FedG-ELITE: Distributed Intelligence Evolution for Software-Defined Vehicular Networks

Research Report on Enhanced Routing Protocol Based on Federated Graph Learning

Architecture Evolution Based on ELITE (IEEE TMC 2023)

November 19, 2025

Abstract

With the rapid development of Intelligent Transportation Systems (ITS), Vehicular Networks (VNs) have become the core nervous system connecting vehicles, road infrastructure, and pedestrians. To address the challenges posed by high vehicle dynamics, frequent network topology changes, and diverse Quality of Service (QoS) requirements, the Software-Defined Vehicular Network (SDVN) architecture has emerged. SDVN achieves centralized control and global optimization of the network by decoupling the control plane from the data plane. The existing ELITE scheme has alleviated traditional routing problems to a certain extent by introducing digital twins and hierarchical routing.

However, ELITE is still limited by the scalability dilemma of tabular reinforcement learning, the lack of generalization ability to new topologies, and the data privacy and bandwidth bottlenecks caused by centralized training. Addressing these limitations, this report proposes a comprehensively upgraded architecture schemeFedG-ELITE (Federated Graph-ELITE). This scheme retains the hierarchical routing and multi-objective fuzzy fusion framework of ELITE but introduces Graph Neural Networks (GNN) to reconstruct state representation, utilizes Deep Reinforcement Learning (DRL) to replace tabular learning, and adopts a Federated Learning (FL) architecture to implement distributed collaborative training. This paper details the system architecture of FedG-ELITE, the mathematical modeling based on GraphSAGE, the workflow of the federated graph learning algorithm, and performance expectations, aiming to provide theoretical support for the next generation of highly reliable and privacy-preserving intelligent vehicular network routing protocols.

Index Terms Software-Defined Vehicular Network (SDVN), Federated Learning, Graph Neural Network (GNN), Deep Reinforcement Learning, Digital Twin, ELITE, Routing Protocol.

1 Introduction and System Background Analysis

1.1 Research Background

With the rapid development of Intelligent Transportation Systems (ITS), Vehicular Networks (VNs) have become the core nervous system connecting vehicles, road infrastructure, and pedestrians. To address the challenges posed by high vehicle dynamics, frequent network topology changes, and diverse Quality of Service (QoS) requirements, the Software-Defined Vehicular Network (SDVN) architecture has emerged. SDVN achieves centralized control and global optimization of the network by decoupling the control plane from the data plane, providing new opportunities to solve local optimum and link breakage problems in traditional distributed routing protocols.

1.2 Status of ELITE Scheme

In this field, the ELITE (Intelligent Digital Twin Hierarchical Routing) scheme represents the current state-of-the-art. This scheme innovatively introduces Digital Twin (DT) technology to construct a virtual network space within the SDVN controller. It utilizes Multi-Agent Reinforcement Learning (RL) to parallelly train policies for different routing metrics (such as delay, packet loss rate, hop count) and performs multi-objective fusion through Fuzzy Logic. ELITE adopts a hierarchical routing structure, where the controller is responsible for macroscopic path planning based on junction sequences, while vehicles execute location-based greedy forwarding within road segments.

1.3 Core Technical Bottlenecks of ELITE Architecture

However, a deep analysis of the ELITE architecture reveals that although it alleviates the computational pressure of traditional SDN routing to some extent, it is still constrained by the following core technical bottlenecks, which are particularly prominent in large-scale, ultra-dense future urban traffic scenarios:

1. **The Curse of Dimensionality in Tabular Reinforcement Learning:** ELITE employs tabular Q-learning for policy training. In tabular RL, the state space typically consists of discrete junction IDs. As the scale of the urban road network expands, the number of state-action pairs grows exponentially. This results in an extremely large Q-table that is difficult to converge within a limited time and imposes tremendous pressure on the controller's storage resources.

