Lagrange Coded Federated Learning (L-CoFL) Model for Internet of Vehicles

Weiquan Ni[1,5], Shaoliang Zhu[1], Md Monjurul Karim[2], Alia Asheralieva[1,2], Jiawen Kang[3], Zehui Xiong[4], and Carsten Maple[5]

[1]*Department of Computer Science and Engineering*, *Southern University of Science and Technology*, Shenzhen, China,
[2]*Research Institute of Trustworthy Autonomous Systems*, *Southern University of Science and Technology*, Shenzhen, China,
[3]*School of Automation*, *Guangdong University of Technology*, Guangzhou, China,
[4]*Pillar of Information Systems Technology and Design*, *Singapore University of Technology and Design*, Singapore,
[5]*WMG*, *University of Warwick*, Coventry, UK,
e-mail:{Weiquan.Ni@warwick.ac.uk, 12132381@mail.sustech.edu.cn, karim@mail.sustech.edu.cn, aasheralieva@gmail.com, kjwx886@163.com, zehui_xiong@sutd.edu.sg, CM@warwick.ac.uk}

***Abstract*—In Internet-of-Vehicles (IoV), smart vehicles can efficiently process various sensing data through federated learning (FL) - a privacy-preserving distributed machine learning (ML) approach that allows collaborative development of the shared ML model without any data exchange. However, traditional FL approaches suffer from poor security against the system noise, e.g., due to low-quality training data, wireless channel errors, and malicious vehicles generating erroneous results, which affects the accuracy of the developed ML model. To address this problem, we propose a novel FL model based on the concept of Lagrange coded computing (LCC) - a coded distributed computing (CDC) scheme that enables enhancing the system security. In particular, we design the first L-CoFL (Lagrange coded FL) model to improve the accuracy of FL computations in the presence of low-quality training data and wireless channel errors, and guarantee the system security against malicious vehicles. We apply the proposed L-CoFL model to predict the traffic slowness in IoV and verify the superior performance of our model through extensive simulations.**

***Index Terms*—Coded distributed computing (CDC), data privacy, federated learning, Internet of Vehicles (IoV), machine learning, security**

I. Introduction

In the IoV, smart vehicles are empowered to collect environmental data through heterogeneous sensors, which can be data provision to perform ML tasks and produce accurate ML models for traffic prediction, road condition analysis and route planing [1]. However, traditional ML approaches require data exchange between the vehicles and fusion centres, e.g., cloud processors and edge servers installed at the base stations (BSs) or roadside units (RSUs), which train the ML model based on data sent from the vehicles [2]. Clearly, such approaches are vulnerable to privacy leaks and security breaches, since the data can be intercepted and/or altered during the process of data exchange [3].

To tackle this issue, the vehicles can use FL [4] that allows training the shared ML model without revealing private data. In particular, with FL, each vehicle only exchanges non-private ML results generated through local training of its own ML model and then, update the shared ML model accordingly until achieving the targeted accuracy [5]. Unfortunately, existing FL approaches are greatly impacted by the system noise, e.g., due to low-quality training data, wireless channel errors, and malicious vehicles generating erroneous results, that can significantly reduce the accuracy of the shared ML model. To address this problem, several approaches have been proposed by integrating FL with supplementary technologies, including intrusion detection methods to remove malicious vehicles, e.g., [6], and blockchain-based solutions to discard erroneous ML results, e.g., [7]. Nonetheless, these methods require verification of ML results produced by vehicles, which is compute-intensive and yields high delay/signalling/energy overheads.

Therefore, unlike existing methods to deal with the system noise, in this paper, we propose another approach. In this approach, we deploy an LCC model - a lightweight CDC model that can provide security against the system noise by introducing computational redundancy to the data processed by the vehicles in the FL-enabled IoV system. The concept of CDC origins from distributed computing systems, in which redundant information is added/encoded into data computed by each distributed device, resulting in that the server can correct erroneous results produced by some devices based on specific decoders and obtain the final computing result [8]. Hence, the key idea of applying LCC in the FL-enabled IoV system is as follows. It uses Lagrange polynomial interpolation to encode data processed by the vehicles, which enables to recover the final ML result (at the fusion centre) by the Reed-Solomon decoder even though some of the returned results are erroneous due to the system noise.

However, the decoding of ML results cannot be performed in the existing LCC model with the Reed-Solomon decoder because it can only be applied to computing tasks denoted as polynomial functions. Hence, to apply the LCC model, the ML models that are typically based on the non-polynomial (i.e., artificial neural network [ANN]) functions must be transformed to polynomial functions in advance. In addition, compared to distributed computing, the process of data encoding in FL systems cannot be conducted in the fusion centre because the training data are distributed on the vehicles. To solve these two problems, we propose the first Lagrange coded FL model

called L-CoFL to support the integration of LCC with FL for secure FL-enabled IoV systems. In the L-CoFL model, vehicles can encode their training data locally based on the encoding elements given by the fusion centre. More importantly, the ML models of IoV applications can be approximated by the polynomial functions with high accuracy and, thus, can apply with the LCC model.

The main contributions of this paper are as follows.

- We adopt the concept of LCC to design the first Lagrange coded FL model for secure FL-enabled IoV systems, aimed at improving security against the system noise. To facilitate the integration of LCC in the FL-enabled IoV system, the proposed L-CoFL model supports the encoding of training datasets processed by the vehicles.
- To apply the developed L-CoFL model for practical FL applications based on the non-polynomial ANN functions, we propose the accurate low-complexity polynomial approximation.
- We apply the proposed L-CoFL model to practical vehicular applications for traffic slowness prediction based on the ANN functions. We show (through extensive simulations) that our scheme can achieve an ideal approximation accuracy and effectively protect the system from the malicious vehicles.

The rest of the paper is as follows. In Section II, we review related works. In Section III, we present model of the considered FL-enabled IoV system. In Section IV, we derive the polynomial approximation and design the L-CoFL model. In Section V, we develop the implementation of the L-CoFL model for a vehicular application and evaluate its performance in Section VI. Finally, we conclude the paper in Section VII.

