

# Secure Cluster-Based Hierarchical Federated Learning in Vehicular Networks

M. Saeid HaghghiFard, *Graduate Student Member, IEEE* and Sinem Coleri, *Fellow, IEEE*

**Abstract**—Hierarchical Federated Learning (HFL) has recently emerged as a promising solution for intelligent decision-making in vehicular networks, helping to address challenges such as limited communication resources, high vehicle mobility, and data heterogeneity. However, HFL remains vulnerable to adversarial and unreliable vehicles, whose misleading updates can significantly compromise the integrity and convergence of the global model. To address these challenges, we propose a novel defense framework that integrates dynamic vehicle selection with robust anomaly detection within a cluster-based HFL architecture, specifically designed to counter Gaussian noise and gradient ascent attacks. The framework performs a comprehensive reliability assessment for each vehicle by evaluating historical accuracy, contribution frequency, and anomaly records. Anomaly detection combines Z-score and cosine similarity analyses on model updates to identify both statistical outliers and directional deviations in model updates. To further refine detection, an adaptive thresholding mechanism is incorporated into the cosine similarity metric, dynamically adjusting the threshold based on the historical accuracy of each vehicle to enforce stricter standards for consistently high-performing vehicles. In addition, a weighted gradient averaging mechanism is implemented, which assigns higher weights to gradient updates from more trustworthy vehicles. To defend against coordinated attacks, a cross-cluster consistency check is applied to identify collaborative attacks in which multiple compromised clusters coordinate misleading updates. Together, these mechanisms form a multi-level defense strategy to filter out malicious contributions effectively. Simulation results show that the proposed algorithm significantly reduces convergence time compared to benchmark methods across both 1-hop and 3-hop topologies. Under combined noise and gradient ascent attacks in the 1-hop scenario, the proposed algorithm reduces convergence time by 17.1% compared to cosine-similarity and Z-score defenses, and by 8.7% over the combined Z-score and cosine strategy. In the 3-hop setting under the same attacks, our proposed algorithm still leads with a 15.6% reduction against Z-score-only methods and a 7.1% improvement over the combined defense. These results demonstrate that the proposed framework consistently delivers superior resilience, accelerating convergence by up to 17% even under intense adversarial conditions.

**Index Terms**—Hierarchical federated learning, vehicular networks, anomaly detection, dynamic client selection

## I. INTRODUCTION

VEHICULAR Ad hoc Networks (VANETs) are increasingly adopting machine learning (ML) algorithms to improve safety and efficiency in transportation systems. These algorithms use sensor data from LIDAR, RADAR, and vehicle-mounted cameras to enable intelligent decision-making. They

M. Saeid HaghghiFard and Sinem Coleri are with the Department of Electrical and Electronics Engineering, Koc University, Istanbul, Turkey, email: {mhaghghiFard21, scoleri}@ku.edu.tr. This work is supported by the Scientific and Technological Research Council of Turkey Grant number 119C058 and Ford Otosan.

support applications such as location-based services, traffic flow prediction and control, and autonomous driving. Traditionally, centralized ML methods train neural networks using large datasets collected from vehicle edge devices. In contrast, federated learning (FL) moves model training to the edge, reducing transmission overhead, and preserving user privacy. In FL, edge devices send parameter gradients to a central server instead of raw data. The server aggregates these gradients to update the global model, which is then redistributed to the devices for continued training. This process is repeated iteratively until the model converges [1].

In recent years, Hierarchical Federated Learning (HFL) has emerged as a promising approach to address the challenges of deploying FL in vehicular networks. In HFL, a central cloud server coordinates global model training with the assistance of edge servers that act as intermediaries for model aggregation. This hierarchical approach helps mitigate issues related to limited communication resources, vehicular mobility, and data heterogeneity [2], [3]. By enabling vehicles to transmit local model updates to nearby edge servers instead of directly to the cloud, HFL reduces communication overhead and expands coverage. It is also more effective in managing the dynamic nature of vehicular mobility and the non-independent and identically distributed (non-IID) data across different edge servers. Several studies have demonstrated the practical benefits of HFL: [4] demonstrates that HFL can leverage vehicular mobility to accelerate convergence in the presence of data heterogeneity, as higher speeds enable more diverse data exchange and faster data fusion across nodes; [5] proposes a semi-asynchronous HFL framework for Cooperative Intelligent Transportation Systems (C-ITS), improving communication efficiency in the presence of heterogeneous road vehicles; and [6] introduces a Cluster-based HFL (Cb-HFL) approach that uses multi-hop clustering and customized metrics to address both mobility and non-IID data, achieving significant improvements in accuracy and convergence time while maintaining an acceptable level of packet overhead. Despite these advances, security vulnerabilities, especially against adversarial attacks, remain largely unresolved. One of the most critical threats is model poisoning, where adversaries manipulate local model updates sent from vehicles to edge servers or the cloud. The decentralized nature of FL and absence of strict monitoring of individual client updates provide an ideal setting for adversaries to execute such attacks undetected. Since FL assumes that most client updates are benign, the aggregation process is highly vulnerable to slight adversarial modifications, which can accumulate over time and severely disrupt global model convergence [7]–[9].

Several defense strategies have been proposed to address security threats in FL. Among them, anomaly detection mechanisms have been developed to distinguish between malicious and benign model behavior in response to model poisoning attack [10]–[13]. [10] introduces a differential privacy-exploited stealthy model poisoning (DeSMP) attack and proposes a reinforcement learning (RL)-based defense mechanism that dynamically adjusts privacy levels according to statistical metrics derived from client updates. Similarly, [11] presents a client-based defense mechanism called White Blood Cell for FL (FL-WBC), which identifies vulnerable regions in the parameter space during local training and actively perturbs them to neutralize the effect of poisoning. [12] proposes filtering out anomalous models by assessing the similarity in historical update patterns. Complementing anomaly detection, dynamic client selection has been proposed as an additional layer of defense to enhance the robustness of FL systems. This approach optimizes the selection of participating vehicles in each training round based on the output of anomaly detection algorithms. [13] proposes a method that monitors the update history of each client and statistically evaluates its performance using Z-Score. Clients whose updates fall below a predefined performance threshold are eliminated from the aggregation process. [14] introduces a Credit-Based Client Selection (CBCS) method, ranking clients based on the accuracy of their past contributions. Clients with higher credit scores are preferred, while low-scoring clients are scrutinized or eliminated. In [15], CosDefense uses cosine similarity to compare local updates with the global model, flagging and excluding clients whose updates deviate significantly, indicating potential malicious behavior. While eliminating persistently malicious participants through client elimination can reduce their adverse influence, it also presents challenges, particularly in non-IID settings. Due to the inherent heterogeneity of client data distributions, even benign clients may sometimes produce updates that are mistakenly classified as malicious, which reduces participation rates and compromises both performance and fairness. Moreover, existing defense approaches are not designed to exploit the hierarchical structure of FL in dynamic vehicular environments. Implementing dual-layer anomaly detection adds additional computational overhead and increases the likelihood of misclassification, especially given the mobility and variability of vehicular clients. Dynamic client selection in HFL remains underexplored, and its integration with anomaly detection has not been adequately investigated.

This paper proposes a novel framework that integrates dynamic client selection with statistical anomaly detection to effectively defend against model poisoning attacks in an HFL structure deployed over cluster-based VANETs. The approach is designed to enhance both the robustness and reliability of the global model in adversarial and dynamic vehicular environments. The key original contributions of this work are as follows:

- We propose a robust framework that defends against model poisoning attacks by integrating two statistical evaluation methods within an HFL structure for vehicular

networks. Cosine similarity is used to measure the alignment between individual vehicle updates and the global model direction, while Z-score analysis measures deviation in update norms from the collective average. This dual approach enables accurate differentiation between benign and malicious updates.

- We develop a novel dynamic client selection algorithm integrated with statistical anomaly detection to enhance the security of HFL in vehicular networks. Vehicles are clustered and assigned reliability scores based on three key metrics: historical accuracy, contribution frequency, and anomaly records. Historical accuracy prioritizes vehicles consistently demonstrating reliable cooperation during the FL process. The contribution frequency is used to identify vehicles with strong communication links and robust hardware. Meanwhile, anomaly records help pinpoint vehicles that exhibit minimal or no abnormal behavior. High-scoring vehicles are selected for participation, while those with suspicious behavior are temporarily excluded and later reassessed.
- We propose an adaptive thresholding mechanism that continuously fine-tunes detection criteria based on the latest updated statistics. Instead of relying on fixed thresholds, our approach leverages real-time feedback from historical accuracy metrics to dynamically adjust the cosine similarity threshold. This adaptive strategy ensures that our dynamic client selection and filtering processes remain robust against unforeseen attack patterns.
- We introduce a novel cross-cluster consistency check mechanism to further improve the integrity of the FL process in vehicular networks. By comparing aggregated updates across different clusters, the method effectively identifies anomalous clusters that may not be detected at the individual client level.
- We introduce a reliability-based weighted gradient averaging strategy to enhance model aggregation. Vehicles with higher reliability scores are given greater influence during global model updates, thereby reducing the impact of untrustworthy updates.
- We demonstrate through extensive simulations that our proposed framework significantly outperforms previously proposed cosine similarity and Z-score-based attacker detection algorithms in terms of convergence time and resilience under diverse attack scenarios.

The subsequent sections of this paper are structured as follows. Section II provides the system model. Section III introduces the proposed secure HFL using anomaly detection and dynamic client selection algorithms. Section IV presents a comprehensive performance evaluation of the proposed algorithm compared to existing methods, across varying network sizes and transmission ranges. Finally, Section VI presents concluding remarks and future research directions.

