Quantum-Driven Context-Aware Federated Learning in Heterogeneous Vehicular Metaverse Ecosystem

Bishmita Hazarika*, Keshav Singh*, Trung Q. Duong†, and Octavia A. Dobre†
*National Sun Yat-sen University, Taiwan (e-mail: keshav.singh@mail.nsysu.edu.tw)
†Memorial University of Newfoundland, Canada (e-mail: {tduong, o.dobre}@mun.ca)

***Abstract*—In the rapidly evolving domain of vehicular metaverse, this study introduces a cutting-edge quantum-based decentralized and heterogeneity-aware federated learning framework for vehicular metaverse named QV-MetaFL, which stands as a testament to the innovative fusion of quantum computing principles with federated learning (FL). This framework is ingeniously tailored to address the challenges in a vehicular metaverse, offering a cost-efficient and adaptive solution for the dynamic vehicular landscape. QV-MetaFL is strengthened by the quantum sequential-training-program (Q-STP) algorithm, a quantum-based sequential training program that transforms model training, reducing communication costs and adeptly managing vehicle states. Complementing this, the quantum vehicle-context-grouping (Q-VCG) mechanism groups vehicles based on contextual data similarity, effectively tackling the complexities of data heterogeneity. The synergy of Q-STP and Q-VCG culminates in the QV-MetaFL algorithm, a decentralized, efficient, and context-aware quantum federated learning (QFL) process that redefines learning dynamics in the vehicular metaverse. Additionally, our research introduces an innovative composite loss function that amalgamates classical loss metrics with quantum parameter regularization, deftly addressing quantum sensitivity to noise. The effectiveness of the QV-MetaFL framework is rigorously validated through comprehensive simulations, with its performance meticulously compared against various adaptations, showcasing its transformative capabilities within the vehicular metaverse ecosystem.**

I. Introduction

THE next-generation vehicular metaverse presents formidable challenges, including scalability, complexity, data diversity, precision in data management, and security. While classical federated learning (CFL) offers benefits in collaborative model training and data privacy, its suitability for addressing the dynamic and heterogeneous nature of the vehicular metaverse remains uncertain [1]. Despite insights gained from decentralized CFL applications in the Internet of Vehicles (IoV) [2] and industrial metaverse domains, challenges such as data inconsistency, model obsolescence, and scalability issues continue to raise questions about its efficacy within the broader vehicular metaverse context [3].

In this context, quantum federated learning (QFL) emerges as a promising solution to address CFL limitations [4]. Leveraging quantum neural networks (QNN), QFL offers advanced data processing, model precision, and enhanced security, well-suited for the dynamic vehicular metaverse. QFL adoption is essential for future-proofing systems in this rapidly advancing field, mitigating the risk of obsolescence. Yet, practical QFL application faces challenges such as heterogeneous data integration, increased computational demands, and quantum-specific issues [5]. On that note, the authors in [6] have proposed an efficient FL system for the industrial metaverse, addressing challenges like non-informally-identically-distributed (non-i.i.d.) data, learning forgetting, and limited bandwidth; however, its reliance on traditional computing methods limits its applicability in the quantum-centric, dynamic, and data-intensive vehicular metaverse. In [7], the authors propose a decentralized and secure QFL framework, leveraging blockchain to enhance transparency and robustness, but the potential limitation lies in its adaptability to the dynamic network and handling of heterogeneous data inherent. Overcoming these hurdles is pivotal for QFL's evolution into a key component of our future, enhancing the next-generation metaverse's computational capabilities. However, current research has yet to fully explore quantum techniques, particularly in managing diverse contextual data.

Addressing this gap, our study introduces a novel framework: a decentralized, context-aware, quantum-based federated learning model. This framework is designed with scalability in mind, making it suitable for large-scale deployment. It efficiently tackles data heterogeneity challenges while maintaining cost-effectiveness and adapting to the dynamic nature of the vehicular metaverse. The primary contributions of this research can be outlined as follows:

1) We introduce the quantum-based decentralized and heterogeneity-aware federated learning framework for vehicular metaverse (QV-MetaFL framework) which innovatively merges quantum computing principles with federated learning, a novel approach that harnesses the strengths of both domains to optimize learning processes in a distributed vehicular metaverse.
2) Next, we design the quantum sequential-training-program (Q-STP) algorithm, a quantum-based sequential training program that significantly reduces communication costs and ensures robust model training within our QV-MetaFL framework by dynamically managing the operational states of vehicles.
3) Further, we introduce quantum vehicle-context-grouping (Q-VCG), an innovative and scalable quantum-driven mechanism for vehicle grouping based on contextual data similarity to address the data heterogeneity issues in model training.
4) Subsequently, we integrate Q-STP and Q-VCG into the QV-MetaFL framework, leading to the creation of the QV-MetaFL algorithm. This facilitates a decentralized, efficient, and context-aware QFL process.
5) Additionally, our research includes a composite loss function that merges classical loss metrics (mean

square error and Huber loss) with quantum parameter regularization to mitigate the quantum sensitivity to noise.