II. Related Work

FL allows vehicles training their own ML models with their sensing dataset, which their produced ML results are then aggregated and used to update the shared ML model at the fusion centre. As such, FL has several advantages such as protection of data privacy and reduction of data transmission [9]–[11]. Moreover, the existing FL-enabled IoV applications [12]–[14] mainly focused on improving the learning accuracy and reducing the traffic congestion and communication overhead. For example, [12] proposed a multi-layer FL model in which each RSU can select and aggregate ML results produced by local vehicles, and the selected ML results by the RSUs are further aggregated at cloud for updating the shared ML model. The contexts of the vehicles and RSUs, e.g. location, are considered during the selection of ML results. Though the accuracy of shared ML model was improved, privacy-preservation of the ML results was ignored. In [13], a decentralized Byzantine-Fault-Tolerance (BFT) solution was presented for FL-enabled IoV applications, allowing each vehicle to exchange and verify their ML results. Besides, peer-to-peer (P2P) FL was combined with BFT to extend the HyRand protocol [15] to perform publicly verifiable secret sharing among autonomous vehicles (AV). Although the approach in [13] showed no side-effect on the accuracy of shared ML model, the BFT-based consensus methods are time-consuming that require multiple times of communication between the vehicles. Besides, the impact of erroneous ML results on the shared ML model was not discussed.

In addition, the above FL-enabled IoV applications did not consider protection of the system against the system noise. To solve the problem, several works (e.g., [16]–[20]) integrated blockchain with FL to perform secure aggregation. However, these blockchain-based approaches have the following disadvantage. That is, the ML results produced by each vehicle are verified by the set of validators in blockchains where the process of consensus not only consumes enormous computing resources, but also yields a large duration of delay. Hence, such solutions are inapplicable for many delay-sensitive and compute-intensive IoV applications.

To the best of our knowledge, this work is the first attempt to adopt LCC to develop a secure and light model that can protect the FL-enabled IoV systems against the system noise. In detail, only small amount of computing resources is required for the approximation of ML models. In addition, after training their own model, the vehicles only submit output of their updated models (i.e., estimation result) rather than large-scale of model parameter/gradient update to the fusion centre. As such, the data transmission of the ML results aggregation is significantly reduced. Moreover, the privacy of vehicles can be protected effectively in our L-CoFL model with only the estimation results exchanged. In contrast, the vehicles may suffer from the privacy leakage during the exchange of model parameters in traditional FL approaches [21].

III. System Model

A. FL-enabled IoV System

We consider the FL-enabled IoV system with a set of $\mathcal{V}$ vehicles denoted as $\mathbf{V} = \{v_i\}_{i=1}^{\mathcal{V}}$ and a fusion centre, e.g., edge server installed at a BS or RSU near the vehicles. Each vehicle v_i has a local dataset $\mathcal{D}_i = \{(\mathbf{x}_k, \mathbf{y}_k)\}_{k=1}^{\mathcal{K}_i}$, where $(\mathbf{x}_k, \mathbf{y}_k)$ is the k^{th} tuple of feature and label vectors, and $\mathcal{K}_i$ is the size of data owned by vehicle v_i. The fusion centre is responsible for initializing and updating the shared ML model based on the ML results (model parameter or gradient update) of the vehicles, which train their own ML models based on the local datasets.

The system operates as follows. At the r^{th} round of global training, the fusion centre transmits the shared ML model to each vehicle, which $\mathbf{w}^{(r-1)}$ is parameter of the shared ML model updated at the last round. Furthermore, each vehicle v_i inputs the local dataset $\mathcal{D}_i$ into the received shared ML model and produce the local ML result $\mathbf{w}_i^r$ through the iterative gradient descent [22]. We denote $\mathbf{w}_i^{r,t}$ as the local ML result of vehicle v_i at the t^{th} round of local training, and $\mathbf{w}_i^{r,0}$ is the received shared ML model from the fusion centre in current global training. The local training of vehicle is represented as

$$\mathbf{w}_i^{r,t} = \mathbf{w}_i^{r,(t-1)} + \rho \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}; \mathcal{D}_i), \tag{1}$$

where ρ is the learning rate of gradient descent algorithm. $\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}; \mathcal{D}_i)$ is the gradient update based on the local ML re-

sult and dataset, which $\mathcal{L}(\mathbf{w};\mathcal{D}_i)$ is the adopted loss function, e.g., squared-SVM, linear regression, k-means and so on [23]. Furthermore, after $\mathcal{L}(\mathbf{w};\mathcal{D}_i)$ meets the predefined accuracy, the vehicles send back the local ML results to the fusion centre for updating the shared ML model. The update of shared ML model is denoted as

$$\mathbf{w}^r = \frac{1}{\mathcal{V}} \sum_{i=1}^{\mathcal{V}} \mathbf{w}_i^{r,t}, \tag{2}$$

where the local ML results are averaged to produce the shared ML model. Finally, the fusion centre delivers the updated shared ML model to each vehicle again to start the next round of global training until the shared ML model achieves the targeted accuracy.

B. Lagrange Coded Computing Model

As shown in Fig. 1, the system noises, e.g., due to low-quality training data, dishonest computation and wireless channel errors, exist in some vehicles that produce the low-accuracy and erroneous ML results. Based on (2), traditional FL approaches have to average the ML results of the vehicles to update the shared ML model. However, to compute (2), these approaches generally assume that all vehicles are honest and produce the accurate ML results that satisfy the requirements of accuracy. Otherwise, the low-accuracy and erroneous ML results will significantly affect the convergence speed and accuracy of the shared ML model. Unfortunately, the existing methods to solve the above problems (based on blockchains or intrusion detections) consume enormous computing resources and delays to verify accuracy of the ML results and reach consensus in blockchains.

In contrast, CDC has been proposed in large-scale distributed computing to tackle fundamental bottlenecks such as communication efficiency, straggler, security and privacy problems [22]. Among many proposed CDC methods, LCC is the most promising to tackle the problem of untrustable computing forged by malicious devices. The reason is that LCC is easy to implement and can be applied to a great variety of computing tasks, which should be polynomial functions [24]. Here, we uniformly describe the devices in distributed computing as the vehicles in IoV. The implementation of LCC in the distributed computing can be divided into three steps: **Encoding**, **Task Computing** and **Decoding**.

Encoding: Without loss of generality, we define a computing task as a polynomial function $\mathcal{C}$ of degree $deg(\mathcal{C})$, which is computed over an input dataset $\mathbf{X}$ by a set of vehicles in $\mathbf{V}$. In distributed computing systems, the data is owned and processed by the fusion centre before the vehicles compute the task. In order to deliver the input data to the vehicles, the fusion centre partitions the dataset $\mathbf{X}$ into $\mathcal{M}$ batches, that is, $\mathbf{X} = \{\mathbf{X}_m\}_{m=1}^{\mathcal{M}}$. Then, the final result of the computing task is described as $Y = \{y_m = \mathcal{C}(\mathbf{X}_m)\}_{m=1}^{\mathcal{M}}$. However, considering the existence of malicious vehicles, these data are not sent to the vehicles directly. Instead, the data are encoded based on the Lagrange interpolation formula in order to produce data redundancy in each dataset sent to the vehicles. By doing so, the fusion centre can find the accurate final result based on the computing results of the vehicles, even if some of which are erroneous or missed.