2. **Lack of Generalization Ability:** Tabular methods lack inductive bias. The congestion avoidance strategy learned by an agent at a specific junction cannot be transferred to other junctions with similar topological structures. ELITE relies on specific ID records; once the road network topology undergoes minor changes (such as new roads or construction), the old Q-values cannot correspond to the new states, often requiring the entire model to be retrained, which is extremely inefficient in the rapidly changing vehicular network environment.

3. **Data Privacy and Bandwidth Bottlenecks in Centralized Training:** ELITE relies on the full mapping of the physical world by the digital twin, which means that underlying vehicles and Road-Side Units (RSUs) need to continuously upload raw sensing data (position, speed, neighbor lists) to the controller. This centralized

data processing mode not only consumes valuable Vehicle-to-Infrastructure (V2I) wireless bandwidth but also triggers serious user privacy leakage risks, such as vehicle trajectory tracking and user behavior profiling.

1.4 Proposed Solution: FedG-ELITE

Addressing the aforementioned limitations, this report proposes a comprehensively upgraded architecture scheme, FedG-ELITE (Federated Graph-ELITE). This scheme aims to perform the following core improvements while retaining ELITE's excellent "hierarchical routing" and "multi-objective fuzzy fusion" framework:

1. **Introducing Graph Neural Networks (GNN):** Reconstruct state representation and utilize its ability to represent non-Euclidean data to solve the state space explosion problem.
2. **Deep Reinforcement Learning (DRL):** Utilize Deep Q-Network to replace tabular learning to improve fitting capability.
3. **Federated Learning (FL):** Implement distributed collaborative training in the architecture, moving "training" down to the edge to achieve data privacy protection.

2 Architecture Evolution: From Centralized Digital Twin to Federated Graph Intelligence

The core idea of FedG-ELITE is to move the "training" process of routing intelligence down to the edge (vehicles and RSUs), while retaining "aggregation" and "global optimization" in the digital twin control plane. This architectural shift not only solves privacy issues but also leverages the powerful representation capability of Graph Neural Networks for non-Euclidean spatial data. The FedG-ELITE architecture is divided into three logical planes.

2.1 Physical Sensing and Execution Plane (Federated Clients)

Vehicle nodes are no longer simple data senders but Federated Learning Clients with computing capabilities. Their main responsibilities include:

- **Graph Data Construction:** Vehicles sense the local traffic environment through V2V/V2I communication and model it as a Local Subgraph. Nodes represent vehicles or junctions, edges represent communication links, and node features include speed, queue length, historical throughput, etc.
- **Local Model Training:** Each client runs the GNN-DRL model for training using locally collected trajectory and interaction data. GNN is used to extract local topological features, and DRL is used to decide the next-hop routing.
- **Parameter Upload:** After training, clients only upload model parameters (gradients or weights Δ) to the RSU or controller, while raw traffic data remains local, strictly following the principle of "data stays, model moves."

2.2 Edge Computing Plane (Edge Aggregation)

Road-Side Units (RSUs) act as an intermediate aggregation layer, responsible for processing model updates from vehicles within their coverage.

- **Regional Aggregation:** RSUs perform preliminary aggregation (e.g., weighted averaging) on model parameters uploaded by vehicles within their jurisdiction to reduce traffic back to the core network and alleviate backbone network pressure.
- **Feature Enhancement:** RSUs leverage their fixed geographical location advantages to provide passing vehicles with macroscopic features at the junction level (e.g., historical congestion index) to assist vehicles in constructing more complete state graphs.

2.3 Digital Twin Control Plane (Federated Server & Global Twin)

The digital twin maintained by the SDVN controller evolves into the central Parameter Server for federated learning.