6) Finally, we assess the effectiveness of our framework through simulations. The comparative performance analysis of the QV-MetaFL algorithm with its various adaptations offers insightful benchmarks and highlights its capabilities within the framework's ecosystem.

II. System Model

In this study, we explore a suburban vehicular metaverse environment comprising K vehicles, as depicted in Fig. 1, equipped with intelligent agents, each outfitted with quantum processors, real-world data collection sensors, sophisticated AI functionalities, and state-of-the-art connectivity modules. Each of these vehicles, or "clients" in the FL context, is engaged in an intricate interplay of both learning and sharing. They consistently enhance their localized QNN models and occasionally synchronize with a comprehensive global model hosted on an edge server. Multiple edge servers, serving as aggregators, enable independent and concurrent global model updates. Our QV-MetaFL framework operates over t rounds, each divided into r time slots. During these slots, vehicles refine models, apply the quantum parameter shift rule for gradient calculation, and communicate updates to the edge server. Communication decisions are optimized through mode indicators for adaptability and efficiency. The central server models are updated based on all vehicle inputs. Scheduled synchronizations, governed by R_{sync}, guarantee consistent alignment with the global model for all vehicles, irrespective of their individual communication frequencies.

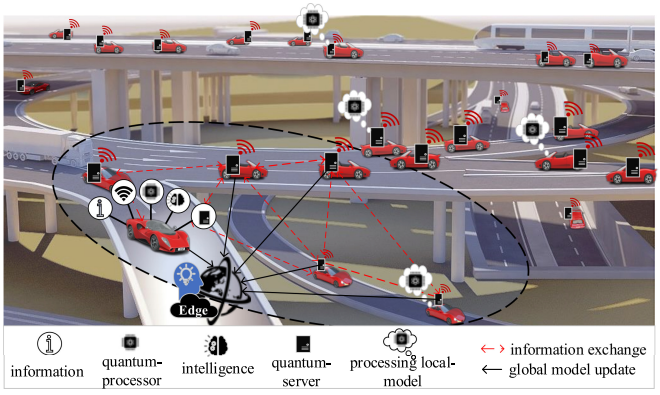

Fig. 1: An illustration of QV-MetaFL based vehicular metaverse framework.

III. Quantum Sequential-Training-Program

In the dynamic realm of vehicular metaverses intersecting with quantum environments, managing communication overhead is crucial. The complexity of quantum data and the ever-evolving vehicular metaverse pose scalability and robustness challenges. An optimized approach is vital to balance communication efficiency with model integrity. To address this, we propose two key operational modes for vehicles within the FL process.

1. Streaming Mode: An operational state in which a vehicle actively participates in quantum streaming for FL.

2. Calibration Mode: An operational state in which a vehicle focuses on stabilizing its local model using quantum techniques without actively participating in quantum streaming.

We formulate the optimization problem, aiming to minimize communication costs, as:

$$
\begin{aligned}
(\mathcal{P}_1): \quad & \min_{M_i, C_i, W_i^{(t)}, U_i^{(t)}} \quad \sum\nolimits_{i=1}^{N} CC_i \\
\text{s.t.} \; & (C.1) \quad M_i \in \{0,1\} \quad \forall i\,, \\
& (C.2) \quad CC_i \geq 0 \quad \forall i \\
& (C.3) \quad L_i(t) = U(L_i(t-1), D_i(t)) \text{ if } M_i = 1, \forall i\,, \\
& (C.4) \quad L_i(t) = S(L_i(t-1), D_i(t)) \text{ if } M_i = 0, \forall i\,, \\
& (C.5) \quad W_i^{(t)}, \text{ if } M_i = 1, \quad \forall i\,, \\
& (C.6) \quad U_i^{(t)}, \text{ if } M_i = 1 \text{ or } M_i = 0, \quad \forall i\,.
\end{aligned} \tag{1}
$$

Here, in constraint $C.1$, M_i serves as a binary indicator for the quantum mode:

$$
M_i = \begin{cases} 1 & \text{if } d_i > \mathbb{T} \text{ and } CC_i \leq \text{Limits}_i \\ 0 & \text{otherwise} \end{cases} \tag{2}
$$

