Privacy-Enhanced Healthcare Monitoring Service Refreshment in Human Digital Twin-Assisted Fabric Metaverse

Yu Qiu, Min Chen, *Fellow, IEEE*, Weifa Liang, *Senior Member, IEEE*, Lejun Ai, Dusit Niyato, *Fellow, IEEE*, and Gang Wei

***Abstract*—Human digital twin bridges humans with digital avatars in the fabric metaverse, assisting users and healthcare professionals with real-time visualization, analysis, and prediction of personal data sensed by fabric sensors. The human digital twin-assisted healthcare monitoring (HHM) service refreshment refers to sending personal health data to corresponding services hosted on nearby edge servers and receiving the results to update local digital avatars continuously. However, the malicious nature and resource limitations of edge servers may lead to user privacy leaks and refreshment timeout, thereby impacting diagnostics. In this paper, we investigate a novel privacy-enhanced HHM service refreshment maximization problem in the fabric metaverse by considering privacy data encryption, model compression, and personalized user requirements. To this end, we first formulate the above issue as an Integer Linear Programming (ILP) problem, and prove its NP-hardness. Then, a resource scheduler named Wiper is designed, consisting of a shallow-deep distiller and an agile refresher library. To enable efficient inference while preserving user privacy, the former replaces violation modules in existing models with approximations and conducts shallow distillation on model layers to meet operation type and depth limits of homomorphic encryption, and then deep distillation on model parameters to decrease end-to-end refreshment delay. Finally, to satisfy user requirements on accuracy and delay during encrypted refreshments while maximizing the throughput of HHM services in offline and online situations with different problem scales, a series of HHM service refreshment algorithms are merged into the latter, including exact, performance-guaranteed approximation, and residual diffusion reinforcement learning algorithms. Theoretical analyses and experiments demonstrate that our algorithms are promising compared with baseline algorithms.**

***Index Terms*—Metaverse, human digital twin, edge computing, healthcare monitoring, service refreshment, accuracy, privacy.**

I. Introduction

FABRIC metaverse shatters traditional time-space constraints in healthcare systems, enabling a human-centered healthcare industry through continuous information exchange among individuals, fabric sensors embedded in living environments, and medical institutions. As a key technology within the fabric metaverse, human digital twin seamlessly bridges the physical and digital worlds, and allows users to experience personalized healthcare services by manipulating a digital proxy known as an avatar. Indeed, the digital avatar is a comprehensive digital representation of an individual with various physical and physiological attributes, displaying current health status and providing extensive historical data for long-term predictive analysis, such as health recommendations, anomaly detection, and interventions [19], [27], [36].

Regularly refreshing the health status of digital avatars to align with their real-world users is vital for ensuring the accuracy and timeliness of diagnosis through human digital twin-assisted healthcare monitoring (HHM) services. Initially, various physical and physiological signals from individuals, such as pressure, electroencephalogram, respiratory sound, and speaking voice, are gathered by various sensors, where flexible fabric sensors are preferred for long-term data acquisition due to their imperceptibility and portability [5], [29]. Following this, the collected personal health data is transmitted to nearby edge servers hosting relevant HHM services empowered by deep neural network (DNN), including sitting posture analysis, lung disease detection, and emotion recognition, for preliminary AI analysis or diagnosis. Finally, the results are relayed back to wearable devices of users to update the real-time health status of their digital avatars, and virtualized in fabric metaverses.

Received 14 January 2025; revised 10 May 2025; accepted 17 June 2025. Date of publication 23 June 2025; date of current version 3 October 2025. This work was supported by in part the Major Research Project of the National Social Science Foundation of China under Grant 23&ZD215, in part by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme under Grant FCP-NTU-RG-2022-010 and Grant FCP-ASTAR-TG-2022-003, in part by the Singapore Ministry of Education (MOE) Tier 1 under Grant RG87/22 and Grant RG24/24, in part by the NTU Centre for Computational Technologies in Finance (NTUCCTF), in part by RIE2025 Industry Alignment Fund - Industry Collaboration Projects (IAF-ICP) under Award I2301E0026, administered by A*STAR, and in part by Alibaba Group NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL). The work of Min Chen was supported by the National Natural Science Foundation of China under Grant 62276109. The work of Weifa Liang was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China with Grant CityU 11202723, Grant CityU 11202824, Grant 7005845, Grant 8730094, and Grant 9380137. Recommended for acceptance by W. Cai. *(Corresponding authors: Min Chen; Gang Wei.)*

Yu Qiu, Min Chen, and Lejun Ai are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China, and also with Pazhou Laboratory, Guangzhou 510640, China (e-mail: csqiuyu@mail.scut.edu.cn; minchen@ieee.org; cs_ailejun@mail.scut.edu.cn).