- **Global Aggregation (FedAvg/FedProx):** Receives local model updates from RSUs or vehicles and executes global aggregation algorithms to generate a general routing model capable of adapting to network-wide features.
- **Multi-Objective Fusion (Legacy Integration):** FedG-ELITE retains the fuzzy logic module of ELITE. The digital twin maintains multiple global GNN models trained for different optimization objectives (high reliability, low latency, load balancing). When there is a data transmission request, the controller uses fuzzy logic to select the most appropriate global model parameters for distribution based on the service type (e.g., safety messages, entertainment streaming), or calculates the macroscopic path directly at the controller end.
- **Model Distribution and Synchronization:** Broadcasts the updated global model parameters back to the physical network to ensure continuous evolution.

3 Mathematical Modeling: Graph-Driven Deep Reinforcement Learning and Federated Optimization

This section will detail the key mathematical models in FedG-ELITE, including the construction of dynamic graphs, GraphSAGE state extraction, the DQN decision process, and the federated aggregation mechanism.

3.1 Dynamic Graph Formulation

We abstract the vehicular network as a time-varying dynamic directed graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$.

- **Node Set:** Contains two types of nodes: junction nodes $J = \{j_1, j_2, ..., j_M\}$ and vehicle nodes $V = \{v_1, v_2, ..., v_N\}$. At the macroscopic routing level, we primarily focus on junction nodes and their connectivity.

- **Edge Set:** Represents the connectivity of physical road segments. If there is a direct road connection between junction i and j, then an edge $e_{ij} \in \mathcal{E}_t$ exists.
- **Feature Matrix X:** Each node possesses a feature vector $x_i^{(t)} \in \mathbb{R}^F$, capturing real-time traffic status:

$$x_i^{(t)} = [\rho_i(t), \overline{v}_i(t), \mathcal{Q}_i(t), PDR_i^{hist}(t), \tau_i(t)]^\top \tag{1}$$

Where:

- $\rho_i(t)$: Vehicle density within the junction coverage (normalized).
- $\overline{v}_i(t)$: Average vehicle speed.
- $\mathcal{Q}_i(t)$: Packet queue length at the current junction.
- $PDR_i^{hist}(t)$: Average delivery ratio within the historical window.
- $\tau_i(t)$: Average traversal time of the road segment (used for delay estimation).

3.2 Inductive State Representation Based on GraphSAGE

To address the issue that tabular states in ELITE cannot generalize, we introduce GraphSAGE (Graph Sample and Aggregate). Unlike traditional Graph Convolutional Networks (GCN), GraphSAGE is inductive; it learns a function to aggregate neighbor features rather than learning fixed embeddings for specific nodes, enabling the model to handle unseen network topologies.

For any junction node v, GraphSAGE generates its embedding vector $h_v^{(K)}$ through K layers of aggregation operations. At the k-th layer ($k \in \{1, ..., K\}$):

1) **Neighbor Aggregation:** Collect information from neighbor nodes $\mathcal{N}(v)$. Considering the differences in link quality in vehicular networks, we adopt a weighted mean aggregator:

$$h_{\mathcal{N}(v)}^{(k)} = AGGREGATE_k(\{h_u^{(k-1)}, \forall u \in \mathcal{N}(v)\}) = \sum_{u \in \mathcal{N}(v)} \alpha_{vu} W_{pool}^{(k)} h_u^{(k-1)} + b_{pool}^{(k)} \tag{2}$$

Where α_{vu} is the attention coefficient (can be calculated by the Graph Attention Network mechanism, reflecting link stability), and W_{pool} and b_{pool} are trainable parameters.

2) **Feature Update:** Concatenate the aggregated neighbor features with the node's own features and update the node representation through a non-linear transformation:

$$h_v^{(k)} = \sigma(W^{(k)} \cdot CONCAT(h_v^{(k-1)}, h_{\mathcal{N}(v)}^{(k)})) \tag{3}$$

Where σ typically uses the ReLU activation function, and $W^{(k)}$ is the weight matrix of the k-th layer. The finally generated embedding vector $z_v = h_v^{(K)}$ fuses the node's own traffic attributes and topological structure information within a K-hop range, serving as the high-dimensional state input for reinforcement learning.

