Hindawi
Security and Communication Networks
Volume 2023, Article ID 9823068, 1 page
https://doi.org/10.1155/2023/9823068

*Retraction*

# Retracted: AFLPC: An Asynchronous Federated Learning Privacy-Preserving Computing Model Applied to 5G-V2X

**Security and Communication Networks**

Received 5 December 2023; Accepted 5 December 2023; Published 6 December 2023



This article has been retracted by Hindawi, as publisher, following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of systematic manipulation of the publication and peer-review process. We cannot, therefore, vouch for the reliability or integrity of this article.

Please note that this notice is intended solely to alert readers that the peer-review process of this article has been compromised.

Wiley and Hindawi regret that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

## References

[1] J. Huang, C. Xu, Z. Ji et al., "AFLPC: An Asynchronous Federated Learning Privacy-Preserving Computing Model Applied to 5G-V2X," *Security and Communication Networks*, vol. 2022, Article ID 9334943, 11 pages, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 9334943, 11 pages
https://doi.org/10.1155/2022/9334943





*Research Article*

# AFLPC: An Asynchronous Federated Learning Privacy-Preserving Computing Model Applied to 5G-V2X

**Jie Huang,[1] Cheng Xu,[2] Zhaohua Ji,[1] Shan Xiao,[1] Teng Liu,[2] Nan Ma,[3] and Qinghui Zhou[4]**

[1]*Beijing Information Technology College, Beijing 100015, China*
[2]*Beijing Key Laboratory of Information Service Engineering, Beijing Union University, Beijing 100101, China*
[3]*Beijing University of Technology, Beijing 100124, China*
[4]*Beijing University of Civil Engineering and Architecture, Beijing 100044, China*

Correspondence should be addressed to Cheng Xu; xc-f4@163.com

Received 18 January 2022; Accepted 11 February 2022; Published 8 March 2022

Academic Editor: Muhammad Arif



Federated learning can effectively protect local data privacy in 5G-V2X environment and ensure data protection in Internet of vehicles environment. The advantages of low delay of 5G network should be better utilized in the vehicle-road cooperative system. But the existing asynchronous federated learning obtains a local model through different node training and completes the update of the global model through the central server. There are problems such as single point of failure, privacy leakage, and deviation of aggregation parameters. In response to the above problems, we proposed a 5G-V2X-oriented asynchronous federated learning privacy-preserving computing model (AFLPC). We used an adaptive differential privacy mechanism to reduce noise while protecting data privacy. A weight-based asynchronous federated learning aggregation update method is proposed to reasonably control the proportion of parameters submitted by users with different training speeds in the aggregation parameters and actively update the aggregation parameters of lagging users, so as to effectively reduce the negative impact on the model caused by the different speed of finding you. Experiments show that the proposed method can effectively ensure the credibility and privacy of asynchronous federated learning in 5G-V2X scenarios and at the same time improve the utility of the model.

## 1. Introduction

In recent years, with the construction of smart cities and intelligent transportation, 5G-V2X-based car networking systems have been rapidly developed. The Internet of vehicles environment contains a wealth of vehicle terminals and roadside sensors. It can perceive image, audio, speed, and other information. These user data are analyzed and processed through machine learning or deep learning models and the processing results are returned. The model used to process user data can be trained using a large amount of labeled data from different users to improve the expression performance and generalization performance of the model. However, due to user privacy and security reasons, these data are inconvenient to upload to the central server.

In response to the problem of privacy data protection, the concept of privacy-preserving computation is proposed. Currently, the application of privacy computing is mainly concentrated in data-driven finance, Internet, and medical and government affairs with a large amount of data. At the same time, cross-institution and cross-industry applications have increased demand, which is mainly concentrated in marketing, joint risk control, smart medical care, and intelligent driving and connected vehicle scenarios.

Google [1] proposed a federated learning algorithm to solve the problem of data requirements for machine learning model training and privacy data protection. As a new type of distributed machine learning framework, federated learning can jointly train machine learning models under the premise of only sharing model parameters with multiple local devices. Specifically, federated learning uses the local computing power and data of mobile devices to train machine learning models and then aggregates the trained model parameters on the server side and uses them as the initial

parameters for the next round of local training and iterates the above process until the final model reaches the maximum great generalization performance. Since all user data is only used for local model training, it can effectively avoid privacy leakage caused by direct data transmission from local devices to edge nodes.

At present, federated learning has been widely used in various fields, such as smart finance, smart medical care, autonomous driving, wireless communication, target detection, etc. However, the existing federated learning mostly relies on the central server to generate global model parameters, which is a centralized architecture, which has problems such as single point of failure, privacy leakage, and performance bottlenecks. In order to solve the above problems, some scholars use encryption methods such as homomorphic encryption [2–5] and secure multiparty computing [6–9] to encrypt model parameters sent by participants to resist model inference attacks. Another part of scholars proposed the method of differential privacy [10, 11], which makes it difficult for attackers to infer the original model parameters by adding disturbance noise to the model parameters. In order to further improve privacy security, Qu et al. [12] designed a hybrid identity mechanism based on digital signature and encryption protocol to prevent attackers from stealing data information stored in the blockchain. Sattler et al. [13] proposed the sparse ternary compression (STC) model, which is designed to meet the requirements of the federated learning environment. STC uses a novel mechanism to extend the existing top-k gradient sparse compression technology to achieve downstream compression and weight update ternary and optimal Golomb coding. Qi et al. [14] applied local differential privacy technology to blockchain federated learning and protected data privacy in the industrial Internet and smart transportation fields by adding noise disturbance to the original data. Short et al. [15] tested whether the accuracy of the model was improved after adding the parameters uploaded by the device based on the verification data set and then screened out reliable updates. Xu et al. [16] proposed an anonymous authentication scheme in the LTE car networking environment, which successfully protected the user's data security.

Federated learning: Because the computing, communication resources, and data volumes available on multiple terminal nodes are usually different, the time for terminal nodes to submit model parameters after each round of local training is different. This will cause the central server to extend the training time due to waiting for slow nodes to upload parameters. Existing work proposes a federated learning method based on exponential moving average. After the central server receives the neural network parameters (weights) sent by a terminal node, the central server weights and averages the saved average weight with the weight sent by the terminal node to obtain a new average weight and returns this weight to the terminal nodes, thereby alleviating the impact of nonindependent distributed data. However, the use of exponential moving average aggregation also has the problem that the exponential moving average method cannot actively control the version gap.