## II. SYSTEM MODEL

### A. Cluster-Based Hierarchical Federating Learning Network Model

The network comprises a fleet of vehicles, each equipped with dual communication interfaces that support both vehicle-to-vehicle (V2V) and vehicle-to-infrastructure or vehicle-to-network (V2I/V2N) communications. These interfaces comply with standards such as IEEE 802.11p for Dedicated Short-Range Communication (DSRC) [16], IEEE 802.11bd for Wireless Access in Vehicular Environments (WAVE) [17], LTE-based Device-to-Device (D2D) [18], and 5G New Radio V2X (NR V2X) [19].

In this dynamic vehicular environment, vehicles form clusters using a clustering algorithm designed to balance data diversity and mobility. This ensures stable cluster head selection and robust model convergence, even under non-IID data distributions [6]. Within each cluster, vehicles take on one of two roles: Cluster Member (CM) or Cluster Head (CH). These roles adapt in real time to changes in network conditions. For example, if the performance of a CH is degraded due to high mobility or poor communication quality, a new CH is elected based on communication stability and computational capacity. This mechanism ensures that clusters remain adaptive and resilient to topology changes and vehicle dynamics.

The system employs hierarchical federated learning with a multi-tiered aggregation structure to optimize communication and training efficiency. Each CM acts as a client that generates model updates based on its local non-IID data. These updates, typically in the form of gradients, are transmitted to their respective CHs, which serve as intermediary aggregators. CHs collect and aggregate updates from the CMs within their cluster and forward the results to the Evolved Packet Core (EPC). The EPC acts as the central aggregation entity, responsible for performing global model aggregation across all clusters. The updated global model is then disseminated back to CHs and subsequently to CMs. This hierarchical structure significantly reduces the communication overhead at the EPC by limiting direct transmissions, while also improving the efficiency and speed of model convergence.

Each vehicle maintains a Vehicle Information Base (VIB), a dynamic data repository containing key operational parameters, including direction, location, velocity, cluster role, and local model updates. The VIB is updated upon operational changes or in response to events like receiving a “*HELLO\_PACKET*” or a model update request. The “*HELLO\_PACKET*” contains parameters such as vehicle direction, position, speed, current cluster affiliation, the connecting node to the cluster, cosine similarity of the FL model parameters, and relative velocity compared to surrounding vehicles. This information supports accurate clustering and ensures effective CH selection and model aggregation.

### B. Attack Model

We consider two types of model poisoning attacks: Gaussian noise and gradient ascent attacks. These attacks pose serious challenges to HFL in vehicular networks due to the highly dynamic nature of these environments. Vehicles frequently

join and leave clusters, giving adversaries repeated opportunities to inject Gaussian noise, either through environmental disturbances or intentional manipulation, and to manipulate gradient updates to compromise system performance [20]. The hierarchical structure of HFL, where CHs aggregate updates from CMs before relaying them to the EPC, adds an additional layer of vulnerability. This additional aggregation step provides adversaries with more opportunities to inject malicious updates. Moreover, the use of wireless communication channels exposes the system to eavesdropping and tampering, facilitating gradient ascent attacks that guide global models away from optimal solutions [21]. Furthermore, because vehicles generate non-IID data under diverse driving conditions, the effects of adversarial modifications are often unevenly distributed, with some clusters more severely impacted than others, compromising model integrity [22].

1) *Gaussian Noise Attack*: In Gaussian noise attacks, adversarial vehicles add random noise to their local updates before sending them to the CH or EPC. This is a form of Byzantine attack, where the adversarial update  $g_{adv}$  from a malicious vehicle is given by

$$g_{adv} = g_{benign} + \mathcal{N}(m, \sigma^2 \mathbf{I}), \quad (1)$$

where  $g_{benign}$  is the genuine gradient update computed on the vehicle’s local data,  $\mathcal{N}(m, \sigma^2 \mathbf{I})$  represents Gaussian noise with mean  $m$  and variance  $\sigma^2$ , and  $I$  is the identity matrix.  $\sigma^2$  controls the intensity of the noise injected.

By introducing Gaussian noise, adversarial vehicles aim to disrupt the learning process by making the aggregated gradients less representative of the true underlying data distribution. This random noise can cause the global model to converge slowly or even diverge, leading to degraded performance. Gaussian attacks are dangerous because the added noise resembles natural variations in gradient updates, making it challenging to detect using simple anomaly detection methods that rely on statistical deviations. The randomness of the noise can mask the malicious intent, especially in networks with high variability among client updates.

2) *Gradient Ascent Attack*: Gradient ascent attack is a more aggressive and targeted adversarial strategy where malicious vehicles deliberately manipulate model updates. Unlike benign gradient descent updates, which aim to minimize the loss function, gradient ascent attacks involve adversaries intentionally maximizing the loss function to steer global model parameters away from their optimal values [22].

The malicious update  $g_{adv}$  in a gradient ascent attack is designed to reverse the intended optimization direction. Let the benign update be given by

$$g_{benign} = -\eta \nabla L(\theta), \quad (2)$$

where  $\nabla L(\theta)$  is the gradient of the loss function with respect to model parameters  $\theta$  and  $\eta > 0$  is the learning rate or scaling factor. To reverse this update, the adversarial client sends:

$$g_{adv} = g_{benign} + 2\eta \nabla L(\theta) = \eta \nabla L(\theta). \quad (3)$$

This formulation clearly shows that the adversarial update is directed along  $\nabla L(\theta)$ , thereby maximizing the loss function.

The gradient ascent attack is particularly effective because it directly counteracts the beneficial updates from honest vehicles. Even a small proportion of adversarial vehicles can have a disproportionately large negative impact on the global model. To counteract this, CHs need to detect and exclude these malicious vehicles from aggregation in HFL.

### III. SECURE HFL USING ANOMALY DETECTION AND DYNAMIC CLIENT SELECTION

We propose a robust approach for dynamic client selection and anomaly detection within a cluster-based HFL framework for vehicular networks. This method is designed to defend against both Gaussian noise and gradient ascent attacks. To detect Gaussian noise attacks, we apply Z-score analysis on the norms of model updates. In typical scenarios, benign updates have norms that cluster around a mean value. When Gaussian noise is introduced, the update norms often deviate significantly from this mean, resulting in Z-scores that exceed a predefined threshold. Such updates are flagged as potentially malicious. To counter gradient ascent attacks, we use cosine similarity to assess the directional alignment between individual vehicle updates and the overall expected model direction. In a standard setting, benign updates are well aligned with the consensus update direction. In contrast, a low cosine similarity indicates that the update is misaligned, suggesting an adversarial attempt to intentionally steer the local or global model away from its intended direction by maximizing the loss function. By combining these two detection methods, we can identify both abnormal magnitudes and directional inconsistencies in updates, key signs of malicious behavior. To further improve detection accuracy, these statistical techniques are combined with vehicle reliability metrics, including historical accuracy, contribution frequency, and anomaly history. These reliability scores guide dynamic client selection, ensuring that only trustworthy and high-quality updates are included in the global model aggregation.

#### A. Detection Criteria

1) *Z-score*: Z-score is used to identify outliers or anomalous behaviors by calculating how many standard deviations the gradient norm of a vehicle deviates from the mean of all vehicle gradient norms. Specifically, for each vehicle, the Z-score of its gradient norm is calculated to assess whether it significantly deviates from the collective average. This statistical approach allows us to detect vehicles whose updates might be corrupted due to faulty sensors, data errors, or malicious intent, such as in Gaussian attacks, where random noise is added to disrupt the learning process. By filtering out these extreme values, the Z-score helps maintain the quality of the aggregated model and prevents performance degradation [23].

For each CM  $k$  at iteration  $i$ , the Z-score  $Z_{Scores_{k,i}}$  is calculated as:

$$Z_{Scores_{k,i}} = \frac{Norms_{k,i} - Mean\_Norm_i}{Std\_Norm_i}, \quad (4)$$

where

$$Norms_{k,i} = \|g_{k,i}\|$$

is the  $\ell_2$ -norm of the gradient update  $g_{k,i}$  from vehicle  $k$  at iteration  $i$ . Here,  $Mean\_Norm_i$  and  $Std\_Norm_i$  denote the mean and standard deviation of the gradient norms across vehicles at iteration  $i$ , respectively. These statistical measures capture the overall distribution of gradient norms, enabling the identification of outlier vehicles whose updates deviate significantly from the average.

For each CH  $k$  at iteration  $i$ , the Z-score  $Z_{Scores_{k,i}}$  is computed in a manner similar to that used for CM. However, the definition of the norm  $Norm_{k,i}$  is adjusted to accommodate the CH's role in the aggregation process. Specifically, the norm for a CH is defined as the Euclidean distance between the CH's local aggregated update  $\theta_{CH,i}$  and the global model  $\theta_{global,i}$  as follows:

$$Norms_{CH,i} = \|\theta_{CH,i} - \theta_{global,i}\|. \quad (5)$$

This formulation allows the Z-score to effectively measure deviations in the aggregated updates relative to the global model. By assessing this deviation, the system can identify abnormal aggregation behavior, which may result from compromised updates contributed by malicious CMs or malicious manipulation by the CH itself.

2) *Cosine Similarity*: Cosine similarity is used to assess the directional consistency of the gradient update of each vehicle relative to the mean gradient of all vehicles by measuring the cosine of the angle between them. A low cosine similarity indicates that the vehicle's update is inconsistent with the collective behavior, potentially signaling adversarial actions like gradient ascent attacks, where clients intentionally push the model parameters away from optimal values. By focusing on the direction rather than the magnitude of the gradients, cosine similarity helps detect such anomalies even when the gradient norms appear normal, adding an additional layer of defense against sophisticated attacks.