3.3 Deep Q-Network (DQN) Routing Decision Model

After obtaining the state representation z, we use a Deep Q-Network (DQN) to replace the Q-table in ELITE.

- **State Space S:** $s_t = z_{current} \oplus z_{destination}$, i.e., the concatenation of the graph embedding vectors of the current junction and the destination junction. This enables the policy to perceive not only the local environment but also to be "goal-oriented."
- **Action Space A:** $a_t \in \mathcal{N}(current)$, i.e., selecting the next junction as a relay.
- **Q-Value Function and Loss Function:** The DQN network $Q(s, a; \theta)$ receives the state vector and outputs Q-values corresponding to all selectable neighbor junctions. The parameters θ here include the parameters of GraphSAGE and the subsequent fully connected layers (MLP).

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta))^2] \tag{4}$$

Where θ^- represents the target network parameters, updated every fixed number of steps to ensure training stability.

3.4 Multi-Objective Reward Shaping

To reproduce and enhance the multi-objective characteristics of ELITE, we designed a parameterized hybrid reward function. Although ELITE uses independent agents, in FedG-ELITE, we can train model variants for different QoS requirements by adjusting reward weights. Define the comprehensive reward R_t:

$$R_t = \omega_1 \cdot R_{PDR} + \omega_2 \cdot R_{Delay} + \omega_3 \cdot R_{Load} + \omega_4 \cdot R_{Hop} \tag{5}$$

The sub-rewards are defined as follows:

- **Packet Delivery Ratio Reward** (R_{PDR})**:** If the packet successfully reaches the destination, reward $+R_{succ}$; if lost due to TTL expiration, penalize $-R_{fail}$. A small positive incentive is given for each successful forwarding hop.
- **Delay Reward** (R_{Delay})**:** $R_{Delay} = -(t_{arrival} - t_{sent})$, or use a penalty based on link quality $R = -(Queue\_Delay + Transmission\_Delay)$.
- **Load Balancing Reward** (R_{Load})**:** $R_{Load} = 1 - \frac{\mathcal{Q}_{next}}{\mathcal{Q}_{max}}$, encouraging the selection of idle junctions.
- **Hop Count Reward** (R_{Hop})**:** Fixed penalty $-C$ for each hop, prompting the agent to find the shortest path.

Reward Scale Alignment and Quantile Adaptation: To avoid a single reward term (e.g., millisecond-level delay values) dominating the entire reward function, it is necessary to perform **normalization** and **weight recalibration** for each component $R_{Delay}, R_{Load}, R_{Hop}$. For example, map R_{Delay} to the $[-1, 1]$ interval. Furthermore, addressing the fixed-intersection fuzzy membership functions used in ELITE (Figure 4a), this

scheme adopts a **Quantile Adaptation** mechanism, dynamically adjusting the threshold for Good/Medium/Bad classifications based on online collected reward distribution statistics (e.g., P_{20}, P_{50}, P_{80} quantiles). It is also recommended to provide reasonable orders of magnitude and sensitivity analysis for α, β (e.g., parameter scanning within the range $[0.1, 0.9]$) to ensure model adaptability to different network conditions.

By adjusting the weight vector $\omega = [\omega_1, \omega_2, \omega_3, \omega_4]$, we can generate various policy models biased towards High Reachability First (HRF), Low Delay First (LDF), or Load Balancing First (LBF) for subsequent invocation by the fuzzy logic module.

4 Federated Graph Learning Algorithm Design

The core innovation of FedG-ELITE lies in utilizing a federated learning framework to train the aforementioned GNN-DRL models. Due to the Non-Independent and Identically Distributed (Non-IID) nature of the vehicular network environment (e.g., traffic patterns in city centers vs. suburbs are vastly different), the standard FedAvg algorithm may face convergence difficulties. Therefore, we introduce an improved aggregation mechanism. We adopt the combined framework of GraphSAGE + DQN + FedProx to approximate the optimal policy. This class of deep federated reinforcement learning methods has been widely verified in related works. However, due to the use of nonlinear function approximation and dynamic graph modeling, we mainly evaluate its performance and stability through extensive simulation experiments, rather than pursuing a rigorous convergence proof.