Most of the above schemes are synchronous federated learning; that is, it is necessary to wait for all participants to complete the local model training before updating the global model. When the number of participants is large, the parallel communication of the local model will cause the shortage of channel resources and reduce the training efficiency of federated learning. For this reason, this paper proposes a privacy-preserving computation method based on the asynchronous federated learning method and fusion of privacy protection algorithms. An adaptive differential privacy mechanism is used to protect parameter privacy, and the clipping threshold can be adjusted adaptively according to the training progress to alleviate the impact of noise on the model. The asynchronous federated learning aggregation update method proposed on this basis eliminates the influence of outdated weights on the global weights and solves the problems of the existing exponential moving average algorithm. In summary, the main contributions of this article are as follows:

(1) An adaptive differential privacy mechanism is proposed, which protects the privacy of parameters while adaptively adjusting the clipping threshold according to the training progress to alleviate the negative impact of noise on the accuracy of the model.

(2) An asynchronous federated learning aggregation update method is designed to eliminate the adverse effects of outdated weights and solve the deficiencies of the exponential smoothing average algorithm.

The structure of this paper is as follows. Chapter 2 discusses related work, Chapter 3 describes the main method details, Chapter 4 gives the experimental results, and Chapter 5 summarizes the methods of this paper.

## 2. Related Work

Privacy computing is an important path to balance data utilization and security. At present, the mainstream privacy computing technology is mainly divided into three directions: cryptographic-based privacy computing represented by multiparty security computing, privacy computing represented by federated learning artificial intelligence algorithm and privacy protection technology and privacy computing represented by trusted execution environment based on trusted hardware. Among them, federated learning and differential privacy technology, as important privacy computing technologies, have been widely concerned.

*2.1. Federated Learning for 5G-V2X.* In distributed scenarios, traditional machine learning algorithms require users to upload data to the data center for training. However, the data may contain private information, and some users are unwilling to share their data. In order to solve this problem, Google proposed federated learning [1], establishing a sharing model between mobile terminals and servers. Each distributed terminal trains a machine learning model based on the local data set and then sends the model parameters to

the central server. The server aggregates all the uploaded parameters to obtain the global model and then sends it to each terminal to update their local model. Lyu et al. [17] constructed a federated learning with fairness and privacy based on the blockchain and ensured the fairness of model training through the mutual credit evaluation mechanism between participants and combined the differential privacy generation adversarial network and triple encryption to ensure accuracy and privacy of model training. Most of these federated learning schemes use synchronized network settings; that is, participants are required to send local model parameters to other participants and at the same time wait for updates before proceeding to the next round of model iteration training, facing problems such as large communication delay, long waiting time, being idle, and waste of computing resources, etc.

Asynchronous federated learning reduces the communication overhead by avoiding the idle waiting time of each participant, and at the same time, it can still guarantee the effectiveness of collaborative training even when some computing nodes fail. Zhao et al. [18] constructed a privacy protection federal learning for smart home devices to predict user needs and habits. The device features extracted by the differential privacy perturbation convolutional neural network protect data privacy. At the same time, a new batch normalization method and reputation incentive mechanism are proposed to improve the accuracy and reliability of model training. Subramanya and Riggio [19] proposed a deep learning model based on the 5G environment and edge devices, including centralized and joint methods, which can perform horizontal and vertical automatic scaling in a multidomain network. The autoscaling problem is modeled as a time-series forecasting problem, which predicts the number of VNF instances in the future based on the expected traffic demand and studies the advantages and disadvantages of federated learning compared with centralized learning methods. Rehman et al. [20] proposed a blockchain-based framework, TrustFed, which provides fairness by detecting and removing attackers from the training distribution. It uses blockchain smart contracts to maintain the reputation of participating devices to force participants to bring positive and honest model contributions and promote the development of blockchain technology and federated learning technology. In general, the current research on federated learning is still insufficient, especially in the operational efficiency of the model and data privacy. There are still many areas worthy of improvement. Xu et al. [21] proposed a spatiotemporal event interaction model in the V2X system based on time periods and grid maps, which promoted the application of this technology in V2X scenarios.

*2.2. Differential Privacy for 5G-V2X.* Differential privacy is a method in cryptography, which aims to provide a way to maximize the accuracy of data query when querying from a statistical database, while minimizing the chance of identifying its records to prevent differential attacks. The basic idea is to add designed noise to the input and output results of the function so that the modification of any single record in the data set will not affect the output results. Therefore, the attacker cannot infer the private information in the data set by analyzing the output results. Soria-Comas et al. [22] studied how to overcome the problem of implementing strict guarantee measures in practice for differential privacy, which will significantly distort data and/or restrict data usage, thereby reducing the analysis utility of differentiated private results. Personal differential privacy is proposed, which is an alternative concept of differential privacy, which provides individuals (even if not to groups of individuals) with the same privacy guarantee as standard differential privacy. This new concept allows the data controller to adjust the distortion based on the actual data set, thereby reducing distortion and improving analysis accuracy. Kim et al. [23] developed a new algorithm for learning new words under local differential privacy (LDP). Unlike existing solutions, the proposed method only generates an LDP report for a single word. This enables the proposed method to use a complete privacy budget to generate reports, and the benefit is that the proposed method provides better utility than existing methods under the same degree of privacy. Hassan et al. [24] used differential privacy to effectively protect modern cyber-physical systems (CPSs) and proposed the CPS differential privacy technology, to solve various CPS problems and data privacy scenarios.

Differential privacy satisfies two characteristics: (1) If the output result of an algorithm meets differential privacy, any operation performed on this result will not cause additional privacy loss. (2) The serialized combination of differential privacy algorithms still meets the properties of differential privacy. The definition of differential privacy and Gaussian mechanism is as follows:

Differential privacy: Let $F$: $D \longrightarrow R$ be a random algorithm; $D$ and $\widetilde{D}$ are two data sets with at most one record different. $O \in R$ is the output of the algorithm $F$ , if the algorithm $F$ satisfies the following formula:

$$\Pr[F(D) = O] \leq e^{\omega} \Pr[F(\widetilde{D}) = O] + \varphi. \tag{1}$$

It is said that $F$ satisfies differential privacy. Among them, $\omega$ is the differential privacy budget. The smaller the value, the higher the degree of privacy protection, but at the same time the greater the accuracy loss for algorithm $F$; $\varphi$ represents the probability that strict differential privacy is allowed to be violated, and the general value is smaller.