Weifa Liang is with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: weifa.liang@cityu.edu.hk).

Dusit Niyato is with the College of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: DNIYATO@ntu.edu.sg).

Gang Wei is with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 511442, China (e-mail: ecgwei@scut.edu.cn).

Digital Object Identifier 10.1109/TMC.2025.3582084

A. Motivations

Although offloading health data to edge servers reduces delay and energy consumption for fabric sensors [30], [31], [34], it also poses privacy risks during refreshment processes, as malicious service providers may exploit user data for profit. Homomorphic encryption (HE) enables secure computation on encrypted data without decryption, preserving privacy while utilizing the server resources [17]. Unlike fully HE requiring intensive bootstrapping to reduce noise from calculations [32], the lightweight leveled HE, such as Cheon–Kim–Kim–Song (CKKS) scheme [6], proves more practical for edge computing. The CKKS supports efficient ciphertext-plaintext operations, enabling faster-encrypted refreshment by encrypting only user data rather than the entire DNN model parameters [7], [8].

However, the privacy-enhanced HHM service refreshment process in the human digital twin-assisted fabric metaverse networks still faces the following difficulties:

- *Both the structure and number of layers in HHM services using DNN models are constrained by the type and depth of limited operations in leveled HE.* Some nonlinear modules in models, namely violation modules including Relu activation or batch normalization functions, cannot be included because only addition and multiplication operations are allowed in HE schemes. In addition, the maximum times of multiplication operations are decided by both the preset noise budget and desired AES-equivalent security level [8], e.g., 128 bits, which also serve as the upper threshold of model layers.
- *The parameters of original DNN models are redundant and massive, affecting the refreshment speed and computing consumption of encrypted HHM services.* Even with compression and reduction of the model layers, redundant parameters in each layer may lead to an intolerable refreshment delay due to the encryption process involving massive arithmetic operations for large prime numbers.
- *Users with small delay requirements may encounter difficulty in offloading health data to nearby servers when limited network resources are arbitrarily allocated.* The HHM service refreshment requests may experience timeout or be discarded on large-scale scenarios due to the limited computing resources of edge servers and dynamic network bandwidth.

Exploring non-trivial trades-off between security, efficiency, and model performance in privacy-enhanced HHM refreshments is very challenging. First, homomorphic encryption ensures user data privacy during inference but limits the types and depth of computations, potentially reducing model accuracy. In contrast, ignoring encryption can preserve inference accuracy but compromise user privacy. Second, the efficiency of secure service refreshment is heavily impacted by both model inference and network conditions. Deploying a compressed model without taking into account of network constraints may result in unaffordable communication delays, thereby offsetting the reduction in model inference delay achieved through compression, and even requiring more computing resources to meet user latency requirements. Thus, tackling the challenge lies in the following two key aspects: (i) how to design and compress the layer number and parameters of original DNN models to satisfy constraints on leveled HE with minimal accuracy loss; and (ii) how to optimally allocate encrypted data to a subset of selected edge servers hosting compressed services to maximize the throughput of HHM refreshment requests, while minimizing the computing resource consumption, subject to network constraints, as well as user delay and accuracy requirements.