For each CM  $k$  at iteration  $i$ , the cosine similarity  $Cos\_Sim_{k,i}$  is computed as:

$$Cos\_Sim_{k,i} = \frac{g_{k,i} \cdot Mean\_grad_i}{\|g_{k,i}\| \cdot \|Mean\_grad_i\|}, \quad (6)$$

where  $g_{k,i}$  is the gradient update from CM  $k$  at iteration  $i$ ,  $Mean\_grad_i = \frac{1}{N} \sum_{j=1}^N g_{j,i}$  is the average gradient vector of all CMs within the cluster at iteration  $i$  and  $N$  is the number of valid CMs in a cluster after Z-score filtering. This calculation measures how closely each CM's gradient aligns with the collective learning direction of the cluster.

For CHs, cosine similarity evaluates the temporal consistency of the CH's aggregated model updates relative to the global model. By comparing the direction of consecutive aggregated updates, we detect anomalies at the aggregation level that may result from malicious activities or significant CH behavior deviations. Specifically, for each CH  $j$  at iteration  $i$ , cosine similarity is calculated between the current aggregated update  $\theta_{j,i}$  and the previous aggregated update  $\theta_{j,i-1}$  relative to the previous iteration global model  $\theta_{global,i-1}$  as follows:

$$Cos\_Sim_{j,i} = \frac{(\theta_{j,i} - \theta_{global,i-1}) \cdot (\theta_{j,i-1} - \theta_{global,i-1})}{\|\theta_{j,i} - \theta_{global,i-1}\| \cdot \|\theta_{j,i-1} - \theta_{global,i-1}\|}. \quad (7)$$

This cosine similarity measures the consistency of the CH's updates over time, ensuring that the direction of model adjustments remains stable and aligned with the global learning objectives.

To further enhance security, the EPC performs a cross-cluster consistency check using cosine similarity to detect collaborative attacks, where multiple compromised CHs may coordinate to submit malicious or misleading updates. If several CHs submit similar updates that are inconsistent with the rest of the clusters, it could indicate a coordinated attack to skew the global model. In such cases, flagging these CHs ensures that the aggregated updates from CHs align with the overall learning objective and helps detect any anomalies or malicious activities at the cluster level. Specifically, after the EPC receives the aggregated model updates  $\theta_i$  from each CH at iteration  $i$ , it performs a consistency check by computing the cosine similarity between the updates of different clusters. The cross-cluster cosine similarity  $Cos\_Sim_{p,q,i}$  between clusters  $p$  and  $q$  is calculated as:

$$Cos\_Sim_{p,q,i} = \frac{(\theta_{p,i} - \theta_{global,i-1}) \cdot (\theta_{q,i} - \theta_{global,i-1})}{\|\theta_{p,i} - \theta_{global,i-1}\| \times \|\theta_{q,i} - \theta_{global,i-1}\|}, \quad (8)$$

where  $\theta_{p,i}$  and  $\theta_{q,i}$  are the model updates from CHs of clusters  $p$  and  $q$  at iteration  $i$ , respectively, and  $\theta_{global,i-1}$  is the global model from the previous iteration.

To quantify the overall consistency of a cluster's update with the rest of the network, we calculate the average cross-cluster cosine similarity  $\overline{Cos\_Sim}_{p,i}$  for cluster  $p$  as:

$$\overline{Cos\_Sim}_{p,i} = \frac{1}{C-1} \sum_{\substack{q=1 \\ q \neq p}}^C Cos\_Sim_{p,q,i}, \quad (9)$$

where  $C$  is the total number of clusters. By computing this average, we assess how well the update of the cluster  $p$  aligns with the updates of the other clusters in the same iteration. This mechanism ensures that the clusters that contribute to the global model align with the overall learning objective, improving the robustness and reliability of the FL.

*3) Reliability Metrics:* We integrate the outputs of Z-score and cosine similarity-based anomaly detection into a set of reliability metrics to evaluate and prioritize vehicles based on their historical performance. The first metric, Historical Accuracy, assesses the long-term performance of each vehicle by measuring the average accuracy of its local model over multiple communication rounds using a validation dataset. The second metric, Contribution Frequency, measures how consistently a vehicle's updates have been accepted into the aggregation process based on Z-score and cosine similarity assessments, while also accounting for packet losses due to channel quality, which may prevent a vehicle from contributing at various learning stages and thereby lower its overall frequency of contribution. The third metric, Anomaly Record, tracks the number of times a vehicle's updates have been flagged as anomalous due to significant deviations detected by Z-score or misalignment identified by cosine similarity. By integrating these reliability metrics with the statistical

evaluations from Z-score and cosine similarity, our approach enhances the robustness of anomaly detection and improves the overall reliability of the federated learning process.

Historical accuracy reflects the average performance of the local model of a vehicle in multiple communication rounds. This metric is calculated by the EPC and all CHs using the accuracy of the model on a validation dataset during each round [24]. Accuracy is defined as the percentage of correct predictions made by the model when tested on unseen validation samples. Higher historical accuracy indicates that the vehicle consistently contributes high-quality updates to the global model. The historical accuracy of vehicle  $k$  at round  $i$  is computed as:

$$Historical\_Accuracy_{k,i} = Total\_Accuracy_{k,i}/i = (1/i) * \sum_{n=1}^i (Accuracy\_of\_Contribution_{k,n}), \quad (10)$$

where  $i$  represents the  $i$ th communication round considered as a single FL iteration within the FL procedure,  $Total\_Accuracy_{k,i}$  represents the sum of all  $Accuracy\_of\_Contribution_{k,n}$  values across all communication rounds up to the current round  $i$ , and  $Accuracy\_of\_Contribution_{k,n}$  is the accuracy of vehicle  $k$  at round  $n$ , evaluated by the CH or EPC as:

$$Accuracy\_of\_Contribution_{k,n} = \left( \frac{\sum_{j=1}^{N_{val}} \delta_j}{N_{val}} \right) \times 100\%. \quad (11)$$

where  $D_{val} = \{(x_j, y_j)\}_{j=1}^{N_{val}}$  is the validation dataset,  $x_j$  is the input data,  $y_j$  is the true label for the  $j$ -th sample,  $N_{val}$  is the number of samples in the validation dataset,

$$\delta_j = \mathbb{I}[\hat{y}_j = y_j] = \begin{cases} 1, & \text{if } \hat{y}_j = y_j, \\ 0, & \text{if } \hat{y}_j \neq y_j, \end{cases} \quad (12)$$

$\hat{y}_j = \theta_{k,n}(x_j)$  is the predicted label for the  $j$ -th sample by using the vehicle's updated model  $\theta_{k,n}$ . By calculating the number of correct predictions, we obtain an accurate measure of the vehicle's model performance on unseen data.

Contribution frequency measures how regularly a vehicle participates in FL rounds. A higher frequency implies that the vehicle is consistently active and dependable. This metric, however, is affected not only by the vehicle's willingness to participate but also by its channel quality. Poor channel conditions may lead to failed or delayed transmissions, reducing the effective contribution frequency. In contrast, vehicles with stable and strong communication links are more likely to contribute updates consistently, reinforcing their reliability as participants in the FL process. The contribution frequency of vehicle  $k$  is calculated by

$$Contribution\_Freq_{k,i} = Total\_Contributions_{k,i}/i, \quad (13)$$

where  $Total\_Contributions_{k,i}$  represents the number of contributions made by vehicle  $k$  across all communication rounds up to the current round  $i$  in the FL process.

Anomaly record tracks the frequency of a vehicle's contributions being flagged as anomalous through Z-score and cosine similarity analyses. A high anomaly record suggests that the vehicle may be compromised or unreliable, and thus should be assigned a lower reliability score to protect the integrity of the learning process. The anomaly record of vehicle  $k$  at round  $i$  is calculated by

$$\begin{aligned} \text{Anomaly\_Record}_{k,i} = \\ \text{Total\_Anomalous\_Contributions}_{k,i}/i, \end{aligned} \quad (14)$$

where  $\text{Total\_Anomalous\_Contributions}_{k,i}$  refers to the total number of times the contributions of vehicle  $k$  have been flagged as anomalous across all communication rounds up to the current round  $i$ .

The final reliability score for each vehicle is determined by a weighted sum of the three metrics as given below:

$$\begin{aligned} \text{Reliability\_Score}_{k,i} = \\ (\text{Accuracy\_Weight} \times \text{Historical\_Accuracy}_{k,i}) + \\ (\text{Frequency\_Weight} \times \text{Contribution\_Freq}_{k,i}) - \\ (\text{Anomaly\_Weight} \times \text{Anomaly\_Record}_{k,i}) \end{aligned} \quad (15)$$

Assigning specific weights to these metrics allows for a flexible and context-sensitive evaluation of each vehicle's contributions in FL. This weighting system can be customized to emphasize particular aspects of reliability based on the specific needs and goals of the FL application. As the network evolves, the reliability score is dynamically updated, ensuring that the selection process remains adaptive to changes in vehicle behavior and network conditions. Vehicles with higher reliability scores are prioritized for participation in model training rounds, thereby enhancing the overall robustness and accuracy of the global model. Vehicles identified as anomalous through Z-score and cosine similarity checks have their anomaly records updated, affecting their reliability scores. This feedback mechanism ensures that the system remains responsive to changes in vehicle behavior, reducing the impact of unreliable or malicious vehicles while promoting contributions from those who consistently provide valuable data.