Here, $d_i > \mathbb{T}$ ascertains if the vehicle i is sufficiently distant from the server, surpassing threshold $\mathbb{T}$, to engage the quantum streaming mode. Concurrently, $CC_i \leq \text{Limits}_i$ ensures that vehicle i's communication cost adheres to the limit, $\text{Limits}_i = m$ mbps per vehicle $\times$ vehicle to bandwidth ratio $\times$ OHF, where OHF stands for overhead factor. Both conditions must concur for M_i to be 1, indicating the vehicle's operation in the quantum streaming mode. Conversely, M_i defaults to 0, denoting calibration mode. Additionally, $C.2$ mandates non-negative communication costs. $C.3$ postulates that quantum updates through operation U are applied to the local model $L_i(t-1)$ of vehicle i at round t using data $D_i(t)$ solely when $M_i = 1$. Further, $C.4$ underscores that operation S is triggered when $M_i = 0$ (representing the calibration mode), $C.5$ and $C.6$ respectively elaborate on the nuances of quantum transmissions $W_i^{(t)}$ and quantum update exchanges $U_i^{(t)}$ contingent on the quantum mode.

To address the problem outlined by $\mathcal{P}_1$, it is crucial first to comprehend the communication cost within the framework. While there are a multitude of factors in practical scenarios that determine communication costs, we will center our attention on a select group of pivotal parameters for simplicity and computational efficiency: d_i: the distance between vehicle i and server, b: the available communication bandwidth, s_i: the size of the quantum update or data transmitted by vehicle i, and e: the transmission rate. The communication cost is:

$$
CC_i = \frac{s_i}{e} + \frac{s_i}{b} \cdot d_i\,, \tag{3}
$$

where $\frac{s_i}{e}$ and $\frac{s_i}{b} \cdot d_i$ represent the time it takes to transmit the quantum update locally, and over to the server.

To navigate the inherent challenges and elevated communication costs in QFL, we introduce the quantum-based sequential training program called Q-STP. Tailored for the vehicular metaverse, Q-STP is a dynamic technique that effectively shifts between quantum streaming and calibration modes, respecting the ever-evolving vehicular data landscape. It leverages quantum methodologies to achieve

optimal training, all the while slashing communication overhead. The Q-STP initiative commences with parameter and hyperparameter initialization, setting each vehicle's CC_i to zero. Each iterative round encompasses:

1) **Calculate CC_i:** For each vehicle i, it computes CC_i.
2) **Mode Switching:** From 2, if $M_i = 1$, it activates the quantum streaming mode by setting `STREAMING_MODE_ACTIVE` to 1.
3) **Mode Counter:** Continuously monitoring and switching modes might introduce its own overheads. Therefore we introduce $ModeCounter_i$ as a delay before transitioning from one mode to another for a specific vehicle i. This counter keeps track of the number of consecutive rounds a vehicle meets the conditions for its current mode. A mode switch for vehicle i will only occur once $ModeCounter_i$ reaches a pre-defined transition threshold, $\mathbb{H}$.
4) **QFL:** Depending on the active mode, the algorithm conducts QFL with a certain number of epochs. If $M_i = 1$, it sets `STREAMING_TRAINING` to true; otherwise, it sets `CALIBRATION_TRAINING` to true.

For a comprehensive understanding, the Q-STP is detailed in **Algorithm 1** where the epochs for streaming and calibration are denoted as S_Epochs and C_Epochs, respectively.

Algorithm 1 Q-STP for Communication Cost Minimization

Initialize parameters and hyperparameters.
Initialize $CC_i \leftarrow 0$ for each vehicle i.
Initialize $M_i \leftarrow 0$ for each vehicle i.
Initialize $ModeCounter_i \leftarrow 0$ for each vehicle i.
for each round **do**
  dynamics_metrics ← calculate_dynamics_metrics()
  for each vehicle i **do**
    $d_i \leftarrow dynamics\_metrics$[vehicle]['distance']
    $s_i \leftarrow dynamics\_metrics$[vehicle]['data_size']
    $CC_i \leftarrow$ Equation (3).
    if $dynamics\_metrics[i]$['metric'] $> \mathbb{T}$ **and** $CC_i \leq$ Limits$_i$ **then**
      if $M_i = 0$ **then**
        $ModeCounter_i \leftarrow 0$
      end if
      $M_i \leftarrow 1$
      $ModeCounter_i \leftarrow ModeCounter_i + 1$
    else
      if $M_i = 1$ **then**
        $ModeCounter_i \leftarrow 0$
      end if
      $M_i \leftarrow 0$
      $ModeCounter_i \leftarrow ModeCounter_i + 1$
    end if
    if $ModeCounter_i < \mathbb{H}$ **then**
      Continue
    end if
  end for
  for each vehicle i **do**
    if $M_i = 1$ **then**
      for each epoch in S_Epochs **do**
        STREAMING_TRAINING=TRUE.
      end for
    else
      for each epoch in C_Epoch **do**
        CALIBRATION_TRAINING=TRUE.
      end for
    end if
  end for
  if $dynamics\_change\_detected()$ **then**
    if STREAMING_MODE_ACTIVE=1 **then**
      STREAMING_MODE_ACTIVE ← 0
    else
      STREAMING_MODE_ACTIVE ← 1
    end if
  end if
end for