4.1 Federated Learning Workflow

The system consists of N clients (RSUs/Vehicles) and a central server (DT Controller). The global model parameters are denoted as $\mathcal{W}_g$, and the local parameters of the i-th client are denoted as $\mathcal{W}_i$.

1) **Step 1: Model Distribution:** At the beginning of communication round t, the server broadcasts the current global model $\mathcal{W}_g^t$ to the selected subset of clients $\mathcal{K}_t$.

2) **Step 2: Local Training:** Each client $i \in \mathcal{K}_t$ uses locally collected trajectory data $\mathcal{D}_i$ to train the model for E epochs. To prevent the local model from overfitting on heterogeneous data and deviating from the global optimum, we introduce the FedProx regularization term in the loss function:

$$\min_{\mathcal{W}_i} \mathcal{L}_i(\mathcal{W}_i) + \frac{\mu}{2}||\mathcal{W}_i - \mathcal{W}_g^t||^2 \tag{6}$$

Where μ is the regularization coefficient, limiting the local update step size.

3) **Step 3: Model Upload and Aggregation:** Clients upload the updated parameters $\mathcal{W}_i^{t+1}$. The server performs weighted aggregation. Considering differences in RSU coverage and traffic flow, the aggregation weight p_i depends not only on data volume but also combines the node's topological importance (Degree Centrality) or traffic load.

Aggregation Normalization and Federated Frequency: The specific aggregation formula should be defined as:

$$W_g^{t+1} = \sum_{i \in \mathcal{K}_t} \frac{p_i}{\sum_{j \in \mathcal{K}_t} p_j} W_i^{t+1} \tag{7}$$

Where the calculation of weight p_i is recommended to combine data volume and centrality, and undergo **normalization/logarithmic transformation**, i.e., $p_i = \alpha \log(|\mathcal{D}_i|) + \beta \text{Norm}(\text{Centrality}_i)$, to prevent nodes with massive data from dominating the model. In addition, addressing the high-frequency communication overhead in vehicular networks, the upload frequency strategy is clarified as "upload once every T seconds or every N steps," and an **adaptive frequency reduction** mechanism is introduced (reducing upload frequency when the model tends to converge). Meanwhile, it is recommended to use **Quantization** or **Sparsification** techniques to compress uploaded gradient parameters, further reducing bandwidth consumption.

4.2 Core Algorithm Pseudocode

To clearly demonstrate the execution logic of FedG-ELITE, detailed pseudocode for the core algorithm is provided below.

4.3 Secure Aggregation and Differential Privacy (SA / DP)

To further complete the privacy threat model, regarding sensitive trajectory data, this scheme suggests adding a **Secure Aggregation (SA)** protocol to ensure the server can only see the aggregated gradients and cannot infer individual user updates. Simultaneously, a **Differential Privacy (DP)** option is provided, adding Gaussian noise before uploading gradients to defend against privacy threats such as Membership Inference Attacks.

5 System Implementation and Deployment Considerations

5.1 Decoupling of Training and Inference

In FedG-ELITE, Training and Inference are performed asynchronously.

- **Training:** A periodic background process. Vehicles utilize onboard computing units for local Epoch training when idle or charging (for electric vehicles) and upload updates when the network is idle.
- **Inference:** A real-time foreground process. When a data packet arrives, the vehicle directly calls the current version of the GNN-DQN model, outputting the next-hop decision in milliseconds without incurring extra communication latency.

5.2 Integration with Fuzzy Logic

A highlight of the original ELITE scheme is the use of fuzzy logic to handle multi-objective conflicts. FedG-ELITE does not discard this module but upgrades its input.

