A Verifiable Privacy-preserving Machine Learning Prediction Scheme for Edge-enhanced HCPSs

Xiong Li, *Member, IEEE*, Jiabei He, Pandi Vijayakumar, *Senior Member, IEEE*, Xiaosong Zhang, and Victor Chang, *Senior Member, IEEE*

***Abstract*—As a highly integrated industrial system, Human Cyber-physical Systems (HCPSs) provide accurate and high-quality services for Industry 5.0. In HCPSs, machine learning (ML) prediction provides reliable prediction results for users based on matured models, while security and privacy protection are considerable issues. In this paper, based on the modified Okamoto-Uchiyama homomorphic encryption, we propose a verifiable privacy-preserving machine learning prediction scheme for the edge-enhanced HCPSs, which outputs the verifiable prediction results for users without privacy leakage. Specifically, a batch of prediction results can be verified at one time, which improves the efficiency of verification. Security analysis shows that our scheme protects the privacy of inputs, ML model and prediction results. The experiment results demonstrate that the edge computing architecture remarkably alleviates the computational burden of the cloud server. Furthermore, compared with other related schemes, our scheme shows the best execution efficiency, and batch verification optimizes the performance by about 15% compared with single verification at the same scale.**

***Index Terms*—Industry 5.0, Human cyber-physical systems, Machine learning prediction, Privacy-preserving, Edge computing, Batch verification**

I. Introduction

With the rapid development of information technology, human society is now stepping into Industry 5.0. Different from Industry 3.0 and 4.0, Industry 5.0 specifically emphasizes the personalization, *i.e.*, more and more personalized productions can be created by combining emerging and advanced techniques, such as the Internet of Things (IoT) [1], machine learning (ML) [2] and cloud computing [3]. As a kind of system that highly integrates these mature technologies, the Human Cyber-physical System (HCPS) [4] is a promising framework for the realization of Industry 5.0. For example, ML prediction service can been seen as an application in HCPSs, where the ML models are trained based on large-scale data collected by IoT devices from humans and physical systems, and provide consumers with accurate, personalized and high-quality prediction services. To further develop the potential of HCPSs, cloud computing can be applied in the ML prediction service system to handle large-scale and complex calculations. However, due to its unbearable service delay and low bandwidth utilization, the centralized cloud computing architecture gradually cannot meet the ever-increasing data processing requirements of Industry 5.0. In order to improve the performance of HCPSs, edge computing [5] can be introduced to reduce the delay and burden of the cloud.

The security and privacy problems are gradually emerging as the complexity of the ML services in HCPSs continues to grow, which become obstacles in the way of Industry 5.0. In such complex and ever-changing environments, frequent interactions among various entities increase the probability of the hostile attacks from malicious entities. If the privacy information, including users' sensitive data in training phase and the parameters of ML model, is leaked to the adversaries, it will cause non-negligible damage to the entities in HCPSs. Many researchers have studied the security and privacy protection issues in ML. Generally, there are three techniques to ensure the security and privacy in ML schemes [6], *i.e.*, Differential Privacy (DP), Secure Multi-party Computation (SMC) and Homomorphic Encryption (HE), and we review some related work as below.

In 2017, Zhang *et al.* [7] proposed a privacy-preserving ML scheme based on DP. However, the noise added in training data may cause divergence of the model. Li *et al.* [8] designed a privacy-preserving ML scheme with multiple data providers, where the cloud server only adds noise once in the encrypted users' data sets with ϵ-DP to prevent privacy from leakage to the data analyst. But the combination of the DP and double decryption cryptosystem causes the loss of both data precision and time efficiency. Arachchige et al. [9] devised a DP-based privacy-preserving federated ML framework in 2020. Before the private models are updated to the central authority, the DP is applied to each local model to protect the privacy of the local trainer. However, the interaction with the blockchain in each round of training increases the response time of the training. Generally, DP-based mechanisms sacrifice the precision and availability of data for better performance, and the DP-based privacy-preserving ML schemes are difficult to leverage the balance between the level of privacy protection and the efficiency of model convergence.

This work was supported supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62072078. *(Corresponding author: Victor Chang, Pandi Vijayakumar.)*

X. Li is with the Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: lixiong@uestc.edu.cn).

J. He is with with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China. (e-mail: hnust_hjb@mail.hnust.edu.cn)

P. Vijayakumar is with the Department of Computer Science and Engineering, University College of Engineering Tindivanam, Tindivanam, Tamilnadu 604001, India (e-mail: vijibond2000@gmail.com).

X. Zhang is with the Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China and with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, Guangdong 518040, China (e-mail: johnsonzxs@uestc.edu.cn).

Victor Chang is with the School of Computing and Digital Technologies, Teesside University, Middlesbrough TS1 3BX, UK. (E-mail: ic.victor.chang@gmail.com).

In Bonawitz *et al.*'s [10] SMC-based privacy-preserving federated ML scheme, the pseudo-random numbers are used to shield the gradient data to be updated, and then are removed by SMC during the aggregation phase. In this scheme, the communication cost of SMC may become unacceptable as the number of trainers increases. Liu *et al.* [11] proposed a privacy-preserving deep learning scheme, where the users have to communicate with the cloud server through SMC to activate every neural node in each layer. It reduces the efficiency of communication and increases the transmission delay. In the privacy-preserving federated learning scheme proposed by Xu *et al.* [12], the trainers negotiate the parameters of ϵ-DP with the SMC to reduce the noise to a controllable size in the aggregated model. However, in SMC-based schemes, the strict requirements for data confidentiality and privacy will inevitably bring about a large amount of communication and calculation overhead.

Li *et al.* [13] proposed an ML scheme with two HE cryptosystems to protect the privacy of the users' data sets. Although it supports model training based on different public keys, too many extra encryption and decryption operations in each iteration increase the computational burden. Yang *et al.* [14] proposed a HE-based naive bayesian disease risk prediction scheme for online medical treatment. The scheme is communication efficient due to the adoption of bloom filters and polynomials. Zhang *et al.* [15] proposed a privacy-preserving disease prediction scheme based on a single layer perceptron classifier model in the medical care system. Specifically, it takes the matrices encryption to support verification. In 2019, Truex *et al.* [16] proposed a hybird privacy-preserving federated learning scheme by combining threshold HE and SMC, but it spends too much communication cost for the model accuracy. Niu *et al.* [17] designed a verifiable privacy-preserving ML prediction services for support vector machine (SVM) model. With the modified BGN [18] cryptosystem they designed, the scheme guarantees different types of privacy protection in SVM machine learning prediction services. HE-based schemes have advantages in data accuracy and computability, but some of the HE cryptosystems have the problem of ciphertext expansion, which are not efficient in communication and storage.

From the above review, most the related work focus on the privacy protection of model training, and rarely studied the security and privacy of the prediction. Besides, the centralized cloud computing structure is gradually not suitable for large-scale client applications in HCPSs of industrial 5.0. Inspired by these questions, we study the security and privacy issues of the ML prediction services in edge-enhanced HCPSs, where the edge computing is used to reduce the load and latency of cloud computing. Based on the modified Okamoto-Uchiyama (OU) cryptosystem [19], we design a verifiable privacy-preserving ML prediction scheme for edge-enhanced HCPSs, *i.e.*, VPMLP. The contributions of the paper are summarized as follows:

1) To pave the way for the verifiable privacy-preserving ML prediction scheme of the edge-enhanced HCPSs, the OU encryption [19] is appropriately modified to support the verifiability of ciphertext.

