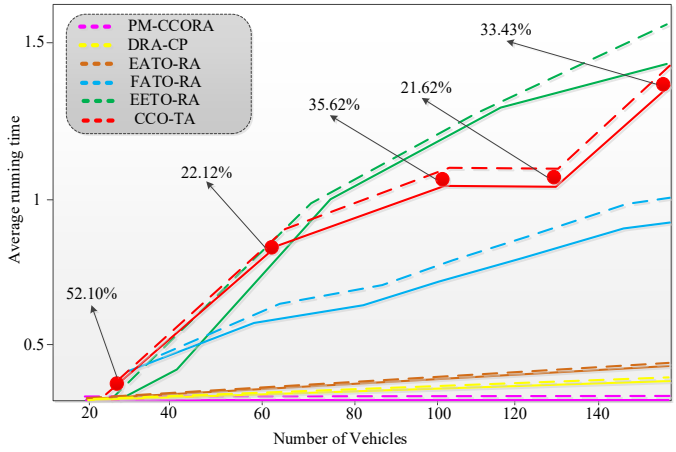


Fig 13. Existing Works and Proposed Model Average Running Time in terms of Number of Vehicles

Running Time of Algorithm

Figure 6 shows the average running times of seven algorithms with respect to the number of cars using two devices: a strong device (D_s) with an Intel Core i7-12700H 2.70 GHz processing and 16.0 GB RAM memory, and a weak device (D_w) with an Intel Core i7-9750H 2.60 GHz processor and 16.0 GB RAM memory. This aids in emphasizing the processor's impact. The data illustrate how each algorithm's average running time rises as the number of vehicles does. Notably, EL, NVO, ECO, and NCO exhibit lower temporal complexity than ExO, OPOPRA, and CCO-TA because they make explicit task offloading decisions; however, this reduction comes at the sacrifice of system performance and efficiency. OPOPRA has a substantially higher time complexity due to its random price-growing incentive methods and laborious one-to-one matching. Furthermore, CCO-TA outperforms ExO in dense networks, albeit at a higher time complexity, with a linear increase in average execution time with the number of cars. When executed on Ds, the average running time decreases by 19.28% to 52.02%, indicating the impact of device capability. In somewhat dense networks, CCO-TA outperforms direct decision-making systems in social welfare, vehicle utility, and server utility, while having a higher temporal complexity than ExO. Moreover, the complexity of the suggested approach increases linearly with the number of vehicles and executes in polynomial time. The technique might provide further time savings if the real VEC nodes have more powerful processing power available.

Conclusion

In conclusion, computation offloading and task allocation play a critical role in the optimization of resource utilization, enhancement of computational efficiency within the IoV paradigm. As discussed, the complexity of IoV environments necessitates efficient strategies for distributing computing tasks among vehicles and edge/cloud infrastructure. However, existing approaches are plagued by inherent limitations such as resource constraints, energy inefficiency, and network reliability issues, underscoring the need for innovative solutions to alleviate these challenges.To address these limitations, we have proposed a novel framework termed CCO-TA, which harnesses the power of emerging technologies including VEC, mmWave, and 6G. By integrating these advanced technologies, our framework aims to revolutionize computation offloading and task allocation processes in vehicular cloud environments.The CCO-TA framework encompasses several key components designed to optimize task allocation and enhance system efficiency. Firstly, the DP-KNN algorithm enables the clustering of AVs based on task emergency, vehicle mobility, and density. Subsequently, each cluster selects a CH with superior computation capabilities, which is entrusted with executing critical tasks through VEC while less critical tasks are offloaded to RSUs.

Furthermore, RSUs engage in collaborative offloading among themselves within their range using the BCVO algorithm, considering vehicle mobility patterns. Additionally, the LPN, a DL algorithm comprising LSTM encoder, convolution layers, and LSTM decoder, is employed for accurate load prediction in RSUs, thereby facilitating efficient task allocation.Moreover, communication among vehicles (V2V) is managed effectively through mmWave technology, which offers high bandwidth and low latency communication capabilities, ensuring seamless data exchange among vehicles. Furthermore, 6G technology is leveraged for overall

communication management within the IoV environment, reducing communication overhead and enhancing network reliability.Finally, simulation results validate the efficacy of the proposed CCO-TA approach, demonstrating superior system utility, resource optimization, and efficiency compared to existing methods, particularly under heavy system workloads. This underscores the potential of our framework to address the prevailing challenges in computation offloading and task allocation within IoV networks, paving the way for more robust and efficient vehicular cloud environments in future.