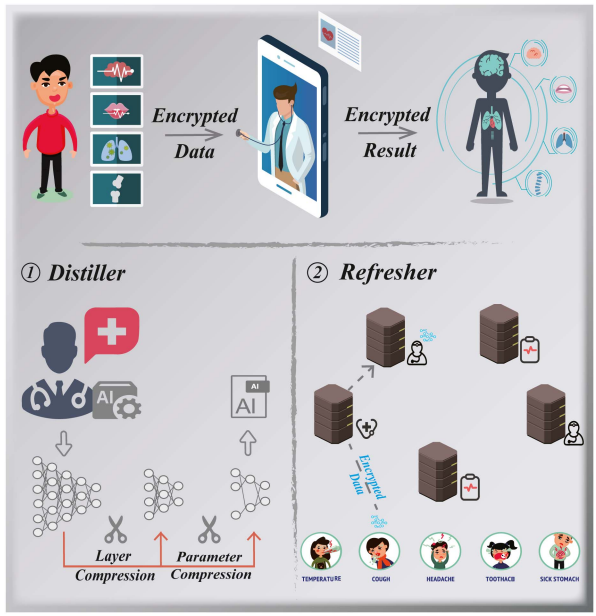


Fig. 1. The wiper system for human digital twin-assisted fabric metaverse network.

To tackle the aforementioned challenge (i), the distiller first replaces violation modules in existing DNN models with approximate ones (using only addition and multiplication) to satisfy operation constraints in leveled HE, and then applies model distillation to compress the number of layers and parameters of these approximate models, further aligning with depth constraints in leveled HE. To address challenge (ii), we propose an agile refresher library for offline and online scenarios, which includes exact algorithms, performance-guaranteed approximation algorithms, and residual diffusion reinforcement learning algorithms. A chosen algorithm from the library processes encrypted HHM refreshment requests efficiently, without privacy leaks while adaptively meeting diverse requirements on accuracy and delay.

B. Contributions

The novelty of this paper lies in the first pilot exploration of privacy-enhanced HHM service refreshments in human digital twin-assisted fabric metaverse networks, by jointly considering privacy data encryption, model compression, and user service requirements. A resource scheduler named `Wiper` is developed as shown in Fig. 1, consisting of a shallow-deep distiller and an agile refresher library.

The main contributions of this paper are as follows.

- To the best of our knowledge, we are the first to study the privacy-enhanced HHM service refreshment maximization problem that jointly considers data encryption, model compression, and personalized user requirements. We formulate the problem using an integer linear programming (ILP) solution. We also show the NP-hardness of the offline version of the problem.
- To standardize inference operation type and depth by adapting in leveled HE, we propose a shallow-deep distiller that first replaces violation modules with approximate modules in DNN models. The distiller then employs layer-compression and parameter-compression techniques to reduce both model layers and parameters, enhancing the encrypted inference speed.
- To address the HHM service refreshment maximization problem, we develop a suite of refreshers, which includes an ILP solution if the problem size is small; otherwise an approximation algorithm with an approximation ratio of $\min\{2, \mathcal{K} \cdot \Xi + 1\}$ for the problem. We also design an online, three-layer residual diffusion reinforcement learning algorithm for dynamic request admissions, which deals with bandwidth fluctuations and unknown delays.
- To evaluate the proposed algorithms, we conduct comprehensive experiments, by demonstrating that `Wiper` enhances accuracy by 2.34% to 9.24% during model distillation, improving throughput by 37% to 113% in offline scenarios, and achieving up to 4.7% improvement in online scenarios compared to the state-of-the-art baselines.

The rest of this article is organized as follows. Section II reviews related works. Section III introduces the human digital twin-assisted metaverse network, service delay, computing resources, and other mathematical models. Section IV introduces the shallow-deep distiller, and a series of agile refreshers and mathematical analyses are developed. Section V evaluates the performance of the proposed algorithms. Section VI concludes the paper.

II. Related Work

In this section, we survey key technologies related to HHM service refreshments, aiming to further evolve it towards immersive and secure.

Human Digital Twin: There are several studies in the topic of human digital twins. For instance, Chen et al. [3] provided an overview of networking architecture and key technologies in human digital twins for personalized healthcare for the first time. Zhou et al. [39] then designed sensor fusion and pose calibration algorithms to process data from a hub node and inertial measurement units for inertial capture of human motion digital twin. To further address synchronization problems, Okegbile et al. [26] developed a reliable data connectivity scheme in human digital twins and humans by integrating differential privacy, federated multi-task learning, and blockchain. Similarly, Liang et al. [21] investigated a multiple service model refreshment problem by devising an online algorithm that strives for non-trivial tradeoffs between the accumulative freshness of all models and the total refreshment cost. Li et al. [20] optimized digital twin freshness via efficient synchronization scheduling, proposing a generic framework. Moreover, Chen et al. [4] presented two intelligent algorithms to process electrocardiograph waves and WiFi signals for graphical monitoring and healthcare prediction in smart homes with human digital twins. Zheng et al. [38] proposed a similarity range query scheme using a partition-based tree structure to secure index data for healthcare monitoring over digital twin cloud platforms. Chen et al. [2] proposed and illustrated the application platform of mobile artificial intelligence-generated content and human digital twin for customized surgery and personalized medication.