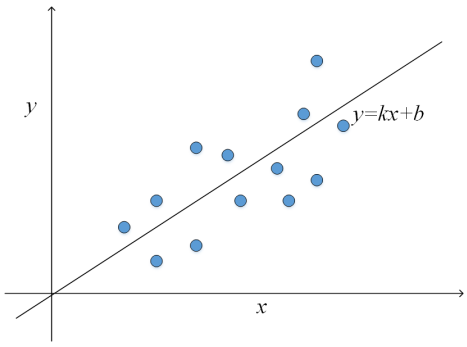


Fig. 1: An example of one-dimension linear regression

2) A verifiable privacy-preserving ML prediction scheme (VPMLP) for edge-enhanced HCPSs is designed. It can not only perform machine learning prediction in a privacy-preserving way, but also support the verifiability of prediction results.
3) The correctness analysis of VPMLP and its verification/batch verification shows that VPMLP can perform ML predict correctly. Besides, the security of VPMLP is also analyzed, and the privacy of inputs, model and results can be protected.
4) The performance of VPMLP is compared with the other four related schemes through experiments, and the results show that VPMLP is the best in term of time efficiency. In addition, the batch verification saves around 15% running time compared to the single verification.

The rest of this paper is arranged as follows. Section II introduces some preliminary knowledge used in our scheme. The system and security models are described in detail in Section III. The proposed scheme and the corresponding correctness and security analysis are give in Section III and Section V, respectively. Section VI simulates the scheme and compares it with other related schemes. Finally, Section VII draws the conclusion of the full paper.

II. Preliminaries

In this section, we briefly introduce the basic knowledge and techniques used in our scheme.

A. Linear Regression

Linear regression (LR) [20] is a kind of supervised machine learning, which is widely used in data analysis and prediction by describing the relationship between the independent variables and the dependent variables. Generally, it fits the data points with a straight line or a curve to minimize the sum of the distances between the curve and the points.

In this paper, the linear regression model is used to make predictions according to the user data. Take the one-dimension linear regression as an example, which can be seen in Fig. 1. Assuming that $y = kx + b$ is a well-trained linear regression model, where the parameters k and b reveal the relationship between the dependent and independent variables. Then, y can

be represented as the multiplication of model vector $W = \begin{bmatrix} k & b \end{bmatrix}$ and input vector $X = \begin{bmatrix} x & 1 \end{bmatrix}$:

$$y = W \cdot X^T = \begin{bmatrix} k & b \end{bmatrix} \cdot \begin{bmatrix} x \\ 1 \end{bmatrix}.$$

Therefore, given an input vector $X' = \begin{bmatrix} x' & 1 \end{bmatrix}$, the LR model outputs the prediction result y' according to $y' = W \cdot X'^T$.

B. OU Homomorphic Encryption

OU encryption [19] is a somewhat homomorphic encryption algorithm. It contains three algorithms, *i.e.*, key generation, encryption and decryption, and we briefly describe them as below.

(1) $(pk, sk) \leftarrow KeyGen(\kappa)$: Given a security parameter κ, the key generation algorithm $KeyGen(\kappa)$ outputs a key pair (pk, sk). Select two big prime numbers p, q satisfying $|p| = |q| = \kappa$ and calculate $N = p^2 q$. Then, choose $g \in \mathbb{Z}_N^*$ where the order of g^{p-1} mod p^2 is p. Calculate $h = g^N$ mod N for the convenience of encryption. It outputs the public key $pk = (N, g, h)$ and the secret key $sk = (p, q)$.

(2) $C \leftarrow Enc_{pk}(m)$: Given the public key pk and plaintext $m \in [0, 2^{\kappa-1})$, a random number $r \in \mathbb{Z}_N$ is generated, and then the plaintext can be encrypted to a ciphertext C as:

$$C = g^m \cdot h^r \bmod N.$$

(3) $m \leftarrow Dec_{sk}(C)$: Given the ciphertext C, a function $L(x) = \frac{x-1}{p}$ mod p is defined and the plaintext m can be recovered according to the secret key sk as follow:

$$\alpha = L(g^{p-1} \bmod p^2)^{-1} \bmod p,$$
$$m = L(C^{p-1} \bmod p^2) \cdot \alpha \bmod p.$$

OU encryption has the homomorphism characteristics, *i.e.*, given two ciphertexts $Enc_{pk}(m_1)$ and $Enc_{pk}(m_2)$ encrypted with the same public key pk, we have

$$Enc_{pk}(m_1) \cdot Enc_{pk}(m_2) = Enc_{pk}(m_1 + m_2),$$
$$Enc_{pk}(m_1)^{m_2} = Enc_{pk}(m_1 \cdot m_2).$$

For more information about OU encryption, please refer to [19].

C. Modified OU Encryption

To support the function of batch verification of our scheme, we modified the OU encryption (MOU), and it differs from the original OU encryption in two algorithms:

(1) $(pk, sk) \leftarrow KeyGen(\kappa)$: Besides the p, q and g, an additional small prime number s is selected, and $N = p^2 qs$ is computed. Correspondingly, the public key and secret key are $pk = (N, g, h, s)$ and $sk = (p, q)$, respectively.

(2) $C \leftarrow Enc_{pk}(m)$: two random numbers $a, b \in \mathbb{Z}_p^*$ are generated, and the plaintext m can be encrypted by MOU as $C = g^m h^{am+b}$ mod N.

III. System and Security Models

In Industry 5.0, the ML prediction in HCPSs can provide customers with high-accurate prediction services based on the trained models and their owned data. In this section, we introduce the system and security models of VPMLP for edge-enhanced HCPSs.

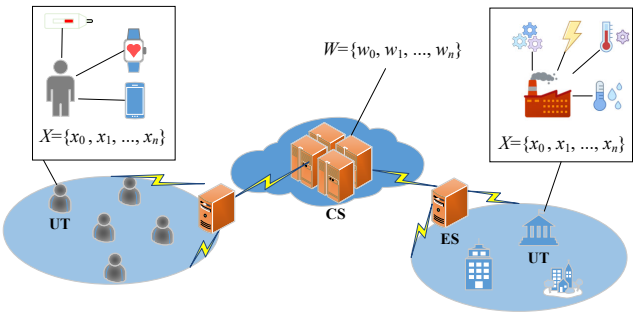


Fig. 2: System model

A. System Model

The system model of VPMLP for edge-enhanced HCPSs is shown in Fig. 2, which contains three kinds of entities, *i.e.*, cloud server (CS), edge server (ES) and the user terminal (UT).

(1) CS: The CS is an ML prediction service provider that holds a LR model $W = \{\omega_i | i \in [0, n]\}$. To provide real-time accurate prediction services, the CS keeps high frequency iterations of the model. Besides, a set of random numbers $R = \{r_i | i \in [0, n]\}$ should be generated to calculate the masked model $W' = \{w'_i = w_i + r_i | i \in [0, n]\}$, which is stored in each ES. Moreover, considering the privacy problems of the model, the number of times that each UT can query the model is limited.

(2) ES: ESs bridge UTs and the CS, and cooperate with the CS to provide the ML prediction services for the UTs. Here, the main purpose of the ESs in the architecture is to mitigate the load and response delay of the CS, and reduce the UTs' computational overhead. To achieve this purpose, each ES holds the masked model $W' = \{w'_i = w_i + r_i | i \in [0, n]\}$.