Gaussian mechanism: If the $L_2$ norm is used to calculate the sensitivity of function $f$, the differential privacy calculation can be realized by adding Gaussian noise to the output of function $f$. The calculation formula of the Gaussian mechanism is shown in

$$F(D) = f(D) + n\left(0, (\Delta f \delta)^2 I\right). \tag{2}$$

Among them, Gaussian noise is a Gaussian distribution with a mean value of 0 and a covariance of $(\Delta f \delta)^2 I$, and $I$ is the identity matrix.

*2.3. Exponential Moving Average Method for 5G-V2X.* The aggregation update mechanism on the federated learning center server is mainly divided into the federated average algorithm and the exponential moving average

algorithm. The federated average algorithm performs a weighted average of the weights of all terminal nodes in each round of training. It needs to wait for the slowest terminal node. In order to overcome the above problems, FedProx [25] used the central server to adjust the required rounds of local training on different performance terminal nodes, reducing the gap between the fast node and the slow node, and limiting the terminal node to the central server during multiple rounds of local training. Too many changes to the weight reduce the impact of fast nodes on the central server. Federated matching average [26] is based on the permutation invariance of neural network parameters. Before averaging multiple nodes, they are matched first, and then weighted average is performed, which makes each model aggregation more effective and increases the convergence of the model. The exponential moving average aggregation update method is more efficient when the communication delay is high and heterogeneous, so it can also be used in federated learning. Using the exponential moving average method, the central server aggregates this update weight with the average weight reserved by the central server, sliding through the index to level. The uniform aggregation update method can avoid the central server's waiting for slow nodes. Aiming at neural networks, [27] proposed a method of updating different levels of DNN network training at different frequencies to improve the convergence speed of the entire model. By updating the shallow neural network at a higher frequency, the generalization ability of the model on differently distributed data sets can be improved. This method can use less communication to update more rounds, thereby improving the convergence speed of the model. Reference [28] combines synchronization and asynchrony for the slow convergence of exponential moving average method. It is believed that different layers of neural network can use different update methods to improve model convergence speed. The same neural network layer among multiple terminal nodes uses synchronous update. And the asynchronous method is used for update between the layers.

The above work has made improvements on the basis of the exponential moving average method, which alleviates the problem of the exponential moving average fast and slow node gap, but it does not effectively use the weight information uploaded by the terminal nodes. And the slow node's delayed submission of parameters hinders the convergence of the central server model and cannot completely eliminate the influence of outdated weights on the central server.

## 3. AFLPV: Asynchronous Federated Learning Privacy Protection Model

This chapter mainly describes the asynchronous federated learning method proposed in this article.

*3.1. An Adaptive Differential Computing Method for Privacy Computing in 5G-V2X.* All equipment terminals are trained on the local data machine to obtain the model gradient and upload it to the central server. Differential privacy technology has a smaller computational expense and is more suitable for edge devices in 5G-V2X scenarios. Qi et al. [14] use local differential privacy technology to add noise to the original training data to protect privacy, but it will cause a greater loss of model accuracy. Therefore, this paper proposes an adaptive differential privacy protection mechanism suitable for V2X scenarios, which can flexibly adjust the cropping threshold according to the training process, thereby reducing the adverse effect of noise on the accuracy of the model.

The traditional PMSProp optimization algorithm mainly speeds up the convergence speed by adjusting the step size. The iteration formula is as follows:

$$\begin{aligned} E\left[g^2\right]_i &\longleftarrow (1-\lambda)E\left[g^2\right]_{i-1} + \lambda\left(g_i\right)^2 \\ \phi_i &\longleftarrow \phi_{i-1} - \chi\frac{g_t}{\sqrt{E\left[g^2\right]_i + \xi_0}}. \end{aligned} \tag{3}$$

Among them, $\phi_i$ represents the model parameters during the $i$ round of training, $g_i$ represents the model gradient, $\chi$ is the learning rate, and $\xi_0$ is to ensure that the divisor is not zero, and $10^{-8}$ is generally taken. $E[g^2]_{i-1}$ is used to estimate the cumulative square of the historical gradient. In view of the continuity and gradual nature of the optimization process, historical gradients can usually be used to estimate the value of the current gradient. Therefore, $E[g^2]_{i-1}$ in the RMSProp optimization algorithm can be regarded as a priori knowledge of the current gradient.

The algorithm in this paper uses prior knowledge $E[\widehat{g}^2]_{i-1}$ on the basis of RMSProp optimization algorithm to predict the global gradient $\widehat{g}_i$ of this round and uses it as the cropping threshold $A_i$ of this round, namely, $A_i = \theta\sqrt{E[\widehat{g}^2]_{i-1}}$, where $\theta$ is the local cropping silver, and the formula of prior knowledge $E[\widehat{g}^2]_{i-1}$ is as follows:

$$\begin{aligned} E\left[\widehat{g}^2\right]_0 &= 0 \\ E\left[\widehat{g}^2\right]_{i-1} &\longleftarrow (1-\lambda)E\left[\widehat{g}^2\right]_{i-2} + \lambda\left(\widehat{g}_{i-1}\right)^2. \end{aligned} \tag{4}$$

Prior knowledge $E[\widehat{g}^2]_0 = 0$ in the first round of training will lead to $A_i = \theta\sqrt{E[\widehat{g}^2]_{i-1}} = 0$, which cannot be used for gradient clipping. Therefore, another prior threshold $G$ is set: when the prior knowledge of the gradient is insufficient at the initial stage of training, the gradient clipping threshold is set to a fixed value $A$; when the training continues until the prior knowledge meets $E\langle[\widehat{g}^2]_{i-1}\rangle G$ , the gradient clipping threshold is set to $A_i = \theta\sqrt{E[\widehat{g}^2]_{i-1}}$.