To encode the data, the fusion centre initially selects $\mathcal{M}$ distinct elements, i.e., $\{\ell_m\}_{m=1}^{\mathcal{M}}$, for each batch of data, followed by generating a Lagrange interpolation formula $\mathcal{H}$ of degree $deg(\mathcal{H}) \leq (\mathcal{M}-1)$ as follows:

$$\mathcal{H}(z) = \sum_{m=1}^{\mathcal{M}} \mathbf{X}_m \sum_{n=1, n\neq m}^{\mathcal{M}} \frac{z-\ell_n}{\ell_m - \ell_n}. \tag{3}$$

In (3), we can find that $\mathcal{H}(\ell_m) = \mathbf{X}_m$. Furthermore, the fusion centre selects $\mathcal{V}$ distinct elements, i.e., $\{\rho_i\}_{i=1}^{\mathcal{V}}$, for each vehicle. Note that all the selected elements should be different, that is, $\{\ell_m\}_{m=1}^{\mathcal{M}} \bigcap \{\rho_i\}_{i=1}^{\mathcal{V}} = \varnothing$. Finally, the fusion centre encodes the data for each vehicle as follows:

$$\widetilde{\mathbf{X}}_i = \mathcal{H}(\rho_i), i \in \mathbf{V}. \tag{4}$$

Task Computing: After encoding the data, the fusion centre sends the encoded input data $\widetilde{\mathbf{X}}_i$ and computing task $\mathcal{C}$ to each vehicle. Then, the vehicles compute the task on the data as follows:

$$\widetilde{\mathbf{Y}}_i = \mathcal{C}(\widetilde{\mathbf{X}}_i), i \in \mathbf{V}. \tag{5}$$

Note that the vehicles cannot decode the encoded data into the raw data because they have no idea on the elements $\{\ell_m\}_{m=1}^{\mathcal{M}}$ and $\{\rho_i\}_{i=1}^{\mathcal{V}}$. The vehicles return the local computing results $\widetilde{\mathbf{Y}}_i$ after they finish the task computing.

Decoding: To obtain the final result, the fusion centre has to decode the computing results of the vehicles. Here, we consider two different assumptions regarding the vehicles. The first assumption is that all vehicles are honest and produce the accurate computing results. As such, even though some of computing results are straggling, the final result can be easily decoded by the Lagrange interpolation method [25]. Another assumption is that some of the vehicles are dishonest and produce the erroneous computing results. In this case, the fusion centre has to exploit Reed-Solomon decoder to compute and reconstruct the polynomial function $\mathcal{C}(\mathcal{H}(z))$ [24]. To reconstruct the polynomial function of degree $deg(\mathcal{C}(\mathcal{H}(z))) = deg(\mathcal{C})deg(\mathcal{H}(z)) \leq (\mathcal{M}-1)deg(\mathcal{C})$, it requires evaluations on $(\mathcal{M}-1)deg(\mathcal{C})+1$ of different elements. Moreover, we assume that the number of computing results from the malicious vehicles is $\mathcal{E}$. For each malicious result, Reed-Solomon decoder requires evaluations at two additional elements. Finally, if the following requirement is satisfied, the fusion centre can decode and obtain the accurate final result:

$$(\mathcal{M}-1)deg(\mathcal{C}) + 2\mathcal{E} + 1 \leq \mathcal{V}. \tag{6}$$

The system is called $\mathcal{E}$-secure if the fusion centre can decode the final result successfully with the existence of $\mathcal{E}$ malicious vehicles. Based on the Reed-Solomon decoder, the fusion centre can find all coefficients of the polynomial function of

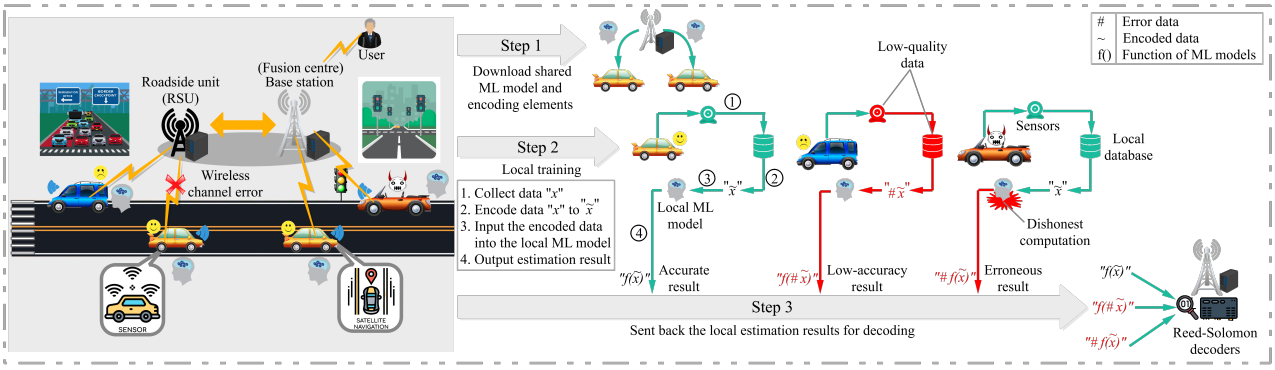

Fig. 1. Lagrange coded federated learning (L-CoFL) model for IoV.

$\mathcal{C}(\mathcal{H}(z))$, from which the accurate final result can be found according to:

$$\mathbf{Y} = \{\mathbf{Y}_i\}_{i=1}^{\mathcal{M}} = \{\mathcal{C}(\mathbf{X}_i)\}_{i=1}^{\mathcal{M}} = \{\mathcal{C}(\mathcal{H}(\ell_i))\}_{i=1}^{\mathcal{M}}. \quad (7)$$

IV. L-CoFL: Lagrange Coded FL Model for FL-enabled IoV Systems

To enhance the security of FL-enabled IoV systems, we propose the novel FL model by applying the concept of LCC. Moreover, the proposed L-CoFL model is designed to improve the accuracy of shared ML models in the presence of low-quality trained data, wireless channel errors, and malicious vehicles producing erroneous results. Different with distributed computing, the training dataset in FL are located at the vehicles and, thus, cannot be encoded at the fusion centre. In addition, the shared ML models in FL are generally non-polynomial functions and, therefore, cannot be applied with LCC and Reed-Solomon decoder directly to deal with the erroneous ML results. In the following, we detail the L-CoFL model regarding solutions of the above problems.