(3) UT: With the data vector $X = \{x_i | i \in [0, n]\}$ collected by IoT devices, the UT queries the CS for the prediction result $W \cdot X$, which is output by the LR model W based on the UT's data X via the corresponding ES. Meanwhile, the UT can verify the integrity of the prediction result.

B. Security Model

In this part, we introduce the adversary model and the security requirements of VPMLP for edge-enhanced HCPS.

1) Adversary Model: In VPMLP, we assume that all the entities in the HCPS are honest-but-curious, *i.e.*, they execute the operations according to VPMLP honestly, but are still curious about the privacy of other entities. Besides, the external adversaries (EAs) may want to acquire some valuable information during the execution of the scheme. Therefore, the following capabilities are defined for the malicious actions of the entities:

(1) CS: The CS is curious about UT's data vector X and prediction result $W \cdot X$. Naturally, the CS can adopt the communication messages to recover them. Moreover, the CS may collude with ESs to obtain UT's private information.

(2) ES: As the bridges between the CS and UTs, ESs are curious about the information of both parties, *i.e.*, the input vector X, the prediction result $W \cdot X$ and the model W. To acquire the privacy information above, the ESs can collude with the CS.
(3) UT: Due to the commercial or technical value, UTs are interested in the CS's prediction model W. A UT can try to recover the model based on its input data, and the corresponding prediction result $W \cdot X$ and verification code. But, in our model, we assume that a UT does not collude with other UTs or ESs.
(4) EA: As an external adversary, through the insecure network environments of HCPSs, EAs may undermine the privacy of each entity by eavesdropping or even tampering messages during the execution of the scheme.

2) Security Requirements: Based on above defined adversary model, to ensure the prediction services operating in a secure and privacy-preserving manner, VPMLP should meet the following security requirements:

(1) *Input privacy*: The input vector $X = \{x_i | i \in [0, n]\}$ is the private data of the UT. The plaintext of the input vector X must not be revealed by ESs or CS in order to protect the UT's privacy, while the UT can get the corresponding prediction result $W \cdot X$.
(2) *Model privacy*: The well-trained model W has great commercial value, so the model should be kept by the CS secretly. To ensure the privacy of the model, ESs cannot recover the model W from the masked model W', and UTs also have no way to reveal it from the prediction result or verification code.
(3) *Prediction result privacy*: The prediction result may contain some important information about the UT, which is also sensitive. To guarantee the privacy of the prediction result, the CS and ESs are not able to get the prediction result, while the UT can acquire it.
(4) *Prediction result integrity*: In the dangerous and ever-changing Internet environment, tampering attacks are unavoidable. Therefore, to make sure the integrity of the prediction results, the UTs can not only get the prediction results, but also check if they have been tampered.

IV. The proposed scheme: VPMLP

In this section, VPMLP is introduced in detail, which contains five phases, *i.e.*, system initialization, query construction, prediction, request recovery and verification phases. For the ease of understanding, the notations used in VPMLP are summarized in the TABLE I.

TABLE I: Notations

Notation	Description
κ	The security parameter
(N, g, h, s)	The public key pk
(p, q)	The secret key sk
n	The number of dimensions of the data vector
r_i	The i-th random number in R
w_i	The i-th model parameter in W
w_i'	The masked model parameter w_i
$H(\cdot)$	A hash function
x_i	The i-th element in user's data vector X
s_j	The small prime number s of the j-th UT
g_{s_j}	The generator of s_j
v_1	The part calculated locally
v_2	The part calculated by the CS
g_c, N_c	The common parameter for a batch of query
$org_a b = c$	The order of a mod b is c

A. System Initialization Phase

In this phase, each entity initializes the relevant parameters.

(1) Each UT generates a public and secret key pair $(pk, sk) \leftarrow KeyGen(\kappa)$ based on MOU, and publishes the pk.
(2) The CS generates a vector $R = \{r_i | i \in [0, n]\}$ for each version of the LR model, where all the r_i are random numbers in the range $[1, 2^{\kappa-1})$. Then, the CS masks the LR model $W = \{\omega_0, \omega_1, \ldots, \omega_n\}$ with the vector R as $W' = \{\omega_i' = \omega_i + r_i | i \in [0, n]\}$, and outsources the masked model W' to each ES.
(3) ESs store the masked model W' sent by the CS.

B. Query Construction Phase

For a data vector X, the UT performs the following steps to form a query for the prediction service.

1) In a period of time, IoT devices collect a n-dimension data vector $X = \{x_i | i \in [0, n]\}$ from the humans or physical systems, where $x_0 = 1$, and the vector X is UT's input of prediction service.
2) The UT calculates $a = H(x_0 || x_1 || \ldots || x_n)$ and chooses a random number $b \in [1, 2^{\kappa-1})$. Then, each element x_i in X is encrypted as below:

$$Enc_{pk}(x_i) = g^{x_i} h^{a x_i + b} \bmod N.$$

The UT constructs a query as $Enc_{pk}(X) = \{Enc_{pk}(x_i) | i \in [0, n]\}$ and sends it to the ES.

C. Prediction Phase

On receiving the prediction queries from UTs, the ESs and the CS perform the following processes to generate the encrypted prediction results.

(1) The ES forwards the query $Enc_{pk}(X)$ to the CS, and calculates $C_{p1} = \prod_{i=0}^{n} Enc_{pk}(x_i)^{\omega_i'}$.
(2) After receiving $Enc_{pk}(X)$, the CS computes $C_{p2} = \prod_{i=0}^{n} Enc_{pk}(x_i)^{(N - r_i)} = Enc_{pk}(\sum_{i=0}^{n}(N - r_i) \cdot x_i)$ according to $R = \{r_i | i \in [0, n]\}$, and sends C_{p2} back to the ES.
(3) The ES aggregates the C_{p1} and C_{p2} to get the prediction result in encrypted form $C_{result} = \prod_{i=0}^{n}(Enc_{pk}(x_i)^{\omega_i'} \cdot Enc_{pk}(x_i)^{N - r_i}) = Enc_{pk}(\sum_{i=0}^{n}(\omega_i + N)x_i)$, and returns C_{result} to the UT.

D. Result Recovery Phase

When receiving C_{result}, the UT decrypts it based on its secret key sk of MOU, and gets the prediction result $W \cdot X = \sum_{i=0}^{n} \omega_i x_i \bmod p = Dec_{sk}(Enc_{pk}(\sum_{i=0}^{n}(\omega_i + N)x_i))$.

E. Verification Phase

To ensure the integrity of the prediction result, VPMLP allows the UT to verify if it is tampered. Besides the normal verification, VPMLP also supports batch verification, *i.e.*, a batch of prediction results can be verified in one time. We introduce this phase as below.

1) Single Verification: For each query, the UT verifies the integrity of the prediction result as follow.

(1) The UT sends a verification request of the C_{result} to the CS, and the CS provides a verification code $V_c = h^{\sum_{i=0}^{n}\omega_i} \bmod N$ to the UT.
(2) The UT computes $C'_{result} = g^{\sum_{i=0}^{n}(\omega_i+N)x_i} \cdot h^{a\sum_{i=0}^{n}\omega_i x_i + N\sum_{i=0}^{n}(ax_i+b)} \cdot V_c^b \bmod N$.
(3) The UT checks if $C_{result} = C'_{result}$. If so, the integrity of the result is confirmed. Otherwise, the UT refuses to accept the result.