Homomorphic Encryption: Doan et al. [9] extensively compared and evaluated various HE algorithms, and introduced implementation specifications in algorithm libraries. Building on this, Hu et al. [12] studied a sparse code multiple access algorithm based on HE to deal with the drastic drop in communication performance and stealing private information problems. Gupta et al. [11] proposed a HE-based IoT communication system incorporating elliptic curve cryptography, dynamically adjusting the key strength of the curve according to the remaining energy of devices and the importance of data topics in edge-cloud environments. Further advancing the field, Disabato et al. [8] introduced a distributed architecture relying on HE for supporting DNN inferences with encrypted data in cloud centers. Chien et al. [7] improved the adversarial representation learning algorithm with HE for privacy-preserving machine learning of IoT devices to achieve the predictor and encrypted encoder in the cloud. Nguyen et al. [25] proposed a HE-based federated learning framework to address the straggling problem, where slow devices may hinder learning speed, by securely offloading part of the training tasks to cloud-edge environments to maximize profit. Huang et al. [13] developed a secure two-party neural network inference system that reduces rotation operations for linear layers and uses lean primitives to enhance communication for non-linear layers.

Unlike the studies in [2], [4], [21], [26], [39] that focused on secure and immutable network environments, malicious server providers are considered in the `Wiper` system. In contrast to the aforementioned studies [7], [8], [11], [12], [13], [25] that primarily addressed a single user, model, and limited operation types in privacy-preserving machine learning, the `Wiper` system addresses scenarios with multiple DNN models, large numbers of users, and HE constraints on operation types and depth simultaneously. Although the study in [38] shares the similar edge medical network background with ours in this paper, it primarily focused on privacy-protecting information queries, rather than real-time refreshments of digital avatars. In this paper, we investigate a privacy-enhanced HHM service refreshment maximization problem in human digital twin-assisted metaverse networks by jointly considering data encryption, model compression, and personalized user requirements. The proposed method in this paper enables efficient end-to-end privacy-preserving inference in fabric metaverse environments. Specifically, it (i) protects both raw health data and inference results, (ii) reduces inference accuracy loss through distillation-assisted homomorphic encryption that can accelerate inference and reduce energy consumption of model itself, and (iii) enables

TABLE I
NOTATIONS

Notation	Definition
$\mathcal{G}$	The human digital twin-assisted metaverse network
$\mathcal{V}$	The set of edge servers in $\mathcal{G}$
C_j	The computing resource capacity of an edge server v_j
$\mathcal{E}$	The set of link in $\mathcal{G}$ connecting edge servers
B_l and D_l	The bandwidth and distance of link e_l
$\mathcal{M}$	The set of DNN models required by HHM services
$F_{p,q}$	The floating-point calculation of a model $m_{p,q}$ in $\mathcal{M}$
γ_i	The HHM refreshment request issued by user i
η_i	The proportional coefficient of computing resource allocated to user i
$\mathcal{D}_i(X)$	The polynomial representation of health data $\mathcal{D}_i$
$\mathcal{C}_i(X)$	The encrypted polynomial representation of health data $\mathcal{D}_i$
$T^d_{ed,i}$	The encoding delay of health data on devices for user i
$T_{et,i}$	The encryption delay of encoded data on devices for user i
$T^m_{ed,i}$	The encoding delay of model required by user i on edge servers
$T_{ec,i}$	The encryption calculation delay of encrypted data on edge servers
$T_{u,i}$	The upload delay of encrypted data for user i
$T_{d,i}$	The download delay of encrypted results for user i
β	The set of factors associated with scaling
δ	The set of factors associated with bias

a refresher library that processes encrypted HHM refreshment requests efficiently while adaptively meeting diverse requirements on accuracy and delay.