*4) Adaptive Thresholding Mechanism:* We introduce an adaptive thresholding mechanism for the cosine similarity metric to enhance the robustness of our anomaly detection process. This mechanism dynamically adjusts the cosine similarity threshold based on each vehicle's historical accuracy, ensuring that vehicles with consistently high performance are held to stricter standards. High-performing CMs might become less careful over time, leading to a gradual decline in update quality. Without adaptive thresholding, these vehicles may continue to pass the anomaly detection checks despite a drop in performance, as the static thresholds do not adjust to their historical contribution levels. The adaptive cosine similarity threshold  $\text{Cosine\_Sim\_Threshold}_{\text{adaptive},k}$  for vehicle  $k$  is defined as:

$$\begin{aligned} \text{Cosine\_Sim\_Threshold}_{\text{adaptive},k} = \\ \text{Cosine\_Sim\_Threshold}_{\text{adaptive},k} - \delta, \end{aligned} \quad (16)$$

where  $\delta$  is a small positive adjustment parameter controlling the degree of threshold tightening. When a vehicle's  $\text{Historical\_Accuracy}_{k,i}$  exceeds  $\text{High\_Threshold\_Up}$  and the cosine similarity threshold is above the floor threshold  $\text{High\_Threshold\_Down}$ , the threshold is reduced by  $\delta$ . This adjustment continues until the threshold reaches  $\text{High\_Threshold\_Down}$ , at which point further reduction stops. This adjustment imposes stricter anomaly detection criteria on high-performing vehicles, ensuring that even minor deviations from their typical update patterns are detected.

*5) Weighted Gradient Averaging:* We implement a weighted gradient averaging mechanism to enhance the robustness and effectiveness of the model aggregation process in FL. This method assigns weights to each CM's gradient updates based on their computed reliability scores, ensuring that more reliable participants have a greater influence on the global model. Specifically, after the anomaly detection and reliability score adjustment steps, the aggregated gradient  $G_i$  is computed as a weighted average of the accepted gradients:

$$G_i = \frac{\sum_{k=1}^N \text{Reliability\_Score}_{k,i} \times g_{k,i}}{\sum_{k=1}^N \text{Reliability\_Score}_{k,i}}, \quad (17)$$

where  $N$  represents the number of CMs in the cluster under that CH,  $\text{Reliability\_Score}_{k,i}$  and  $g_{k,i}$  denote the reliability score and gradient of the  $k$ th CM during the  $i$  communication round, respectively. It then updates its model parameters using a gradient descent step:

$$\theta_{CH,i} = \theta_{CH,i-1} - \eta G_i, \quad (18)$$

where  $\theta_{CH,i-1}$  represents the CH's model parameters carried over from the previous round. This approach effectively amplifies the positive impact of trustworthy CMs while diminishing the influence of those with lower reliability. The normalization by the sum of reliability scores ensures that the aggregated gradient maintains an appropriate scale.

At the cluster level, a similar weighted averaging is performed when aggregating the updates from CHs at the EPC. The global model parameters  $\theta_{global,i}$  at iteration  $i$  are updated using the weighted average of the CHs' model parameters  $\theta_{CH,i}$ :

$$\theta_{global,i} = \frac{\sum_{CH=1}^C \text{Reliability\_Score}_{CH,i} \times \theta_{CH,i}}{\sum_{p=1}^C \text{Reliability\_Score}_{CH,i}}, \quad (19)$$

where  $\text{Reliability\_Score}_{CH,i}$  is the reliability score of CH after any adjustments due to cross-cluster consistency checks or anomaly penalty factors,  $\theta_{CH,i}$  is the local update from CH at iteration  $i$  and  $C$  is the number of clusters. This weighted averaging process enhances the overall robustness and effectiveness of the FL by prioritizing contributions from trustworthy clusters.

#### B. Dynamic Anomaly Detection and Reliability-based Client Selection (DARCS) Algorithm

In the proposed HFL framework for vehicular networks, the CHs and the EPC have distinct responsibilities, each supported by dedicated algorithms. CHs serve as intermediaries between CMs and the EPC. They manage communication within

their clusters, perform local model aggregations, and execute anomaly detection mechanisms tailored to their respective clusters. The CH algorithm focuses on dynamic client selection and anomaly detection at the CM level, using individual vehicle reliability scores. On the other hand, the EPC serves as the central coordinator of the entire network. It aggregates model updates from all CHs to refine the global model and performs higher-level anomaly detection, including cross-cluster consistency checks and reliability evaluations of CHs based on aggregated metrics. Unlike CHs, the EPC algorithm operates at the cluster level and processes aggregated rather than individual vehicle updates. By employing different algorithms proposed for the specific functions and data accessibility of the CHs and EPC, this hierarchical approach effectively mitigates risks posed by malicious actors and unreliable data sources at different levels of the network hierarchy.

Algorithm 1 operates within the Cluster-based HFL framework and incorporates mechanisms to enhance robustness against unreliable or adversarial gradient updates from CMs at the CH. The primary challenge is efficiently aggregating these local updates while ensuring malicious contributions do not compromise the overall model convergence. The CH initializes the reliability scores for each CM based on the data in the VIB (Lines 1-2). At the beginning of each communication round, the algorithm assesses the participation eligibility of each CM. If a CM's  $Block\_Flag_{CM}$  is set to *TRUE* and the  $block\_duration$  is less than the predefined  $Unblock\_Time$ , it indicates that the CM has been previously identified as malicious and is still within its blocking period. In this case, the  $Unblock\_Time$  is incremented, and the CH skips further processing for this CM, proceeding to the next. If the condition is not met—meaning the CM is either not blocked, or the blocking duration has expired—the algorithm resets the  $Block\_Flag_{CM}$  to *FALSE*, allowing the CM to participate in the current round, and the CH receives the update  $g_{CM,i}$  from the CM (Lines 3-7). In the subsequent stage, all CMs that have passed the blocking check are considered eligible. These eligible CMs are then sorted in descending order based on their reliability scores, and the top  $SELECTED\_CLIENT\_PERCENTAGE$  are designated as *Selected\_Clients*. This selection process prioritizes the most reliable CMs for participation in the current training round, enhancing the overall model quality (Lines 8-9).

Next, Z-scores are calculated for the gradient norms of each CM, indicating how far a particular CM's gradient is from the mean in terms of standard deviations (Line 11). If the absolute value of  $Z\_Scores_{CM,i}$  is less than the  $Z\_Score\_Threshold$ , the algorithm computes the cosine similarity between each valid CM gradient and the mean gradient to further ensure robustness (lines 12-13). In contrast, if the absolute value of  $Z\_Score_{CM,i}$  exceeds  $Z\_Score\_Threshold$ , it indicates an outlier or anomalous behavior. In this case, the algorithm sets the  $Block\_Flag_{CM}$  to *TRUE*, initializes the block duration timer to zero, increments the  $Total\_Anomalous\_Contributions_{CM,i}$  counter, and proceeds to skip this CM, moving on to the next CM in the cluster (Lines 14-17).

If the absolute difference between  $Cos\_Sim_{CM}$  in

---

**Algorithm 1:** Dynamic Anomaly Detection and Reliability-based Client Selection (DARCS) Algorithm at the CH

---

```

1 foreach  $CM \in VIB$  do
2   Initialize  $Reliability\_Score_{CM,i}$  based on VIB;
3   if  $Block\_Flag_{CM} = TRUE$  and
4      $block\_duration_{CM} < Unblock\_Time$  then
5       Increment  $block\_duration_{CM}$ ;
6       Continue to next  $CM$ ;
7     else
8       Reset  $Block\_Flag_{CM}$  to FALSE;
9   Sort all  $CM$  with  $Block\_Flag_{CM} = FALSE$  by
10     $Reliability\_Score_{CM,i}$  in descending order;
11   Select top
12     $SELECTED\_CLIENT\_PERCENTAGE$  of
13    vehicles as Selected_Clients;
14 foreach  $CM \in Selected\_Clients$  do
15   Compute  $Z\_Scores_{CM,i}$  by using (4);
16   if  $|Z\_Score_{CM,i}| < Z\_Score\_Threshold$  then
17     Compute  $Cos\_Sim_{CM,i}$  by using (6);
18   else
19     Set  $Block\_Flag_{CM} = TRUE$  and
20       $block\_duration_{CM} = 0$ ;
21     Increment
22       $Total\_Anomalous\_Contributions_{CM,i}$ ;
23     Continue to next  $CM$ ;
24   if  $|Cos\_Sim_{CM,i-1} - Cos\_Sim_{CM,i}| >$ 
25     $Cosine\_Sim\_Threshold_{adaptive,CM}$  then
26     Reset CM parameters using  $Memory_{CM,i-1}$ 
27   else
28     Increment  $Total\_Contributions_{CM,i}$ ;
29     Calculate  $Accuracy\_of\_Contribution_{CM,i}$ ;
30     Compute  $Historical\_Accuracy_{CM,i}$  by
31     using (10),  $Contribution\_Freq_{CM,i}$  by
32     using (13),  $Anomaly\_Record_{CM,i}$  by
33     using (14);
34     Update  $Reliability\_Score_{CM,i}$  by using (15);
35     Record updated metrics in VIB;
36   if  $Historical\_Accuracy_{CM,i} \geq$ 
37      $High\_Threshold\_Up$  and
38      $Cosine\_Sim\_Threshold_{adaptive,CM} >$ 
39      $High\_Threshold\_Down$  then
40     Adjust  $Cosine\_Sim\_Threshold_{adaptive,CM}$ 
41     by using (16);
42   Compute model update  $\theta_{CH,i}$  by using (17) and (18);
43   foreach  $CM \in VIB$  do
44     Record all parameters in  $Memory_{CM,i}$ ;
45   Send updated model parameters  $\theta_{CH,i}$  to all CMs and
46   EPC;

```

---

the current and previous communication rounds exceeds  $\text{Cosine\_Sim\_Threshold}_{\text{adaptive},i}$ , it indicates a substantial change that may serve as a potential sign of malicious intent, where a CM introduces significant deviations to disrupt the learning process. Upon detecting this anomaly, the CM's parameters are reset using  $\text{Memory}_{CM,i-1}$ , which contains the stored model parameters from the previous round. A memory mechanism replicates the most recent CM structure, ensuring that previous CM parameters remain accessible within the system. This approach prevents abrupt omissions of a CM based solely on cosine similarity differences, which could otherwise disrupt the learning process (Lines 18-19).