2) Batch verification: For a set of the queries sent to the same ES in a period of time, batch verification can be applied to improve efficiency. VPMLP supports two types of batch verification. In case 1, a batch of prediction results from the same UT can be verified for one time, while, in case 2, a batch of prediction results from different UTs can be verified with one verification code.

Case 1: Assuming that the UT wants to verify the integrity of a batch of t prediction results $\{Enc_{pk}(W_j \cdot X_j)|j \in [1,t]\}$ encrypted with the same random number b, where j denotes j-th query, the procedures of verification are executed as below.

(1) The UT aggregates the prediction results as:

$$C^{+}_{result} = \prod_{j=1}^{t} C_{result,j} = \prod_{j=1}^{t} Enc_{pk}(W_j \cdot X_j) \bmod N$$

Then, the UT sends the batch verification request $(\{ID_{w_j}|j \in [1,t]\}, pk)$ to the CS, where ID_{w_j} is the identification of W_j.
(2) On receiving the request, the CS calculates a verification code $V_c^{+} = h^{\sum_{j=1}^{t}\sum_{i=0}^{n} w_{j,i}} \bmod N$, and returns it back to the UT.
(3) When gets V_c^{+}, the UT computes $C^{+''}_{result} = g^{\sum_{j=1}^{t} W_j \cdot X_j} h^{b(n+1)Nt + \sum_{j=1}^{t} a_j (W_j \cdot X_j + \sum_{i=0}^{n} x_i)} \bmod N$ and $C^{+'}_{result} = C^{+''}_{result} \cdot (V_c^{+})^b \bmod N$.
(4) The UT checks whether $C^{+}_{result} = C^{+'}_{result}$. If they are equal, the UT confirms the integrity of all the prediction results in this batch. Otherwise, some of the prediction results in this batch must be tampered.

Case 2: Let U denote a set of t UTs, who send the verification requests to a ES so that they can verify a batch of prediction results based on the same model. In this case, based on UTs' verification requests, a common verification request is formed by the ES and forwarded to the CS. Accordingly, the CS generates a common verification code. At last, after the ES transforms the common code into the local codes, each prediction result of UT can be verified.

To form the common request, the following algorithm $ComGen(\cdot)$ is executed between the ES and UTs.

(1) Each $UT_j \in U$ chooses a generator g_{s_j} of $\mathbb{Z}^*_{s_j}$, and sends the verification request $(g_{s_j}, pk_j = (N_j, g_j, h_j, s_j))$ to the ES. Specifically, to ensure the correctness of batch verification, any two s_js in the batch should be different according to the Chinese Remainder Theory.
(2) On receiving the requests, the ES computes:

$$N_c = \prod_{j=1}^{t} s_j,\ g_c = \sum_{j=1}^{t} M_j M'_j g_{s_j},$$

where $M_j = \frac{N_c}{s_j}$ and $M'_j = M_j^{-1} \bmod s_j$.
(3) Then, the ES sends a common verification request (g_c, N_c) to the CS.

Once receiving (g_c, N_c), the CS calculates a common verification code $V_c^* = g_c^{\sum_{i=0}^{n} w_i} \bmod N_c$, and sends it back to the ES. After receiving V_c^*, the ES executes the following processes to transform the common code into the local codes:

(1) The ES solves a small-scale discrete problem with baby-step-giant-step algorithm [21] for each UT_j, *i.e.*, finds a $k_j \in \mathbb{Z}^*_{s_j}$ satisfying $g_{s_j}^{k_j} = g_j \bmod s_j$.
(2) The ES computes the local verification code $V^*_{c,j} = (V_c^*)^{k_j} \bmod s_j$ for each UT_j respectively, and sends it to the corresponding UT_j.

On receiving the code $V^*_{c,j}$ from the ES, to verify the integrity of the prediction result $C_{result,j} = g_j^{\sum_{i=0}^{n}(w_i+N_j)x_{j,i}} h_j^{\sum_{i=0}^{n}(w_i+N)(a_j x_{j,i}+b_j)} \bmod s_j$, each $UT_j \in U$ calculates $C'_{result,j} = g^{\sum_{i=0}^{n}(w_i+N)x_{j,i}} h^{b(n+1)N_j + \sum_{i=0}^{n}(w_i+N_j)x_{j,i}} (V^*_{c,j})^{b_j} \bmod s_j$. If $C_{result,j} = C'_{result,j}$, the UT outputs $flag_j = True$, and otherwise, $flag_j = False$. Only if all $UT_j \in U$ output $flag_j = True$, where $j \in [1,t]$, the integrity of this batch of queries are confirmed.

V. Theoretical Proof

In this section, we first prove the correctness of VPMLP, and then show that VPMLP achieves the aforementioned security requirements.

A. Correctness

We prove the correctness of the prediction service and the verification of VPMLP as below.

1) Correctness of Prediction Service: Since the VPMLP is based on MOU, the correctness of MOU can ensure the correctness of VPMLP. As introduced in Section II, the algorithms $KeyGen(\cdot)$ and $Enc_{pk}(\cdot)$ in MOU are modified. To prove the correctness of MOU, we first prove the correctness of the encryption and decryption algorithms, and then show that MOU also maintains homomorphic characteristics.

Given a key pair (pk, sk) generated by $KeyGen(\cdot)$ of MOU and a ciphertext $C = g^m h^{ax_i+b} \bmod N$ encrypted with pk, let m' denote the output of $Dec_{sk}(C)$. Then we have

$$\begin{aligned} m' =& Dec_{sk}(C) \\ =& \frac{(g^{m(p-1)}h^{(am+b)(p-1)}) - 1 \bmod p^2}{g^{p-1} - 1 \bmod p^2} \bmod p. \end{aligned} \tag{1}$$

Because $ord_{g^{p-1}}p^2 = p$, we can obtain $(1 + rp)^N = 1 \bmod p^2$, where $r \in \mathbb{Z}_N^*$. Then, Eq. (1) can be transformed into

$$m' = \frac{(1 + rp)^m(1 + rp)^{N(am+b)} - 1 \bmod p^2}{(1 + rp) - 1 \bmod p^2} \bmod p. \tag{2}$$

Some fractions in Eq. (2) can be replaced by polynomial expansions. So we have

$$(1 + rp)^m = 1 + mpr \bmod p^2, \tag{3}$$

$$(1 + rp)^{N(am+b)} \bmod p^2 = 1 \bmod p^2. \tag{4}$$

By bringing Eq. (3) and Eq. (4) into Eq. (2), we can get

$$m' = \frac{(1 + mrp) - 1 \bmod p^2}{rp \bmod p^2} \bmod p = m \bmod p.$$

Thus, the ciphertext of MOU can be decrypted correctly.

Besides the above features, the MOU also inherits the homomorphism characteristics from OU encryption, *e.g.*, given two ciphertexts $Enc_{pk}(m_1), Enc_{pk}(m_2)$ under the same public key, we have

$$\begin{aligned} & Enc_{pk}(m_1) \cdot Enc_{pk}(m_2) \\ =& g^{m_1}h^{am_1+b_1} \cdot g^{m_2}h^{am_2+b_2} \bmod N \\ =& g^{m_1+m_2}h^{a(m_1+m_2)+b_1+b_2} \bmod N \\ =& Enc_{pk}(m_1 + m_2) \end{aligned}$$

and

$$\begin{aligned} & Enc_{pk}(m_1)^{m_2} \\ =& (g^{m_1}h^{am_1+b_1})^{m_2} \bmod N \\ =& g^{m_1m_2}h^{am_1m_2+bm_2} \bmod N \\ =& Enc_{pk}(m_1 \cdot m_2). \end{aligned}$$

Therefore, the MOU also has the homomorphism characteristics.