IV. Quantum Vehicle-Context-Grouping

Although the Q-STP process reduces communication costs, data heterogeneity in QFL presents a challenge that impedes its ultimate efficiency. Vehicles ideally need to maintain similar data distributions to achieve the best performance outcomes. Given the vast scope of the vehicular metaverse, a scalable and decentralized solution becomes imperative. With this in mind, we present an optimization problem, focusing on training vehicles based on data context similarity as

$$
\begin{aligned}
(\mathcal{P}_2): \quad & \max_{\Omega_{ij}} \sum_{i=1}^{V} \sum_{\substack{j=1 \\ j\neq i}}^{V} \texttt{similarity\_score}_{(i,j)} \cdot \Omega_{ij} \\
\text{s.t.} & (C.7) \quad \Omega_{ij} \in \{0,1\} \quad \forall i \neq j, \forall (i,j) \in \{1,\ldots,K\}, \\
& (C.8) \quad \Omega_{ij} = \Omega_{ji} \quad \forall (i,j) \in \{1,\ldots,K\}, \\
& (C.9) \sum_{\substack{j=1 \\ j\neq i}}^{K} \Omega_{ij} = 1, \forall i \in \{1,\ldots,K\}, \\
& (C.10) \quad \Omega_{ii} = 0, \forall i \in \{1,\ldots,K\}, \qquad (4)
\end{aligned}
$$

where the contextual data are based on the vectors-Geographical Location `(lat, long)`: (V_1, V_2), `(speed)`: V_3, X-coordinate: V_4, Y-coordinate: V_5, Bandwidth: V_6.

To navigate the diverse data landscape in the dynamic metaverse and to address $\mathcal{P}_2$, we introduce a quantum-driven vehicular context grouping mechanism, called Q-VCG which groups vehicles by their contextual data similarities. Each formed group then embarks on independent quantum training on distinct edge servers. By harnessing the power of data similarity, we optimize QFL and tackle the inherent challenges posed by data diversity. This strategy culminates in a decentralized, scalable training system. Remarkably, its design allows for the modulation of group sizes based on the count of vehicles, ensuring consistent adaptability to the ever-evolving vehicular metaverse. We determine the similarity of data between vehicles, i and j, using the cosine similarity metric among their datasets, similar to [8]. The magnitude is calculated as $||\mathbf{V}|| = \sqrt{V_1^2 + V_2^2 + V_3^2 + V_4^2 + V_5^2 + V_6^2}$. Next, we compute the normalization of each factor such that $\hat{\mathbf{V}}_n$ denotes the normalized n^{th} factor ($n = 6$). Thus, to compute the cosine similarity between vehicles i and j, we first calculate the dot product of each normalized vector, followed by their cumulative Euclidean norms $\hat{\mathbf{V}}_i$ and $\hat{\mathbf{V}}_i$. Finally, the cosine similarity score between vehicles i and j is: $\texttt{cos\_sim}_{(i,j)} = \frac{\cdot}{\|\hat{V}_i\| \cdot \|\hat{V}_j\|}$. This iterative process determines similarity for each vehicle pair, with a higher $\texttt{cos\_sim}_{(i,j)}$ score indicating greater contextual data similarity. Furthermore, Q-VCG adheres to the following conditions:

- The maximum number of groups, denoted as $\mathcal{G}$, does not exceed $\beta \times K$, where β represents the proportion of the total vehicles.
- A group must comprise at least $K_{\min}$ vehicles. If a vehicle

i is not part of a group with a minimum of $K_{\min}$ vehicles, it will not engage in the training.
- During a specific training round t, a vehicle, denoted as i, remains exclusive to one group.
- Vehicles i and j share a group only if their similarity score surpasses a predetermined threshold, `Sim_Thres`. This threshold is configured based on a specific percentile $\mathbb{P}$ of all the similarity values recorded in that round, ensuring adaptability to potential shifts in data distribution.

The comprehensive Q-VCG methodology is elaborated upon in **Algorithm 2**.