III. Problem Statement

In this section, we provide system models for a fabric metaverse network, user requirements, and HHM services model, and present precise problem definitions. We also introduce notions and notations, which are listed in Table I.

A. Fabric Metaverse Model

A human digital twin-assisted metaverse network can be modeled by an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the set of edge servers and $\mathcal{E}$ is the set of optic links between edge servers. Each edge server $v_j \in \mathcal{V}$ with computing capacity C_j is co-located with an access point (AP), and each link $e_l \in \mathcal{E}$ with length D_l has bandwidth capacity B_l. In addition, there are $|\mathcal{M}|$ human digital twin-assisted healthcare monitoring (HHM) services provided by the metaverse network, which help to diagnose health statuses of patients. Each HHM service is associated with a service model pool $\mathcal{M}_p$, in which each model $m_{p,q}$ has a similar structure but with a different number of parameters, leading to a different accuracy $A_{p,q}$ [29], [30]. We assume that all models in $\mathcal{M}$ are cached in external memory of edge servers.

B. Human Digital Twin-Assisted Healthcare Services

The refreshment of human data twin is crucial, which relies on real-time diagnostic results of health information to update the statuses of local digital avatars [38]. A user can issue a service request including both a specified service quality requirement and health data collected by body sensors from intelligent fabrics [29]. This request is then offloaded to a nearby edge server for processing, such as personalized disease diagnosis. The processed result is synchronized with digital avatars constantly, and shown in interactive interfaces of their wearable devices. The HHM service refreshment request from a user i can be modeled as a tuple $\gamma_i = (\mathcal{L}_i, \mathcal{M}_i, \mathcal{A}_i, \mathcal{T}_i, \mathcal{D}_i)$, where $\mathcal{L}_i$ is location coordinates of user i, $\mathcal{M}_i$ is the type of services of health monitoring requested, $\mathcal{A}_i$ is the model accuracy requirement of user i, and $\mathcal{T}_i$ is the end-to-end delay requirement of refreshing the digital avatar of user i. Notice that when a user service type and accuracy requirements are specified, the system is capable of selecting a DNN model $m_{i,q}$ from that service library $\mathcal{M}_i$ such that the accuracy of the chosen model is no less than the specified requirement. In addition, $\mathcal{D}_i$ is the input data for the chosen model $m_{i,q}$.

C. Service Delay Model

To mitigate the risk of personal information leakage from malicious edge servers or clouds, digital avatars are stored in wearable devices locally and refreshed periodically, using the homomorphic encryption (HE) technology such as the CKKS scheme. The Ring Learning With Errors problem (RLWE) is the core of security assurance for the CKKS scheme, where a ciphertext is added with "noise" to avoid hacking [16]. Subsequently, the encrypted data is uploaded to a nearby server for services via wireless networks. Finally, the encrypted results are downloaded and decrypted, using the private key to refresh the digital avatar. Thus, various service delays in data encryption, computation, and transmission are considered as follows.

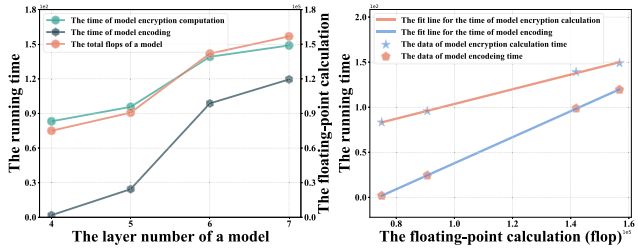
Data encoding and encryption delay: Given a piece of data $\mathcal{D}$, such as health data and model parameter, must be encoded in plaintext polynomial $\mathcal{D}(X)$ before encryption computation. For instance, given an integer $M = 8$ and a vector of health data $\mathcal{D}_i = [d_{i,0}, d_{i,1}, d_{i,2}, d_{i,3}]$, the Euler function is $\phi(M) = N = 4$, the Mth cyclotomic polynomial is $1 + X^4$, and the Mth root ξ_M of unity is $e^{\frac{2i\pi}{8}}$ through calculation [6]. The Vandermonde matrix formed by the above variables is expressed as

