GMR
The Original

PCO-VN: DESIGNING PREDICTIVE COMPUTATIONAL OFFLOADING MODEL FOR VEHICULAR NETWORKS USING HYBRID MOBILE EDGE AND CLOUD COMPUTING 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, rajesswari45@rediffmail.com
[2]Professor and Head Department of Artificial Intelligence and Data Science Faculty of Engineering Karpagam Academy of Higher Education Coimbatore, Tamil Nadu, India, journals856@gmail.com
*Corresponding Author: rajesswari45@rediffmail.com

ABSTRACT
A Computational Offloading Model for Vehicular Networks leveraging Hybrid Mobile Edge and Cloud Computing Technologies presents an innovative approach to address the computational constraints of vehicles in dynamic environments. By integrating mobile edge computing (MEC) and cloud computing, this model aims to offload computational tasks from vehicles to nearby edge servers or remote cloud data centers, thereby enhancing the efficiency and scalability of vehicular networks. However, despite its promising benefits, several limitations exist. By creating a predictive offloading model with cloud and hybrid mobile edge computing technologies, this study seeks to optimize the computation offloading workflow in the vehicular cloud environment. In this study, Predictive Computation Offloading in Vehicular Network (PCO-VN) which begins by examining where RSUs should be placed about vehicles, treating RSUs as mobile edge servers. Artificial Humming Bird Optimization (AHBO), an optimization algorithm that maximizes efficiency by considering multiple variables, is used to achieve effective RSU placement. When RSUs are positioned, the surrounding cars group together to create vehicular cloudlets, which are areas where AVs and RSUs can exchange resources and data by communicating within their respective communication ranges. An Attention-based Asymmetry Encoder-Decoder Network (AEDNet) is used in this framework to forecast the trajectory of autonomous vehicles (AVs) while considering a variety of parameters to improve prediction accuracy. Afterwards, RSUs use a Multi-Criteria Decision Making (MCDM) algorithm to rate the AVs within their range from low to high moving AVs based on the trajectory prediction results. Deep Reinforcement Learning (DRL) technique is used to do virtualized compute offloading based on the ranking outcomes. To be more precise, the DRL agent finds AVs that are available for unloading first. A virtualized computing server is constructed for timely offloading, ensuring effective resource use; if all AVs are overburdened, the task is offloaded to RSUs. The proposed PCO-VN model is evaluated in terms of proving its efficiency, in which PCO-VN is outperformed other existing works.

KEYWORD*:* Predictive Computation Offloading, Trajectory Prediction, Optimal RSU Placement, DRL, AV's Ranking, Internet of vehicles.

I. INTRODUCTION

The advent of the Internet of Vehicles (IoV) has instigated a profound shift in transportation systems, heralding an era of heightened intelligence, enhanced security, and superior efficacy in mobility solutions. Within the intricate fabric of IoV networks, the seamless integration of vehicles [1], road infrastructure, and cutting-edge communication technologies facilitates real-time data exchange [2] and astute decision-making. At the core of this transformative landscape lies computation offloading, a pivotal process orchestrating the transfer of computational workload from resource-constrained vehicles to more potent cloud or edge servers. This strategic redistribution not only bolsters system performance but also substantially curtails energy consumption [3]. To fully harness the expansive computational resources available in the cloud or at the network edge, computation offloading techniques must be judiciously employed within IoV networks. By delegating tasks such as data processing, sensor fusion, and decision-making to centralized servers, vehicles can effectually conserve energy, prolong battery life [4]-[6], and optimize system efficiency. Moreover, sophisticated algorithms and applications such as real-time traffic management, autonomous driving, and predictive maintenance thrive when computational tasks are intelligently offloaded to robust computing platforms [7].

Among the array of strategies for optimizing compute offloading in IoV networks, trajectory prediction-based offloading emerges as a particularly promising avenue. This methodology harnesses advanced machine learning

algorithms [8] and predictive analytics to forecast the future trajectories of vehicles through an exhaustive analysis of historical data, prevailing conditions, and relevant environmental factors. By accurately predicting vehicle movements and discerning emerging traffic patterns [9][10], trajectory prediction empowers proactive decision-making regarding computation offloading [11]. For instance, vehicles approaching congested areas or intersections can proactively offload computational tasks to nearby servers in anticipation of heightened processing demands. Similarly, vehicles embarking [12][13] on extended journeys can strategically offload tasks to ensure seamless operation throughout the duration of the trip, optimizing both resource utilization [14] and system performance. However, notwithstanding the manifold potential advantages of trajectory-based computation offloading, several challenges persist, particularly concerning the optimal placement of Roadside Units (RSUs) [15]-[17] within IoV networks. RSUs serve as indispensable communication hubs, facilitating seamless data exchange between infrastructure and vehicles. Yet, suboptimal RSU placement can significantly impede the effectiveness of computation offloading endeavors. For instance, inadequate RSU coverage may precipitate communication delays, thereby compromising the accuracy of real-time trajectory prediction systems. Additionally, the uneven distribution of RSUs across roadways may exacerbate server congestion and overload, thwarting efficient task offloading efforts.

Moreover, a notable constraint of trajectory-based computation offloading pertains to the prioritization of autonomous vehicles within IoV networks. The ranking of computational activities in autonomous vehicles [18] is typically predicated on various criteria, including system preferences, resource availability, and proximity to critical events. However, without precise trajectory prediction capabilities, the ranking of autonomous vehicles may be suboptimal, leading to inefficient resource allocation and performance degradation. For instance, vehicles with inaccurate trajectory predictions may erroneously receive higher task priorities than those with more reliable predictions, resulting in unnecessary delays and bottlenecks in the offloading process. In conclusion, computation offloading constitutes a cornerstone of IoV networks, pivotal to maximizing their efficiency and performance. Predictive trajectory offloading [19] represents a potent strategy for proactive task management and enhanced system responsiveness. However, to fully realize the promise of trajectory-based computation offloading in IoV networks, challenges such as suboptimal RSU placement and constraints on autonomous vehicle ranking must be addressed. By surmounting these obstacles, IoV networks can attain heightened intelligence, scalability, and dependability, paving the way for a new era of connected and self-driving mobility [20]. To alleviate these issues, we have introduced effective computation offloading approach based on trajectory prediction in IoV networks.

A. Research Contribution

To achieve effective and predictive computation offloading, in this work we have proposed several novel contributions which are detailed as follows,

- ***Innovative RSU Placement Strategy:*** The proposed PCO-VN model employs Artificial Humming Bird Optimization (AHBO), a novel optimization algorithm, to strategically place Road Side Units (RSUs) within vehicular environments. By treating RSUs as mobile edge servers, this approach maximizes efficiency by considering multiple variables, thereby enhancing the effectiveness of computation offloading.
- ***Trajectory Prediction with Enhanced Accuracy:*** Leveraging an Attention-based Asymmetry Encoder-Decoder Network (AEDNet), the PCO-VN model forecasts the trajectories of autonomous vehicles (AVs) within vehicular networks. By considering a variety of parameters, this framework significantly improves prediction accuracy, enabling more precise decision-making regarding computation offloading.
- ***Dynamic Virtualized Compute Offloading:*** The PCO-VN model incorporates a Deep Reinforcement Learning (DRL) technique for dynamic virtualized compute offloading based on trajectory prediction results and AV ranking outcomes. This innovative approach ensures effective resource utilization by creating virtualized computing servers for timely offloading, thereby optimizing system performance and scalability within vehicular networks.

B. Paper Organization

The following paper are structured as follows: In Section II, the associated efforts are expounded upon, offering a comprehensive examination of the current computation offloading model and highlighting its research gaps. Section III emphasizes the research methodology, supplemented by relevant theoretical, diagrammatical, and mathematical explanations. Section IV presents the experimental findings, detailing dataset specifications and evaluation results juxtaposed with the most recent iterations of computation offloading models. Finally, Section V provides the concluding remarks on the proposed research.

II. LITERATURE SURVEY

The burgeoning field of Internet of Vehicles (IoV) has witnessed a surge in research aimed at optimizing computation offloading strategies to leverage the potential of edge and cloud computing paradigms. In this review, we delve into five seminal research papers that contribute significantly to this domain, offering insights into reliable and efficient computation offloading techniques tailored for IoV environments. [21] This study addresses the critical issue of reliability in computation offloading within edge-enhanced Internet of Vehicles (IoV) environments. It introduces a comprehensive framework that ensures dependable offloading techniques, crucial for real-time applications in IoV. The proposed solution's adaptability and efficiency are further enhanced through the integration of software-defined networking (SDN). [22] Furthermore, the study focuses on maximizing various factors such as latency, energy utilization, and cost, presenting an advanced computation offloading strategy

tailored for Internet of Vehicles applications. By striking a balance between competing objectives and leveraging both cloud and edge computing resources, the suggested strategy enhances the overall performance of IoV systems, contributing significantly to the advancement of vehicular computing paradigms. [23] This article pioneers the use of deep reinforcement learning techniques to address the intricate challenges of resource allocation and joint computation offloading within Internet of Vehicles (IoV) environments. By harnessing the collaborative capabilities of edge and cloud computing, the proposed approach dynamically adjusts to evolving network conditions, optimizing resource consumption, and enhancing Quality of Service (QoS) for IoV applications. [24] Furthermore, this research presents a holistic approach to enhancing the scalability and performance of IoV systems by integrating intelligent edge computing capabilities with computation offloading and caching strategies. Through deliberate offloading of processing tasks and caching frequently accessed data at the network edge, the proposed method minimizes latency, reduces bandwidth usage, and optimizes overall resource efficiency. This comprehensive approach offers promising prospects for improving the efficiency and effectiveness of IoV deployments in real-world scenarios. [25] This study presents a decentralized method of multiuser computation offloading that is designed especially for Internet of Vehicles (IoV) scenarios that are supported by fog computing infrastructure. DEFT mitigates the effects of network dynamics and node failures while optimizing compute offloading decisions through effective task allocation and coordination methods, guaranteeing strong performance in dynamic Internet of Vehicles situations.