- **Original Input:** Numerical values in the Q-table (Q-values).
- **New Input:** Action probability distributions (Softmax Probability) output by multiple specialized models (PDR model, Delay model).

Algorithm 1 FedG-ELITE Client Local Training (ClientUpdate)

Require: Global model weights $\mathcal{W}_{global}$, Local dataset $\mathcal{D}_k$ (Replay Buffer), Learning rate η, Regularization coefficient μ
Ensure: Updated local weights $\mathcal{W}_k$

Function ClientUpdate ($\mathcal{W}_{global}$):
Initialize local model $\mathcal{W}_k \leftarrow \mathcal{W}_{global}$
// Data Collection Phase (Running Continuously)
Observe current graph state $G_t = (V, E)$ and node features X_t
GNN Forward Pass: Generate embeddings
$H_v \leftarrow$ GraphSAGE_Forward $(G_t, X_t, \mathcal{W}_k)$
$S_t \leftarrow$ Concatenate $(H_{current}, H_{dest})$
DQN Action Selection (Epsilon-Greedy):
if Random () $< \epsilon$ **then**
    $a_t \leftarrow$ Random_Neighbor()
else
    $Q_{values} \leftarrow$ DQN_Forward $(S_t, \mathcal{W}_k)$
    $a_t \leftarrow ArgMax(Q_{values})$
end if
Execute a_t, observe reward r_t, new state S_{t+1}
Store transition (S_t, a_t, r_t, S_{t+1}) into $\mathcal{D}_k$
// Training Phase
for epoch = 1 to E **do**
    Sample Random Batch $(\mathcal{D}_k)$
    Loss $\leftarrow 0$
    for each transition (s, a, r, s_{next}) in Batch **do**
        // Target Q-value Calculation
        $Q_{pred} \leftarrow$ DQN_Forward $(s, \mathcal{W}_k)$ $[a]$
        $Q_{target} \leftarrow r + \gamma \max($ DQN_Forward $(s_{next}, \mathcal{W}_{k.target}))$
        $L_{mse} \leftarrow$ HuberLoss (Q_{pred}, Q_{target})
        // Loss with FedProx Regularization Term
        $L_{prox} \leftarrow \frac{\mu}{2}||\mathcal{W}_k - \mathcal{W}_{global}||^2$
        Loss $\leftarrow Loss + L_{mse} + L_{prox}$
    end for
    Update $\mathcal{W}_k$ using Adam(∇ Loss)
end for
return $\mathcal{W}_k$

Algorithm 2 FedG-ELITE Server Global Aggregation and Fusion (ServerExec)

Require: Client set $\mathcal{K}$, Rounds T_0, Fusion rules F_t
Ensure: Deployed Policy

Initialize global models for different objectives: $\mathcal{W}_{PDR}, \mathcal{W}_{Delay}, \mathcal{W}_{Load}, \mathcal{W}_{Hop}$
for round $t = 1$ to T_0 **do**
  // Parallel Federated Training for each objective model
  for each Obj in {PDR, Delay, Load, Hop} **do**
    Select active client subset $\mathcal{S}_t \subset \mathcal{K}$
    Broadcast $\mathcal{W}_{Obj}$ to $\mathcal{S}_t$
    Received_Updates $\leftarrow$ []
    Total_Weight $\leftarrow 0$
    for client k in $\mathcal{S}_t$ (Parallel Execution) **do**
      $\mathcal{W}_k \leftarrow k$.ClientUpdate ($\mathcal{W}_{Obj}$)
      // Calculate Aggregation Weight
      $p_k \leftarrow$ CalculateWeight (k.Traffic Load, k.Graph Degree)
      Received_Updates.append($(\mathcal{W}_k, p_k)$)
      Total_Weight $\leftarrow$ Total_Weight + p_k
    end for
    // Weighted Aggregation
    $\mathcal{W}_{Obj}^{new} \leftarrow \sum(p_k \cdot \mathcal{W}_k)/Total\_Weight$
    $\mathcal{W}_{Obj} \leftarrow \mathcal{W}_{Obj}^{new}$
  end for
end for
Policy Deployment (Integration with ELITE Fuzzy Logic):
Function HandleRouting Request(Request_Type, Source, Dest):
$Prob_{PDR} \leftarrow$ GNN_Infer(Source, Dest, $\mathcal{W}_{PDR}$)
$Prob_{Delay} \leftarrow$ GNN_Infer(Source, Dest, $\mathcal{W}_{Delay}$)
$Prob_{Load} \leftarrow$ GNN_Infer(Source, Dest, $\mathcal{W}_{Load}$)
// Fuzzy Logic Fusion
Final_Score $\leftarrow$ FuzzyInference ($Prob_{PDR}, Prob_{Delay}, Prob_{Load}$, Request_Type)
return ArgMax(Final_Score)