$$\begin{bmatrix} 1 & \xi_M & \ldots & \xi_M^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \xi_M^7 & \ldots & \xi_M^{(2N-1)(N-1)} \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{N-1} \end{bmatrix} = \begin{bmatrix} d_{i,0} \\ \vdots \\ d_{i,N-1} \end{bmatrix},$$

where $\{\alpha_0, \ldots, \alpha_{N-1}\}$ is original coefficients for plaintext polynomial $\mathcal{D}(X)$ waiting to be computed. These coefficients are then processed through a scaling factor and rounding to yield the final integer polynomial coefficients $\hat{\alpha}$. In essence, data encoding converts data $\mathcal{D} \in \mathbb{C}^{\frac{N}{2}}$ into plaintext space $\mathbb{Z}[X]/(X^N + 1)$, whereas data decoding translates plaintext polynomial $\mathcal{D}(X) = \sum_{i=0}^{N-1} \hat{\alpha}_i \cdot X^i \in \mathbb{Z}[X]/(X^N + 1)$ back into data space $\mathbb{C}^{\frac{N}{2}}$. The calculation amount mainly comes from solving the Vandermonde matrix and the data volume, thus the encoding delay $T^d_{ed,i}$ of health data for user i is

$$T^d_{ed,i} = \beta_{ed} \cdot \frac{|\mathcal{D}_i|_n}{W_i} + \delta_{ed}, \tag{1}$$

where β_{ed} and δ_{ed} are the scaling and bias factors of encoding delay for the calculated $\hat{\alpha}$, respectively. W_i is the computing capacity of wearable devices for user i, and $|\mathcal{D}_i|_n$ represents the data number of $\mathcal{D}_i$.



(a) The effect of layer number (b) The linear fitting function

Fig. 2. The links among layer, delay, and flops in SPANet. The fitting function among model flops, encryption, and encoding time are $y = 0.000816x + 21.956949$ and $y = 0.001442x - 106.371549$, respectively.

Encoded health data $\mathcal{D}_i(X)$ is further encrypted by a public key pk on wearable devices, while the model parameters are still only at the stage of being encoded on the edge server. This is because the CKKS scheme supports faster arithmetic operations between ciphertexts and plaintexts. Given a security parameter λ and the depth upper limit L, the associated parameters including secret key sk, public key $pk = (-a \cdot sk + e, a)$, and evaluation key are generated, where e is a small noisy polynomial inserted to guarantee security hardness of Ring Learning With Errors problem, and a, e, and sk are extracted from $Z_q[X]/(X^N + 1)$ [13]. In essence, encrypting using pk entails computing $(\mathcal{D}_i(X) - a \cdot sk + e, 0 + a)$, while decryption using sk involves calculating $\{[\mathcal{D}_i(X) - a \cdot sk + e] + [(0 + a) \cdot sk]\} = \mathcal{D}_i(X) + e$. Formally, the ciphertext polynomial $\mathcal{C}_i(X)$ of data $\mathcal{D}_i(X)$ by a secret key sk has a decryption structure form $\langle \mathcal{C}_i(X), sk \rangle = \mathcal{D}_i(X) + e \ (mod\ q)$. Thus, the data encryption delay $T_{et,i}$ for user i is formulated as follows:

$$T_{et,i} = \beta_{et} \cdot \frac{|\mathcal{D}_i(X)|_n}{W_i} + \delta_{et}, \tag{2}$$

where β_{et} and δ_{et} are the scaling and bias factors of encryption delay during encryption of encoded data $\mathcal{D}_i(X)$, respectively.

Model encoding and encryption computation delay: When an edge server encodes model parameters and receives the encrypted health data, privacy-enhanced inference from the required HHM services begins to diagnose. Each model contains numerous and heterogeneous parameters, necessitating massive relinearization and rescaling operations after homomorphic additions and multiplications to reduce the growth rate of noise e during encryption computation. Quantifying the exact delay generated throughout the model encoding and computation is challenging, so we propose an experimental modeling approach as shown in Fig. 2. The experimental data verifies that the floating-point calculation (flops) $F_{p,q}$ of a model $m_{p,q}$ has linear relationships with both model encoding delay and encryption computation delay under a given standard computing resource $C_{sta} < C_j$. Consequently, the encoding $T^m_{ed,i}$ and encryption computation delay $T_{ec,i}$ for user i with the amount $\eta_{i,j} \cdot C_j$ of computing resource are modeled,