Therefore, in the training of theory $i$ in this article, the process of the device locally cropping gradient $g_{n,i}$ and adding noise is shown in the following equation:

$$\widetilde{g}_{n,i} = \frac{g_{n,i}}{\max\left[1, \left\|g_{n,i}\right\|_2 / C_i\right]} + \text{noise}\left(0, C_i^2\delta^2\right)$$

$$\text{where } C_i = \begin{cases} C, & \text{when} E\left[\widehat{g}^2\right]_{i-1} < G \\ \partial\sqrt{E\left[\widehat{g}^2\right]_{t-1}}, & \text{when} E\left[\widehat{g}^2\right]_{i-1} > G \end{cases} \tag{5}$$

It can be seen from this formula that as the model continues to converge, the local clipping threshold $C_i$ will decrease as $\sqrt{E[\widehat{g}^2]_{t-1}}$ decreases. As a result, the noise $\zeta \sim \text{noise}(0, (C_i\delta)^2 I)$ on the gradient becomes smaller and smaller, which accelerates the convergence of the model.

*3.2. Aggregation Algorithm Based on Federated Learning Applied in 5G Environment.* In view of the exponential moving average method, the frequent submission of weights by fast nodes causes the aggregated model parameters to deviate from the convergence direction of the models on other nodes, and the lagging submission of slow nodes hinders the convergence of the central server model. This paper proposes a weight profile asynchronous federated learning aggregation update method. The weight summary-based aggregation method retains the latest weights of the complete terminal nodes through the central server, so that each update can completely replace the impact of the previous weights, making the overall update more effective.

In the federated learning system, there are multiple terminal nodes and a central server. The overall system framework is shown in Figure 1.

In the terminal nodes, each terminal node has local data that can be used for training, and the data is unbalanced, nonindependent, and identically distributed. The central server does not directly participate in data processing and iterates continuously by aggregating the models trained by the terminal nodes. Train a model $M$, each piece of data has $k$ dimension, $F(\cdot)$ is the loss function, and the iteration process is as

$$F(x) = \frac{1}{n} \sum_{i \in [n]} E_{s^i \sim D^i} f\left(x; s^i\right), \tag{6}$$

$s^i$ is the sampling of the local data set $D^i$, and $E_{s^i \sim D^i} f(x; s^i)$ is the data set expectations of different devices. Because the data is not independent and distributed, the data sets of different devices have different expectations.

On the central server side, the exponential moving average aggregation method is to send the neural network parameters to the central server after a fixed round of local training and wait for the latest neural network parameters. When the parameters are received, the exponential moving average is used to aggregate into the new weight as

$$W_{e+\rho+1} \longleftarrow (1-\lambda) \cdot W_{e+\rho} + \lambda \cdot W_{e_i}, \tag{7}$$

where $W_{e+\rho}$ is the reserved weight of the central server during aggregation, $W_{e_i}$ is the weight uploaded by the terminal node $i$ to the server, $e_i$ is the updated version of the weight of the $i$-th terminal node, and the terminal node uses the exponential coefficient $\lambda$ to obtain the latest weight $W_{e+\rho+1}$ of the next round of the server and send it to the terminal node that sent the weight. The terminal node will continue to train when it receives the weight from the central server and repeat the above process.

The ultimate goal of federated learning is to minimize the average loss function and train a model with the best generalization performance, namely, $\min_{x \in R^d} F(x)$. It can be seen that the optimization goal of federated learning is the smallest shown in the whole; that is, the best overall convergence performance is sought, and it is not biased toward certain nodes. In the traditional aggregation method, the fast nodes are updated frequently, and the slow nodes are updated lagging. When the slow node completes a round of training and obtains $W_{e_i}$, the latest weight $W_{e+\rho}$ is on the central server. When the version gap is too large, the update will affect the central server as a whole. The weight is invalid and even produces negative effects. If the central server receives a delayed update of the weight, the negatively affected weight will be merged into the latest weight of the central server, and the impact can only be reduced by multiple updates afterwards. After multiple rounds of updates, this negatively affected update will still be retained and will not completely disappear. Even if the sliding average method uses version gap weighting to reduce the update proportion of the gradient delay update, there will still be part of the gradient delay aggregated into the whole. In the weight, and in the subsequent aggregation process, this bad update aggregated to the central server will also be treated as the latest weight to obtain the same weighting ratio as the other weights, and as the scale of the node expands, the probability of the slow node generated will also correspond, and the difference between the proportion of fast nodes and slow nodes in the average weight of the central server will increase as the equipment heterogeneity increases. This is inconsistent with the federal learning optimization goal to minimize the average loss. This aggregation method is more biased to data distribution of fast nodes.

In this paper, the aggregation method based on the weight summary retains the latest weights of the complete working nodes. Assume the weight summary $W = \langle w_1^t, w_2^t, \ldots, w_k^t \rangle$ at time $t$, where $w_1^t$ represents the working node 1 and saves the summary of version $t$ in the parameter server. Assuming that node 1 sends update $w_1^{t+1}$ at the next moment, the weight summary will update the corresponding position of node 1, which is $W = \langle w_1^{t+1}, w_2^t, \ldots, w_k^t \rangle$. In this way, outdated update weights can be completely removed when nodes are updated, and frequent updates of fast nodes can only update themselves, so that the gradient delay of slow nodes can be removed in the next effective update. And the parameters of each parameter server are stored separately, rather than simply aggregated together. In this way, the latest weights of all nodes are retained, instead of the exponential moving average method. And it can effectively limit the frequent update of fast-running nodes, so that the overall weight cannot be seriously biased toward the fast node training data through frequent updates, which improves the overall convergence performance of the model.

As shown in Figure 2, the method in this paper can ensure that each node has the same weight during aggregation and storage, so that each node does not bias the data distribution of fast nodes during aggregation. When the lagging weight is updated, the method in this paper can completely remove the influence of the lagging weight.