By utilizing traffic flow prediction models inside edge computing infrastructure, this study tackles the complex trade-offs between consumption of energy and latency in compute offloading for IoV systems [26]. The suggested system meets latency restrictions and maximizes energy efficiency while optimizing offloading decisions based on real-time traffic estimates, improving the overall sustainability and dependability of IoV installations. [27] In order to facilitate compute offloading in Internet of Vehicles (IoV) networks, this research provides a resource allocation strategy that makes use of Deep Reinforcement Learning (DRL). The suggested method adjusts to changing network conditions by learning the best offloading policies by interaction with the surroundings, maximizing resource use and enhancing Quality of Service (QoS) for Internet of Vehicles (IoV) applications. [28] This work presents a new resource allocation strategy for Internet of Vehicles (IoV) systems by integrating Deep Reinforcement Learning with edge computing infrastructure. The suggested methodology optimizes offloading decisions to improve system performance and scalability in edge computing contexts by utilizing deep reinforcement learning techniques to assign computing resources dynamically. [29] Using Deep Reinforcement Learning to adaptively offload work to edge nodes, this research provides a dynamic edge computation offloading technique for Internet of Vehicles (IoV) settings. In dynamic IoV settings, the suggested method minimizes latency and maximizes resource usage by optimizing offloading decisions by utilizing real-time feedback and environmental observations. [30] This study presents a complete system that combines content caching at the network edge with compute offloading, with a focus on the next generation of 6G-enabled IoV. The suggested method reduces latency and bandwidth usage, which improves the performance and scalability of IoV applications in 6G settings. It does this by proactively caching frequently requested data and shifting processing duties to edge nodes.

This research introduces a collaborative optimization framework for computation offloading in IoV, leveraging both edge and cloud computing resources [31]. By jointly optimizing offloading decisions across the edge and cloud layers, the proposed approach enhances resource utilization, minimizes latency, and improves overall system efficiency, thereby facilitating seamless integration of IoV applications with diverse computing infrastructures. [32] Utilizing the concept of minority game, this paper proposes a dynamic task offloading mechanism tailored for IoV environments in cloud-edge computing scenarios. By dynamically allocating tasks based on the minority strategy, the proposed approach optimizes resource utilization, mitigates congestion, and enhances system scalability, thereby improving the overall performance of IoV applications in dynamic and heterogeneous computing environments. [33] Addressing the energy consumption challenges in IoV, this research presents an energy-aware designed for dependent applications in offloading scheme of software-defined IoV environments with fog computing architecture. By considering the interdependencies between applications and optimizing offloading decisions accordingly, the proposed scheme minimizes energy consumption while meeting application-specific requirements, thereby enhancing the sustainability and reliability of IoV deployments. [34] Focusing on the synergy between edge computing and 5G networks, this paper proposes an efficient computation offloading scheme tailored for IoV applications. By leveraging edge computing resources and the high-speed connectivity offered by 5G networks, the proposed approach minimizes latency, optimizes resource utilization, and enhances overall system performance, thereby unlocking the full potential of IoV in next-generation network environments. [35] Introducing a game-theoretic framework coupled with fuzzy neural network for distributed task offloading in IoV, this research offers a sophisticated solution to optimize resource allocation and offloading decisions. By modeling the offloading process as a game and employing fuzzy neural networks for decision-making, the proposed approach enhances system efficiency, improves QoS, and fosters collaboration among IoV entities in edge computing environments.

This research focuses on leveraging 5G NR-V2X communication for task offloading and resource allocation in IoV environments [36]. By exploiting the low-latency and high-bandwidth capabilities of 5G NR-V2X communication, the proposed approach enhances real-time data exchange and computation offloading decisions, thereby improving system efficiency and responsiveness in dynamic vehicular scenarios. [37] Introducing a distributed computation offloading approach based on DRL, this paper addresses offloading decisions in

Intelligent Connected Vehicles (ICV). By leveraging deep reinforcement learning techniques, the proposed approach enables ICVs to autonomously optimize offloading decisions based on local observations, thereby enhancing system scalability, reliability, and adaptability to diverse network conditions. [38] This research presents a comprehensive analysis of computation offloading in MEC-enabled IoV networks, focusing on average energy efficiency and learning-based maximization. By evaluating offloading strategies and leveraging learning-based approaches, the proposed framework optimizes energy efficiency and enhances system performance, paving the way for sustainable and efficient IoV deployments. [39] Introducing a resource allocation scheme based on Stochastic Actor-Critic (SAC) algorithm, this paper addresses computation offloading in IoV networks. By modeling the offloading process as a stochastic decision-making problem, the proposed approach optimizes resource allocation decisions, improving system efficiency, and scalability while adapting to dynamic network conditions and user preferences. [40] Focusing on multi-access edge computing (MEC) enabled IoV environments, this research presents a joint computation offloading and scheme of data caching. By integrating computation offloading with data caching at the network edge, the proposed approach minimizes latency, reduces bandwidth consumption, and enhances overall system performance, facilitating seamless integration of IoV applications with edge computing infrastructure. [41] proposed a task offloading method and implemented handover for enhancing service performance. In this work, trajectory prediction is performed in edge-adapted vehicular network thereby refining task offloading. Collectively, these research papers contribute significantly to advancing computation offloading strategies for IoV, offering innovative solutions tailored for the unique requirements and challenges encountered in vehicular environments. Through their pioneering methodologies and experimental validations, these studies pave the way for more efficient, resilient, and intelligent IoV deployments, driving forward the realization of connected and autonomous vehicular ecosystems.

III. PCO-VN FRAMEWORK

In this work, we tends to optimize the workflow of computation offloading in the vehicular cloud environment by designing the predictive offloading model using cloud and hybrid mobile edge computing technologies. The entities involved in this work are Autonomous Vehicles (AVs), Road Side Units (RSUs), vehicular cloudlet, and core cloud computing server. Initially, we perform RSU placement in the vehicular environment as the RSUs can acts as the mobile edge server. Figure 1 represents the overall architecture of proposed PCO-VN model for computation offloading in IoV networks.

A. Optimal RSU Placement

For effective RSU placement, we utilized an optimization algorithm named Artificial Humming Bird Optimization (AHBO) by considering several criteria. After that, the vehicles in the environment form the vehicular cloudlet where the AVs and RSUs communicate within the communication range for data and resource sharing. Here, n birds are arbitrarily positioned on n nectar sources which is described as,

$$\mathfrak{X}_i = Low + r.(Up - Low) \quad i = 1, \dots, n \tag{1}$$

From the above equation, $\mathfrak{X}_i$ affords optimal solution i.e. food source position, lower and upper bound value are denoted as Low and Up respectively. r is the random vector selected among 0 and 1. Visiting table handled through hummingbirds is afforded by following mathematical equation,

$$HB_{i,j} = \begin{cases} 0 & if\ i \neq j \\ nul & i = j \end{cases} \quad i = 1, \dots, n; j = 1, \dots, n \tag{2}$$

;

Here, $i = j\ HB_{i,j} =$ null infers where hummingbirds obtained from nectar of its source specific nectar; $i = j\ HB_{i,j} = 0$ infers the present incarnation of i-th hummingbird which retrieved j-th food source. AHBO functions in following four foraging which are articulated as follows,

Guided Foraging: Basically, all kinds of birds fly in all directions, however hummingbirds can fly diagonally and axially. Therefore, diagonal, axial and omnidirectional flight of $\mathfrak{D}$ dimension which is written in following equation,

$$\mathfrak{D}^{(i)} = \begin{cases} 1 & if\ \ i = rani([1,d]) \\ 0 & else \end{cases} \quad i = 1, \dots, d \tag{3}$$

$$\mathfrak{D}^{(i)} = \begin{cases} 1 & if\ i = \mathfrak{P}(j),\ \ j \in [1,k], \mathfrak{P} = ranper(k), k \in [2, r_1 \cdot (d-2)+1] \\ 0 & else \end{cases} \quad i = 1, \dots, d \tag{4}$$

$$\mathfrak{D}^{(i)} = 1 \quad i = 1, \dots, d \tag{5}$$

Where $rani([1,d])$ provided the arbitrary value among 1 to d, $ranper(k)$ gives arbitrary integers permutations from 1 to k, r_1 computes arbitrary number of range [0,1]. The mathematical equation for designing behaviour of guided foraging and likely food source,

$$h_i(t+1) = \mathfrak{X}_{i,tar}(t) + a \cdot \mathfrak{D} \cdot \left(\mathfrak{X}_i(t) - \mathfrak{X}_{i,tar}(t)\right) \tag{6}$$

$$a \sim N(0,1) \tag{7}$$

Here, $\mathfrak{X}_i(t)$ provides the i-th nectar source location in time t, $\mathfrak{X}_{i,tar}(t)$ is the guide conforms to a standard distribution, serving as a reference for the intended nectar source location targeted by the i-th hummingbird. Expressed with a mean of 0 and a standard deviation of 1, denoted as $N(0,1)$ in Eq. (7), this signifies how hummingbirds leverage their distinct flying abilities to adjust their positions around the desired nectar source, resembling directed foraging. The latest update for the i-th food supply location is outlined as follows:

$$\mathfrak{X}_i(t+1) = \begin{cases} \mathfrak{X}_i(t) & f(\mathfrak{X}_i(t+1)) \le f(h_i(t+1)) \\ h_i(t+1) & f(\mathfrak{X}_i(t+1)) \le f(h_i(t+1)) \end{cases} \quad (8)$$