For example, for the next hop, the PDR model predicts a success probability of 0.9 (High), and the Delay model predicts a congestion probability of 0.8 (High). The fuzzy controller synthesizes the final score of the node based on the rules of the current service "video streaming" (insensitive to delay but requires high throughput). This design preserves the flexibility of ELITE's service awareness.

Output Calibration and Data-Driven Fine-tuning: Before feeding Softmax probabilities into the fuzzy module, it is recommended to perform **Temperature Scaling** to solve the "Over-confident" problem of neural network output probabilities. Furthermore, it is suggested to change the original fixed fuzzy rules to **Data-Driven Fine-tuning**, where the weight parameters of fuzzy rules can be adaptively adjusted with changes in the network environment, rather than being hard-coded.

5.3 Inference Overhead Budget

To ensure the online availability of the scheme, specific overhead budgets must be provided. It is recommended to set: GNN layers $K \leq 2$ or 3, neighbor sampling count (fan-out) approximately 10, and embedding dimension controlled within 64-128 dimensions. Model size should be controlled at the MB level, and vehicle-side forward inference time should meet the real-time requirements of vehicular networks (e.g., $< 20ms$). Specific target values versus measured values should be provided to demonstrate feasibility.

6 Simulation Verification Strategy

To verify the effectiveness of FedG-ELITE, it is recommended to use SUMO (Simulation of Urban Mobility) to generate realistic urban traffic flows, combined with NS-3 or OMNET++ for network communication simulation.

- **Datasets:** Use real urban road network maps (e.g., Manhattan or San Francisco maps exported from OpenStreetMap).
- **Evaluation Metrics:**
    1. Convergence Speed: Number of training rounds required to reach stable PDR (FedG-ELITE is expected to be faster than Tabular Q-learning).
    2. Communication Overhead: Total bytes of uploaded data (expected to be significantly reduced).
    3. Generalization Test: Train in Area A, test in Area B, evaluate performance degradation (FedG-ELITE is expected to have minimal degradation).
    4. Privacy Leakage Analysis: Use Membership Inference Attack to test resistance capabilities.

7 Deep Comparative Analysis of Enhanced Scheme vs. Original ELITE Scheme

This section compares FedG-ELITE with the baseline ELITE scheme from three dimensions: architecture performance, privacy protection, and adaptability, clarifying the necessity and advantages of the improvements.

7.1 Generalization & Scalability

- **ELITE:** Relies on discrete IDs. If a new junction is added to the network, or a vehicle enters a never-visited area, the Q-table has no corresponding record and must start exploration from scratch, causing a sharp performance drop in the initial stage (Cold Start problem).
- **FedG-ELITE:** Based on inductive learning of GNN. The model learns general rules like "how to judge routing quality based on neighbor features," rather than rote memorization of "Junction ID5 to ID6 is good." Therefore, when encountering a new junction, as long as its features (traffic flow, speed) can be obtained, the model can immediately generate effective embeddings and make reasonable decisions, possessing strong Zero-shot Transfer capability.