After filtering out unreliable CMs based on cosine similarity, the algorithm updates the contribution records for the remaining valid CMs. It calculates the  $\text{Accuracy\_of\_Contribution}_{CM,i}$ , updates the  $\text{Total\_Accuracy}_{CM,i}$ , and computes the  $\text{Historical\_Accuracy}_{CM,i}$ . It also computes the  $\text{Contribution\_Freq}_{CM,i}$  and  $\text{Anomaly\_Record}_{CM,i}$  to assess each CM's reliability over time. The  $\text{Reliability\_Score}_{CM,i}$  is updated using these metrics, incorporating weights for accuracy, frequency, and anomaly records. This score reflects the CM's trustworthiness and influences its future participation and impact on the model aggregation (Lines 20-24). Subsequently, these updates are added to the VIB for further decision-making and calculation (Line 25).

The next step applies adaptive thresholding by adjusting the cosine similarity threshold for the high-performing CMs. This mechanism dynamically adjusts the cosine similarity threshold based on each CM's historical accuracy, ensuring that CMs with consistently high performance are held to stricter standards. Specifically, if a CM's historical accuracy surpasses a predefined  $\text{High\_Threshold\_Up}$ , it indicates that the CM has consistently contributed reliable updates over time. The algorithm tightens the cosine similarity threshold for these high-performing CMs using (16) until it is not less than a predefined parameter  $\text{High\_Threshold\_Down}$  to ensure sustained high-performance (Lines 26-27). The CH aggregates the accepted updates from CMs using weighted averaging based on reliability scores (Line 28). The current parameters of each CM are then stored in a memory buffer denoted as  $\text{Memory}_{CM,i}$ . Finally, the CH sends the average aggregated update  $\theta_{CH,i}$  to the EPC and receives the global model update in return. It distributes the updated model parameters to all CMs in the cluster, completing the training round (Lines 29-31).

Algorithm 2 operates within the same framework and improves robustness against unreliable or adversarial model updates from CHs at the EPC. The primary challenge is to efficiently aggregate these cluster-level updates while ensuring that malicious contributions do not compromise the convergence of the global model. Initially, the algorithm iterates over each CH in the VIB and manages blocked CHs the same way as Algorithm 1 (Lines 1-6). It then calculates  $\text{Norms}_{CH,i}$ , to quantify the deviation of each CH from the global model, updating the mean and standard deviation of these norms to build a statistical profile of the updates of the CHs. Using this

---

**Algorithm 2:** Dynamic Anomaly Detection and Reliability-based Client Selection (DARCS) Algorithm at the EPC

---

```

1 foreach  $CH \in VIB$  do
2   if  $\text{Block\_Flag}_{CH} = \text{TRUE}$  and
3      $\text{block\_duration}_{CH} < \text{Unblock\_Time}$  then
4       Increment  $\text{block\_duration}_{CH}$ ;
5       Continue to next CH;
6   else
7     Reset  $\text{Block\_Flag}_{CH}$  to FALSE;
8     Compute  $\text{Norms}_{CH,i}$  by using (5);
9     Compute  $\text{Z\_Score}_{CH,i}$  by using (4);
10    if  $|\text{Z\_Score}_{CH,i}| < \text{Z\_Score\_Threshold}$  then
11      Compute  $\text{Cos\_Sim}_{CH,i}$  by using (7);
12    else
13      Set  $\text{Block\_Flag}_{CH} = \text{TRUE}$  and
14         $\text{block\_duration}_{CH} = 0$ ;
15      Increment
16         $\text{Total\_Anomalous\_Contributions}_{CH,i}$ ;
17      Continue to next CH;
18    if  $|\text{Cos\_Sim}_{CH,i-1} - \text{Cos\_Sim}_{CH,i}| >$ 
19       $\text{Cosine\_Sim\_Threshold}_{\text{adaptive},CH}$  then
20        Reset CH parameters using  $\text{Memory}_{CH,i-1}$ 
21    else
22      foreach pair of CHs  $(p, q)$  where  $p \neq q$  do
23        Compute  $\text{Cos\_Sim}_{p,q,i}$  by using (8);
24      foreach  $p$  do
25        Compute  $\overline{\text{Cos\_Sim}}_{p,i}$  by using (9);
26        if  $\overline{\text{Cos\_Sim}}_{p,i} <$ 
27           $\text{Cosine\_Sim\_Threshold}_{\text{cross}}$  then
28            Set  $\text{Block\_Flag}_{CH} = \text{TRUE}$  and
29               $\text{block\_duration}_{CH} = 0$ ;
30            Increment
31               $\text{Total\_Anomalous\_Contributions}_{CH,i}$ ;
32            Continue to next CH;
33        else
34          Increment  $\text{Total\_Contributions}_{CH,i}$ ;
35          Calculate  $\text{Accuracy\_of\_Contribution}_{CH,i}$  by
36          Compute  $\text{Historical\_Accuracy}_{CH,i}$  by
37            using (10),  $\text{Contribution\_Freq}_{CH,i}$  by
38            using (13),  $\text{Anomaly\_Record}_{CH,i}$  by
39            using (14);
40          Update  $\text{Reliability\_Score}_{CH,i}$  by using (15);
41          Record updated metrics in VIB;
42        if  $\text{Historical\_Accuracy}_{CH,i} \geq$ 
43           $\text{High\_Threshold\_Up}$  and
44           $\text{Cosine\_Sim\_Threshold}_{\text{adaptive},CH} >$ 
45           $\text{High\_Threshold\_Down}$  then
46            Adjust  $\text{Cosine\_Sim\_Threshold}_{\text{adaptive},CH}$ 
47            by using (16);
48    Aggregate the accepted model parameters from CHs;
49    Compute global model update  $\theta_{global,i}$  by using (19);
50    foreach  $CH \in VIB$  do
51      Record all parameters in  $\text{Memory}_{CH,i}$ ;
52    Send updated global model parameters  $\theta_{global,i}$  to all
53    CHs;

```

---

profile, the algorithm computes Z scores for each CH model update (Lines 7–8) to identify outliers and track anomaly records (Line 9). For CHs passing the Z-score check, the cosine similarity between each CH’s update and the global model parameters is computed (Line 10). If a CH shows a significant deviation in cosine similarity between consecutive rounds, the algorithm reverts the parameters of that CH to the most recent trusted state, mirroring Algorithm 1 but with a specific index for CH vehicles (Lines 15–16).

The algorithm then performs a cross-cluster consistency check to ensure consistent aggregated updates from different CHs. For each pair of CHs, the cross-cluster cosine similarity is computed to measure the directional alignment between their updates. For each CH, it calculates the average cross-cluster cosine similarity with all other CHs. If a CH’s average cross-cluster cosine similarity is below the predefined threshold  $Cosine\_Sim\_Threshold_{cross}$ , it indicates that the CH’s update is inconsistent with those from other clusters, possibly due to anomalies or adversarial behavior. In such cases, the algorithm sets the  $Block\_Flag_{CH}$  to *TRUE*, resets the  $block\_duration_{CH}$  to zero, increments the  $Total\_Anomalous\_Contributions_{CH,i}$  counter, and skips processing this CH (Lines 17–25). After filtering out unreliable CHs via these Z-score and cosine similarity evaluations, the algorithm updates the contribution records of each remaining CH following the same procedure as Algorithm 1 (Lines 26–31). Next, adaptive thresholding is applied by adjusting the cosine similarity threshold for high-performing CHs. If a CH’s  $Historical\_Accuracy_{CH,i}$  surpasses a predefined  $High\_Threshold\_Up$ , it indicates consistently high performance (Lines 32–33).

The EPC then aggregates the accepted model parameters from the remaining CHs to update the global model. The aggregation is performed using weighted averaging based on the CHs’ reliability scores, ensuring that more reliable CHs have a more significant influence on the global model update. This weighted aggregation promotes the contributions of trustworthy CHs while mitigating the impact of less reliable ones, enhancing the overall performance and convergence of the global model (Lines 34–35). The subsequent step focuses on saving the current parameters of each CH into a designated memory buffer, referred to as  $Memory_{CH,i}$  (Lines 36–37).

Finally, the EPC sends the updated global model parameters back to all CHs for distribution to their respective CMs, completing the training round. This dissemination ensures all participants have the latest model parameters, enabling them to continue the local training and updating process in subsequent rounds (Line 38).

#### IV. PERFORMANCE EVALUATION

The simulations evaluate the effectiveness of the proposed Dynamic Anomaly Detection and Reliability-based Client Selection (DARCS) algorithm when integrated with the Cluster-based HFL (CbHFL) framework presented in [6]. Our approach builds on CbHFL’s dynamic clustering mechanism, which adapts to highly mobile vehicular environments and enables multi-hop transmission of local updates to CHs. We

compare the performance of the DARCS-enhanced CbHFL framework against four benchmark approaches. The first benchmark is the original CbHFL framework without any attacks. In this benchmark, vehicles are grouped into CMs and CHs, while the EPC acts as the central server for global model aggregation. This baseline serves as a reference for an effective HFL environment in a non-IID, highly dynamic vehicular setting. The second benchmark is a cosine similarity-based attacker detection algorithm (CosDefense) [15], adapted for CbHFL. It identifies malicious vehicles by computing cosine similarity scores between the global model’s last-layer weights and local updates, excluding vehicles with significant deviations from the average. The third benchmark is Z-Score Detection [13], also adapted for CbHFL, which detects outliers based on Z-score analysis and removes updates that exceed a predefined threshold. The fourth benchmark combines cosine similarity and Z-score methods but does not incorporate reliability scores, adaptive thresholding, or cross-cluster consistency checks. This variant helps to isolate and evaluate the impact of additional mechanisms proposed in DARCS.