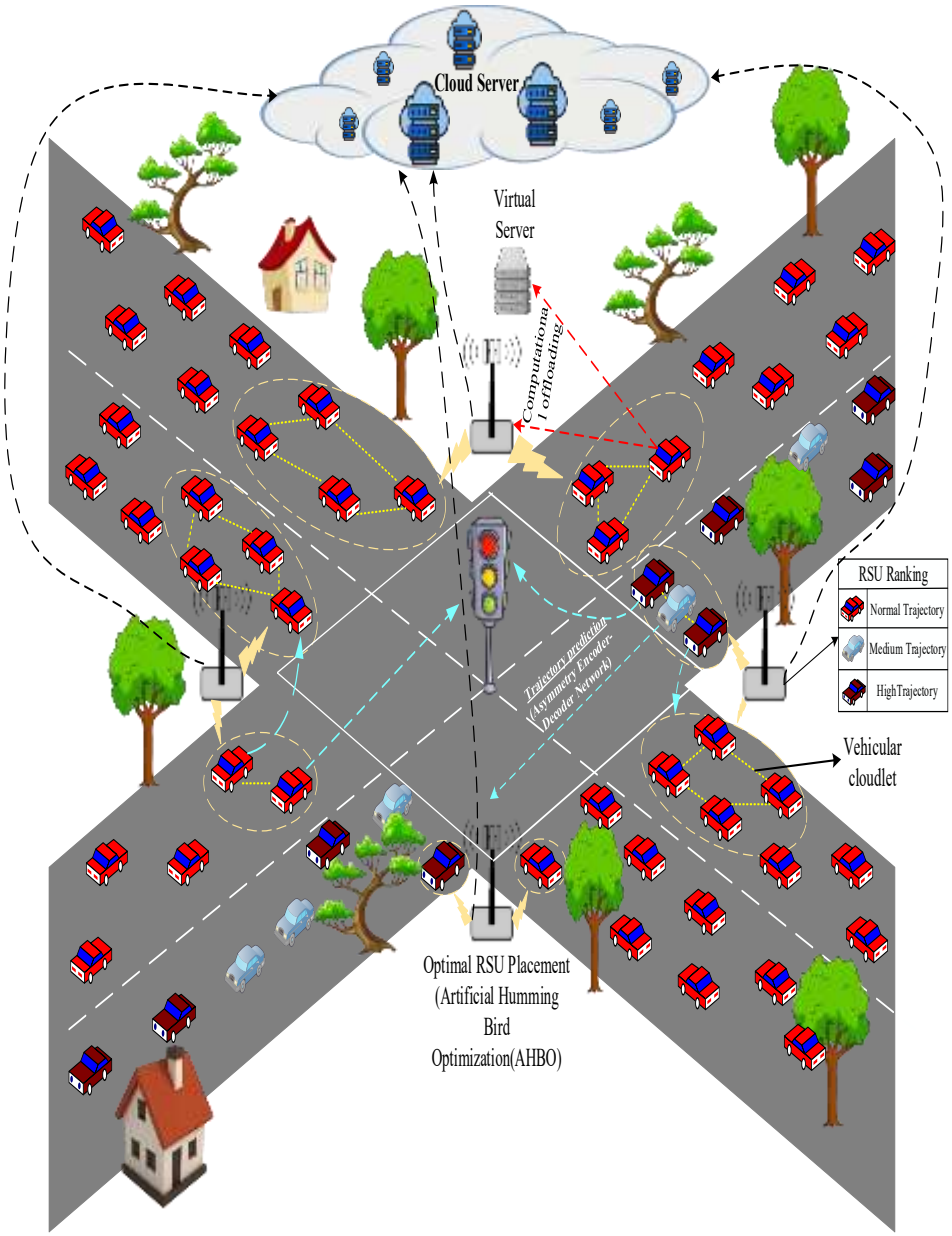

Fig.1 Overall Architecture of PCO-VN Model for Predictive Computation offloading

As per equation (8), when the rate of nectar refill surpasses its current pace, the hummingbird abandons its present food source in favor of feeding at the hummingbird feeder. This decision is influenced by the food supply, as described in Equation (6), and the AHBO-guided foraging strategy.

Territorial Foraging: This bird is more likely to look for an efficient nectar source than to explore other available food sources after spending time at the nectar source of target and finishing off the food from the f litter. Because of this, a hummingbird may simply relocate to a neighbouring location inside its own territory, where it may find a different nectar supply that is possibly better than the one it now uses. The mathematical formula for simulating the local search in their territorial foraging strategy and potential nectar supply is as follows:

$$h_i(t+1) = \mathfrak{X}_i(t) + b \cdot \mathfrak{D} \cdot \left(\mathfrak{X}_i(t) - \mathfrak{X}_i(t)\right) \quad (9)$$

$$b \sim N(0,1) \quad (10)$$

The parameter b represents a territorial component, which conforms to the normal distribution $N(0,1)$, with a mean of 0 and a standard deviation of 1. Each hummingbird possesses the capability to swiftly discern any changes in the nearby food supply by utilizing its distinctive flight characteristics. After the completion of the territorial foraging method and the AHBO territorial foraging approach, adjustments should be made to the visiting table.

Migration Foraging: While a hummingbird finds its preferred feeding location depleted, it will seek out a distant food supply. The AHBO algorithm introduces a migration coefficient, dictating the bird's migration to a newly generated food source across the entire search space if the conjunction rate surpasses this coefficient. Currently positioned at the food source with the poorest rate of nectar-refilling, the hummingbird will transition to the new

source upon migration, automatically updating its visiting table while discontinuing the use of the old one. Algorithm 4 outlines the migrating foraging strategy of AHBO. Hummingbird migration, shifting from the source with the lowest nectar replenishment rate to a freshly generated, randomly selected source, can be designated as follows:

$$\mathfrak{X}_{nrr}(t+1) = Low + r.(Up - Low) \quad (11)$$

Here, $\mathfrak{X}_{nrr}$ denotes the refilling rate of nectar in food source of weakest populations. By this way, the RSU are optimally placed for provided the resource appropriately.

B. Trajectory Planning

After that, the vehicles in the environment forms the vehicular cloudlet where the AVs and RSUs communicate within the communication range for data and resource sharing. Within that, we perform trajectory prediction for the AVs using Attention-based Asymmetry Encoder-Decoder Network (AEDNet) by considering several metrics. We've devised an encoder–decoder model featuring an asymmetric structure, illustrated in Fig. 2. This asymmetry is evident in both the scale of the neural network and the methods applied. The encoder transforms variable-length input sequences into fixed-length context vectors, while the decoder forecasts trend changes following the input sequence. Typically, simpler deep learning models sacrifice performance on complex datasets for reduced memory consumption and faster training. To address this, we've adjusted the model's complexity by modifying the number of hidden layers. Our proposed asymmetric model features a simpler encoder and a complex decoder employing GRU implementations and gating mechanisms, resulting in superior performance through optimized computation, memory usage, and efficiency. Additionally, asymmetry extends to the application methods; the encoder excludes dropout, whereas the decoder incorporates it.

Attention Mechanism-based Model Optimization Mechanism: The encoder–decoder model excels in sequence prediction tasks nevertheless struggles with lengthy time series, resulting in reduced accuracy. Its variable-length nature yields unstable prediction performance, influenced by input sequence lengths. This instability stems from uneven information distribution within the context vector, with the decoder lacking full input sequence details. Additionally, fixed-length contexts hinder accurate predictions for varying input lengths. An attention mechanism to address this limitation, inspired by human information processing. This mechanism enables prioritization of relevant information, enhancing prediction accuracy. The global attention mechanism structure is depicted in Fig. 2, facilitating item importance determination through attention weights. The attention weight is acquired by associating the output of hidden layer $\mathfrak{y}_{t-1}$ in decoder and one of encoder in prior circumstances which is calculated as,

$$Sco(\mathfrak{y}_t, \overline{\mathfrak{y}_s}) = \mathcal{V}_a{}^T tanh(\mathfrak{W}a[\mathfrak{y}_t; \overline{\mathfrak{y}_s}]) \quad (12)$$

Next, this weight is adjusted to conform with the normalization procedure outlined in Equation (13):

$$a_t(s) = \frac{e^{Sco(\mathfrak{y}_t, \overline{\mathfrak{y}_s})}}{\sum_{s'} e^{Sco(\mathfrak{y}_t, \overline{\mathfrak{y}_s})}} \quad (13)$$

Here, $\overline{\mathfrak{y}_s}$ represents the output of single hidden layers within the encoder. The attention weight determines the weighted sum of overall layer outputs of encoder hidden, contributing to the formation of the context vector in the attention mechanism depicted by the purple box. The computation formula for this process is defined in Equation (14).

$$c_t = \sum_{s'} a_t(s') * \overline{\mathfrak{y}_{s'}} \quad (14)$$

To generate the expected outcome, the context vector c_t from the attention mechanism and the hidden decoder output at the specific time step are both inputted into a linear NN model. This model, depicted by a green box, performs the computation outlined in Equation (4).

$$\overline{\mathfrak{y}_t} = tanh(\mathfrak{W}a[c_t; \mathfrak{y}_t]) \quad (15)$$

The attention mechanism within the encoder-decoder model shares similarities with the attention principle observed in human information processing. To enhance prediction accuracy, the decoder focuses on input data relevant to the prediction task. Through the attention mechanism, the decoder's output is influenced by both the context and the input sequence, thereby mitigating the challenges posed by fixed-duration contexts and variations in decoding information. The central idea revolves around determining the significance of each item within the input sequence for the decoder's prediction and generating the output accordingly. By transmitting a new context vector to the decoder, the insights gathered through the attention mechanism can significantly enhance the accuracy of the model's Remaining Useful Life (RUL) prediction for zinc-ion batteries.

Hyperparameter optimization: Table 1 presents the Bayesian optimization algorithm. Initially, a random set of observational data is extracted from the DL model to initialize the model of Bayesian statistical. Subsequently, candidate hyperparameter points are derived using the posterior distribution of the model of Bayesian statistical. Lastly, the sampling function is utilized to identify the following set of optimal hyperparameters. To employ the Bayesian statistical model, begin by incorporating the unique set of actual sampling data into the preliminary observation dataset. Then, apply the hyperparameters identified by the sampling function to the real DL model. Then, determine the actual sampling value equivalent to the hyperparameter point. The model's acquisition and update function estimates the optimal locations for hyperparameters. As N iterations progress through

hyperparameter estimation and actual model sampling, the data in the experiential dataset steadily increases. Ultimately, the optimal hyperparameter is the one corresponding to the ideal sampled value of the model.

In Bayesian statistical models, the black-box function F is typically approximated using a Gaussian process, where hyperparameters serve as independent variables. This is demonstrated by Equation (16):

$$\wp(\mathrm{F}) = gp\ (\mathrm{F}; \vartheta, H) \qquad (16)$$

Equation (17) depicts the posterior distribution of the observation dataset $d = (x, y)$.

$$\wp(\mathrm{F}|d) = gp\ (\mathrm{F}; \vartheta_{\mathrm{F}|d}, H_{\mathrm{F}|d}\ ) \qquad (17)$$

Where H represents the covariance matrix and ϑ denotes the mean vector of the probability distribution.