Algorithm 2 Q-VCG for Optimal Vehicular Grouping

```
Initialize parameters and hyperparameters.
Initialize G empty lists for grouped vehicles.
Initialize sim_values as an empty list.
for each round do
    for each vehicle i do
        for each vehicle j ≠ i do
            Calculate cos_sim_(i,j).
            Append cos_sim_(i,j) to sim_values.
        end for
    end for
    Sim_Thres ← percentile(sim_values, ℙ)
    for each vehicle i do
        if vehicle i is not already in a group then
            Create group G and add vehicle i to it.
            for each potential group member (vehicle j) do
                Calculate cos_sim_(i,j).
                if cos_sim_(i,j) > Sim_Thres then
                    Assign vehicle j to vehicle i's group.
                end if
            end for
        end if
    end for
end for
```

V. Decentralized QFL For Vehicular Metaverse

We leverage the Q-STP and Q-VCG algorithms and design a quantum-based decentralized and heterogeneity-aware federated learning framework for vehicular metaverse named QV-MetaFL framework. In this framework, the QV-MetaFL algorithm is designed to handle data heterogeneity in a cost-efficient way while maintaining a robust quantum environment.

In the QV-MetaFL framework, each vehicle i has a local model (L_i) initialized. This model is represented by the parameterized quantum circuit $L_i(\boldsymbol{\theta}_i) = U_{\text{QNN}}(\boldsymbol{\theta}_i)$ in the context of QNN, where θ_i represents the set of trainable parameters for vehicle i and U_{QNN} denotes the parameterized quantum unitary (circuit) for the QNN of vehicle i. Next, updating the local model involves quantum circuits with parameter shifts for gradient calculations given as

$$\Delta\boldsymbol{\theta}_i = -\eta\nabla\mathcal{L}_{\text{local}}(L_i)\,, \quad \boldsymbol{\theta}_{i\text{new}} = \boldsymbol{\theta}_i + \Delta\boldsymbol{\theta}_i\,, \tag{5}$$

where $\nabla\mathcal{L}_{\text{local}}(L_i)$ is the gradient of the local loss with respect to the parameters of the QNN, η is the learning rate and $\boldsymbol{\theta}_{i\text{new}}$ are the updated parameters after a given update step. Next, we derive the local gradients by applying (9), which is derived using *Theorem* 1 provided below:

Theorem 1 (Quantum Parameter Shift Rule): Given a quantum circuit with a loss function $\mathcal{L}_{\text{local}}(\theta)$, the gradient with respect to its parameter θ_{in} can be approximated as:

$$\frac{\partial\mathcal{L}_{\text{local}}}{\partial\theta_{in}} \approx \frac{\mathcal{L}_{\text{local}}(\theta_{in}+\frac{\pi}{2}) - \mathcal{L}_{\text{local}}(\theta_{in}-\frac{\pi}{2})}{2}. \tag{6}$$

Proof: Starting with the general concept of a derivative:

$$\lim_{h\to 0}\frac{f(x+h)-f(x)}{h} = f'(x). \tag{7}$$

From the central finite difference approximation:

$$\frac{\partial f(x)}{\partial x} \approx \frac{f(x+h)-f(x-h)}{2h}. \tag{8}$$

By applying (8) with $h = \frac{\pi}{2}$, specifically valid for parameterized quantum circuits due to their unitary nature [9]:

$$\frac{\partial\mathcal{L}_{\text{local}}}{\partial\theta_{in}} \approx \frac{\mathcal{L}_{\text{local}}(\theta_{in}+\frac{\pi}{2}) - \mathcal{L}_{\text{local}}(\theta_{in}-\frac{\pi}{2})}{2}. \tag{9}$$

■

Next, the global model is aggregated using the gradient:

$$\Delta\boldsymbol{\theta}_{G_g} = -\eta\nabla\mathcal{L}_{\text{global}}(M_{G_g})\,, \tag{10}$$

$$\boldsymbol{\theta}_{G_g\text{new}} = \boldsymbol{\theta}_{G_g} + \Delta\boldsymbol{\theta}_{G_g}\,, \tag{11}$$

where $\boldsymbol{\theta}_{G_g}$ is the set of trainable parameters for the global model of the group G_g. The step r mod $R_{sync} = 0$ ensures that vehicles that do not send updates in every time slot can still sync with the global model periodically, where R_{sync} is the synchronization time. Consequently, to counteract the noise sensitivity inherent in quantum frameworks, we utilize a composite loss function, blending mean square error (MSE) for precision and Huber loss for outlier resilience. This approach balances prediction accuracy with robustness to anomalies. Thus, the loss function is formulated as:

$$\mathcal{L}_{\text{total}}(y,\hat{y}) = \gamma_1\cdot\text{MSE}(y,\hat{y}) + \gamma_2\cdot\text{Huber}(y,\hat{y}) + \gamma_3\cdot\mathcal{R}(\boldsymbol{\theta})\,, \tag{12}$$

where y is the actual output, $\hat{y}$ is the predicted output, $\{\gamma_1,\gamma_2,\gamma_3\}$ are the hyperparameters that balance the three components of the loss, and $\mathcal{R}(\boldsymbol{\theta})$ is the regularization term (using L2 regularization) penalizing large values of the parameters to handle data heterogeneity.