Based on the correctness of MOU and its homomorphism characteristics, the correctness of VPMLP can be ensured.

2) Correctness of Verification: In this section, we prove the correctness of the verification phase in VPMLP, including single verification and two types of batch verification.

Single Verification: To verify the prediction result $C_{result} = g^{\sum_{i=0}^{n}(w_i+N)x_i}h^{\sum_{i=0}^{n}(w_i+N)(ax_i+b)} \bmod N$ with a verification code V_c provided by the CS, the UT computes

$$\begin{aligned} & C'_{result} \\ =& g^{\sum_{i=0}^{n}(\omega_i+N)x_i} \cdot h^{a\sum_{i=0}^{n}\omega_i x_i + N\sum_{i=0}^{n}(ax_i+b)} \cdot (V_c)^b \bmod N \\ =& g^{\sum_{i=0}^{n}(\omega_i+N)x_i}h^{a\sum_{i=0}^{n}\omega_i x_i + b\sum_{i=0}^{n}w_i + N\sum_{i=0}^{n}(ax_i+b)} \bmod N \\ =& g^{\sum_{i=0}^{n}(w_i+N)x_i}h^{\sum_{i=0}^{n}(w_i+N)(ax_i+b)} \bmod N. \end{aligned}$$

Therefore, if the prediction result is not tampered, we have $C_{result} = C'_{result}$, and the single verification of VPMLP is correct.

Batch Verification in Case 1: To verify a batch of t queries from the same UT, the UT first constructs $C^+_{result} = g^{\sum_{j=1}^{t}\sum_{i=0}^{n}(w_{j,i}+N)x_{j,i}}h^{\sum_{j=1}^{t}\sum_{i=0}^{n}(w_{j,i}+N)(a_ix_{j,i}+b)} \bmod N$. Then, with the batch verification code V_c^+, the UT computes

$$\begin{aligned} & C^{+'}_{result} \\ =& C^{+''}_{result} \cdot (V_c^+)^b \bmod N \\ =& g^{\sum_{j=1}^{t}\sum_{i=0}^{n}(w_{j,i}+N)x_{j,i}}h^{b(n+1)Nt+\sum_{j=1}^{t}a_j(W_j\cdot X_{j,i}+\sum_{i=0}^{n}x_{j,i})}. \\ & h^{\sum_{j=1}^{t}\sum_{i=0}^{n}w_{j,i}} \bmod N \\ =& g^{\sum_{j=1}^{t}\sum_{i=0}^{n}(w_{j,i}+N)x_{j,i}}h^{\sum_{j=1}^{t}\sum_{i=0}^{n}(w_{j,i}+N)(a_ix_{j,i}+b)} \bmod N. \end{aligned}$$

Therefore, $C^+_{result} = C^{+'}_{result}$ if none of the prediction results have been tampered, and the correctness of batch verification in case 1 is ensured.

Batch Verification in Case 2: To verify a batch of t queries from different UTs, firstly, the common verification code V_c^* should be transformed into UT_j's local code $\{V^*_{c,j}|j \in [1, t]\}$ as:

$$\begin{aligned} V^*_{c,j} =& (V_c^*)^{k_j} \bmod s_j \\ =& (g_c^{\sum_{i=0}^{n}w_i})^{k_j} \bmod s_j \\ =& g_{s_j}^{k_j\sum_{i=0}^{n}w_i} \bmod s_j \\ =& g_j^{\sum_{i=0}^{n}w_i} \bmod s_j \end{aligned}$$

where the Chinese Remainder Theory ensures $g_{s_j} \equiv g_c \bmod s_j$ for each $UT_j \in U$. Additionally, because all $g_{s_j} \in \mathbb{Z}^*_{s_j}$ are generators, it ensures that for any $UT_j \in U$, there is a k_j that satisfies $g_{s_j}^{k_j} = g_j \bmod s_j$.

To locally verify the prediction result $C_{result,j} = g_j^{\sum_{i=0}^{n}(w_i+N_j)x_{j,i}}h_j^{\sum_{i=0}^{n}(w_i+N)(a_jx_{j,i}+b_j)} \bmod s_j$ with the local code $V^*_{c,j}$, each $UT_j \in U$ computes $C'_{result,j}$ as:

$$\begin{aligned} & C'_{result,j} \\ =& g_j^{\sum_{i=0}^{n}(\omega_i+N_j)x_{j,i}} \cdot h_j^{a_j\sum_{i=0}^{n}\omega_i x_{j,i} + N_j\sum_{i=0}^{n}(a_jx_{j,i}+b_j)}. \\ & (V_{c,j})^{b_j} \bmod s_j \\ =& g_j^{\sum_{i=0}^{n}(\omega_i+N_j)x_{j,i}}. \\ & h_j^{a_j\sum_{i=0}^{n}\omega_i x_{j,i} + b\sum_{i=0}^{n}x_i + N_j\sum_{i=0}^{n}(a_jx_{j,i}+b_j)} \bmod s_j \\ =& g_j^{\sum_{i=0}^{n}(\omega_i+N_j)x_{j,i}} \cdot h_j^{\sum_{i=0}^{n}(\omega_i+N_j)(a_jx_{j,i}+b_j)} \bmod s_j \end{aligned}$$

We have $C_{result,j} = C'_{result,j}$ if none of the prediction results of $UT_j \in U$ have been tampered. Therefore, the correctness of batch verification in case 2 is guaranteed.

In summary, the three types of integrity verification in VPMLP are correct.

B. Security Analysis

Based on the OU encryption, we designed the MOU to support VPMLP. We prove the security of MOU and VPMLP in the section.

1) Security of MOU: **Theorem 1.** Cracking the ciphertext of MOU encryption is computationally infeasible if and only if Integer Factorization Problem (IFP) is computationally infeasible.

By calculating $pk = (N' = N/s, g, h = g^{N'})$ and $C = (g^m h^{am+b} \bmod N) \bmod N'$, the public key and

ciphertext of MOU can be transformed into that of OU. If cracking the OU encryption is computationally infeasible, the MOU encryption is also computationally infeasible at the same scale. In the following part of this section, the ciphertexts and public keys mentioned are generated by the algorithms from OU cryptosystem.

Definition 1. Define a kind of factorization problem that given (N', κ), find prime numbers (p, q) to satisfy $N' = p^2 q$. If for a large enough κ and any algorithm A can solve the problem $(p, q) \leftarrow A(N', \kappa)$ in polynomial time with negligible probability ϵ, i.e., $Pr[A(1^\kappa, N') = (p, q)] < \epsilon$, the factorization problem is computationally infeasible.

Definition 2. Given the set of public key $\{pk_j = (N', g_j)\}$ generated by $KeyGen(\kappa)$ and the ciphertext $C = Enc_{pk}(m) \bmod N'$. If any algorithm Adv can only output the correct plaintext m in polynomial time with a negligible probability $Pr[Adv(1^\kappa, N', g, C) = m] < \epsilon$, cracking the encryption function is computationally infeasible.