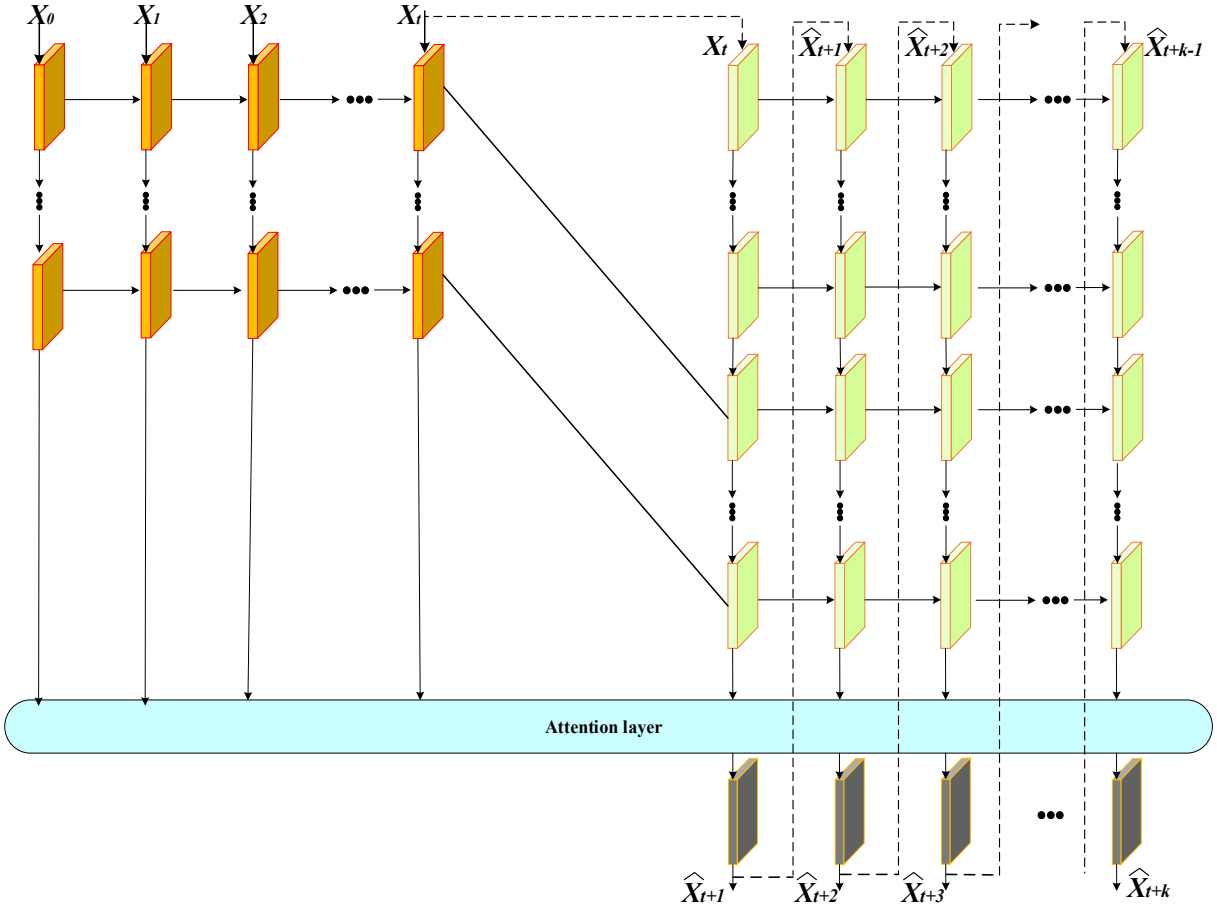

Fig.2 Illustration of AEDNet

The posterior distribution serves as a basis for computing the estimated loss value for each hyperparameter point within the context of Bayesian decision theory. Subsequently, the hyperparameter associated with the finest point among all appraised values is selected. A widely used sampling technique for optimizing hyperparameters of the supreme value of the black-box function is the Upper Confidence Bound (UCB). Equation (18) presents the formula for its calculation.

$$a_{UCB}(x; \gamma) = \vartheta(x) + \gamma\beta(x) \qquad (18)$$

If the hyperparameter optimization problem concerns minimizing the value rather than maximizing it using the Upper Confidence Bound (UCB) method, the calculation formula should be revised to Eq. (8).

$$a_{UCB}(x; \gamma) = \vartheta(x) - \gamma\beta(x) \qquad (19)$$

Here, γ denotes a trade-off parameter, while $\vartheta(x)$ and $\beta(x)$ represent the calculated mean and variance, respectively, of the posterior distribution of the Gaussian process. The estimated optimal hyperparameter is determined by selecting the highest (or lowest) value $a_{UCB}(x; \gamma)$, and this hyperparameter is then utilized by the model in a single resampling. Under certain conditions, the Upper Confidence Bound (UCB) method may converge towards the global optimum value of the model when compared to alternative sampling functions. Consequently, the sampling function employed for Bayesian optimization in this study is UCB.

C. AVs Ranking using MCDM

Based on trajectory prediction results, the RSUs ranked the AVs from low to higher moving AVs from the RSU range using ***Multi-Criteria Decision Making (MCDM) algorithm***. The utilization of the MCDM method is essential for selecting an optimal AVs ranking, considering various evaluation criteria. However, when confronting biased data, such as class imbalance, where a majority of data prevails, the selection of the optimal model based solely on one or a few evaluation criteria, such as accuracy or precision, becomes inadequate. To

address this issue, we explore MCDM, which integrates multiple evaluation factors with varying degrees of influence, offering a comprehensive solution. Notably, certain evaluation criteria, like accuracy and precision, are anticipated to carry higher importance, while others, such as false positive and false negative rates, hold lesser significance. The combination of Entropy and TOPSIS methods stands out as a widely adopted approach in multicriteria decision making. Entropy is employed to determine the weight of each assessment criterion, while TOPSIS utilizes a decision matrix to manage these weights, ultimately yielding the best-performing model. TOPSIS offers several advantages, including its adaptability to a broad spectrum of alternatives and attributes, its user-friendly nature, and its consistent processing steps for any given number of attributes. The decision matrix, pivotal to the TOPSIS approach, is constructed by applying the definition of evaluation criteria values for each choice as defined following mathematical equation.,

$$\partial = \begin{matrix} \forall_1 \\ \forall_2 \\ \vdots \\ \forall_m \end{matrix} \begin{pmatrix} C_1 & C_2 & \cdots & C_n \\ Y_1 & Y_2 & \cdots & Y_{1n} \\ Y_{21} & Y_{22} & \ddots & Y_{2n} \\ Y_{m1} & Y_{m2} & \cdots & Y_{mn} \end{pmatrix} \quad (20)$$

Here, $\forall_1, \forall_2, \ldots, \forall_m$ is denoted as substitutions to ranking as per the estimate criteria and $C_1, C_2, \ldots, C_n$. Y_{ij} is the substitutions score $\forall_i$ correlated to criterion C_j. The weighting of criteria is established through an entropy-based system, which quantifies the information embedded within the decision matrix, a prerequisite for the TOPSIS methodology's development. Alongside quantifying data through entropy, proportional weights are also computed to ensure comprehensive analysis. Algorithm 1 succinctly encapsulates the entire process of determining the weight for each assessment criterion, designed for ease of understanding. Here, Y_{ij} represents the value of the j-th criterion in the i-th option, ∂ given that there are m choices and criteria in total. The process encompasses standardization of indices, element-wise projection, entropy assessment of the j-th index, and weight calculation for each criterion, constituting various stages within the procedure. To executing the initial two TOPSIS procedures: normalization of the decision matrix and generation of the weighted decision matrix. The subsequent formulas can be employed to ascertain the ideal worst and ideal best solutions:

$$\nabla^{+} = \left\{ \left(\max_j \nabla_{ij} \,|j \in \mathcal{J}_{+} \right), \left(\min_j \nabla_{ij} \,|j \in \mathcal{J}_{-} \right);\ \mathfrak{U}_i \right\} \quad (21)$$

$$\nabla^{-} = \left\{ \left(\min_j \nabla_{ij} \,|j \in \mathcal{J}_{+} \right), \left(\max_j \nabla_{ij} \,|j \in \mathcal{J}_{-} \right);\ \mathfrak{U}_i \right\} \quad (22)$$

In this context, the criteria exerting positive and negative impacts are represented by $\mathcal{J}_{+}$ and $\mathcal{J}_{-}$, respectively. Step four involves computing the distance between each feasible option and both the ideal negative and positive solutions. Subsequently, step 5 evaluates the relative proximity to the ideal solution, while step 6 ranks the evaluation alternatives based on the corresponding level of closeness.

D. DRL-based Predictive Computational Offloading

With those ranking results, the virtualized computation offloading is performed using ***Deep Reinforcement Learning (DRL) algorithm*** as shown in figure 3. More clearly, the DRL agent initially looks ups the available AVs for offloading, if the all the AVs are overloaded then the task is offloaded to RSUs otherwise virtualized computing sever is created for timely offloading. Three categories can be identified for Reinforcement Learning (RL) algorithms, according to the RL control framework outlined above:

(i) Value repetition (e.g., deep $\mathbb{Q}$ network [DQN]);
(ii) Policy repetition (e.g., deterministic policy gradient [DPG]);
(iii) Actor-critic approach.