Reference

1. Ji, B., Zhang, X., Mumtaz, S., Han, C., Li, C., Wen, H., & Wang, D. (2020). Survey on the internet of vehicles: Network architectures and applications. IEEE Communications Standards Magazine, 4(1), 34-41.

2. Kamble, S. J., &Kounte, M. R. (2020). Enabling technologies for internet of vehicles. In Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI-2018) (pp. 257-268). Springer International Publishing.

3. Muhammad, K., Ullah, A., Lloret, J., Del Ser, J., & de Albuquerque, V. H. C. (2020). Deep learning for safe autonomous driving: Current challenges and future directions. IEEE Transactions on Intelligent Transportation Systems, 22(7), 4316-4336.

4. Duan, W., Gu, J., Wen, M., Zhang, G., Ji, Y., & Mumtaz, S. (2020). Emerging technologies for 5G-IoV networks: applications, trends and opportunities. IEEE Network, 34(5), 283-289.

5. Long, J., Luo, Y., Zhu, X., Luo, E., & Huang, M. (2020). Computation offloading through mobile vehicles in IoT-edge-cloud network. EURASIP Journal on Wireless Communications and Networking, 2020, 1-21.

6. Liu, T., Tang, L., Wang, W., He, X., Chen, Q., Zeng, X., & Jiang, H. (2022). Resource allocation in dt-assisted internet of vehicles via edge intelligent cooperation. IEEE Internet of Things Journal, 9(18), 17608-17626.

7. Faccio, M., Granata, I., & Minto, R. (2024). Task allocation model for human-robot collaboration with variable cobot speed. Journal of Intelligent Manufacturing, 35(2), 793-806.

8. Huang, X., He, L., Chen, X., Wang, L., & Li, F. (2021). Revenue and energy efficiency-driven delay-constrained computing task offloading and resource allocation in a vehicular edge computing network: A deep reinforcement learning approach. IEEE Internet of Things Journal, 9(11), 8852-8868.

9. Hou, X., Ren, Z., Wang, J., Cheng, W., Ren, Y., Chen, K. C., & Zhang, H. (2020). Reliable computation offloading for edge-computing-enabled software-defined IoV. IEEE Internet of Things Journal, 7(8), 7097-7111.

10. Wang, J., Zhu, K., & Hossain, E. (2021). Green Internet of Vehicles (IoV) in the 6G era: Toward sustainable vehicular communications and networking. IEEE Transactions on Green Communications and Networking, 6(1), 391-423.

11. Abbas, N., Sharafeddine, S., Mourad, A., Abou-Rjeily, C., & Fawaz, W. (2022). Joint computing, communication and cost-aware task offloading in D2D-enabled Het-MEC. Computer Networks, 209, 108900.

12. Wang, J., Zhu, K., & Hossain, E. (2021). Green Internet of Vehicles (IoV) in the 6G era: Toward sustainable vehicular communications and networking. IEEE Transactions on Green Communications and Networking, 6(1), 391-423.

13. Panda, S. K., Lin, M., & Zhou, T. (2022). Energy-efficient computation offloading with DVFS using deep reinforcement learning for time-critical IoT applications in edge computing. IEEE Internet of Things Journal, 10(8), 6611-6621.

14. Gao, H., Liu, C., Li, Y., & Yang, X. (2020). V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability. IEEE Transactions on Intelligent Transportation Systems, 22(6), 3533-3546.

15. Zhang, Z., Li, C., Peng, S., & Pei, X. (2021). A new task offloading algorithm in edge computing. EURASIP Journal on Wireless Communications and Networking, 2021(1), 17.

16. Liu, L., Zhao, M., Yu, M., Jan, M. A., Lan, D., & Taherkordi, A. (2022). Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks. IEEE Transactions on Intelligent Transportation Systems, 24(2), 2169-2182.

17. Hou, X., Ren, Z., Wang, J., Cheng, W., Ren, Y., Chen, K. C., & Zhang, H. (2020). Reliable computation offloading for edge-computing-enabled software-defined IoV. IEEE Internet of Things Journal, 7(8), 7097-7111.

18. Khan, A. A., & Zakarya, M. (2021). Energy, performance and cost efficient cloud datacentres: A survey. Computer Science Review, 40, 100390.

19. Xu, X., Huang, Q., Yin, X., Abbasi, M., Khosravi, M. R., & Qi, L. (2020). Intelligent offloading for collaborative smart city services in edge computing. IEEE Internet of Things Journal, 7(9), 7919-7927.

20. Alam, M. Z., & Jamalipour, A. (2022). Multi-agent DRL-based Hungarian algorithm (MADRLHA) for task offloading in multi-access edge computing Internet of Vehicles (IoVs). IEEE Transactions on Wireless Communications, 21(9), 7641-7652.