$$T^m_{ed,i} = \frac{C_{sta}}{\eta_{i,j} \cdot C_j} \cdot (\beta_{em} \cdot F_{i,q} + \delta_{em}), \tag{3}$$

$$T_{ec,i} = \frac{C_{sta}}{\eta_{i,j} \cdot C_j} \cdot (\beta_{ec} \cdot F_{i,q} + \delta_{ec}), \tag{4}$$

where β_{ec} and δ_{ec} are the scaling and bias factors of encryption computation delay, respectively, and $\eta_{i,j}$ is the percentage of computing resource allocated to user i for server v_j.

Data uploading and downloading delay: During digital avatar refreshments, encrypted data is uploaded to edge servers for processing, and the processed result is encrypted and downloaded to wearable devices. The transmission ($T_{ut,i}$ and $T_{dt,i}$) and propagation ($T_{up,i}$ and $T_{dp,i}$) delays are the main bottlenecks in upload delay $T_{u,i}$ and download delay $T_{d,i}$, which are affected by network bandwidth, congestion condition, and the data volume. These delays for user i are modeled as follows:

$$T_{u,i} = T_{ut,i} + T_{up,i} = \sum_{e_l \in \mathcal{P}_{i,j}} \left(\frac{\zeta \cdot D_l}{3 \times 10^8} + \frac{|\mathcal{C}_i(X)|_m}{B_l} \right), \tag{5}$$

$$T_{d,i} = T_{dt,i} + T_{dp,i} = \sum_{e_l \in \mathcal{P}_{i,j}} \left(\frac{\zeta \cdot D_l}{3 \times 10^8} + \frac{|\mathcal{C}_i(X)|_m}{B_l / \beta_{dc}} \right), \tag{6}$$

where $\mathcal{P}_{i,j}$ is the shortest path from user i to server v_j in $\mathcal{G}$, and 3×10^8 m/s is the signal propagation speed. ζ, representing the network congestion coefficient, is used to correct transmission delay. β_{dc} is the scaling factor of data volume after encryption. $|\mathcal{C}_i(X)|_m$ is the memory size of $\mathcal{C}_i(X)$. After the wearable device receives the encrypted diagnostic result $\mathcal{C}_i$, it decrypts and decodes the result to obtain final data for updating digital avatar. Decoding and decryption delays are neglected because the data volume of $\mathcal{C}_i(X)$ is much smaller than that of $\mathcal{D}_i$.

Overall, the privacy-enhanced service refreshment for user i by using server v_j should experience the total delay $T_{i,j}$, including data encoding and encryption, model encoding, data uploading, encryption computation, and data download delays.

$$T_{i,j} = T^d_{ed,i} + T^m_{ed,i} + T_{et,i} + T_{u,i} + T_{ec,i} + T_{d,i}. \tag{7}$$

D. Computing Resource Demand Model

To ensure that the refreshment process strictly meets the user delay requirement $\mathcal{T}_i$, the minimum threshold of end-to-end delays must satisfy the following inequality,

$$\mathcal{T}_i \geq T^d_{ed,i} + T^m_{ed,i} + T_{et,i} + T_{u,i} + T_{ec,i} + T_{d,i}. \tag{8}$$

The actual refreshment delay $T_{i,j}$ for each user is nondeterministic due to multiple factors as stated in $Lemma$ 1, resulting in fluctuations in allocated computing resources.

Lemma 1: Given the delay requirement $\mathcal{T}_i$ of user i, the amount $\eta_{i,j}$ of computing resource allocated to the user fluctuates with server attributes such as location and total resources, model attributes including input size and model parameters, and network attributes like bandwidth.