When using the method in this paper to iterate the model training, suppose there is a parameter server under an

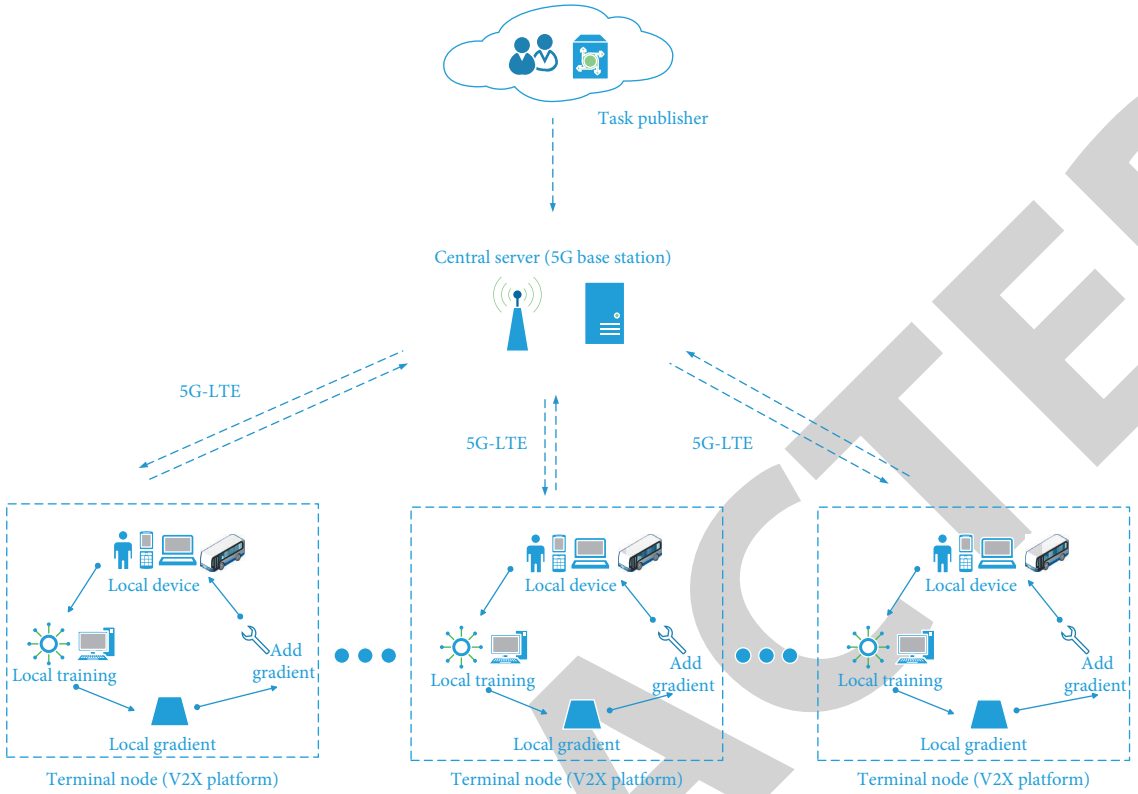


Figure 1: Asynchronous Federated Learning System Architecture, in which multiple terminals containing different user populations are connected to each other over 5G networks, and each terminal node continues to communicate with the central node over 5G networks.

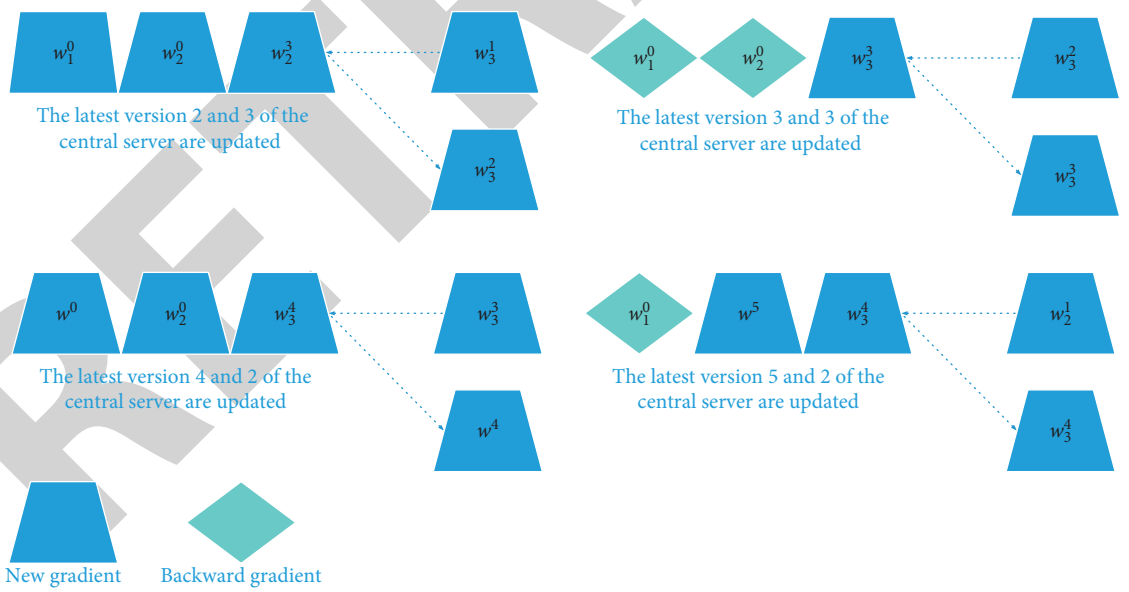


Figure 2: This method ensures that each node has the same weight in the aggregation storage, so that each node will not be biased to the data distribution of fast nodes in the aggregation.

asynchronous federated learning, randomly select working nodes, and distribute the parameter server to the neural network of the working nodes during the initialization phase, initial neural network parameters, local training rounds, etc. Then enter the aggregation update stage. When the working node completes the specified number of iterations, it will upload its own node number $i$ $d$ and neural network parameters $w$ to the parameter server. The parameter server stores this weight in the corresponding position of the node label, and if there is already a value in the

position of the node, it will be overwritten. After that, the average of the latest weights of all known nodes is calculated as the initial value of the next iteration. The sending function is used to send the updated weight to the working node. In this algorithm, the average value of the latest weight is sent to the node number.

Weighted aggregation mechanism: The introduction of a weighted mechanism can better determine the hyperparameters of the global sum according to the actual situation of different data to adapt to different actual situations and can record with the parameter server according to the latest version provided on different nodes during aggregation. According to the version gap, increase the weight of updated nodes with a smaller version gap, and reduce the weight of nodes with a larger version gap. Different from the traditional weighted average mechanism strategy, the parameter server plus global aggregation only considers the overall weight saved by the parameter server and the version gap of the node gradient passed to do weighted aggregation. The weighted aggregation method in this article will consider all the nodes saved by the parameter server. The version gap and the gap between the latest versions are weighted and aggregated, and as the latest version increases, the weight ratio of each node stored on the server is also dynamically changed. The formula is as follows:

$$p(i) = \left(k + v - k_i + 1\right)^{-\partial}. \tag{8}$$

Among them, $p(i)$ represents the proportion of nodes whose weight summary label is $i$, and $\partial$ is the hyperparameter of the version proportion, $\partial \in (0, 1)$.