21. Yang, B., Cao, X., Bassey, J., Li, X., & Qian, L. (2020). Computation offloading in multi-access edge computing: A multi-task learning approach. IEEE transactions on mobile computing, 20(9), 2745-2762.

22. Zhang, Z., & Zeng, F. (2022). Efficient task allocation for computation offloading in vehicular edge computing. IEEE Internet of Things Journal, 10(6), 5595-5606.

23. Jiang, H., Dai, X., Xiao, Z., & Iyengar, A. K. (2022). Joint task offloading and resource allocation for energy-constrained mobile edge computing. IEEE Transactions on Mobile Computing.

24. Tan, L., Kuang, Z., Zhao, L., & Liu, A. (2021). Energy-efficient joint task offloading and resource allocation in OFDMA-based collaborative edge computing. IEEE Transactions on Wireless Communications, 21(3), 1960-1972.

25. Tang, H., Wu, H., Zhao, Y., & Li, R. (2021). Joint computation offloading and resource allocation under task-overflowed situations in mobile-edge computing. IEEE Transactions on Network and Service Management, 19(2), 1539-1553.

26. Alfakih, T., Hassan, M. M., Gumaei, A., Savaglio, C., & Fortino, G. (2020). Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA. IEEE Access, 8, 54074-54084.

27. Zhou, J., & Zhang, X. (2021). Fairness-aware task offloading and resource allocation in cooperative mobile-edge computing. IEEE Internet of Things Journal, 9(5), 3812-3824.

28. Yan, J., Bi, S., & Zhang, Y. J. A. (2020). Offloading and resource allocation with general task graph in mobile edge computing: A deep reinforcement learning approach. IEEE Transactions on Wireless Communications, 19(8), 5404-5419.

29. Wang, F., Xu, J., & Cui, S. (2020). Optimal energy allocation and task offloading policy for wireless powered mobile edge computing systems. IEEE Transactions on Wireless Communications, 19(4), 2443-2459.

30. Zhao, M., Yu, J. J., Li, W. T., Liu, D., Yao, S., Feng, W., ... & Quek, T. Q. (2021). Energy-aware task offloading and resource allocation for time-sensitive services in mobile edge computing systems. IEEE Transactions on Vehicular Technology, 70(10), 10925-10940.

31. Zhang, Q., Gui, L., Hou, F., Chen, J., Zhu, S., & Tian, F. (2020). Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud RAN. IEEE Internet of Things Journal, 7(4), 3282-3299.

32. Dai, Y., Zhang, K., Maharjan, S., & Zhang, Y. (2020). Edge intelligence for energy-efficient computation offloading and resource allocation in 5G beyond. IEEE Transactions on Vehicular Technology, 69(10), 12175-12186.

33. Yu, Z., Gong, Y., Gong, S., & Guo, Y. (2020). Joint task offloading and resource allocation in UAV-enabled mobile edge computing. IEEE Internet of Things Journal, 7(4), 3147-3159.

34. Tong, Z., Deng, X., Ye, F., Basodi, S., Xiao, X., & Pan, Y. (2020). Adaptive computation offloading and resource allocation strategy in a mobile edge computing environment. Information Sciences, 537, 116-131.

35. Li, C., Cai, Q., Zhang, C., Ma, B., & Luo, Y. (2021). Computation offloading and service allocation in mobile edge computing. The Journal of Supercomputing, 77(12), 13933-13962.

36. Chu, W., Yu, P., Yu, Z., Lui, J. C., & Lin, Y. (2022). Online optimal service selection, resource allocation and task offloading for multi-access edge computing: A utility-based approach. IEEE Transactions on Mobile Computing.

37. Yuan, H., & Zhou, M. (2020). Profit-maximized collaborative computation offloading and resource allocation in distributed cloud and edge computing systems. IEEE Transactions on Automation Science and Engineering, 18(3), 1277-1287.

38. Chang, Z., Liu, L., Guo, X., & Sheng, Q. (2020). Dynamic resource allocation and computation offloading for IoT fog computing system. IEEE Transactions on Industrial Informatics, 17(5), 3348-3357.

39. Chai, R., Song, X., & Chen, Q. (2020). Joint task offloading, CNN layer scheduling, and resource allocation in cooperative computing system. IEEE Systems Journal, 14(4), 5350-5361.

40. Zhang, G., Zhang, S., Zhang, W., Shen, Z., & Wang, L. (2021). Joint service caching, computation offloading and resource allocation in mobile edge computing systems. IEEE Transactions on Wireless Communications, 20(8), 5288-5300.