Proof: Assume that given the public key set $\{pk_j = (N', g_j)\}$ and the ciphertext set $\{C\}$ where the elements are encrypted with the corresponding public key in $\{pk_j\}$, algorithm Adv can output the correct m within polynomial time with non-negligible probability.

Assume that the factorization problem defined above is computationally feasible, and that MOU is not secure, which means the algorithm Adv can calculate m from $C = Enc_{pk}(m)$ with non-negligible probability. So, we can construct an algorithm A to solve the factorization problems as below under the help of Adv with non-negligible probability in polynomial time.

(1) For a N', A randomly generates the different $g_j \in \mathbb{Z}^*_{N'}$, where g_j satisfies $ord_{g_j^{p-1}} p^2 = p$ with high probability $(p-1)/p$, and generates a set of public key $\{pk_j = (N', g_j)\}$.

(2) As the ciphertext can be presented as $C = g^{m+N'(am+b)} = g^z \bmod N'$, A uniformly chooses $z' \in \mathbb{Z}_{N'}$ and computes a set of the pk_j and ciphertext pair $\{(pk_j, C'_j = g_j^{z'} \bmod N')\}$.

(3) Define $ord_{g_j} p^2 = pp'$ and $ord_{g_j} q = q'$, *i.e.*, $p' | p-1$ and $q' | q-1$. The distribution of correct ciphertext C can be expressed as $[z_1 \equiv z \bmod p, z_2 \equiv z \bmod lcm(p', q')]$, where $gcd(p, lcm(p', q')) = 1$ and $gcd(q, lcm(p', q')) = 1$. Similarly, for the distribution $[z'_1 \equiv z' \bmod p, z'_2 \equiv z' \bmod lcm(p', q')]$ of the forged ciphertext C'_j, if $z'_1 = z_1$, the distribution of z_2, z'_2 will be statistically close. In this way, Adv can output the corresponding plaintext m according to the forged ciphertext $C'_j = g^{z'} \bmod N'$ with non-negligible probability in polynomial time.

(4) For the plaintext m and $C'_j = g^{z'} \bmod N'$, it should satisfy that $m < 2^{\kappa-1}$ and $z' \equiv m \bmod p$. Because $z' \in \mathbb{Z}_{N'}$ is selected from uniform distribution, $z' \geq 2^{\kappa-1}$ with a high probability. Then, A can obtain the answer of $gcd(z' - m, N')$, which is one of p, p^2 and pq. Because $gcd(z' - m, N')$ is a multiple of p, we can infer that $(z' - m) < N'$ and obtain one factor directly (when $gcd(z - m, N') = p$), or by dividing N' with $gcd(z - m, N')$, *i.e.*, $q = N'/p^2$ or $p = N'/pq$.

Hence, algorithm A can solve the factorization problem with a non-negligible possibility in polynomial time. However, the widely acknowledged truth is that IFP is computationally infeasible, which means that the assumptions are quite opposite, *i.e.*, cracking OU encryption is computationally infeasible, so, cracking MOU encryption is also computationally infeasible and as hard as IFPs. For more detailed information about the proof, please refer to [19].

2) Input Privacy: To ensure the privacy of the input, the UT first encrypts the input data X with the public key pk of the MOU, and sends the encrypted request $Enc_{pk}(X)$ to the ES and the CS for the prediction result. The only way to reveal the plaintext input X is to decrypt the ciphertext $Enc_{pk}(X)$. However, according to the above analysis, the security of the MOU is equivalent to solving large integer factorization problem $N = p^2 qs$. Without the corresponding secret key sk, any external adversary EA or even the ES colluding with the CS, cannot recover the plaintext with non-negligible probability when N is big enough. Therefore, no other entity can reveal UT's input data X, and the input privacy can be well protected.

3) Model Privacy: In VPMLP, the model W is related to the masked model W', the prediction result $W \cdot X$ and verification code $V_c = g^{\sum_{i=0}^{n} w_i}$. As we can see from the description of VPMLP, the masked model W' is stored in ESs. Since $W' = \{w_i + r_i | i \in [0, n]\}$ is masked by the random number set $R = \{r_i | i \in [0, n]\}$, the ESs have no way to get the original model W without knowing the random number set R. Besides, as defined in the adversary model and the description of VPMLP, no collusion among the UTs, and each UT just can request a certain number of queries. Therefore, the UTs also cannot figure out the original model W. Furthermore, if an adversary gets the verification code $V_c = h^{\sum_{i=0}^{n} w_i}$, the $\sum_{i=0}^{n} w_i$ is protected by the intractability of discrete logarithm problem, and W also cannot be revealed by the adversary. In a word, VPMLP guarantees the model privacy.

4) Result Privacy: As we can see from the description of VPMLP, the prediction result $C_{reslut} = Enc_{pk}(\sum_{i=0}^{n}(\omega_i + N)x_i)$ is returned to the UT when UT initiates a query, which is generated based on both ES's contribution C_{p1} and CS's contribution C_{p2}. The generation of the prediction result utilizes the homomorphic properties of MOU, and any external adversary EA or even the CS colluding with the ES, cannot reveal the final prediction result $W \cdot X$ without knowing sk. Therefore, VPMLP ensures the privacy of prediction results.

5) Result Integrity: As we can see from the description of the scheme, based on the homomorphic properties of the MOU, the processes of VPMLP are executed in the ciphertext form. The integrity verification is an important security requirement for VPMLP, which allows the UT to check if the encrypted prediction result has been modified by the adversary. As defined in the adversary model, the CS and ESs execute the scheme honestly, so we just consider the tampering attacks launched by the external adversary EA here. Based on the correctness analysis of verification, we can know that if the external adversary EA tampers the encrypted prediction result C_{result}, it will be checked by the UT since $C_{result} \neq C'_{result}$.

Therefore, VPMLP ensures the integrity of prediction results.

VI. Performance Evaluation

The performance is an important factor in evaluating the services of HCPSs, which reflects the throughputs and efficiency of the systems to a certain extent. In this section, we analyze the performance of VPMLP from the complexity and the simulation experiments, and compare the prediction service of VPMLP with other related schemes, *i.e.*, PPML [22], SML [23], PPEL [24] and MVP [17]. Meanwhile, we evaluate the computational efficiency improvement of the batch verification compared to the single verification. Besides, the simulations of these schemes are implemented in python 3.7 on Linux platform with quad-core i7 7500U 2.7GHz and 24G RAM.

A. Complexity

As we know, the computational complexity of HE-based privacy-preserving ML schemes partially depends on the efficiency of the used cryptographic algorithms. Therefore, we can roughly evaluate the efficiency of these schemes through them. To facilitate the comparison of different schemes, we select three security levels according to the recommended reference key length standard [25] in the simulations, which is shown in TABLE II. In these schemes, PPML [22] , SML [23], PPEL [24] are based on Paillier [26], while MVP [17] and VPMLP are respectively based on modified BGN (MBGN) [17] and MOU. For these basic cryptosystems, the execution times of their operations are tested and shown in TABLE III.