In order for the weight expectation to remain unchanged, it needs to be normalized. As shown in formula (9), $p(i)$ represents the weight of the actual weight.

$$p(i) = \frac{p(i)}{\sum_i^n \left(k + v - k_i + 1\right)^{-\partial}}. \tag{9}$$

Based on the above formula, the weighted aggregation mechanism in the algorithm of this paper is more flexible. It can aggregate each weight on the weight summary with an appropriate weight. The formula is as follows:

$$W_{e+\rho+1} = \sum_i^k \left(p(i) \cdot W_{e_i}\right). \tag{10}$$

## 4. Experimental Results and Analysis

In this experiment, first introduce the data set and environmental parameters in Section 4.1, and then introduce the overall experimental comparison results in Section 4.2.

*4.1. Experimental Setup.* This paper has carried out experimental verification through cluster simulation experiments. The operating system environment is Ubuntu 20.04.1 LTS, and the hardware configuration is Intel i7-8700K CPU, GTX 1080T GPU, and 16 GB RAM. The entire experiment is based on the distributed machine learning framework, Parallel-SGD, and the experiment is carried out in a cluster environment, with a total of 8 working nodes.

The data set uses two data sets, MNIST and CIFAR-100, and the first two data sets are processed with non-independent and identical distribution before being distributed to different nodes. The MNIST data set contains 60000 examples for training and 10,000 examples for testing. Each instance is a $28 \times 28$ single-channel picture with 10 different categories. The CIFAR-100 data set contains 50,000 examples for training and 10,000 examples for testing. Each instance is a $32 \times 32$ RGB channel image, with 10 different categories, and labels include "aircraft" and "dog". There are 10 types of universal objects such as "cars" and so on. All data sets are made to conform to nonindependent and identical distribution when they are allocated to different working nodes.

In the experiment, let $G = 10^{-6}$ in adaptive differential privacy. On the MNIST data set, the initial cropping threshold $C = 4$, and privacy budget $\delta = 2$; on the CIFAR-100 data set $C = 3$ and $\delta = 4$, E. In order to simulate the distributed environment of federated learning, it is assumed that there are 20 local devices in the system, and the experimental data set is randomly and evenly divided into 20 parts and distributed to each device as a local data set. The device takes a batch size of 64 for local training, and the default accuracy is 64 bit.

*4.2. Results and Analysis*

*4.2.1. Privacy Budget Consumption.* In order to measure the effect of the adaptive differential privacy mechanism in reducing privacy budget consumption in this algorithm, the differential privacy federated learning algorithm (DPFL) [29] is used for comparison. The privacy budget consumed when the two algorithms reach the specified accuracy rate is recorded, as shown in Table 1, where $\xi_D$ and $\xi_A$ represent the privacy budget consumed by DPFL and the algorithm in this paper, respectively.

It can be seen from Table 1 that when the accuracy is the same, the algorithm in this paper reduces the privacy budget by 37% and 29% on the MNIST and CIFAR-100 datasets on average compared to DPFL. In a representative CIFAR100 dataset, the proposed method reduces losses by an average of 20% compared with traditional methods. It shows that the adaptive differential privacy method can effectively reduce the consumption of privacy budget. The data in Table 1 show that the algorithm achieves the purpose of enhancing privacy calculation and reducing consumption through the adaptive differential privacy mechanism.

*4.2.2. The Accuracy of the Adaptive Differential Privacy Protection Mechanism.* In order to measure the impact of the differential privacy mechanism on the accuracy of the algorithm, given the same privacy budget, the accuracy of the algorithm in this paper is compared with that of DPFL. At the same time, the original federated learning algorithm is used as a benchmark for comparison. The results are shown in Figures 3 and 4.

Table 1: Comparison of the privacy budget consumed by the algorithm in this paper and DPFL.

| Datasets | Accuracy (%) | $\zeta$ | $\xi_D$ | $\xi_A$ |
|---|---|---|---|---|
| MNIST [30] | 95.2 | $10^{-4}$ | 2.14 | 1.73 |
| | 97.4 | $10^{-4}$ | 3.49 | 1.95 |
| | 98.1 | $10^{-4}$ | 5.26 | 2.82 |
| CIFAR-100 [31] | 69.6 | $10^{-4}$ | 4.63 | 3.41 |
| | 72.3 | $10^{-4}$ | 6.32 | 4.23 |
| | 73.7 | $10^{-4}$ | 8.91 | 6.17 |

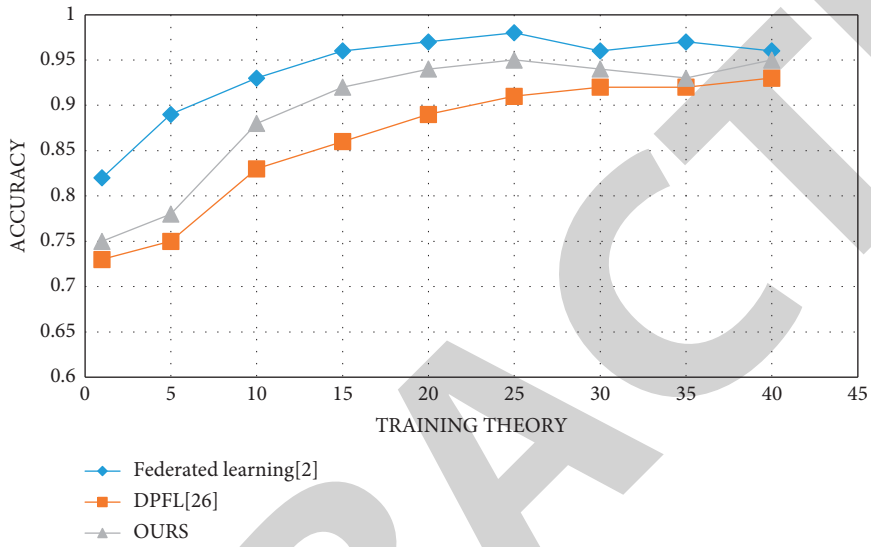


Figure 3: Comparison of the accuracy of the three algorithms in the MINIST data set shows that our method has higher accuracy.