In this investigation, the DDPG algorithm, which amalgamates the assets of DQN and DPG, is employed. The experimental state within the RL controller is often high-dimensional, particularly for dynamic systems characterized by nonlinear and stochastic reservations, including as computation offloading approach. However, when estimating the action-value function The traditional value iteration techniques, such as $\mathbb{Q}$-learning an esteemed off-policy approach that utilizes the greedy policy $\psi(\mathfrak{s}_h) = \text{argmax}_{a_h} \mathbb{Q}(\mathfrak{s}_h, a_h)$in discrete action spaces—encounter challenges related to dimensionality. In this context, the DQN method harnesses the formidable deterministic nonlinear fitting capability of neural networks. Equation (23) indicates that the network is parametrized by $\phi^{\mathbb{Q}}$ to approximate $\mathbb{Q}(\mathfrak{s}_h, a_h)$. This estimation is subsequently refined by minimizing the loss function through the temporal-difference learning method.

$$\mathcal{R}(\phi^{\mathbb{Q}}) = \mathbb{W}_{\mathfrak{s}_h \sim \varphi^{\alpha}, a_h \sim \alpha, \tau_h \sim \mathbb{W}}[(\mathbb{Q}(\mathfrak{s}_h, a_h | \phi^{\mathbb{Q}}) - X_h)^2] \quad (23)$$

Where φ^{α} represents the discounted visitation distribution for a probabilistic action distribution, and α denotes any stochastic behavior policy with Ornstein-Uhlenbeck noise added. The target reward for the current action $\psi(\mathfrak{s}_{h+1})$ is defined as $X_h = \tau(\mathfrak{s}_h, a_h) + \partial \mathbb{Q}(\mathfrak{s}_{h+1}, \psi(\mathfrak{s}_{h+1}) | \phi^{\mathbb{Q}})$. However, employing a greedy strategy at each time step would lead to an extensive optimization process. Instead, in this scenario, DPG defines the policy performance as illustrated by Equation (24), utilizing a neural network parametrized by ϕ^{p} to characterize the existing deterministic policy $\psi(\mathfrak{s}_h)$.

$$\mathcal{M}^{\phi^p} = \mathbb{W}_{s_h-\varphi^\alpha}\left[\mathbb{Q}\left(s_h, \psi(s_h|\phi^p)\right)\right] \quad (24)$$

Here, $\psi(s_h|\phi^p)$ represents the NN weighted by ϕ^p that defines the controller's selected policy. Equation (24) demonstrates how the gradient descent approach can be utilized to estimate the parameter ϕ^p in Equation (25).

$$\nabla_{\phi^p}\mathcal{M}^{\phi^p} = \frac{1}{\mathbb{N}}\sum_{i=h}^{\mathbb{N}} \nabla_a \mathbb{Q}(s_h, \psi(s_h)|\phi^p)|_{\nabla_{\phi^p}} \nabla_{\phi^p}\psi\left((s_h|\phi^p)\right) \quad (25)$$

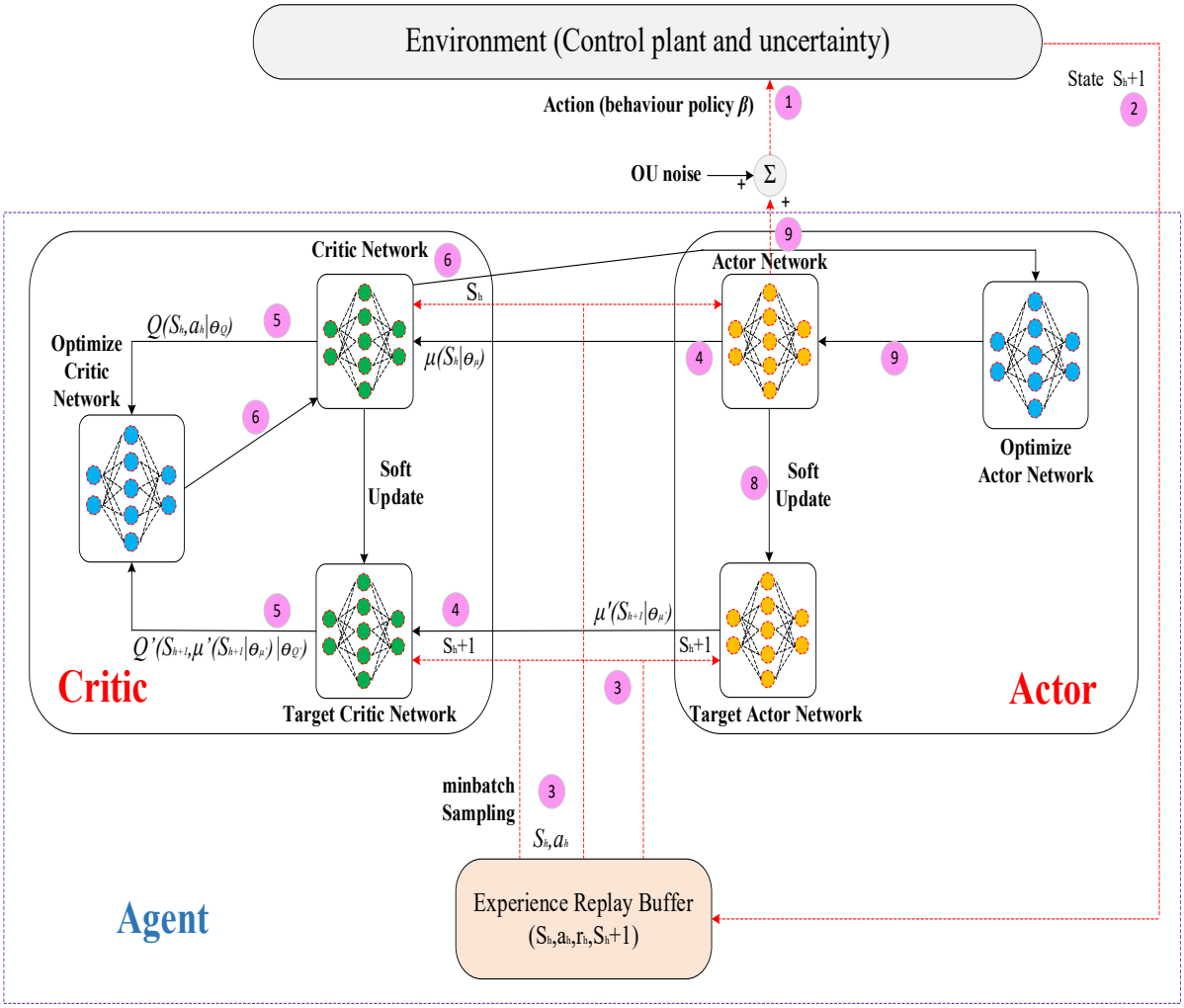

Fig.3 DDPG based Predictive Computation Offloading

Here, $\nabla_{\phi^p}\mathcal{M}^{\phi^p}$ denotes the sampled policy gradient, and ∇ represents the gradient operation. Typically, several optimizers as per gradient descent, including as Adam method, are employed to optimize the sampled policy gradient. In summary, DDPG merges DQN and DPG to attaining faster and additional constant conjunction. Figure 3 demonstrates how DDPG incorporates the following networks:

(i) Actor network ψ (same as DPG);

(ii) Target critic network $\mathbb{Q}'$ (parametrized by $\phi^{\mathbb{Q}'}$);

(iii) Target actor network ψ'(parametrized by $\phi^{\psi'}$);

(iv) The Critic network $\mathbb{Q}$ is identical to DQN.

The weights of the target networks are updated using the following soft-update strategy:

(i) $\phi^{\mathbb{Q}} \leftarrow \omega\phi^{\mathbb{Q}} + (1+\omega)\phi^{\mathbb{Q}}$; and

(ii) $\phi^{\psi'} \leftarrow \omega\phi^{\psi} + (1+\omega)\phi^{\psi}$. Here, the soft update factor ω (approximately 1) controls the rate of updating, leading to a slower process. Similar to the loss function $\mathcal{R}(\phi^{\mathbb{Q}})$ is minimized to train the weights of the critic network $\mathbb{Q}$. However, in the context of DDPG, utilizes the target networks to compute the action-value function in the target reward, given by $X_h = \tau(s_h, a_h) + \partial\mathbb{Q}'(s_{h+1}, \psi(s_{h+1})|\phi^{\mathbb{Q}})$. Equation (26), which demonstrates the recasting of the loss function $\mathcal{R}(\phi^{\mathbb{Q}})$, illustrates this process.

$$\mathcal{R}(\phi^{\mathbb{Q}}) = \mathbb{W}_{s_h\sim\varphi^\alpha, a_h\sim\alpha, \tau_h\sim\mathbb{W}}\left[\left(\mathbb{Q}(s_h, a_h|\phi^{\mathbb{Q}}) - \left[\tau_h + \partial\mathbb{Q}'\left(s_{h+1}, \psi'\left(s_{h+1}|\phi^{\psi'}\right)\middle|\phi^{\phi^{\mathbb{Q}'}}\right)\right]\right)^2\right] \quad (26)$$

Where $(\mathfrak{s}_{h+1}|\phi^{\mathbb{Q}'})$ is the action-value function approached by the network of target critic with weights $\phi^{\mathbb{Q}'}$, assumed state $\mathfrak{s}_{h+1}$ and action ψ. $\psi'(\mathfrak{s}_{h+1}|\phi^{\psi'})$ is the action approached by the target actor network with weights $\phi^{\psi'}$, given state $\mathfrak{s}_{h+1}$. Utilizing the experience replay buffer, a mini-batch of records can be selected as the training involvement, aimed at mitigating serial correlations among samples while efficiently storing a limited number of tuples. The complete progression of the DDPG algorithm stages is illustrated after the random initialization of the actor network $\psi(\mathfrak{s}_h|\phi^{\psi})$ and the critic network $\mathbb{Q}(\mathfrak{s}_h, a_h|\phi^{\mathbb{Q}})$ (stages 1-9 are depicted in Figure 3).

IV EXPERIMENTAL RESULTS

In this section, simulations are conducted to validate the impact of the DDPG-based offloading approach on the functionality of our proposed loading system task. Initially, we present the parameter configurations and simulation scenarios. Subsequently, we analyze the results obtained from the simulation.

A. Simulation Setup

We utilize internet resources to access traveler information, particularly relying on traffic trajectory data derived from the DC competition's traffic c line access time prediction, available at https://js.dclab.run/v2/cmptDetail.html?id=318. This data, encompassing historical taxi trajectory information, is publicly accessible, containing details such as cab ID, location coordinates, traveling speed, driving direction, and timestamp. Considering the presence of surrounding vehicles, we assume that vehicles passing through the handover region of two neighboring Roadside Units (RSUs) do so at a realistic travel pace. With an 800-meter communication range, the 300-meter space between the two RSUs is designated as the handover area. Vehicles are tasked with various assignments, prioritized by processing time within a specified delay limit, as they traverse this handover area from the starting point to the destination in each iteration. Initially, we designate $z_m = 0$ as the origin of task creation for the vehicle and determine the traveling speed based on processing data. Table 1 provides a snapshot of the data processing outcomes.

Given the dynamic nature of the state space across time slots, the mission vehicle selects the optimal loading scheme to process generated tasks, constituting a training step. The iteration concludes as the vehicle exits the handover area. Post data processing, channel gains for Vehicle-to-RSU (V2R) and Vehicle-to-Vehicle (V2V) communication environments are derived from the dataset. The job size $\mathfrak{L}_m$ is generated in MB for task m, with each Mb corresponding to $\mathfrak{C}_m = 500$. CPU frequency allocation is proportional to job load. Cooperative vehicles, or RSUs, dedicate their computational capabilities to task execution. Table 2 summarizes the simulation parameters. We operate within the ow1.13.1 environment version. Both the behavior Q network and DDPG of our Deep Reinforcement Learning (DRL) model consist of two layers. The RL experience pool size is set at 2000, with 32 samples being extracted per iteration. A greedy probability of 0.9 and a reward decay factor of 0.9 are utilized.