The designed L-CoFL model is shown in Fig. 1. There exist multiple vehicles and a group of edge servers installed in BSs and RSUs, in which one of the edge servers near the users is the fusion centre and the rest of them act as the relay nodes between the fusion centre and vehicles. The vehicles can collect data by the on-board sensors, e.g., information regarding traffic lights and traffic jams, and gather data from the IoT devices by the roadside, e.g., information about air quality and monitor. As such, more vehicles are involved, more data the shared ML models can be trained with. The operation of the L-CoFL model presented in Fig. 1 can be divided into three steps as follows.

Step 1: To protect privacy of the vehicles, the training dataset must be encoded locally. Therefore, the fusion centre predefines and generates encoding elements for each vehicle. We assume that the number of features in a piece of trained data is $\mathcal{M}$, and the datasets in each vehicle are split into $\mathcal{M}$ batches for encoding. Then, the fusion centre generates $\mathcal{M}$ distinct elements $\{\ell_m\}_{m=1}^{\mathcal{M}}$ for each batch of data in the vehicles. Meanwhile, the fusion centre assigns each vehicle a distinct element $\{\rho_i\}_{i=1}^{\mathcal{V}}$. By doing so, the vehicles can encode the local dataset based on (3) and (4) independently after they get the encoding elements. Moreover, to apply LCC and Reed-Solomon decoder, approximation approaches such as the Least-square approximation [26], Taylor series [27] and Chebyshev polynomial [28] have to be introduced to approximate the shared ML models (usually non-polynomial functions) by polynomial functions based on Weierstrass approximation theorem (*Theorem 1*) which states that any continuous function can be approximated by a polynomial function in the defined closed interval.

Theorem 1 (Weierstrass approximation). We assume that $\mathcal{R}(x)$ is a continuous function defined on the interval $[x_1, x_2]$. For every σ, there is a polynomial function $\mathcal{Q}(x)$ satisfying $|\mathcal{R}(x) - \mathcal{Q}(x)| \leq \sigma$ for all x in $[x_1, x_2]$.

Furthermore, while considering a specific approximation method, the applicable domains of the adopted method must be considered and match with ranges of the encoded data that are decided by the selected encoding elements and raw data. Here, we take Taylor expansion as an example of the approximation method. Specifically, Taylor expansion of logistic function converges to [-1,1], which is the applicable domain of Taylor expansion to approximate a complex function when the input values are extremely large. Therefore, the encoded data have to be restricted in the domain of [-1,1] by exploiting normalizing methods. In general, the selection of approximation method and the generation of encoding elements should be conducted together while considering the specific applications and the range of training data. More details are presented in our proposed L-CoFL model based vehicular application for traffic slowness prediction in the next section. After making a decision on the approximation method $\mathcal{A}(x)$, the fusion centre sends it with the encoding elements and shared ML model to each vehicle.

Step 2: After getting the messages from the fusion centre, each vehicle splits and encodes local dataset by using the Lagrange interpolation formula and encoding elements based on (3) and (4). Then, the vehicles must approximate the non-polynomial functions $\mathcal{N}(x)$ in the shared ML model, e.g., activation function in each neuron of neural network model, by the polynomial function $\mathcal{P}(x)$. The approximating way is that the vehicles uniformly select k elements $\{x_i\}_{i=1}^{k}$ from the defined domain of $\mathcal{N}(x)$, and produce the corresponding outputs $\{\mathcal{N}(x_i)\}_{i=1}^{k}$. After that, the set of points $\{(x_i, \mathcal{N}(x_i))\}_{i=1}^{k}$ is used to produce $\mathcal{P}(x)$ by using $\mathcal{A}(x)$, and thus the activation function in each neuron is replaced by the polynomial function. Note that the number of selected elements k is decided by the vehicles according to their contributed computing resource, which the more points selected lead to the more accurate approximation. In addition, the vehicles only have to approximate the shared ML model at the first time they receive it. After that, the vehicle inputs the encoded data into and trains the approximated model based on the iterative gradient algorithms as (1) until achieving the predefined accuracy. Different with the existing FL approaches, in this paper, the vehicles only upload the encoded final estimation results, i.e., outputs of the updated ML models (instead of parameters and gradient updates of the ML models) to the fusion centre. Thus, the communication cost can be significantly reduced.

Step 3: Due to the existence of system noises, the L-CoFL model decodes the accurate estimation results by using Reed-Solomon decoder when the requirement of (6) is met. After collecting all estimation results from the vehicles, the decoder exploit the results and the selected elements $\{\rho_i\}_{i=1}^{\mathcal{V}}$ to reconstruct the approximating polynomial functions of the updated shared ML model. To this end, the most important thing is to construct an error polynomial $e(x)$ to locate error position of the received estimation results by using Forney's algorithm [29], Berlekamp–Welch algorithm [30] and so on. Moreover, the received estimation results are exploited to construct an incoming polynomial $\mathcal{I}(x)$ based on the Lagrange interpolation formula. Then, the approximating polynomial function of the shared model, is reconstructed by $\mathcal{C}(x) = \mathcal{I}(x) + e(x)$ [29]. Finally, the accurate estimation results are obtained by the Reed-Solomon decoders and sent back to each vehicles. As such, the inaccurate estimation result produced with the system noises can be removed.

The **time complexity/computing cost** of L-CoFL model is analyzed in Proposition 1 below.

Proposition 1. The time complexity of the L-CoFL model is $\mathcal{O}(\mathcal{V}(\mathcal{M}^2 + \mathcal{A}(x)) + \mathcal{M} + \mathcal{V}^3)$*, where* $\mathcal{O}(\mathcal{A}(x))$ *is the time complexity of the adopted approximation method* $\mathcal{A}(x)$.