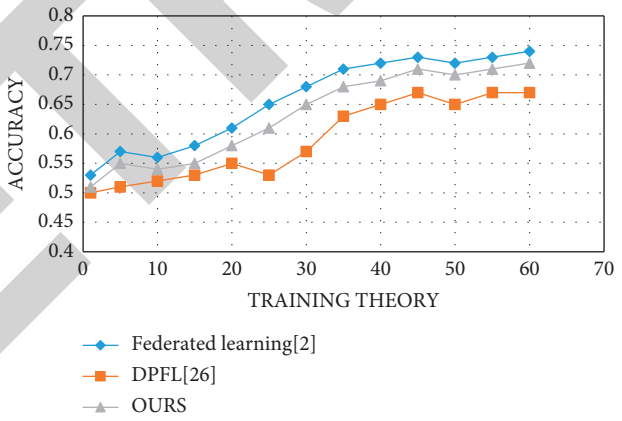


Figure 4: Comparison of the accuracy of the three algorithms in the CIFAR-100 data set shows that the accuracy of the method in this paper is more stable and higher than that of the traditional method.

According to Figure 3, on the MNIST data set, the original federated learning algorithm achieved an accuracy of 96.3%, the accuracy of the algorithm in this paper reached 95.5%, and the accuracy of DPFL reached 93.7%. As shown in Figure 4, on the CIFAR-100 data set, the original federated learning algorithm achieved an accuracy of 74.3%, while the accuracy of the algorithm in this paper reached 72.2%, and the accuracy of DPFL reached 67.8%. In summary, through the adaptive differential privacy mechanism, the algorithm in this paper can achieve a higher accuracy rate under the same privacy budget, which is only slightly lower than the original federated learning algorithm. Therefore, the algorithm in this paper is suitable for 5G-V2X application scenarios that require high accuracy and privacy protection.

*4.2.3. Weight Summary Aggregation Method.* We tested the difference in the loss reduction of asynchronous federated learning based on exponential moving average Afed [32], Aed_Hinge [28], and the method in this paper on the

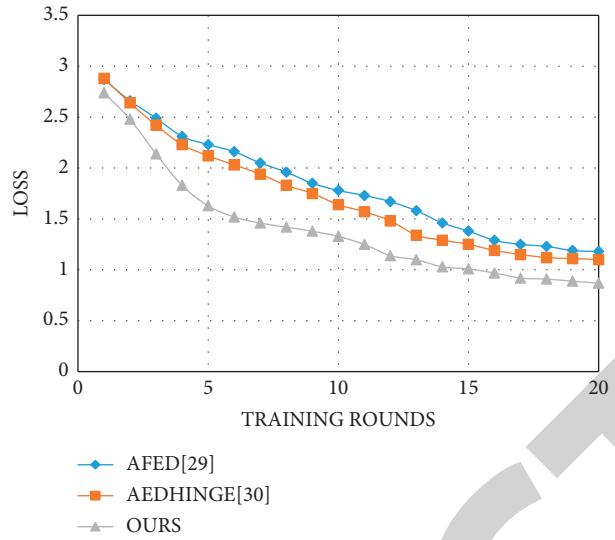


Figure 5: Afed, Aed_Hinge, and our method training efficiency on the MNIST dataset, compared with the traditional methods; our method loss was reduced by 10% on average.

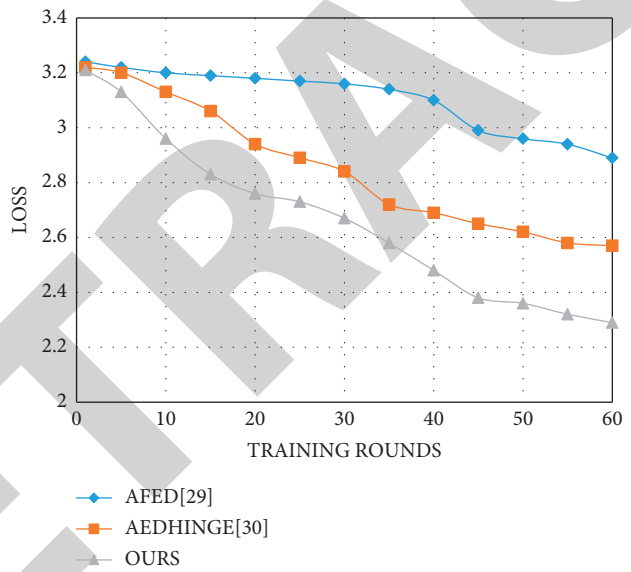


Figure 6: Afed, Aed_Hinge, and our method training efficiency on the CIFAR-100 dataset compared with traditional methods; the method loss in this paper was reduced by 9% on average.

MNIST dataset. Aed_Hinge and the method in this paper both drop faster than traditional asynchronous federated learning in the same round of loss function. As shown in Figure 5, the loss of the Aed_Hinge algorithm compared with the Afed algorithm on the MNIST data set is reduced by an average of 4.95%, and the method compared with the algorithm in this paper is reduced by an average of 15.35% on the MNIST data set. It can be seen from Figure 5 that the algorithm of this paper has a faster convergence speed. The Aed_Hinge algorithm is almost consistent with Afed when the gap in the early version is not significant. When the gap is more significant in the later version, the gap between Aed_Hinge and Afed gradually opens up. The algorithm of this paper is relatively stable and exceeded Afed and Aed_Hinge algorithms in the whole process.

After that, we tested the difference in loss reduction of asynchronous federated learning based on exponential moving average Afed, Aed_Hinge, and this method on the CIFAR-100 dataset. Aed_Hinge and this method are faster than traditional asynchronous federated learning in the same round of loss function. As shown in Figure 5, the loss of Aed_Hinge algorithm compared with Afed algorithm on the CIFAR-100 data set is reduced by an average of 5%, and the loss of the method compared with Afed algorithm in this paper is reduced by an average of 7.26% on the CIFAR-100 data set. It can be seen from Figure 6 that the algorithm of this paper has a faster convergence speed. The slower loss of Afed in the middle of training is due to the widening of the version gap of the fast full node. The update of the slow node reduces the overall convergence performance of the model.

The Aed_Hinge algorithm is due to the slow nodes punished by weight, which can alleviate the problem of slow convergence in the middle and late stages to a certain extent, but the method in this paper has a more obvious effect on the Aed_Hinge algorithm due to its weight summary mechanism in the later stage of the experiment.