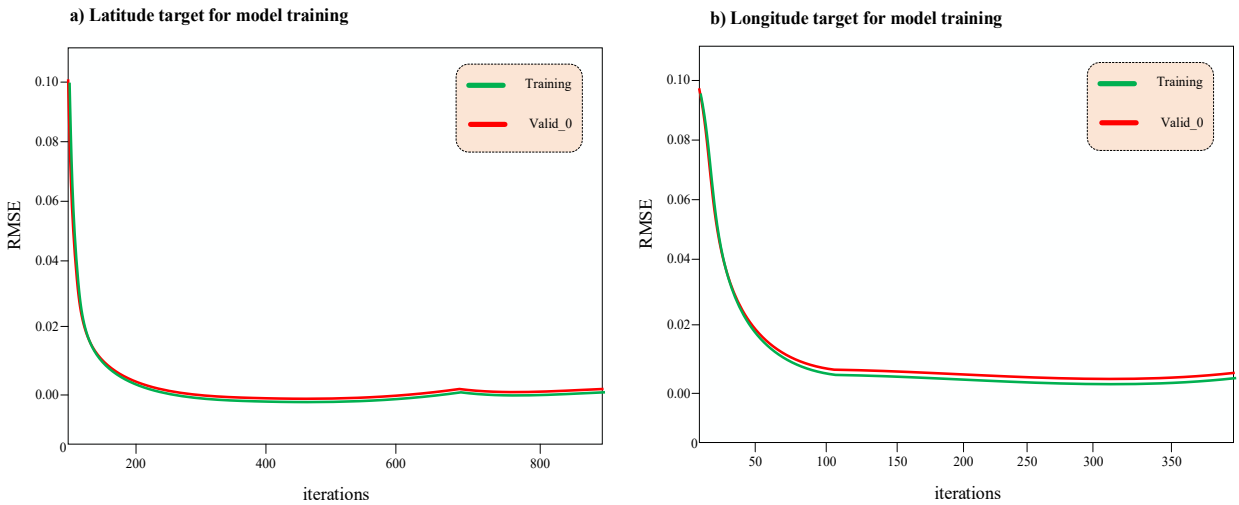

Fig.4 Model Training Latitude and Longitude Target

Table 1 Default Setup of Parameters

Description	Value
V2R, V2V Spectrum bandwidth	10, 6MHz
V2R, V2V Transmit power	20, 20dBm
Noise of background	-19dbm
Task execution in maximum delay	0.1s
Backhaul link in transmission power	1W

Backhaul link in transmission rate	5Mb
Cost of unit in backhaul link	2units/Mbps
Proceeds of task execution	2units/Mbps
Proceeds of access in task transmission	2units/Mbps
Occupancy fee of V2R bandwidth	7.0units/MHz
Occupancy fee of V2V bandwidth	4.7units/MHz
Autonomous vehicles energy consumption	70GHz/W
RSU energy consumption	40GHz/W
Local energy consumption	70GHz/W
Autonomous vehicles expenditure estimation	0.06W/units
RSU expenditure estimation	0.10W/units
Local expenditure estimation	0.04W/units
Autonomous vehicles maximum frequency estimation	30GHz
RSU's maximum frequency estimation	40, 60GHz

B. Simulation Results

(i) Trajectory Planning

We begin by feeding the dataset into the model for training. Since the proposed model necessitates a 1-D numpy array, we partition the longitude and latitude coordinates into separate 1-D numpy arrays for the training phase. The training procedure entails employing two distinct GBM models, as depicted in Figure 4. Here, Vad_0 denotes the RMSE value of the validation set, while $train$ signifies the RMSE value of the training set. It's worth noting that there's negligible variance in the training error observed between longitude and latitude inputs for the AEDNet model. This uniformity arises from both inputs leveraging the same extracted features during model training. Consequently, the square root of the RMSE of the current longitude and latitude can serve as a substitute for the current position, defined as,

$$RMSE_{pos} = \sqrt{RMSE_{lat}^2 + RMSE_{Long}^2} \qquad (27)$$

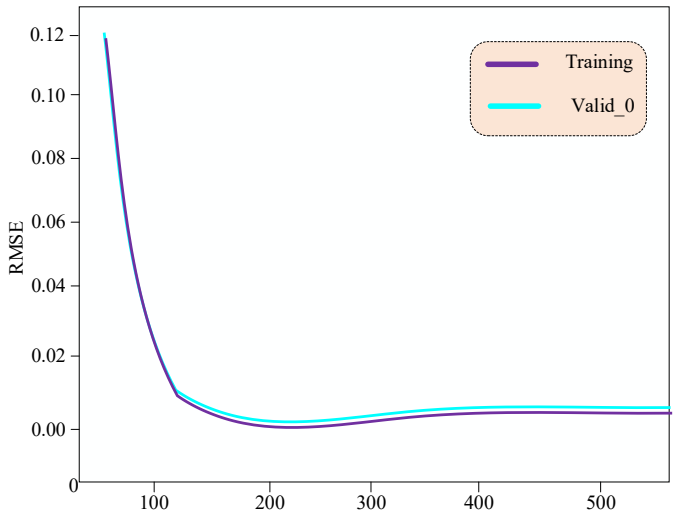

Fig.5 Offloading Approach

Figure 5 illustrates the training process of the AEDNet model, showcasing a comparison between the RMSE values of the training and validation sets. Notably, while these values are comparable, it's evident that the RMSE of the validation set exceeds that of the training set. This observation generally indicates the accuracy of the trajectory forecast. Figure 6 illustrates the advantages of the AEDNet -based TO scheme utilizing various V2V cooperative vehicles for trajectory prediction. These vehicles are selected to undertake loading tasks, with varying distances among the mission vehicle and their computational capabilities. Through trajectory prediction, we confirm that vehicle A remains the optimal choice, consistent. Additionally, three more vehicles are selected for comparison from the simulation dataset: Vehicle C, boasting greater computational power than Vehicle B but with a longer average distance to the mission vehicle; Vehicle B, with a shorter average distance but less computational power; and Vehicle D, which tends to lag behind the mission vehicle. Analysis of Fig. 6 reveals that the TO

Scheme integrating AEDNet and trajectory prediction yields the highest reward. It is evident that opting for Vehicle B, Vehicle C, or Vehicle D would not maximize profit for the cooperative vehicle. Therefore, the selection of a cooperative vehicle for V2V communication is intricately linked to trajectory prediction.

(ii) DRL based Computational Offloading

To implement the proposed PCO-VN-based TO scheme, we establish five baseline schemes as follows:

- Q-learning TO Scheme (Q-learn): This scheme utilizes Q-learning for determining the loading strategy.
- Sarsa TO Plan (Sar-Sch): Sarsa algorithm is employed to formulate the loading plan.
- Delivering to RSU (RSU-Off): Assignments are exclusively delivered to RSU.
- Processing locally (Loc-Off): All processing tasks are performed locally.
- Operating cooperative vehicles only (CoV-OFF): Responsibilities pertaining solely to operating cooperative vehicles are assigned.

The relationship between the number of iterations and the rewards obtained by various loading techniques is illustrated in Figure 7. Each loading scheme maintains a fixed learning rate of 0.01. The total number of iterations is set to 1000, and learning commences for each scheme after 25 iterations. Around 220 iterations, the PCO-VN-based TO scheme reaches its optimal value, while both the Sar-Sch and the Q-learn achieve their optimal values after about 100 iterations. Notably, the PCO-VN-based TO scheme demonstrates significantly superior performance compared to previous benchmark systems. Moreover, the optimal values attained by the Sar-Sch and the Q-learn are comparable. This parity arises because the PCO-VN-based TO scheme employs the most efficient loading technique for each time slot, utilizing a neural network to replace and updating its weights through training on small batches of data from memory replay. Conversely, the RSU-Off, Loc-Off, and CoV-Off schemes fail to achieve optimal system performance within the given time frame.

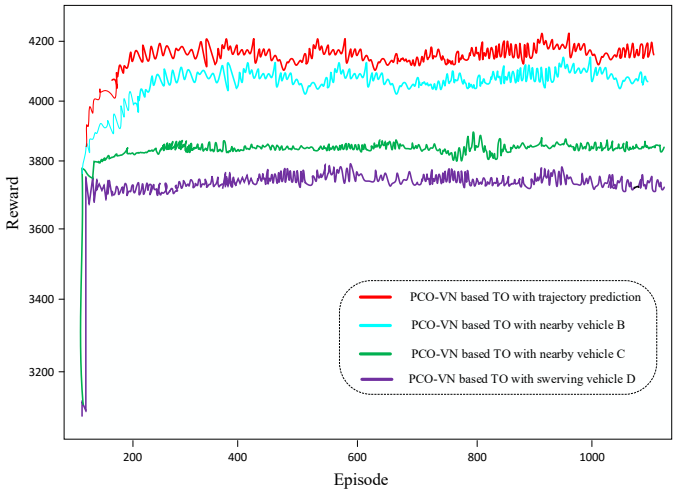

Fig.6 PCO-VN based Computation Offloading with Diverse V2V Collaborative Vehicle in Trajectory Prediction

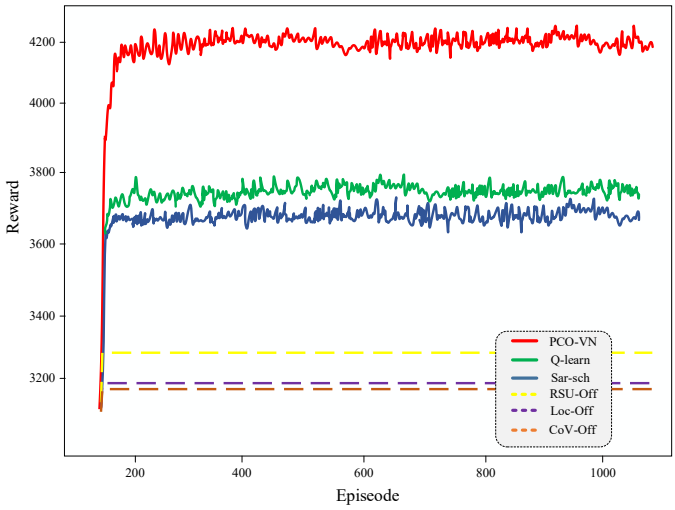

Fig.7 Reward of Diverse Offloading Approach

The execution times of various loading strategies are depicted in Figure 8, using the same simulation parameters as in Figure 7. Notably, while the Q-learn exhibits slightly longer execution times, the PCO-VN-based TO

Scheme's execution time is comparable to that of the Sarsa loading scheme. Additionally, RSU-Off, Loc-Off, and CoV-Off—having shorter execution times—do not necessitate training of the action network. Figure 10 illustrates the relationship between the number of iterations and the rewards of the PCO-VN -based TO Scheme, with varying learning rates. The learning rates of PCO-VN are adjusted to 0.01, 0.002, 0.001, and 0.0005. It is apparent that higher learning rates lead to quicker convergence, ultimately reaching the same optimal value as predicted. Proper adjustment of the learning rate directly enhances PCO-VN's convergence speed.