In **Algorithm 3**, the QV-MetaFL algorithm determines convergence dynamically through the reduction rate of the loss value. Let $\mathcal{L}_p$ and $\mathcal{L}_c$ represent the previous and current loss values, respectively. The percentage decrease, $\Delta\mathcal{L}$, is defined as: $\Delta\mathcal{L} = \frac{\mathcal{L}_p-\mathcal{L}_c}{\mathcal{L}_p}$. Convergence is deemed achieved when $\Delta\mathcal{L}$ falls below a set threshold $\mathcal{W}$ for Y consecutive rounds, employing a counter mechanism for tracking.

VI. Experiment

This section outlines our experimental setup.

A. Dataset Preparation

To prepare data for quantum computation, classical dataset representations are first converted to a quantum-encoded format. Utilizing the MNIST dataset, we extract 100 images of digits '**0**', '**1**', and '**2**' through the Tensorflow Python library. Given our emphasis on a vehicular context, we derive vehicular attributes from these MNIST images as follows:

Algorithm 3 QV-MetaFL framework with Q-STP and Q-VCG

Initialize global parameters and hyperparameters.
Initialize $\mathcal{G}$ empty lists for grouped vehicles.
Initialize global models M_G for each group.
Initialize local models L_i for each vehicle i.
Initialize CC_i and mode indicators M_i for each vehicle i.
for each round **do**
  Calculate vehicle dynamics using **Q-STP**.
  Form vehicle groups using **Q-VCG**.
  for each group G_g **do**
    Initialize the edge server for group G_g.
    Sync the global model M_{G_g} with the edge server.
    for each time slot r **do**
      for each vehicle i in group G_g **do**
        Update the local model L_i.
        Calculate local gradients.
        if $M_i = 1$ **then**
          Send gradients to the edge server.
          Update global model M_{G_g} using received gradients.
          Send model update to the vehicle i.
        end if
      end for
      if r mod $R_{sync} = 0$ **then**
        for each vehicle i in group G_g **do**
          if $M_i = 0$ **then**
            Send gradients to the edge server.
            Update M_{G_g} using received gradients.
            Send model update to all vehicles in the group.
          end if
        end for
      end if
      function CONVERGENCE(G_g, $\mathcal{L}_p$, counter)
        Calculate $\mathcal{L}_c$ for group G_g.
        $\Delta\mathcal{L} = (\mathcal{L}_p - \mathcal{L}_c)/\mathcal{L}_p$
        if $\Delta\mathcal{L} < \mathcal{W}$ **then**
          counter = counter + 1
          if counter $\geq$ Y **then return** True
          elsereturn False
          end if
        else
          counter = 0 **return** False
        end if
      end function
    end for
  end for
end for

1) Speed: Determined by the image's average pixel intensity: $\frac{\sum_{i=1}^{28}\sum_{j=1}^{28}\text{PixelValue}(i,j)}{28\times 28}$.
2) Coordinate: (X, Y) coordinates are obtained from the image digit's center of the mass. Here, $X = \frac{\sum_{i=1}^{28}\sum_{j=1}^{28} i\times\text{PixelValue}(i,j)}{\text{TotalPixelValue}}$ and $Y = \frac{\sum_{i=1}^{28}\sum_{j=1}^{28} j\times\text{PixelValue}(i,j)}{\text{TotalPixelValue}}$.
3) Distance: Computed as the distance between the image's center and the digit's center of mass, given by $\sqrt{(X-14)^2+(Y-14)^2}$.
4) Available Bandwidth: This is the count of pixels exceeding the threshold of 128. It is calculated as $\sum_{i=1}^{28}\sum_{j=1}^{28}\Theta(\text{PixelValue}(i,j) - 128)$, using the Heaviside step function, $\Theta()$.

With vehicular features sourced from MNIST images, we then translate this data into quantum states using the Cirq and TensorFlow Quantum (TFQ) libraries [10]. Each vehicular attribute undergoes encoding into a quantum state via a sequence of Ry rotation gates. The rotation angle, θ, is set relative to the attribute value, ensuring that the quantum format represents the vehicular attribute accurately. This is expressed as $|\psi\rangle = R_y(\theta)|0\rangle$, with $\theta = \kappa \times$ FeatureValue and κ as a normalization constant. The quantum vehicular dataset is then partitioned into 80% for training and 20% for testing.