## 5. Conclusion

This paper proposes a privacy computing model (AFLPV) for 5G-V2X scenarios through asynchronous federation learning, to solve the potential privacy leakage problem at the edge of network in 5G V2X environment, based on asynchronous federated learning and privacy computing method. Privacy computing in the Internet of vehicles environment has been effectively improved. Design an adaptive differential privacy mechanism to protect parameter privacy and reduce the impact of noise on model accuracy. And in order to overcome the aggregation problem of asynchronous federated learning based on exponential moving average in actual use, the aggregation ratio is biased toward fast nodes, and the outdated weights of slow nodes cannot be eliminated in time. An aggregation strategy based on weight summaries is proposed. The weight summaries retain the latest weights of working nodes, and all working nodes have the same weight ratio. The weight summary allows each working node to only update its own summary part, which limits the impact of high-frequency updates of fast nodes on the overall weight, and can promote faster convergence of the model on the parameter server. Experiments on the MNIST and CIFAR-100 datasets show that the algorithm in this paper can achieve privacy protection with high accuracy and solves the problem of reduced model convergence speed caused by the difference in node training speed and improves the efficiency of asynchronous federated learning and training. The method proposed in this paper is mainly for 5G V2X environment, and other environments are even suitable for us to further explore. In the future, we will also carry out tests and explorations on other equipment terminals based on 5G to further improve the scope of application and detection speed of the model.

## Data Availability

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This paper is a general project of science and technology plan of Beijing Municipal Education Commission (KM202110857002), the Key Project of Beijing Information Technology College (XY-YN-07-201902), the National Natural Science Foundation of China (Grant nos. 62102033, 61871038, and 61931012), and the Beijing Natural Science Foundation (Grant no. 4222025).

## References

[1] C. Xu, H. Wu, Y. Zhang, S. Dai, H. Liu, and J. Tian, "A real-time complex road AI perception based on 5G-V2X for smart city security," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–11, 2022.

[2] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[3] W. Y. B. Lim, N. C. Luong, D. T. Hoang et al., "Federated learning in mobile edge networks: a comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[4] C. Xu, K. Chen, M. Zuo, H. Liu, and Y. Wu, "Urban fruit quality traceability model based on smart contract for Internet of things," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 9369074, 2021.

[5] C. Xu, H. Liu, Z. Pan, W. Li, and Z. Ye, "A group authentication and privacy-preserving level for vehicular networks based on fuzzy system," *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 2, pp. 1547–1562, 2020.

[6] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[7] X. Ma, F. Zhang, X. Chen, and J. Shen, "Privacy preserving multi-party computation delegation for deep learning in cloud computing," *Information Sciences*, vol. 459, pp. 103–116, 2018.

[8] J. Duan, J. Zhou, and Y. Li, "Privacy-Preserving distributed deep learning based on secret sharing," *Information Sciences*, vol. 527, pp. 108–127, 2020.

[9] C. Xu, H. Liu, Y. Zhang, and P. Wang, "Mutual authentication for vehicular network in complex and uncertain driving," *Neural Computing & Applications*, vol. 32, no. 1, pp. 61–72, 2020.

[10] Q. Feng, D. He, Z. Liu, H. Wang, and K.-K. R. Choo, "SecureNLP: a system for multi-party privacy-preserving natural language processing," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3709–3721, 2020.

[11] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*vol. 9, no. 3-4, pp. 211–407, 2014.

[12] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, "A blockchained federated learning framework for cognitive computing in industry 4.0 networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2964–2973, 2020.

[13] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. Data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020.

[14] Y. Qi, M. S. Hossain, J. Nie, and X. Li, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Generation Computer Systems*, vol. 117, pp. 328–337, 2021.

[15] A. R. Short, H. C. Leligou, M. Papoutsidakis, and E. Theocharis, "Using Blockchain Technologies to Improve Security in Federated Learning Systems," in *Proceedings of the 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1183–1188, IEEE, Madrid, Spain, July 2020.

[16] C. Xu, X. Huang, M. Ma, and H. Bao, "An anonymous handover authentication scheme based on LTE-A for vehicular networks," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[17] L. Lyu, J. Yu, K. Nandakumar et al., "Towards fair and privacy-preserving federated deep models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.

[18] Y. Zhao, J. Zhao, L. Jiang et al., "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2020.

[19] T. Subramanya and R. Riggio, "Centralized and federated learning for predictive VNF autoscaling in multi-domain 5G networks and beyond," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 63–78, 2021.

[20] M. H. u. Rehman, A. M. Dirir, K. Salah, E. Damiani, and D. Svetinovic, "TrustFed: a framework for fair and trustworthy cross-device federated learning in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8485–8494, 2021.

[21] Cheng Xu, Hengjie Luo, Hong Bao, and Pengfei Wang, "STEIM: A Spatiotemporal Event Interaction Model in V2X Systems Based on a Time Period and a Raster Map," *Mobile Information Systems*, vol. 2020, Article ID 1375426, 2020.

[22] J. Soria-Comas, J. Domingo-Ferrer, D. Megias, and D. Megías, "Individual differential privacy: a utility-preserving formulation of differential privacy guarantees," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1418–1429, 2017.

[23] S. Kim, H. Shin, C. Baek, S. Kim, and J. Shin, "Learning new words from keystroke data with local differential privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 3, pp. 479–491, 2020.

[24] M. U. Hassan, M. H. Rehmani, and J. Chen, "Differential privacy techniques for cyber physical systems: a survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 746–789, 2020.

[25] T Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federatedoptimization in Heterogeneous networks," in *in Proceedings of the 3rd Machine Learning and Systems (MLSys) Conference*, Austin, TX, USA, 2020.

[26] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated Learning with Matched averaging," Febraury 2020, https://arxiv.org/abs/2002.06440.

[27] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 4229–4238, 2019.

[28] Z. Chai, Y. Chen, A. Anwar, L. Zhao, Y. Cheng, and H. Rangwala, "Fedat: A Communication-Efficient Federated Learning Method with Asynchronous Tiers under Non-iid data," October 2020, https://arxiv.org/pdf/2010.05958.

[29] R. C. Geyer, T. Klein, and N. Moin, "Differentially private federated learning: a client level perspective," 2017, https://arxiv.org/abs/1712.07557.

[30] Li. Li Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141-142, 2012.

[31] A. Krizhevsky and G. Hinton, *Learning Multiple Layers of Features from Tiny Images*, p. 7, 2009.

[32] J. Li, Y. Meng, L. Ma et al., "A federated learning based privacy-preserving smart healthcare system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, 2021.