##### A. Simulation Setup

The simulations utilize Python, with vehicle mobility generated by Simulation of Urban Mobility (SUMO) and data streaming via KAFKA. This setup creates a realistic vehicular network scenario for FL. SUMO models individual driver behaviors and traffic dynamics, while KAFKA is utilized to transmit packets and model parameters, enabling communication among vehicles. FL models are built using PyTorch.

The simulation environment replicates a one km<sup>2</sup> area with two-lane roads, where vehicle movements are modeled using a Poisson process. Vehicles have speeds ranging from 10 m/s to 35 m/s, capturing a variety of mobility patterns. IEEE 802.11p is employed for vehicle-to-vehicle communication, while 5G NR is used for vehicle-to-infrastructure communication. For vehicle-to-vehicle links, the Winner+ B1 propagation model is applied as described in [25]. For vehicle-to-infrastructure communication, the Friis propagation model estimates signal attenuation and propagation characteristics between vehicles and 5G NR base stations, following [26]. FL training uses the MNIST dataset with stochastic gradient descent (SGD), under a non-IID data distribution across vehicles. The main evaluation metrics are model accuracy, indicating the predictive performance of the model [24], and convergence time, representing the number of communication rounds required to achieve model stability. Convergence is defined using a threshold  $\epsilon$ , where learning is considered complete if the improvement in model performance falls below  $\epsilon$  over three consecutive rounds [27].

A comprehensive set of simulations is conducted to evaluate the contribution of each component of the proposed algorithm under three scenarios: (i) 25 vehicles with a 100-meter transmission range, (ii) 25 vehicles with a 500-meter transmission range, and (iii) 50 vehicles with a 100-meter transmission range. In each case, 20% of the vehicles are designated as attackers, employing harsh fake additive

Gaussian noise (mean = 2, variance = 0.3), gradient ascent attacks, or a combination of both. The evaluation models a worst-case scenario where malicious vehicles begin attacking from the first communication round and continue without interruption. Both a single-hop configuration, allowing direct CM–CH communication, and a 3-hop setup, involving data relaying through intermediate CMs, are considered to assess performance under different network topologies. Key parameter settings for this scenario include a vehicle selection ratio of 75% per training round, an unblock time fixed at 5 rounds, and specific thresholds: a  $Z\_Score\_Threshold$  of 3, an  $Cosine\_Sim\_Threshold_{adaptive,k}$  of 0.90,  $High\_Threshold\_Up$  of 95%,  $High\_Threshold\_Down$  of 0.2, an Adjustment Factor ( $\delta$ ) of 0.05, and a  $Cosine\_Sim\_Threshold_{cross}$  of 0.9. The weighting of each parameter towards the reliability score is equal.

### B. Performance Evaluation of Proposed Algorithm

In this section, we evaluate the impact of incorporating adaptive thresholding and cross-cluster consistency checks into the proposed algorithm. The goal is to demonstrate their effectiveness and highlight the necessity of integrating both components. The analysis focuses on a scenario with 25 vehicles operating at a 100-meter transmission range in a single-hop configuration. By comparing the performance of the algorithm with and without these features, we provide a detailed assessment of their contributions to the overall robustness and security of the framework.

Figure 1 illustrates the impact of adaptive thresholding on the performance of the proposed algorithm for various types of attacks: high noise, gradient ascent, and a combination of both. Under Gaussian noise attack, static thresholding exhibits an accuracy drop of roughly 5% from the 95% baseline, while adaptive thresholding shows a less significant decrease of around 1%. The sharp dip under static thresholding, reaching as low as 75%, is completely mitigated by adaptive thresholding. Under gradient ascent attacks, static thresholding results in a loss of approximately 6%, whereas adaptive thresholding restricts the decline to nearly 2%. Even in the most severe combined noise and gradient ascent scenario, static thresholding causes a drop of almost 7%, while adaptive thresholding confines the loss to approximately 3%. These results clearly demonstrate that static thresholds allow performance degradation of up to 7%, whereas adaptive thresholding, by leveraging each vehicle’s historical reliability, limits accuracy loss to within 1–3% of the baseline, thereby preserving overall system stability under all attack types.

Figure 2 demonstrates the impact of the cross-cluster consistency check on the performance of the proposed algorithm, evaluated under the same scenarios and attack conditions. Incorporating the cross-cluster consistency check enhances system resilience by identifying deviations from the mean average cosine similarity across clusters. If a CH or a cluster’s weighted average exhibits malicious behavior significantly deviating from the global model, this anomaly is immediately detected and mitigated by excluding the malicious entity. As illustrated in the figure, under Gaussian noise attack, omitting

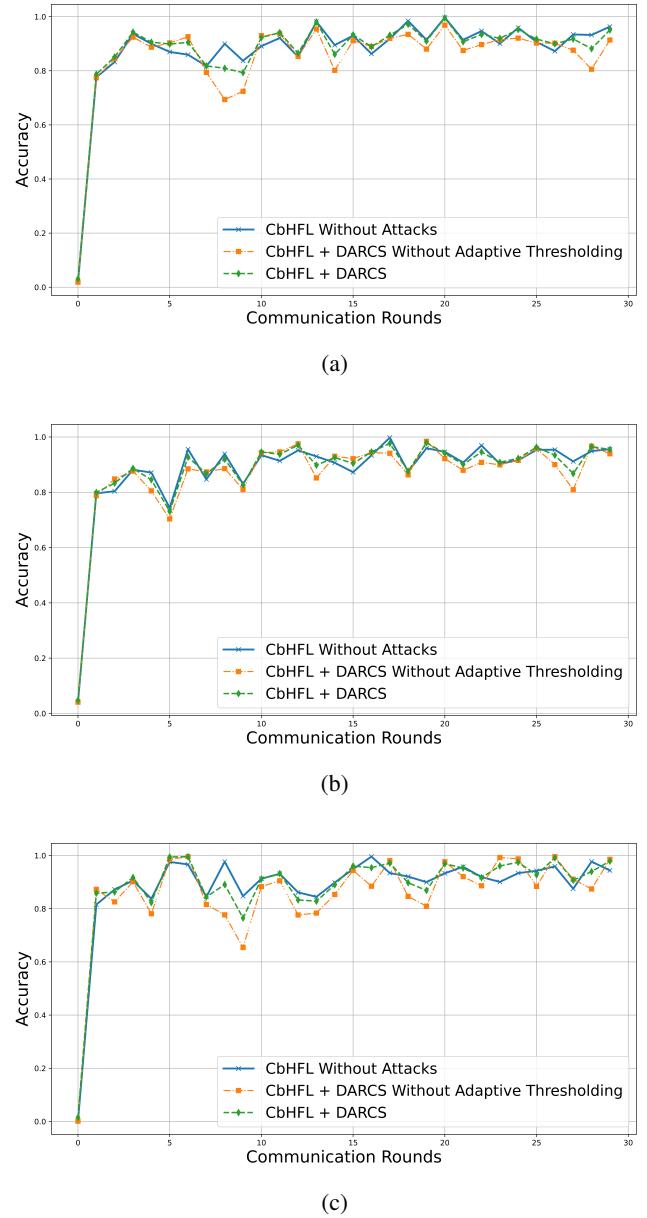
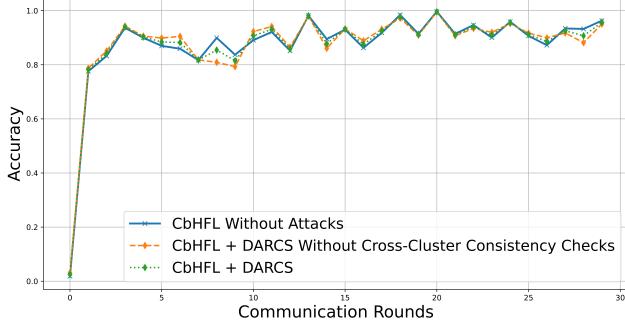
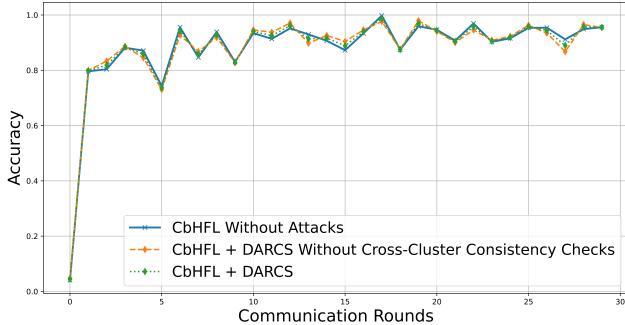


Figure 1: Comparison of accuracy at the EPC with and without adaptive thresholding. a) Noise attack with Mean= 2, Variance= 0.3, b) Gradient ascent attack, c) Noise and gradient ascent attack.