TABLE II: Security standard of key length

Security Level	**80-bits (SL1)**		**112-bits (SL2)**		**128-bits (SL3)**	
Integer Factoring	1024		2048		3072	
Discrete Logarithm	Key 160	Group 1024	Key 224	Group 2048	Key 256	Group 3072

From the TABLE III, we can see that the encryption and decryption operations in all three cryptosystems are much more time-consuming than homomorphic addition and somewhat homomorphic multiplication operations. In addition, the encryption and decryption performance of MOU is better than that of Paillier in all three security levels, and the execution time of encryption in MOU and Paillier is much more than the time of decryption. In MBGN, the encryption operations are highly efficient with the help of professional optimization of cryptographic library PBC, while the self-designed decryption operations consume a huge amount of the execution time.

In order to approximately evaluate the efficiency of these schemes, TABLE IV shows the amount of homomorphic operations required to provide a prediction service in each scheme. Combining TABLE III and TABLE IV, VPMLP is quite competitive among the five schemes in time complexity. Among the five schemes, VPMLP and PPEL[24] have the least numbers of encryption and decryption operations, which saves a large percentage of time. MVP [17] also has the least numbers of encryption and decryption operations, however, the decryption of MVP [17] is inefficient, which takes far more execution time than that in MOU and Paillier [26]. In PPML [24] and SML [23], two non-collusive servers are used to protect model privacy, so more additional computational overhead is required to complete the task. In theoretical analysis, VPMLP and PPEL [24] have lower-level time consumption compared to other schemes.

TABLE IV: Comparison of homomorphic operation

Protocol	**Enc**	**Dec**	**Add**	**Mul**
VPMLP	n	1	$2n+1$	$2n$
PPML [22]	$3n+2$	2	$2n+2$	$2n$
SML [23]	$2n$	1	$4n$	$4n$
PPEL [24]	n	1	n	n
MVP [17]	n	1	n	n

Another aspect of complexity is communication efficiency. The communication cost of each entity in these schemes is shown in TABLE V, where n and κ denote the size of the query vector and the key length in each scheme, respectively. Besides, the length of the ciphertext in each scheme is also considered, *e.g.*, the ciphertext length of MBGN [17], MOU and Paillier [26] are $2\kappa, 3\kappa$ and 4κ, respectively. From the TABLE V, we can see that VPMLP is better than PPML [22] and SML [23] in communication efficiency, and not as good as PPEL [24] and MVP [17], but it is still at a relatively reasonable level. The extra communication cost of VPMLP is caused by the adoption of the edge-enhanced architecture, which reduces the burden of the CS significantly. Comprehensively, VPMLP has advantages in time and communication.

TABLE V: Communication cost

Protocol	UT	CS/**Server**	ES	**Total**
VPMLP	$3n\kappa$	3κ	$3(n+1)\kappa$	$(6n+6)\kappa$
PPML [22]	$8n\kappa$	$(8n+16)\kappa$		$(16n+16)\kappa$
SML [23]	$8n\kappa$	$(8n+8)\kappa$		$(16n+8)\kappa$
PPEL [24]	$4n\kappa$	4κ		$(4n+4)\kappa$
MVP [17]	$2n\kappa$	2κ		$(2n+2)\kappa$

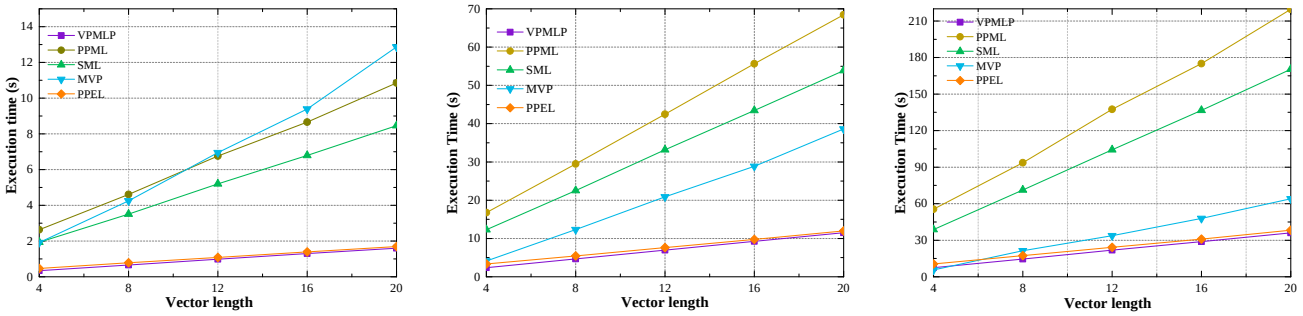
B. The Experiment

In order to show the computational efficiency of each scheme more intuitively, we simulate the time cost of performing one prediction service for all schemes, and also evaluate the efficiency improvement of the batch verification in VPMLP.

1) Execution Time Comparison: The simulations are executed at the different security levels with different sizes of the query vectors, and the result is shown in Fig. 3. Because some of the schemes are designed for neural network services, for the sake of fairness, these schemes are adjusted to the same scale of LR in the same form. Figures 3(a), 3(b) and 3(c) reflect the similar tendencies on the execution time of these schemes under different security levels. We can see that VPMLP and PPEL [24] are the most effective schemes, which is consistent with the analysis in the complexity part. Compared with VPMLP and PPEL [24], the execution time of

TABLE III: The execution time for the homomorphic operations (ms)

	80-bits (SL1)				112-bits (SL2)				128-bits (SL3)			
	Enc	Dec	Add	Mul	Enc	Dec	Add	Mul	Enc	Dec	Add	Mul
MOU	183.53	14.74	0.03	0.45	1345.44	107.89	0.10	1.68	4246.16	324.85	0.20	3.47
Paillier [26]	219.19	107.83	0.05	0.82	1498.88	775.58	0.18	2.98	4793.79	2397.39	0.34	5.91
MBGN [17]	4.29	4047.68	0.04	0.62	8.84	7566.62	0.08	1.35	15.63	15173.78	0.13	2.30



(a) Execution time at security level 1 (b) Execution time at security level 2 (c) Execution time at security level 3