7.2 Privacy & Communication

- **ELITE:** Belongs to centralized cloud training. Requires all vehicles to upload detailed movement trajectories and interaction logs to construct a high-fidelity digital twin. This not only brings huge uplink bandwidth pressure (Raw Data Volume) but also exposes vehicles to the risk of trajectory privacy leakage.
- **FedG-ELITE:** Achieves "data available but invisible." Vehicles only process raw data locally and only upload model gradients processed by encryption or differential privacy. The size of gradient data (usually KB level) is far smaller than raw sensing data streams, significantly reducing communication overhead.

7.3 Handling Non-IID Data

- **ELITE:** Centralized training assumes all data is uniformly mixed on the server side (IID), ignoring the differences in traffic patterns across different regions, which may lead to poor model performance in certain special areas (such as frequently congested school zones).
- **FedG-ELITE:** By adding the FedProx regularization term to the local loss function, it explicitly addresses the systematic heterogeneity of data. It allows the model to retain adaptability to local specific traffic patterns while adapting to global trends, improving robustness.

8 Conclusion

This report addresses the deficiencies of the existing ELITE routing scheme in terms of scalability, generalization ability, and privacy protection, proposing a deep improvement scheme FedG-ELITE based on federated graph learning. By introducing GraphSAGE, we transform the physical topology of the road network into vector representations in a high-dimensional feature space, thoroughly solving the curse of dimensionality of tabular reinforcement learning in large-scale road networks and endowing the routing protocol with inductive generalization capabilities for dynamic topologies. By introducing the federated learning framework, we reconstruct the functional positioning of the digital twin,

Table 1: Core Feature Comparison: ELITE vs. FedG-ELITE

Dimension	**ELITE (Baseline)**	**FedG-ELITE (Proposed)**
Core Algorithm	Tabular Q-Learning + Fuzzy Logic	GNN (GraphSAGE) + DQN + Federated Learning
State Space	Discrete Junction ID	Continuous Graph Embedding Vectors
Training Architecture	Centralized (DT Controller)	Distributed (Vehicle/RSU Clients + Server Aggregation)
Privacy Protection	Low (Requires uploading raw data)	High (Transmits only model parameters/gradients)
Network Adaptability	Weak (Topology change requires retraining)	Strong (Inductive learning, adapts to dynamic topology)
Bandwidth Consumption	High (Transmits sensing logs)	Low (Transmits model weights)

transforming it from a single simulation center to an aggregation center for network-wide intelligence, achieving collaborative intelligent evolution without violating user privacy. At the same time, the fuzzy logic fusion mechanism is retained and optimized to ensure refined support for multi-modal QoS services. Theoretical analysis and architectural design demonstrate that FedG-ELITE not only technologically aligns with the development trends of "Edge Intelligence" and "Privacy Computing" in vehicular networks but also possesses significant advantages in reducing bandwidth costs and improving decision robustness in actual deployment.

References

[1] L. Zhao, Z. Bi, A. Hawbani, K. Yu, Y. Zhang, and M. Guizani, "ELITE: An Intelligent Digital Twin-Based Hierarchical Routing Scheme for Softwarized Vehicular Networks," IEEE Transactions on Mobile Computing, vol. 22, no. 9, pp. 5231-5247, Sept. 2023.

[2] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Proc. NIPS, 2017.

[3] T. Li et al., "Federated optimization in heterogeneous networks," in Proc. MLSys, 2020.

[4] L. Zhao et al., "Routing schemes in software-defined vehicular networks: Design, open issues and challenges," IEEE Intell. Transp. Syst. Mag., vol. 13, no. 4, pp. 217-226, 2021.

[5] X. Wang et al., "Federated Learning with Graph-Based Aggregation for Traffic Forecasting," arXiv preprint arXiv:2507.09805, 2025.

[6] "Federated Learning for Privacy-Preserving Intrusion Detection in VANETs," ResearchGate, 2025.