Proof: In *Step 1*, the fusion centre has to select the encoding elements $\{\ell_m\}_{m=1}^{\mathcal{M}}$ and $\{\rho_i\}_{i=1}^{\mathcal{V}}$ for the vehicles, which the time complexity is $\mathcal{O}(\mathcal{M} + \mathcal{V})$. In *Step 2*, each vehicle has to encode the data by using the Lagrange interpolation formula at $\mathcal{M}$ elements and, therefore, the time complexity is $\mathcal{O}(\mathcal{M}^2)$. After that, the shared ML models are approximated by each vehicle based on the approximation method $\mathcal{A}(x)$ at k points, with the time complexity $\mathcal{O}(\mathcal{A}(x))$. For example, the time complexity of Least-square approximation, Taylor series and Chebyshev polynomial is $k{*}deg(f)^2$, in which f is the polynomial function of the approximation methods. As such, the total time complexity of local training is $\mathcal{O}(\mathcal{V}(\mathcal{M}^2{+}\mathcal{A}(x)))$. In *Step 3*, the fusion centre updates the shared ML model by decoding the local estimation results based on the Reed-Solomon decoder. The time complexity of Reed-Solomon decoder is $\mathcal{O}((\mathcal{Z} + 2\mathcal{E})^3)$, where $\mathcal{Z}$ and $\mathcal{E}$ are the recover threshold and number of erroneous estimation results, respectively. However, since the upper bound of $(\mathcal{Z} + 2\mathcal{E})$ is $\mathcal{V}$, the time complexity of the Reed-Solomon decoder can be calculated by $\mathcal{O}(\mathcal{V}^3)$. Thus, the way to compute the time complexity of the L-CoFL model is to sum up the above time complexities, which is $\mathcal{O}(\mathcal{V}(\mathcal{M}^2 + \mathcal{A}(x)) + \mathcal{M} + \mathcal{V}^3)$. ■

V. L-CoFL Model based Traffic Flow Prediction

In this section, we present a practical vehicular application for traffic slowness prediction based on the L-CoFL model with the aim to clarify the method to decide the approximation function and encoding elements as follows. Specifically, we exploit a multi-layer neural network (NN) model for this application. From (3) and (4), we can find that the input data of vehicle v_i can be written as

$$\widetilde{\mathbf{X}}^{\mathbf{i}} = (p_1, ..., p_m, ..., p_{\mathcal{M}})(\mathbf{X}_1^i, ..., \mathbf{X}_m^i, ..., \mathbf{X}_{\mathcal{M}}^i)^T, \quad (8)$$

where $p_m = \Pi_{n=1, n\neq m}^{\mathcal{M}} \frac{\rho_i - \ell_n}{\ell_m - \ell_n}$, and the sum of p_m is always 1, that is, $\sum_{m=1}^{\mathcal{M}} p_m = 1$ regardless of the values of encoding elements (i.e., ρ, ℓ). Moreover, if $\mathcal{M} > 1$, there are always some of $p_m < 0$. Thus, the encoded data $\widetilde{\mathbf{X}}^{\mathbf{i}}$ is generally bigger than each raw data $\mathbf{X}_m^i$. However, if each raw data can be normalized into the domain of [-1,1], the encoded data, i.e., $\sum_{m=1}^{\mathcal{M}} |p_m| \mathbf{X}_m^i$, can also be restricted into the domain of $[-D, D]$, where $|p_m|$ is the absolute value of p_m, and D is a constant bigger than one. As such, the condition of $\sum_{m=1}^{\mathcal{M}} |p_m| \leq D$ should be satisfied while the fusion centre generates the encoding elements, i.e.,

$$\widetilde{\mathbf{X}}^{\mathbf{i}} = (p_1, ..., p_{\mathcal{M}})(\mathbf{X}_1^i, ..., \mathbf{X}_{\mathcal{M}}^i)^T \in [-D, D], \forall i \in \mathbf{V} \iff |p_1| + ... + |p_{\mathcal{M}}| \leq D. \quad (9)$$

In this case, since the domain of encoded data is confirmed, the maximal approximation error is also known and the approximation method can be decided. Based on the above analysis, we adopt the least-square approximation as the approximation method in this application since it achieves small approximation error in a specific domain if the degree of polynomial function is determined when compared with other approximation methods [26]. In this method, the coefficients of the produced polynomial function are constructed to fit the non-polynomial function in the NN model at a given degree. Importantly, based on the least-square approximation,

the domain of input data of the produced polynomial function can be restricted by the coefficients generation functions of this method to a specific domain, which can fit to the targeted domain in (9).

To apply the approximation method in this application, the following function in the NN model should be approximated:

$$\mathcal{F}(\widetilde{\mathbf{X}}^{\mathbf{i}}) = \frac{1 - e^{-\mathbf{w}(\widetilde{\mathbf{X}}^{\mathbf{i}})^T}}{1 + e^{-\mathbf{w}(\widetilde{\mathbf{X}}^{\mathbf{i}})^T}}. \tag{10}$$

The approximation way is described in *Step 2* of the operation procedure in section IV. After approximation, each vehicle inputs the encoded data into the approximating NN model, which is trained until the model achieves the predefined accuracy. The accuracy of the approximating NN model in the local training is computed by the loss function based on the cross entropy:

$$\mathcal{L} = -(yln\pi + (1-y)ln(1-\pi)), \tag{11}$$

where y the real label corresponding to the input data, and π is the estimation result that is computed by $\pi = \frac{1+f(x)}{2}$. $f(x)$ is the approximating polynomial function of (10). Finally, the vehicles upload the final estimation result π to the fusion centre for decoding and obtain the accurate prediction results.

VI. Performance Evaluation

In this section, we evaluate performance of the proposed L-CoFL model that is designed and implemented on MATLAB using the ML toolbox. Besides, we consider a vehicular scenario with a single fusion center and 100 randomly distributed vehicles. The fusion center is initialized within a BS where the vehicles are placed randomly within 500 meter coverage of or switch between different BSs or RSUs. The dataset used in the evaluation are collected from vehicular environment in urban area [31]. In the dataset, there are 16 individual features to define the traffic pattern such as hour, immobilized bus, broken truck, vehicle excess, accident victim, running over, fire vehicles, occurrence involving freight, incident involving dangerous freight, lack of electricity, fire, point of flooding, manifestations, defect in the network of trolleybuses, tree on the road, semaphore off and intermittent semaphore.