Param.	Value	Param.	Value	Param.	Value
$\mathbb{T}$	0.75 km	t	50	γ_1	0.5
$\mathbb{H}$	5	r	20	γ_1	0.2
S_Epochs	5	R_{sync}	10	γ_3	0.05
C_Epoch	2	area	1km^2	$\mathcal{W}$	0.1
β	0.2	Y	5	η	0.01
$K_{\min}$	5	κ	$[0,1]$	$\mathbb{P}$	80
K	100	m	0.5	OHF	1.2

TABLE I: Simulation Parameters

B. Experimental Setup

We set up a quantum environment using Quantum Toolbox [11], TFQ, and an FL setup using TensorFlow Federated (TFF) in Python, utilizing the GPU runtime on Google Colab Pro [5], [7]. Quantum computations involve qubits, which can computationally challenge classical systems due to their complexity. To ensure the Colab runtime manages our computations, we use a modest dataset of 100 vehicles with 6 features. This dataset gets encoded into 100 quantum circuits, each operating on 6 qubits. To simplify computations and allow processing without a dedicated quantum processor, we transform the issue into a classification task, assessing the efficiency of the proposed QV-MetaFL framework. We aim to predict a vehicle's '*speed*' using quantum vehicular data. Since we have symbolically tied speed to an average pixel intensity; we classify vehicles as '*Fast*' (brightness above the threshold of 128) or '*Slow*' (below the threshold of 128). Given that our experiments occur in a simulated environment on a conventional computer, and the computational complexity can rise exponentially, we opt to train a compact QNN. This QNN has an input layer of 6 qubits (matching our data), a first hidden layer of 3 qubits, a second hidden layer of 3 qubits, and an output layer of 2 qubits, aligning with our classification model. The other parameters used in this study are given in **Table I**.

C. Simulation Results

i. **Training Accuracy:** Fig. 2 compares the training accuracy of the QV-MetaFL framework over various rounds. The graph shows a clear advantage of the complete QV-MetaFL over its version without the Q-STP and Q-VCG components, emphasizing the importance of Q-STP and Q-VCG in improving model learning and convergence. Examining the loss functions, the main loss function of QV-MetaFL combines MSE and Huber losses with a regularization component. The graph indicates that omitting either MSE or Huber affects the model differently, with the absence of the latter resulting in slightly better performance than without the former. This suggests Huber loss's potential superiority in managing outliers. The high accuracy levels shown are typical for training data, which naturally tends to show positive results.

ii. **Testing Accuracy:** Fig. 3 illustrates the testing accuracy for the QV-MetaFL framework and its various adaptations over different rounds. The graph distinctly highlights the

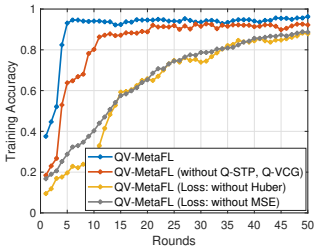

Fig. 2: Training accuracy vs. Rounds.

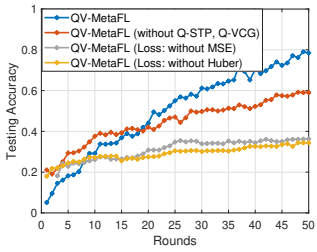

Fig. 3: Testing accuracy vs. Rounds

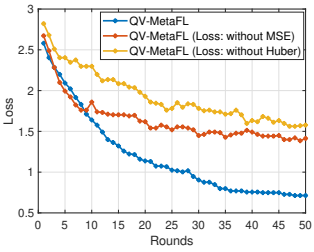

Fig. 4: Loss vs. Rounds.

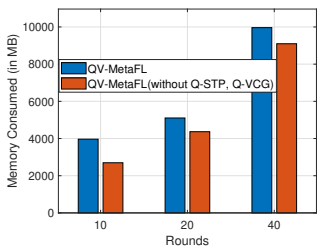

Fig. 5: Memory consumption vs. Rounds.

superior performance of the complete QV-MetaFL model compared to its versions missing the Q-STP and Q-VCG components, underscoring the significance of Q-STP and Q-VCG in boosting the model's testing accuracy. The graph also reveals the effects of omitting either the MSE or Huber components. Notably, the QV-MetaFL without the Huber loss marginally outperforms the one lacking MSE, indicating that the Huber loss might be better equipped to deal with outliers and testing data inconsistencies.

iii. **Loss:** Fig. 4 displays the loss metrics over rounds for the QV-MetaFL model and its modified versions. The pronounced decrease in loss for the main QV-MetaFL model highlights its adeptness in learning, harnessing the full spectrum of its components. Versions omitting the MSE or Huber loss components exhibit distinct trends. The model without MSE tends to stabilize faster, pointing to the MSE's pivotal role in sharpening the model's accuracy. On the other hand, the version lacking the Huber loss shows an inconsistent loss curve, highlighting its importance in addressing data irregularities or outliers. This visual data emphasizes the value of robust loss functions in ensuring peak model outcomes.