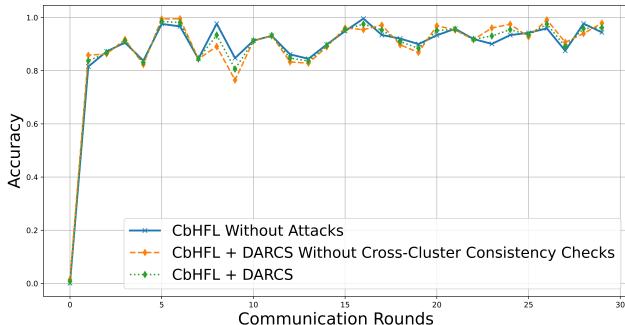
the cross-cluster consistency check results in an accuracy drop of roughly 6% from the 95% baseline, while incorporating the consistency check limits the decrease to about 3%. Without the check, the deepest accuracy dip, reaching 80%, is entirely smoothed out by the consistency check. Under gradient ascent attacks, performance without the consistency check declines by around 5%, whereas including the check reduces the drop to about 2%. In the combined noise and gradient ascent scenario, the no-check configuration leads to a degradation of nearly 7%, while the consistency check confines the loss to approximately 3%. These results demonstrate that, without cross-cluster consistency checks, the system is unable to detect



(a)



(b)



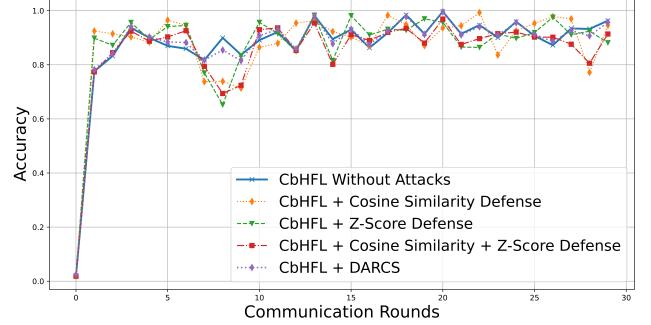
(c)

Figure 2: Comparison of accuracy at the EPC with and without cross-cluster consistency check. a) Noise attack with Mean= 2, Variance= 0.3, b) Gradient ascent attack, c) Noise and gradient ascent attack.

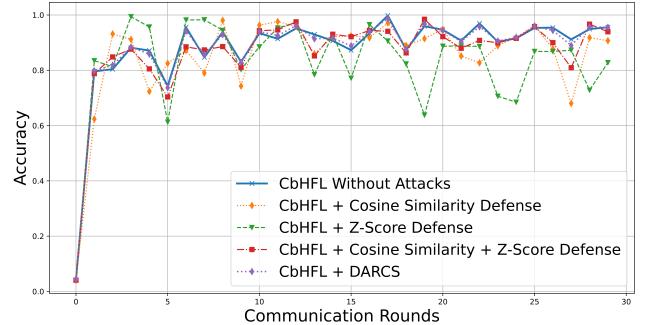
coordinated malicious updates, leading up to 7% accuracy loss. In contrast, verifying inter-cluster alignment through the consistency check limits the degradation to within 2–3% of the baseline, preserving system stability across all attack types.

### C. Performance Comparison of Proposed Algorithm with Benchmark Algorithms

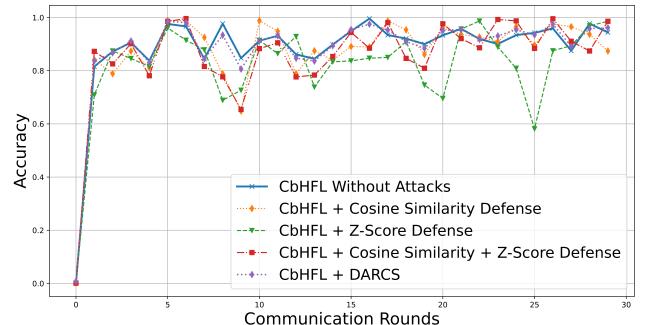
Figure 3 shows the accuracy at the EPC under various attack scenarios in a 1-hop communication setup with 25 vehicles and a 100-meter transmission range. The baseline performance of CbHFL without any attacks achieves approximately 95%



(a)



(b)



(c)

Figure 3: Comparison of accuracy at the EPC with benchmark algorithms for 25 vehicles, 100-meter transmission range, and 1-hop communication: a) Noise attack with Mean= 2, Variance= 0.3, b) Gradient ascent attack, c) Noise and gradient ascent attack.

accuracy, representing the ideal state. Under severe adversarial conditions—particularly when high Gaussian noise is combined with gradient ascent attacks—benchmark methods experience substantial performance degradation, resulting in significant accuracy fluctuations and system instability. Under pure Gaussian noise, the Z score only approach exhibits a drop in accuracy of approximately 12% from the baseline, while the cosine similarity only method shows a slight decrease of around 10%. Even when these defenses are combined, the aggregated accuracy remains about 8% lower than the ideal. Under gradient ascent attacks, the cosine similarity

Table I: Convergence time under varying attack types, number of vehicles, transmission ranges, and convergence threshold  $\epsilon$  for 1-hop communication.

25 Vehicles, Tx Range = 100 Meters						
		CbHFL Without Attacks	CbHFL + Cosine Similarity Defense	CbHFL + Z-Score Defense	CbHFL + Cosine Similarity + Z-Score Defense	CbHFL + DARCS
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	61	72	76	69	63
	$\epsilon = 0.005$	65	75	$\infty$	74	68
	$\epsilon = 0.001$	73	$\infty$	$\infty$	$\infty$	74
Gradient Ascent Attack	$\epsilon = 0.01$	53	68	74	57	55
	$\epsilon = 0.005$	58	73	$\infty$	68	61
	$\epsilon = 0.001$	69	74	$\infty$	74	72
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	62	74	$\infty$	70	65
	$\epsilon = 0.005$	70	$\infty$	$\infty$	74	73
	$\epsilon = 0.001$	74	$\infty$	$\infty$	$\infty$	74
25 Vehicles, Tx Range = 500 Meters						
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	59	66	71	67	59
	$\epsilon = 0.005$	58	70	$\infty$	68	63
	$\epsilon = 0.001$	65	$\infty$	$\infty$	$\infty$	68
Gradient Ascent Attack	$\epsilon = 0.01$	47	62	68	52	49
	$\epsilon = 0.005$	55	67	75	62	58
	$\epsilon = 0.001$	64	70	$\infty$	69	69
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	58	68	70	65	61
	$\epsilon = 0.005$	66	73	$\infty$	71	66
	$\epsilon = 0.001$	69	$\infty$	$\infty$	74	69
50 Vehicles, Tx Range = 100 Meters						
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	68	80	81	75	71
	$\epsilon = 0.005$	73	81	$\infty$	82	75
	$\epsilon = 0.001$	77	$\infty$	$\infty$	$\infty$	80
Gradient Ascent Attack	$\epsilon = 0.01$	59	75	80	63	59
	$\epsilon = 0.005$	62	75	$\infty$	74	66
	$\epsilon = 0.001$	72	81	$\infty$	78	77
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	68	77	$\infty$	76	71
	$\epsilon = 0.005$	73	$\infty$	$\infty$	81	77
	$\epsilon = 0.001$	77	$\infty$	$\infty$	$\infty$	78

method outperforms the Z-score approach, with accuracy reductions of approximately 10% versus 15%, respectively. When both attack types are compounded, benchmark methods incur losses ranging from 12% (cosine similarity) to as much as 17% (Z-score). In contrast, our proposed algorithm (CbHFL+DARCS) consistently maintains performance very close to the baseline, with only a minimal drop of approximately 2–3% even under the most challenging conditions. Across all attack types, DARCS delivers the highest resilience, followed by the combined Z-score and cosine similarity defense, followed by the cosine similarity-only method, with Z-score-only performing worst. Under pure Gaussian noise, our proposed algorithm best mitigates random perturbations, while Z-score-only suffers the greatest degradation. In the gradient ascent scenario, DARCS again leads, the combined defense comes second, and Z-score-only remains the weakest. When attacks are compounded, DARCS still holds within 3% of the baseline, the combined defense offers partial mitigation, and Z-score-only incurs the most severe drop. Moreover, similar trends are observed in a 3-hop environment, though performance degradation in benchmark techniques is even more pronounced due to reduced inter-cluster diversity in larger clusters. Overall, these results clearly demonstrate that while conventional benchmark techniques provide only partial protection, our DARCS algorithm delivers superior robustness and stability, achieving performance within just 2–3% of the baseline, thus offering a highly reliable solution for dynamic vehicular HFL environments.

Table I compares the convergence times of the proposed algorithm with benchmark methods under various attack scenarios, number of vehicles, transmission ranges, and  $\epsilon$  values in a 1-hop communication setup. The proposed algorithm consistently achieves significantly faster convergence compared to the benchmarks, aligning with the trends observed in Figure 3. Under high noise levels and combined noise and gradient ascent attacks, benchmark methods fail to converge at lower  $\epsilon$  values. In contrast, the proposed algorithm reliably converges in rounds remarkably close to those observed in attack-free scenarios. Furthermore, the proposed algorithm maintains stable convergence times irrespective of attack intensity or conditions. As the number of vehicles increases, the convergence time of the proposed algorithm remains comparable to that of the attack-free scenario. Similarly, in cases where attacks become more distributed due to increased transmission ranges, the algorithm continues to deliver efficient convergence. These results emphasize the effectiveness of the proposed algorithm, which achieves a performance profile remarkably close to that of a no-attack environment, even under diverse and challenging adversarial scenarios.

Table II provides an analysis of convergence times for the proposed algorithm versus benchmark approaches, evaluated under different attack models, vehicle densities, transmission distances, and  $\epsilon$  settings within a 3-hop network configuration. In the 1-hop configuration, vehicles are organized into many small clusters, which enables highly sensitive cross-cluster consistency checks for anomaly detection but provides limited

Table II: Convergence time under varying attack types, number of vehicles, transmission ranges, and convergence threshold  $\epsilon$  for 3-hop communication.