We now describe the result analysis regarding the implementation feasibility of the proposed L-CoFL model based on LCC for vehicular applications. The goal of the application is to predict the urban traffic behavior (slow or fast) in percentage based on the features collected by the vehicles. In the following experiments, we adopt a multi-layer NN models. Moreover, to observe performance of the proposed L-CoFL model, we introduce two comparison models, including *Plain FL model* and *Approximation-only FL model*. The approximation-only model means that the training NN model is approximated by polynomial functions during the training process, without using the Reed-Solomon decoder for removing the impact of erroneous estimation results. Thus, the approximation-only FL model cannot tackle the problems of malicious vehicles. Furthermore, during the approximation of the NN model, the least square approximation is done by

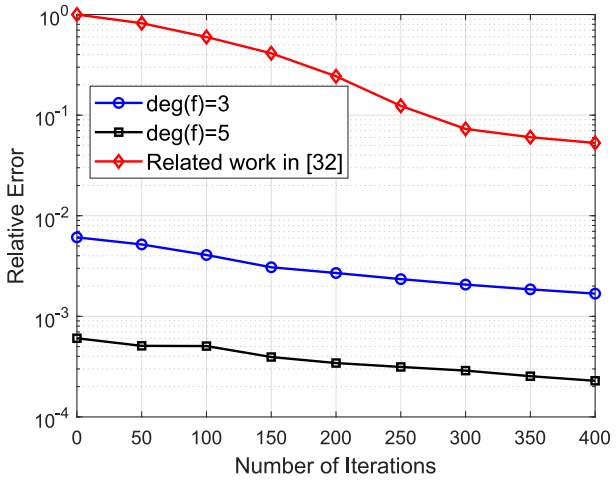

Fig. 2. Convergence of relative error of L-CoFL model with different degrees of the approximation functions and the related work in [32] during the training process, under the condition of no malicious vehicles.

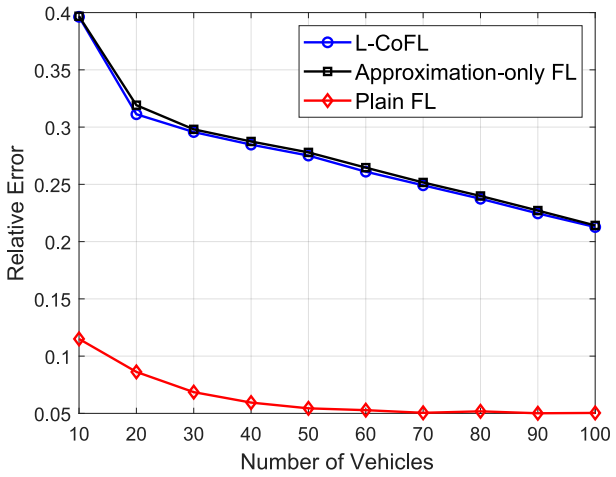

Fig. 3. Relative error of the comparison models without malicious vehicles in the system having different numbers of vehicles.

using 21 sample points uniformly distributed on $[-2, 2]$. In addition, in the following experiments, we propose a metric, i.e., relative error, which is the gap of accuracy between the examined models and the plain FL model without malicious vehicles (the most ideal model).

In Fig. 2, we evaluate the convergence of relative errors with different degrees of approximation functions. Here, we also compare the performance of our scheme with the related work in [32] based on the random linear encoding. In [32], they considered the scenarios with the fixed number (i.e., 24) of vehicles with the aim to mitigate the impact of straggling vehicles. Note that they assume all the vehicles are faithful and, thus, do not use the Reed-Solomon decoder for decoding and approximate the ML models. From the figure, we can find that our scheme with different degrees of approximation function can converge gradually, in which the high degree of the function leads to the high approximation accuracy. Importantly, compared to the related work in [32], our schemes

can achieve lower relative error and faster convergence speed.

In Fig. 3, we evaluate the performance of L-CoFL model in terms of approximation of the NN models. Since we add a random value to input data of plain FL model, the relative error of the model is more than zero, with the aim to better compare three models. From Fig. 3, we can find that the relative errors of L-CoFL model and approximation-only FL model are identical due to the fact that all the vehicles are honest and there is no decoding failure. Moreover, their relative errors become smaller with the number of vehicles. Therefore, the approximation of the NN model is effective when the number of vehicles is large.

Following this, we also introduce malicious vehicles and observe the performance of the comparison models. Fig. 4 shows the convergence of estimation results of the shared NN model during the training process, with 30% of malicious vehicles in the system. In this experiment, we use a test dataset to evaluate the shared NN model and get the estimation result. From Fig. 4, we can find that the estimation results of the shared NN model are more stable based on our proposed L-CoFL model due to the fact that the model can effectively react against the erroneous estimation results from the malicious vehicles with Reed-Solomon decoder. In contrast, the estimation results of the shared NN model based on the plain FL model fluctuate rapidly and are hard to converge.

Furthermore, in Fig. 5, we evaluate the relative error of comparison models with different percentages of malicious vehicles in the system, ranging from 10% to 50%. In Fig. 5, the relative error of the L-CoFL model is the smallest among three models although it increases slightly with the number of malicious vehicles in the system. More importantly, as shown in Fig. 5, relative error of plain FL model and approximated FL model can reach 0.25 during simulations, because they cannot deal with the erroneous estimation results without decoders. In contrast, the relative errors of the L-CoFL model are extremely low until the number of malicious vehicles reaches 40%. Therefore, the L-CoFL model can decode data successfully when the number of malicious vehicles does not exceed the recover threshold.

Fig. 6 shows the average absolute error of the comparison models with different percentages of malicious vehicles in the system. From this figure, the proposed L-CoFL model can secure the system against up to 30% malicious vehicles. Moreover, even when the number of malicious vehicles is higher than 30%, the proposed L-CoFL model still achieves better accuracy than the plain FL and approximation-only FL models.

Fig. 7 shows the probability density function (PDF) of the estimation results of the shared NN model. It compares the estimation results distribution of the proposed model against the accurate FL, plain FL, and approximation-only FL, while considering both malicious and honest vehicles. In this figure, we can find that the L-CoFL model has the largest overlapping area with the accurate FL model in terms of PDF of the estimation results. Besides, the estimation result of the L-CoFL model with highest frequency is also close to the accurate

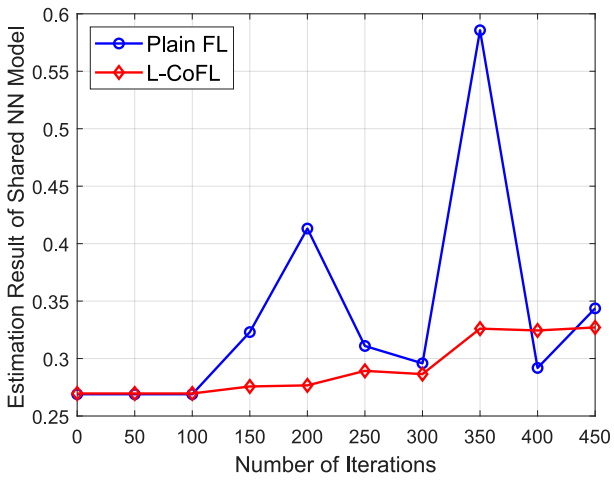

Fig. 4. Convergence of estimation results of the shared NN model during the training process, with 30% of malicious vehicles in the system.