(iii) Comparison on Computational Offloading Existing Works

To evaluate and prove the efficiency of our proposed PCO-VN model, we compared our work with five existing works which are detailed as follows:

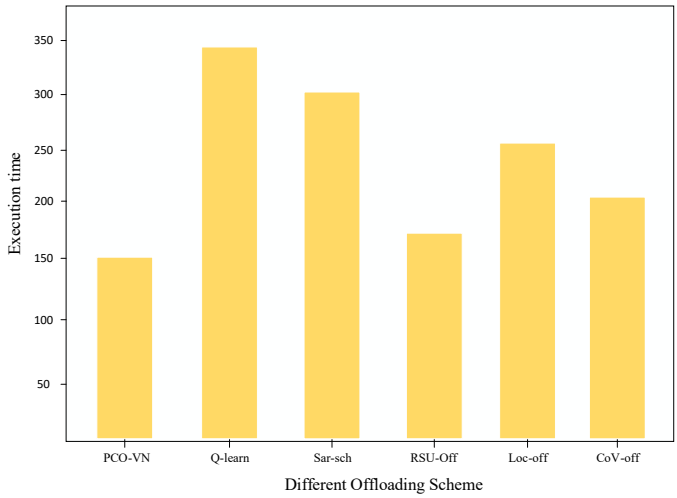

Fig.8 Execution Time for Diverse Offloading Approach

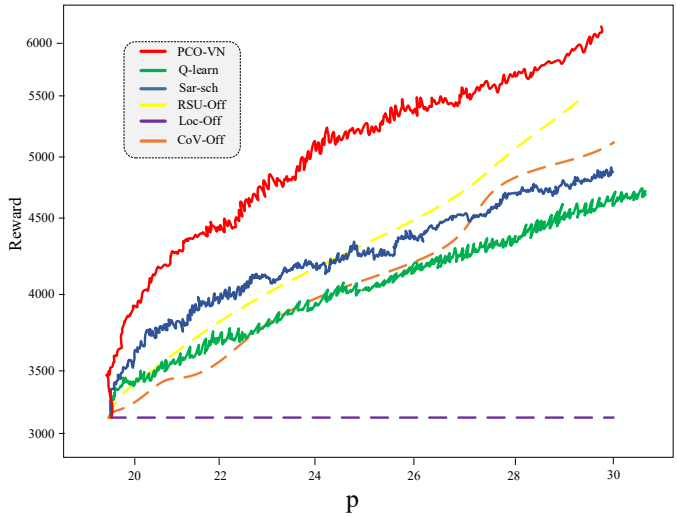

Fig.9 Reward of Diverse Offloading Approach under Transmit Powers

- MOCO [22]: In this research, computational offloading is performed in the basis of multi-objective in cloud-edge environment.
- JCO-DRL [23]: DRL approach is utilized in this work for performing both computation offloading and resource allocation. Besides, for improving offloading performance joint computation offloading is accomplished.
- Co-CO [31]: Collaborative optimization based computation offloading introduced in edge-cloud vehicular network.
- DTO-MG [32]: This work employs minority game for dynamic task offloading in IoV network.
- DQN [41: RL approach-based handover and task offloading is implemented using Q-learning algorithm. Furthermore, trajectory prediction is included in this study.

Figure 7 portrays the correlation between the number of iterations and rewards achieved through diverse loading techniques. Each technique maintains a constant learning rate of 0.01, with a total of 1000 iterations, starting after the initial 25 iterations for each scheme. At approximately 220 iterations, the PCO-VN-based TO scheme reaches its peak performance, surpassing previous benchmarks, while both MOCO and DQN reach their optimal values around 100 iterations. Notably, PCO-VN-based TO scheme outperforms significantly due to its efficient loading technique, employing performing trajectory prediction based co. Besides, DTO-MG performs dynamic offloading

ineffectively due to its ineffective parameters consideration. Conversely, JCO-DRL and Co-CO fail to achieve optimal performance within the given timeframe.

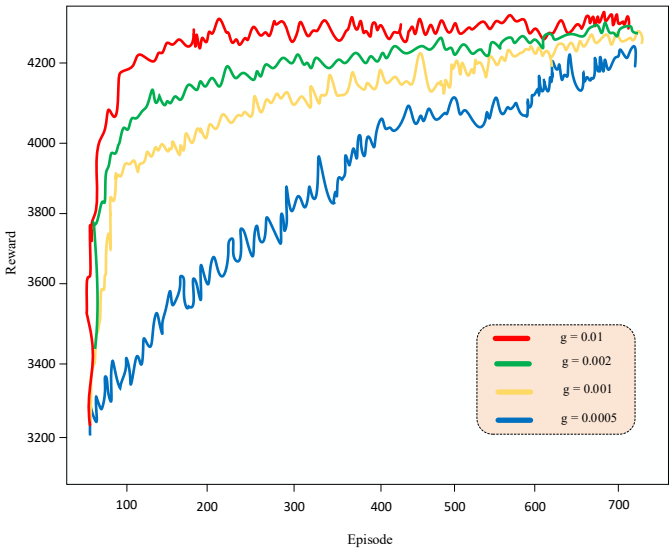

Fig.10 Reward of PCO-VN based Computation Offloading Approach with Diverse Learning Rates

Figure 8 illustrates the execution times of various loading strategies, mirroring Figure 9's simulation parameters. Despite slightly longer execution times for MOCO, the PCO-VN-based TO Scheme's time matches DQN. JCO-DRL, Co-CO and DTO-MG exhibit shorter execution times as they do not require training of the action network. In Figure 10, the relationship between iteration numbers and rewards for the PCO-VN-based TO Scheme is shown with different learning rates: 0.01, 0.002, 0.001, and 0.0005. Higher learning rates lead to faster convergence, reaching the same optimal value predicted. Proper adjustment of the learning rate directly impacts PCO-VN's convergence speed.

C. Model Evaluation

Our proposed model for optimizing computation offloading in the vehicular cloud environment undergoes rigorous evaluation to assess its effectiveness. The model integrates predictive offloading techniques with cloud and hybrid mobile edge computing technologies to enhance efficiency and resource utilization

- Optimized RSU Placement: Leveraging the Artificial Humming Bird Optimization (AHBO) algorithm ensures strategic placement of Road Side Units (RSUs), maximizing coverage and minimizing latency.
- Vehicular Cloudlet Formation: By forming vehicular cloudlets, our model facilitates efficient communication and resource sharing between Autonomous Vehicles (AVs) and RSUs within their communication range.
- Precise Trajectory Prediction: The Attention-based Asymmetry Encoder-Decoder Network (AEDNet) accurately predicts AV trajectories, enabling proactive decision-making and task allocation.
- Ranked AV Selection: Using a Multi-Criteria Decision Making (MCDM) algorithm, AVs are ranked based on mobility within RSU range, optimizing resource allocation and offloading decision.
- Dynamic Offloading with DRL: Employing a Deep Reinforcement Learning (DRL) algorithm enables dynamic offloading decisions. The model intelligently distributes tasks among available resources, be it AVs or RSUs, to minimize latency and maximize throughput.
- Resource Utilization: Through virtualized computation offloading, our model ensures optimal utilization of available resources, preventing overloading and maximizing system efficiency.
- Timely Offloading: In cases where AVs are overloaded, our model swiftly offloads tasks to RSUs or creates virtualized computing servers, ensuring timely execution of tasks.

Overall, the proposed PCO-VN model demonstrates significant effectiveness in optimizing computation offloading within the vehicular cloud environment. By leveraging predictive techniques and advanced algorithms, it enhances resource utilization, minimizes latency, and maximizes overall system efficiency.

IV. CONCLUSION

In conclusion, the presented Computational Offloading Model for Vehicular Networks, leveraging Hybrid Mobile Edge and Cloud Computing Technologies, introduces a pioneering approach to tackle the computational limitations faced by vehicles in dynamic environments. By integrating MEC and cloud computing, this model aims to alleviate the computational burden on vehicles by offloading tasks to nearby edge servers or remote cloud data centers, thereby enhancing the efficiency and scalability of vehicular networks. However, despite the promising benefits, several limitations exist. This study sought to optimize the computation offloading workflow in the vehicular cloud environment through the development of a predictive offloading model employing cloud

and hybrid mobile edge computing technologies. The research began by strategically placing RSUs as mobile edge servers using the AHBO algorithm, which maximizes efficiency by considering multiple variables. Subsequently, vehicular cloudlets were formed, facilitating communication and resource sharing between AVs and RSUs within their respective communication ranges. Trajectory prediction for AVs was conducted using the AEDNet, enhancing prediction accuracy by considering various parameters.

Following trajectory prediction, AVs were ranked within RSU ranges using a MCDM algorithm based on prediction results. Virtualized compute offloading was then performed using DRL, where the DRL agent identified available AVs for offloading, creating virtualized computing servers for timely offloading or offloading tasks to RSUs if all AVs were overloaded. The proposed PCO-VN model was evaluated for efficiency, outperforming existing works in terms of effectiveness. This study demonstrates the potential of predictive offloading models in optimizing computation offloading in vehicular cloud environments. Despite the limitations, the findings underscore the importance of leveraging advanced technologies to enhance the performance and scalability of vehicular networks, paving the way for future advancements in intelligent transportation systems.