It is essential to mention that the presented testing accuracy and loss metrics are relatively conservative. Such results stem from our experimental constraints, where a confined dataset was employed alongside a limited number of rounds. These constraints arose since our conventional processors face challenges in managing the extensive computations tied to quantum procedures. Despite these limitations, the main goal was to gauge the comparative efficiency of our model. The primary QV-MetaFL model's standout performance, even within these confines, affirms its promise and establishes its proficiency in quantum-centric learning scenarios.

iv. **Memory Usage:** Fig. 5 depicts the memory utilization of the QV-MetaFL over different rounds. The complete QV-MetaFL, inclusive of Q-STP and Q-VCG, uses more memory than its simplified version. While the memory difference is not very high, it is evident. The rise in memory usage is due to the complexities associated with quantum processing, especially with the Q-STP and Q-VCG features since additional quantum operations require substantial resources. When comparing this with Fig. 2 and Fig. 3, the inclusion of Q-STP and Q-VCG significantly boosts model performance, highlighting a trade-off between resource use and efficiency.

To address these memory concerns, our future efforts will refine the framework, aiming to implement a quantum memory compression method to optimize QML resource usage.

VII. Conclusion

In this study, we introduced the QV-MetaFL framework, a groundbreaking integration of quantum computing and federated learning tailored for the vehicular metaverse. The framework, strengthened by the Q-STP and Q-VCG mechanisms, innovates a decentralized, efficient QFL approach, streamlining model training, optimizing communication, managing vehicle operations, and intelligently addressing data heterogeneity. Through comprehensive simulations, the QV-MetaFL framework demonstrated its superiority by outperforming its various adaptations in terms of accuracy and efficiency. Although our experiments were constrained by the limitations of classical simulations, the results provide insightful benchmarks and underscore the framework's potential within the vehicular metaverse ecosystem. Looking ahead, future efforts will focus on refining the framework, particularly in optimizing quantum memory usage, thus reinforcing its practicality and efficiency for real-world applications in the vehicular metaverse.

References

[1] Y. Ren, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Quantum collective learning and many-to-many matching game in the metaverse for connected and autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 71, no. 11, pp. 12 128–12 139, Jul 2022.

[2] B. Hazarika and K. Singh, "AFL-DMAAC: Integrated resource management and cooperative caching for URLLC-IoV networks," *IEEE Trans. Intell. Veh.*, 2023.

[3] B. Hazarika, K. Singh, C.-P. Li, A. Schmeink, and K. F. Tsang, "RADiT: Resource allocation in digital twin-driven UAV-aided internet of vehicle networks," *IEEE J. Sel. Areas Commun.*, 2023.

[4] Y. Cao *et al.*, "Hybrid trusted/untrusted relay-based quantum key distribution over optical backbone networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 9, pp. 2701–2718, Mar 2021.

[5] R. Huang *et al.*, "Quantum federated learning with decentralized data," *IEEE J. Sel. Top. Quantum Electron.*, vol. 28, pp. 1–10, Apr 2022.

[6] S. Zeng, Z. Li, H. Yu, Z. Zhang, L. Luo, B. Li, and D. Niyato, "Hfedms: Heterogeneous federated learning with memorable data semantics in industrial metaverse," *IEEE Trans. Cloud Comput.*, 2023.

[7] D. Gurung, S. R. Pokhrel, and G. Li, "Decentralized quantum federated learning for metaverse: Analysis, design and implementation," *arXiv preprint arXiv:2306.11297*, 2023.

[8] L. Zheng, K. Jia, T. Bi, Y. Fang, and Z. Yang, "Cosine similarity based line protection for large-scale wind farms," *IEEE Trans. Ind. Electron.*, vol. 68, no. 7, pp. 5990–5999, Apr 2020.

[9] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, "General parameter-shift rules for quantum gradients," *Quantum*, vol. 6, p. 677, Mar 2022.

[10] M. Broughton *et al.*, "Tensorflow quantum: A software framework for quantum machine learning," *arXiv preprint arXiv:2003.02989*, 2020.

[11] J. R. Johansson, P. D. Nation, and F. Nori, "QuTiP: An open-source Python framework for the dynamics of open quantum systems," *Comput. Phys. Commun.*, vol. 183, no. 8, pp. 1760–1772, Aug 2012.