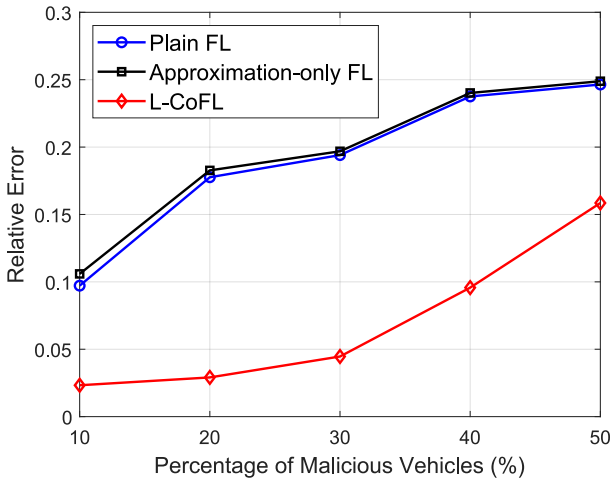

Fig. 5. Relative error of the comparison schemes with different percentages of malicious vehicles in the system.

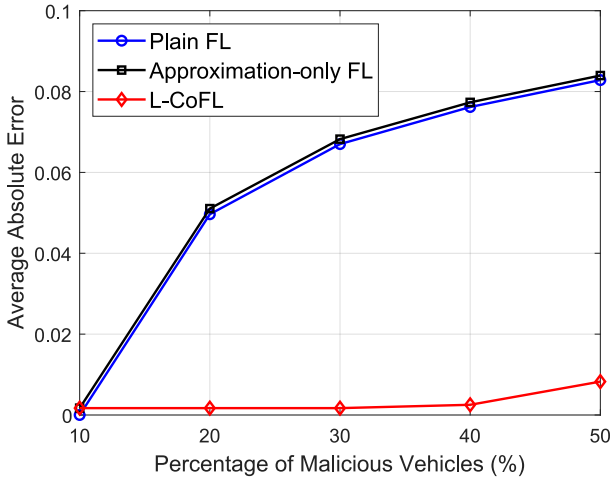

Fig. 6. Average absolute error of the comparison models with different percentages of malicious vehicles in the system.

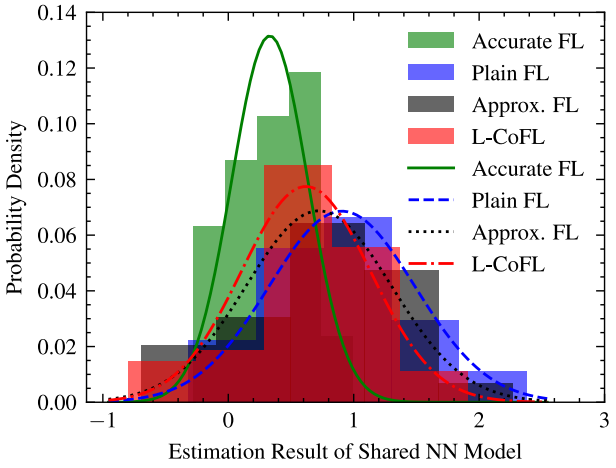

Fig. 7. Comparison of estimation result distribution among the comparison models.

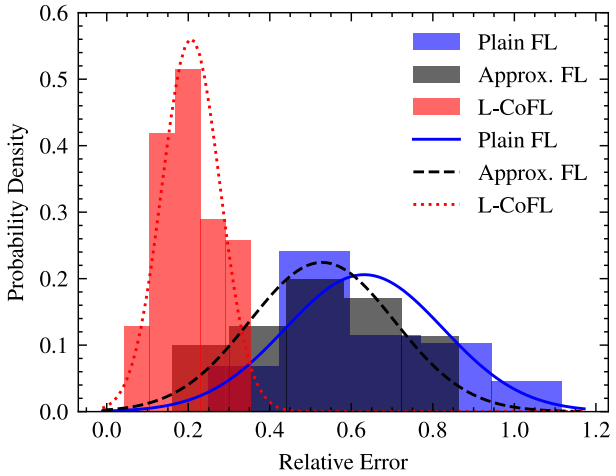

Fig. 8. Comparison of relative error distribution among the comparison models.

estimation results. Therefore, our proposed L-CoFL model can achieve the best estimation results that are closed to the accurate estimation results, compared to the plain FL model and approximation-only FL model.

In Fig. 8, we also show the PDF of relative error between plain FL, approximation-only FL, and L-CoFL. It is easy to find that the relative errors of the L-CoFL are lowest, and most of the relative errors are lower than 0.2 based on the test dataset. In contrast, the relative errors of plain FL model and approximation-only FL are extremely high, i.e., more than 0.3. Therefore, our model can achieve the best performance in terms of model accuracy.

Finally, we evaluate the computing cost/redundancy of our scheme with different degrees of the approximation functions and different rate of malicious vehicles in the system, shown in Fig. 9. Note that the results in the figure are the average computing cost of each piece of data. In Fig. 9, we can find that the higher degree of approximation function incurs more cost to approximate the non-polynomial functions in the ML model. Meanwhile, according to section III-B, the decoder has to evaluate at two additional point for each erroneous ML results. Therefore, we can find from the figure that the more malicious vehicles in the system, the higher computing cost are incurred.

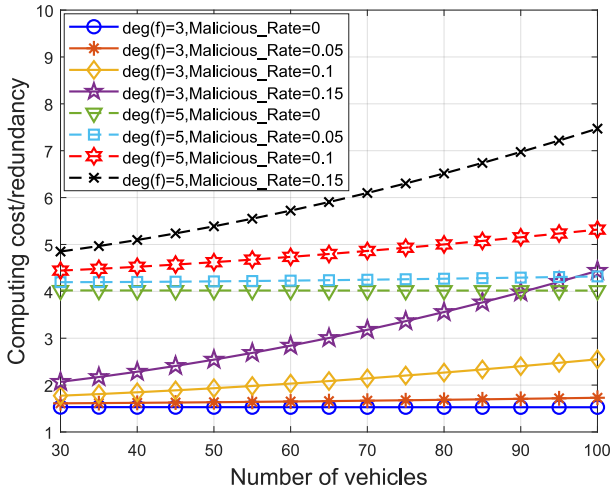

Fig. 9. Computing cost/redundancy with different degrees of approximation function and different rates of malicious vehicles in the system.