REFERENCE

1) Pahadiya, B., & Ranawat, R. (2023, December). A Review of Smart Traffic Operation System for Traffic Control Using Internet of effects & Reinforcement Learning. In *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)* (pp. 1-10). IEEE.

2) Zhao, J., Zhao, W., Deng, B., Wang, Z., Zhang, F., Zheng, W., ... & Burke, A. F. (2023). Autonomous driving system: A comprehensive survey. *Expert Systems with Applications*, 122836.

3) Himeur, Y., Sayed, A., Alsalemi, A., Bensaali, F., & Amira, A. (2023). Edge AI for Internet of Energy: Challenges and perspectives. *Internet of Things*, 101035.

4) Fazel, E., Najafabadi, H. E., Rezaei, M., & Leung, H. (2023). Unlocking the power of mist computing through clustering techniques in IoT networks. *Internet of Things*, *22*, 100710.

5) Khan, A., Gupta, S., & Gupta, S. K. (2022). Emerging UAV technology for disaster detection, mitigation, response, and preparedness. *Journal of Field Robotics*, *39*(6), 905-955.

6) Tran-Dang, H., Krommenacker, N., Charpentier, P., & Kim, D. S. (2022). The Internet of Things for logistics: Perspectives, application review, and challenges. *IETE Technical Review*, *39*(1), 93-121.

7) Walia, G. K., Kumar, M., & Gill, S. S. (2023). AI-empowered fog/edge resource management for IoT applications: A comprehensive review, research challenges and future perspectives. *IEEE Communications Surveys & Tutorials*.

8) prediction-based offloading emerges as a particularly promising avenue. This methodology harnesses advanced machine learning algorithms

9) Kong, X., Wang, J., Hu, Z., He, Y., Zhao, X., & Shen, G. (2024). Mobile Trajectory Anomaly Detection: Taxonomy, Methodology, Challenges, and Directions. *IEEE Internet of Things Journal*.

10) Qu, B., & Li, X. (2023). Optimizing Dynamic Cache Allocation in Vehicular Edge Networks: A Method Combining Multi-Source Data Prediction and Deep Reinforcement Learning. *IEEE Internet of Things Journal*.

11) Kuruvatti, N. P., Habibi, M. A., Partani, S., Han, B., Fellan, A., & Schotten, H. D. (2022). Empowering 6G communication systems with digital twin technology: A comprehensive survey. *IEEE access*, *10*, 112158-112186.

12) Zhumayeva, M., Dautov, K., Hashmi, M., & Nauryzbayev, G. (2023). Wireless energy and information transfer in WBAN: A comprehensive state-of-the-art review. *Alexandria Engineering Journal*, *85*, 261-285.

13) Zhumayeva, M., Dautov, K., Hashmi, M., & Nauryzbayev, G. (2023). Wireless energy and information transfer in WBAN: A comprehensive state-of-the-art review. *Alexandria Engineering Journal*, *85*, 261-285.

14) Singh, D., & Dwivedi, R. K. (2024). Designing blockchain based secure autonomous vehicular internet of things (IoT) architecture with efficient smart contracts. *International Journal of Information Technology*, 1-17.

15) Zhang, S., Li, J., Shi, L., Ding, M., Nguyen, D. C., Tan, W., ... & Han, Z. (2023). Federated Learning in Intelligent Transportation Systems: Recent Applications and Open Problems. *IEEE Transactions on Intelligent Transportation Systems*.

16) Agbaje, P., Anjum, A., Mitra, A., Oseghale, E., Bloom, G., & Olufowobi, H. (2022). Survey of interoperability challenges in the internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *23*(12), 22838-22861.

17) Hussein, N. H., Yaw, C. T., Koh, S. P., Tiong, S. K., & Chong, K. H. (2022). A comprehensive survey on vehicular networking: Communications, applications, challenges, and upcoming research directions. *IEEE Access*, *10*, 86127-86180.

18) Chelbi, N. E., Gingras, D., & Sauvageau, C. (2022). Worst-case scenarios identification approach for the evaluation of advanced driver assistance systems in intelligent/autonomous vehicles under multiple conditions. *Journal of Intelligent Transportation Systems*, *26*(3), 284-310.

19) Sacco, A., Esposito, F., Marchetto, G., & Montuschi, P. (2022). A self-learning strategy for task offloading in UAV networks. *IEEE Transactions on Vehicular Technology*, *71*(4), 4301-4311.

20) Ghosh, J., Kumar, N., Al-Utaibi, K. A., Sait, S. M., & So-In, C. (2024). Reliable data transmission for a VANET-IoIT architecture: A DNN approach. *Internet of Things*, 101129.

21) X. Hou et al., "Reliable Computation Offloading for Edge-Computing-Enabled Software-Defined IoV," in IEEE Internet of Things Journal, vol. 7, no. 8, pp. 7097-7111, Aug. 2020, doi: 10.1109/JIOT.2020.2982292.
22) Xu, Xiaolong & Gu, Renhao & Dai, Fei & Qi, Lianyong & Wan, Shaohua. (2020). Multi-objective computation offloading for Internet of Vehicles in cloud-edge computing. Wireless Networks. 26. 10.1007/s11276-019-02127-y.
23) Huang, J., Wan, J., Lv, B., Ye, Q., & Chen, Y. (2023). Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning. *IEEE Systems Journal*.
24) Ning, Zhaolong & Zhang, Kaiyuan & Wang, Xiaojie & Guo, Liang & Hu, Xiping & Huang, Jun & Hu, Bin & Kwok, Ricky. (2020). Intelligent Edge Computing in Internet of Vehicles: A Joint Computation Offloading and Caching Solution. IEEE Transactions on Intelligent Transportation Systems. PP. 10.1109/TITS.2020.2997832.
25) Ning, Zhaolong & Zhang, Kaiyuan & Wang, Xiaojie & Guo, Liang & Hu, Xiping & Huang, Jun & Hu, Bin & Kwok, Ricky. (2020). Intelligent Edge Computing in Internet of Vehicles: A Joint Computation Offloading and Caching Solution. IEEE Transactions on Intelligent Transportation Systems. PP. 10.1109/TITS.2020.2997832.
26) Ning, Zhaolong & Zhang, Kaiyuan & Wang, Xiaojie & Guo, Liang & Hu, Xiping & Huang, Jun & Hu, Bin & Kwok, Ricky. (2020). Intelligent Edge Computing in Internet of Vehicles: A Joint Computation Offloading and Caching Solution. IEEE Transactions on Intelligent Transportation Systems. PP. 10.1109/TITS.2020.2997832.
27) Hazarika, B., Singh, K., Biswas, S., & Li, C. P. (2022). DRL-based resource allocation for computation offloading in IoV networks. *IEEE Transactions on Industrial Informatics*, *18*(11), 8027-8038.
28) Zhang, Y., Zhang, M., Fan, C. *et al.* Computing resource allocation scheme of IOV using deep reinforcement learning in edge computing environment. *EURASIP J. Adv. Signal Process.* **2021**, 33 (2021). https://doi.org/10.1186/s13634-021-00750-6
29) Yao, L., Xu, X., Bilal, M., & Wang, H. (2022). Dynamic edge computation offloading for internet of vehicles with deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*.
30) Zhou, X., Bilal, M., Dou, R., Rodrigues, J. J., Zhao, Q., Dai, J., & Xu, X. (2023). Edge computation offloading with content caching in 6G-enabled IoV. *IEEE Transactions on Intelligent Transportation Systems*.
31) Zhou, X., Bilal, M., Dou, R., Rodrigues, J. J., Zhao, Q., Dai, J., & Xu, X. (2023). Edge computation offloading with content caching in 6G-enabled IoV. *IEEE Transactions on Intelligent Transportation Systems*.
32) Zhou, X., Bilal, M., Dou, R., Rodrigues, J. J., Zhao, Q., Dai, J., & Xu, X. (2023). Edge computation offloading with content caching in 6G-enabled IoV. *IEEE Transactions on Intelligent Transportation Systems*.
33) Zhai, Yanlong & Sun, Wenxin & Wu, Jianqing & Zhu, Liehuang & Shen, Jun & Du, Xiaojiang & Guizani, Mohsen. (2020). An Energy Aware Offloading Scheme for Interdependent Applications in Software-Defined IoV With Fog Computing Architecture. IEEE Transactions on Intelligent Transportation Systems. 22. 3813 - 3823.
34) Wan, Shaohua & Li, Xiang & Xue, Yuan & Lin, Wenmin & Xu, Xiaolong. (2020). Efficient computation offloading for Internet of Vehicles in edge computing-assisted 5G networks. The Journal of Supercomputing. 76. 10.1007/s11227-019-03011-4. \
35) Xu, X., Jiang, Q., Zhang, P., Cao, X., Khosravi, M. R., Alex, L. T., ... & Dou, W. (2022). Game theory for distributed IoV task offloading with fuzzy neural network in edge computing. *IEEE Transactions on Fuzzy Systems*, *30*(11), 4593-4604.
36) Raza Naqvi, Syed Salman & Wang, Shangguang & Ahmed, Manzoor & Anwar, Muhammad & Mirza, Muhammad & Khan, Wali Ullah. (2021). Task Offloading and Resource Allocation for IoV using 5G NR-V2X Communication. IEEE Internet of Things Journal. PP. 10.1109/JIOT.2021.3121796.
37) Lin, Bing & Lin, Kai & Lin, Changhang & Lu, Yu & Huang, Ziqing & Chen, Xinwei. (2021). Computation offloading strategy based on deep reinforcement learning for connected and autonomous vehicle in vehicular edge computing. Journal of Cloud Computing. 10. 10.1186/s13677-021-00246-6.
38) Ernest, T. Z. H., & Madhukumar, A. S. (2023). Computation Offloading in MEC-Enabled IoV Networks: Average Energy Efficiency Analysis and Learning-Based Maximization. *IEEE Transactions on Mobile Computing*.
39) Hazarika, B., Singh, K., Biswas, S., Mumtaz, S., & Li, C. P. (2022, June). SAC-based resource allocation for computation offloading in IoV networks. In *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)* (pp. 314-319). IEEE.
40) Liu, L., Yuan, X., Zhang, N., Chen, D., Yu, K., & Taherkordi, A. (2023). Joint computation offloading and data caching in multi-access edge computing enabled internet of vehicles. *IEEE Transactions on Vehicular Technology*.