25 Vehicles, Tx Range = 100 Meters						
		CbHFL Without Attacks	CbHFL + Cosine Similarity Defense	CbHFL + Z-Score Defense	CbHFL + Cosine Similarity + Z-Score Defense	CbHFL + DARCS
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	60	70	75	68	60
	$\epsilon = 0.005$	62	73	$\infty$	70	66
	$\epsilon = 0.001$	69	$\infty$	$\infty$	$\infty$	72
Gradient Ascent Attack	$\epsilon = 0.01$	51	66	71	54	51
	$\epsilon = 0.005$	56	69	$\infty$	66	60
	$\epsilon = 0.001$	66	71	$\infty$	74	71
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	59	70	$\infty$	67	63
	$\epsilon = 0.005$	69	$\infty$	$\infty$	73	69
	$\epsilon = 0.001$	71	$\infty$	$\infty$	$\infty$	72
25 Vehicles, Tx Range = 500 Meters						
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	58	62	71	62	59
	$\epsilon = 0.005$	56	68	$\infty$	66	59
	$\epsilon = 0.001$	67	71	$\infty$	70	67
Gradient Ascent Attack	$\epsilon = 0.01$	46	60	65	49	48
	$\epsilon = 0.005$	48	65	68	62	54
	$\epsilon = 0.001$	62	66	$\infty$	68	64
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	53	65	69	61	57
	$\epsilon = 0.005$	64	70	$\infty$	66	65
	$\epsilon = 0.001$	66	74	$\infty$	72	69
50 Vehicles, Tx Range = 100 Meters						
Noise Attack with Mean = 2, Var = 0.3	$\epsilon = 0.01$	64	76	77	70	65
	$\epsilon = 0.005$	67	77	$\infty$	75	69
	$\epsilon = 0.001$	76	$\infty$	$\infty$	$\infty$	76
Gradient Ascent Attack	$\epsilon = 0.01$	57	71	75	60	59
	$\epsilon = 0.005$	61	75	$\infty$	70	62
	$\epsilon = 0.001$	71	77	$\infty$	75	73
Noise + Gradient Ascent Attack	$\epsilon = 0.01$	66	78	$\infty$	72	66
	$\epsilon = 0.005$	71	$\infty$	$\infty$	78	74
	$\epsilon = 0.001$	77	$\infty$	$\infty$	$\infty$	75

aggregation-driven noise suppression. In contrast, the 3-hop hierarchy consolidates vehicles into larger clusters and introduces an additional layer of hierarchical aggregation. This multi-tier structure enhances weighted averaging across both spatial and hierarchical dimensions, thereby more effectively diluting random perturbations and attenuating the impact of adversarial gradient manipulations. Consequently, convergence in the 3-hop topology is marginally faster than in the 1-hop case across all attack scenarios. The larger cluster size reduces variance in the aggregated updates, accelerating convergence under Gaussian noise, while the hierarchical consistency checks inherent to the 3-hop design mitigate gradient ascent attacks by cross-validating updates at multiple aggregation levels. Benchmark methods based solely on Z-score or cosine similarity cannot fully exploit these advantages and thus exhibit slower convergence. Overall, DARCS+CbHFL leverages robust statistical filtering and hierarchical consensus to achieve rapid and stable convergence in adversarial settings. The 3-hop architecture not only preserves sensitivity to anomalies but also significantly enhances resilience through better noise dilution and redundant validation pathways.

## V. CONCLUSION

In this paper, we present a novel framework to improve both the security and integrity of HFL within VANETs. The framework integrates dynamic client selection with robust anomaly detection, ensuring that only the most trustworthy vehicles participate in the learning process. Trustworthiness

is evaluated through a composite reliability score based on historical accuracy, contribution frequency, and past anomaly records. To strengthen resilience against evolving threats, adaptive thresholding dynamically adjusts detection sensitivity, while cross-cluster consistency checks validate update integrity across clusters, effectively mitigating risks from coordinated malicious behaviors. A reliability-based weighted gradient averaging strategy further improves global model robustness by assigning greater influence to consistently high-performing vehicles. Malicious contributions are detected and isolated using a combination of cosine similarity and Z-score analysis. Through comprehensive simulations across various attack scenarios, including fake noise, gradient ascent, and their combination, DARCS consistently outperforms all benchmark defenses, achieving up to a 17% reduction in convergence time and limiting accuracy loss to within 2–3% of the ideal attack-free baseline across both 1-hop and 3-hop topologies. These gains demonstrate the contribution of each component: adaptive thresholding improves performance by 4–6%, while cross-cluster consistency checks add a further 3–5%. By effectively isolating malicious participants and preserving model integrity, DARCS provides a robust, scalable solution for dynamic VANET environments. Future work will focus on enhancing privacy protections and extending the framework to guard against malicious exploitation of model parameters.

## REFERENCES

- [1] C. Feng, D. Feng, Z. Zhao, G. Min, and H. Duan, *Federated Learning with non-IID data in Mobile Edge Computing Systems*, ch. 2, pp. 23–48. John Wiley & Sons, Ltd, 2023.
- [2] L. Liu, J. Zhang, S. Song, and K. B. Letaief, “Client-edge-cloud hierarchical federated learning,” in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2020.
- [3] M. S. H. Abad, E. Ozfatura, D. GUndUz, and O. Ercetin, “Hierarchical federated learning across heterogeneous cellular networks,” in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8866–8870, 2020.
- [4] T. Chen, J. Yan, Y. Sun, S. Zhou, D. Gündüz, and Z. Niu, “Mobility accelerates learning: Convergence analysis on hierarchical federated learning in vehicular networks,” *arXiv preprint arXiv:2401.09656*, 2024.
- [5] Q. Chen, Z. You, and H. Jiang, “Semi-asynchronous hierarchical federated learning for cooperative intelligent transportation systems,” *arXiv preprint arXiv:2110.09073*, 2021.
- [6] M. S. HaghghiFard and S. Coleri, “Hierarchical federated learning in multi-hop cluster-based vanets,” *arXiv preprint arXiv:2401.10361*, 2024.
- [7] G. Xia, J. Chen, C. Yu, and J. Ma, “Poisoning attacks in federated learning: A survey,” *IEEE Access*, vol. 11, pp. 10708–10722, 2023.
- [8] K. N. Kumar, C. K. Mohan, and L. R. Cenkeramaddi, “The impact of adversarial attacks on federated learning: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2023.
- [9] X. Cao and N. Gong, “Mpaf: Model poisoning attacks to federated learning based on fake clients,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, (Los Alamitos, CA, USA), pp. 3395–3403, IEEE Computer Society, jun 2022.
- [10] M. T. Hossain, S. Islam, S. Badsha, and H. Shen, “Desmp: Differential privacy-exploited stealthy model poisoning attacks in federated learning,” in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 167–174, 2021.
- [11] J. Sun, A. Li, L. DiValentin, A. Hassanzadeh, Y. Chen, and H. Li, “Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective,” *arXiv preprint arXiv:2110.13864*, 2021.
- [12] X. You, Z. Liu, X. Yang, and X. Ding, “Poisoning attack detection using client historical similarity in non-iid environments,” in *2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 439–447, 2022.
- [13] N. Galanis, “Defending against data poisoning attacks in federated learning via user elimination,” *arXiv preprint arXiv:2404.12778*, 2024.
- [14] M. Khorramfar, “Securing federated learning model aggregation against poisoning attacks via credit-based client selection,” *WinnSpace Home*, Aug 2023.
- [15] D. N. Yaldiz, T. Zhang, and S. Avestimehr, “Secure federated learning against model poisoning attacks via client filtering,” *arXiv preprint arXiv:2304.00160*, 2023.
- [16] J. B. Kenney, “Dedicated short-range communications (dsrc) standards in the united states,” *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.
- [17] M. Noor-A-Rahim, Z. Liu, H. Lee, M. O. Khyam, J. He, D. Pesch, K. Moessner, W. Saad, and H. V. Poor, “6g for vehicle-to-everything (v2x) communications: Enabling technologies, challenges, and opportunities,” *Proceedings of the IEEE*, vol. 110, no. 6, pp. 712–734, 2022.
- [18] H. Seo, K.-D. Lee, S. Yasukawa, Y. Peng, and P. Sartori, “Lte evolution for vehicle-to-everything services,” *IEEE Communications Magazine*, vol. 54, no. 6, pp. 22–28, 2016.
- [19] H. Bagheri, M. Noor-A-Rahim, Z. Liu, H. Lee, D. Pesch, K. Moessner, and P. Xiao, “5g nr-v2x: Toward connected and cooperative autonomous driving,” *IEEE Communications Standards Magazine*, vol. 5, no. 1, pp. 48–54, 2021.
- [20] R. Shokri and V. Shmatikov, “Privacy-preserving deep learning,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 1310–1321, ACM, 2015.
- [21] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2019.
- [22] A. N. Bhagoji, S. Chakraborty, P. Mittal, and J. Calo, “Analyzing federated learning through an adversarial lens,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1360–1367, AAAI, 2019.
- [23] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.
- [24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [25] M. Sepulcre, M. Gonzalez-Martín, J. Gozalvez, R. Molina-Masegosa, and B. Coll-Perales, “Analytical models of the performance of ieee 802.11p vehicle to vehicle communications,” *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 713–724, 2022.
- [26] 3rd Generation Partnership Project (3GPP), “Study on channel model for frequencies from 0.5 to 100 ghz,” Tech. Rep. TR 38.901, 3GPP, 2017.
- [27] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.