VII. Conclusion

In the paper, we proposed the novel L-CoFL model by integrating LCC with FL to enhance the system security of the FL-enabled IoV systems against the system noise such as low-quality training data, wireless channel errors, and erroneous ML results produced by malicious vehicles. Furthermore, in order to apply LCC to ML tasks of non-polynomial, we developed the simple but efficient approximation method with low degree of approximation function to approximate the non-polynomial ML model by a polynomial ML model. Finally, we applied the L-CoFL model for a vehicular application to predict the traffic slowness, and simulation results showed that the proposed L-CoFL model can not only achieve a high accuracy in the approximation of NN models, but also be effectively to deal with the erroneous ML results produced by the malicious vehicles.

Acknowledgement

This work was supported in part by the Characteristic Innovation Project No. 2021KTSCX110 of Guangdong Provincial Department of Education; in part by the UKRI grants: EP/R007195/1 (Academic Centre of Excellence in Cyber Security Research - University of Warwick), EP/N510129/1 (The Alan Turing Institute), EP/S035362/1 (PETRAS National Centre of Excellence for IoT Systems Cybersecurity) and EP/R029563/1 (Autotrust); in part by the NSFC under grant No. 62102099, and Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China (No. ICT2022B12); in part by the SUTD SRG-ISTD-2021-165, the SUTD-ZJU IDEA Grant (SUTD-ZJU (VP) 202102), and the SUTD-ZJU IDEA Seed Grant

(SUTD-ZJU (SD) 202101). The Corresponding Author is Alia Asheralieva.

REFERENCES

[1] W. Y. B. Lim, J. Huang, Z. Xiong, J. Kang, D. Niyato, X.-S. Hua, C. Leung, and C. Miao, "Towards federated learning in uav-enabled internet of vehicles: A multi-dimensional contract-matching approach," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[2] P. Li, J. Li, Z. Huang, T. Li, C.-Z. Gao, S.-M. Yiu, and K. Chen, "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.

[3] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[4] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23920–23935, 2020.

[5] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent iot applications: A cloud-edge based framework," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 35–44, 2020.

[6] K. Yadav and B. Gupta, "Clustering algorithm to detect adversaries in federated learning," *arXiv preprint arXiv:2102.10799*, 2021.

[7] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "On-device federated learning via blockchain and its latency analysis," *arXiv preprint arXiv:1808.03949*, 2018.

[8] J. S. Ng, W. Y. B. Lim, N. C. Luong, Z. Xiong, A. Asheralieva, D. Niyato, C. Leung, and C. Miao, "A comprehensive survey on coded distributed computing: Fundamentals, challenges, and networking applications," *IEEE Communications Surveys & Tutorials*, 2021.

[9] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. Vincent Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[10] S. B. Prathiba, G. Raja, S. Anbalagan, K. Dev, S. Gurumoorthy, and A. P. Sankaran, "Federated learning empowered computation offloading and resource management in 6g-v2x," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2021.

[11] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," *IEEE Communications Surveys Tutorials*, vol. 23, no. 3, pp. 1759–1799, 2021.

[12] X. Zhou, W. Liang, J. She, Z. Yan, and K. I.-K. Wang, "Two-layer federated learning with heterogeneous model aggregation for 6g supported internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 5308–5317, 2021.

[13] J.-H. Chen, M.-R. Chen, G.-Q. Zeng, and J. Weng, "Bdfl: A byzantine-fault-tolerance decentralized federated learning method for autonomous vehicles," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2021.

[14] Q. Kong, F. Yin, R. Lu, B. Li, X. Wang, S. Cui, and P. Zhang, "Privacy-preserving aggregation for federated learning-based navigation in vehicular fog," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8453–8463, 2021.

[15] P. Schindler, A. Judmayer, N. Stifter, and E. Weippl, "Hydrand: Efficient continuous distributed randomness," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 73–89, 2020.

[16] H. Liu, S. Zhang, P. Zhang, X. Zhou, X. Shao, G. Pu, and Y. Zhang, "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 6073–6084, 2021.

[17] H. Chai, S. Leng, Y. Chen, and K. Zhang, "A hierarchical blockchain-enabled federated learning algorithm for knowledge sharing in internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 3975–3986, 2021.

[18] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Federated learning for data privacy preservation in vehicular cyber-physical systems," *IEEE Network*, vol. 34, no. 3, pp. 50–56, 2020.

[19] Q. Kong, F. Yin, Y. Xiao, B. Li, X. Yang, and S. Cui, "Achieving blockchain-based privacy-preserving location proofs under federated learning," in *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021.

[20] Y. Zou, F. Shen, F. Yan, J. Lin, and Y. Qiu, "Reputation-based regional federated learning for knowledge trading in blockchain-enhanced iov," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2021.

[21] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 2512–2520, IEEE, 2019.

[22] S. Prakash, S. Dhakal, M. R. Akdeniz, Y. Yona, S. Talwar, S. Avestimehr, and N. Himayat, "Coded computing for low-latency federated learning over wireless edge networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 233–250, 2020.

[23] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[24] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1215–1225, PMLR, 2019.

[25] S. Li, S. M. M. Kalan, Q. Yu, M. Soltanolkotabi, and A. S. Avestimehr, "Polynomially coded regression: Optimal straggler mitigation via data encoding," *arXiv preprint arXiv:1805.09934*, 2018.

[26] C. Lanczos and T. Teichmann, "Applied analysis," *Physics Today*, vol. 10, no. 6, p. 44, 1957.

[27] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service.," *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, 2018.

[28] M. Vlcek, "Chebyshev polynomial approximation for activation sigmoid function," *Neural Network World*, vol. 4, no. 12, pp. 387–393, 2012.

[29] A. Sebastian and K. Bonna, "Reed-solomon encoder and decoder." Sakai. https://sakai.rutgers.edu/access/content/user/ak892/Reed-SolomonProjectReport.pdf.

[30] "Reed-solomon codes." Website. http://www.eecs.harvard.edu/ michaelm /CS222/eccnotes.pdf.

[31] R. Ferreira, C. Affonso, and R. Sassi, "Combination of artificial intelligence techniques for prediction the behavior of urban vehicular traffic in the city of sao paulo," in *10th Brazilian Congress on Computational Intelligence (CBIC)-Fortaleza, Ceara Brazil (Nov. 2011)*, pp. 1–7, 2011.

[32] S. Dhakal, S. Prakash, Y. Yona, S. Talwar, and N. Himayat, "Coded federated learning," in *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2019.