Fig. 3: Execution time of the schemes

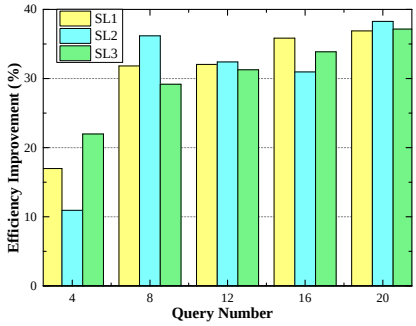


Fig. 4: Efficiency improvement of batch verification in case 1

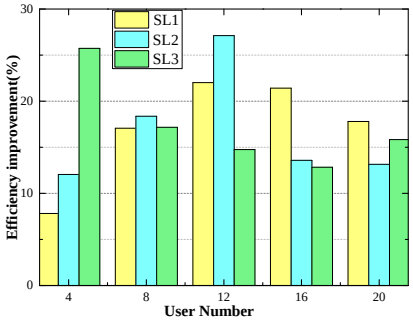


Fig. 5: Efficiency improvement of batch verification in case 2

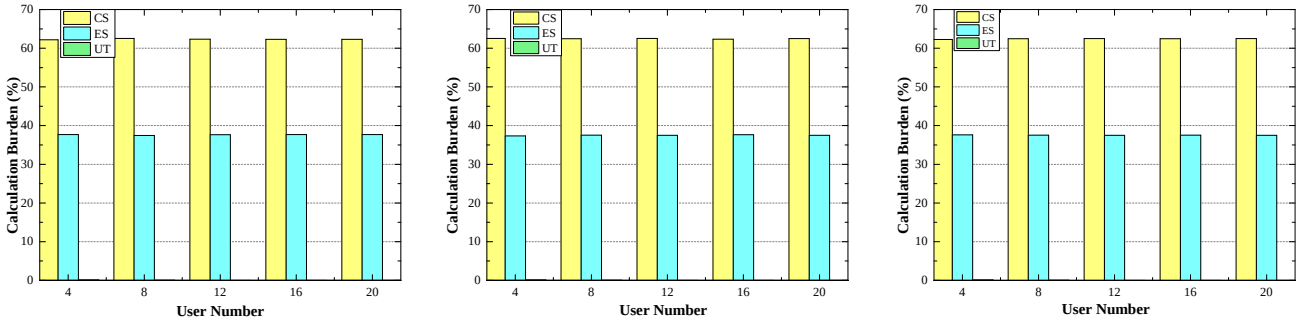
PPML [22] and SML [23] grows rapidly as the vector length increases. While the efficiency of MVP [17] is not influenced so much by the key length. As the security level gets higher, the performance of MVP [17] becomes closer to VPMLP and PPEL [24] from the one with the lowest efficiency in security level 1.

2) Efficiency Improvements of Batch Verification: To present the advantages of batch verification, we compare computational efficiency of batch verification with the single verification, and illustrate the efficiency improvements of batch verification in Fig. 4 and Fig. 5. In case 1, a verification request for a batch of prediction results from one UT is sent to the CS, and the CS returns the verification code. While in case 2, the system provides prediction services for different UTs under the same ES, which is a quite common architecture in reality. From Fig. 4, we can see that the batch verification in case 1 enhances almost 30% of computational efficiency. As shown in Fig. 5, the efficiency improvement in case 2 may not be very significant, because the transformation from the common code to local codes takes some time. But it still saves the communication cost between the CS and ESs, and improves global efficiency by 10% to 20%.

Furthermore, to show the utility of edge computing architecture, the computational load ratio of each entity in prediction and the case 2 of the batch verification processes are counted in Fig. 6. It is simulated under the edge-enhanced architecture with one CS and 32 ESs, where each ES manages the same number of UTs. From Fig. 6, we can see that the ESs share a big percentage of burden from the CS, and the CS and ESs undertake nearly 98% calculation burden. With the edge computing architecture, the system can be more stable and capable to provide personalized services for customers. Therefore, the edge computing is very suitable for the HCPSs applications in Industry 5.0.

VII. Conclusion and Future Work

The ML prediction is one of the key techniques to realize the personalization of Industry 5.0. To support the prediction services in a privacy-preserving and secure manner, we propose a verifiable privacy-preserving ML prediction scheme (VPMLP) for edge enhanced HCPSs based on modified OU cryptosystem. To further improve the performance, we design a batch verification algorithm, which can verify a batch of prediction results with a common verification code. We



(a) Burden percentage at security level 1 (b) Burden percentage at security level 2 (c) Burden percentage at security level 3

Fig. 6: Burden percentage of each entity in VPMLP

theoretically prove the correctness of VPMLP, and the security analysis shows that it guarantees the privacy of multiple parties and even the integrity of the final prediction results. From the experiments and analysis, we can see its advantages on comprehensive performance. Compared with the other four schemes, VPMLP shows high-efficiency in complexity and performance. Especially, the edge computing architecture of VPMLP saves the communication costs of the core networks and alleviates the calculation burden of the cloud in HCPSs.

With the rapid development of information technology and industry, the tendency of personalization in Industry 5.0 is unstoppable. As a highly integrated system of advanced productivity, ML prediction services in HCPSs have great potential to realize personalization. In future work, we will try to design privacy-preserving mechanisms for more complex ML functions and algorithms, *e.g.*, convolutional neural network and memory networks, and make more contributions to the HCPSs and Industry 5.0.

REFERENCES

[1] A. Čolaković and M. Hadžialić, "Internet of things (iot): A review of enabling technologies, challenges, and open research issues," *Computer Networks*, vol. 144, pp. 17–39, 2018.

[2] J. Hegde and B. Rokseth, "Applications of machine learning methods for engineering risk assessment–a review," *Safety science*, vol. 122, p. 104492, 2020.

[3] M. M. Othman and A. El-Mousa, "Internet of things & cloud computing internet of things as a service approach," in *2020 11th International Conference on Information and Communication Systems (ICICS)*. IEEE, 2020, pp. 318–323.

[4] J. Zhou, Y. Zhou, B. Wang, and J. Zang, "Human–cyber–physical systems (hcpss) in the context of new-generation intelligent manufacturing," *Engineering*, vol. 5, no. 4, pp. 624–636, 2019.

[5] I. Sittón-Candanedo, R. S. Alonso, J. M. Corchado, S. Rodríguez-González, and R. Casado-Vara, "A review of edge computing reference architectures and a new global edge proposal," *Future Generation Computer Systems*, vol. 99, pp. 278–294, 2019.

[6] D. Zhang, X. Chen, D. Wang, and J. Shi, "A survey on collaborative deep learning and privacy-preserving," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2018, pp. 652–658.

[7] X. Zhang, S. Ji, H. Wang, and T. Wang, "Private, yet practical, multiparty deep learning," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 1442–1452.

[8] P. Li, T. Li, H. Ye, J. Li, X. Chen, and Y. Xiang, "Privacy-preserving machine learning with multiple data providers," *Future Generation Computer Systems*, vol. 87, pp. 341–350, 2018.

[9] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, "A trustworthy privacy preserving framework for machine learning in industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6092–6102, 2020.

[10] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[11] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 619–631.

[12] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 13–23.

[13] 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.

[14] X. Yang, R. Lu, J. Shao, X. Tang, and H. Yang, "An efficient and privacy-preserving disease risk prediction scheme for e-healthcare," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3284–3297, 2018.

[15] C. Zhang, L. Zhu, C. Xu, and R. Lu, "Ppdp: An efficient and privacy-preserving disease prediction scheme in cloud-based e-healthcare system," *Future Generation Computer Systems*, vol. 79, pp. 16–25, 2018.

[16] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 1–11.

[17] C. Niu, F. Wu, S. Tang, S. Ma, and G. Chen, "Toward verifiable and privacy preserving machine learning prediction," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[18] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of cryptography conference*. Springer, 2005, pp. 325–341.

[19] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *International conference on the theory and applications of cryptographic techniques*. Springer, 1998, pp. 308–318.

[20] S. Weisberg, *Applied linear regression*. John Wiley & Sons, 2005, vol. 528.

[21] A. M. Odlyzko, "Discrete logarithms in finite fields and their cryptographic significance," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1984, pp. 224–314.

[22] X. Ma, X. Chen, and X. Zhang, "Non-interactive privacy-preserving neural network prediction," *Information Sciences*, vol. 481, pp. 507–519, 2019.

[23] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 19–38.

[24] S. Kuri, T. Hayashi, T. Omori, S. Ozawa, Y. Aono, L. Wang, S. Moriai *et al.*, "Privacy preserving extreme learning machine using additively homomorphic encryption," in *2017 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2017, pp. 1–8.

[25] Damien Giry, "Cryptographic keylength recommendation, Subtitle," 2017, https://www.keylength.com/en/4/, Last accessed on 2021-6-25.

[26] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International conference on the theory and applications of cryptographic techniques*. Springer, 1999, pp. 223–238.