Proof: The inequality (8) is simplified further by combining formulas (3) and (4), where the delays $\iota = T^d_{ed,i} + T_{et,i}$ caused by wearable devices are a fixed value,

$$\mathcal{T}_i \geq \frac{C_{sta} \cdot [(\beta_{em} + \beta_{ec}) F_{i,q} + \delta_{em} + \delta_{ec}]}{\eta_{i,j} \cdot C_j} + T_{u,i} + T_{d,i} + \iota.$$

Then, shifting related variables in the above inequality,

$$\mathcal{T}_i - T_{u,i} - T_{d,i} - \iota \geq \frac{C_{sta} \cdot [(\beta_c + \beta_{ed}) F_{i,q} + \delta_{em} + \delta_{ec}]}{\eta_{i,j} \cdot C_j}.$$

Finally, the expression for determining the ratio $\eta_{i,j}$ is derived,

$$\begin{aligned}\eta_{i,j} &\geq \frac{(\beta_{em} + \beta_{ec}) \cdot F_{i,q} + \delta_{em} + \delta_{ec}}{\mathcal{T}_i - T_{u,i} - T_{d,i} - \iota} \cdot \frac{C_{sta}}{C_j} \\ &\geq \frac{(\beta_{em} + \beta_{ec}) \cdot F_{i,q} + \delta_{em} + \delta_{ec}}{\mathcal{T}_i - \frac{2\zeta \cdot |\mathcal{P}_{i,j}|}{3\times 10^8} - \sum_{e_l \in \mathcal{P}_{i,j}} \frac{(1+\beta_{dc}) \cdot |\mathcal{C}_i(X)|_m}{B_l} - \iota} \cdot \frac{C_{sta}}{C_j}.\end{aligned} \tag{9}$$

Thus, allocated resources for user i are changed with server attributes such as $\mathcal{P}_{i,j}$ and C_j, model attributes including $\mathcal{C}_i(X)$ and $F_{i,q}$, and network attributes like B_l and ζ. □

Based on the derived relationships, it is evident that the allocated resources for user i vary with the location of the selected server even if the same model is chosen. Furthermore, the dynamic nature of network bandwidth and congestion conditions introduces additional challenges in meeting the end-to-end delay requirements.

E. Threat Model and User Privacy

In healthcare systems, service providers often operate under an honest-but-curious model: while they faithfully execute requested computation, they also seek to profits from user health data and statuses. When a user initiates a refresh request for its digital twin, the health data is stored on the server hosting the corresponding HHM service. Over time, repeated refreshes lead to the accumulation of the user data across the entire network, forming a health-centric database. Service providers monetize this valuable information by selling (1) the aggregated datasets that reveal population-level disease trends to a third party for market analysis or model training purposes, and (2) inference results of individualized health status inferences to third parties such as pharmaceutical companies. In addition, when the refresh result indicates a disease status, the user will inevitably undergo diagnosis by medical professionals, leading to the disclosure of the previously decrypted data. This gives adversaries an opportunity to perform key recovery attacks by correlating the information (even ciphertext) stored in the cloud with the corresponding plaintext known to the medical professionals.

To mitigate key recovery attacks for encrypted health data, we introduce a differential privacy mechanism on user sides. Specifically, when a user receives an updated disease status and intends to use the decrypted data for diagnosis, differential privacy ensures that his/her sensitive personal information remains protected. The formal definitions and theoretical foundations on such information protection are presented as follows. For the sake of convenience, we give several definitions and a theorem that are essential for privacy and security analysis in the Section IV.

Definition 1 (Rényi Divergence): For two probability distributions P and Q defined over $\mathbb{R}$, the Rényi divergence of order $\widetilde{\alpha} > 0$ is

$$D_{\widetilde{\alpha}}(P \parallel Q) = \begin{cases} \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}, & \widetilde{\alpha} = 1, \\ \frac{1}{\widetilde{\alpha}-1} \log \left(\sum_{x \in \mathcal{X}} P(x)^{\widetilde{\alpha}} Q(x)^{1-\widetilde{\alpha}} \right), & \text{others.} \end{cases}$$

where $P(x)$ denotes the density of P at x.

Definition 2 (ε-Zero-Concentrated Differential Privacy (ϵ-zCDP) [1]): A randomized mechanism $\mathfrak{M} : \mathcal{X}^n \to \mathcal{Y}$ is said to be ε-zCDP if for all $x, x' \in \mathcal{X}^n$ and all $\widetilde{\alpha} \in (1, \infty)$,

$$D_\alpha \left(\mathfrak{M}(x) \parallel \mathfrak{M}(x') \right) \leq \widetilde{\alpha} \cdot \varepsilon.$$

Definition 3: Let $\varepsilon > 0$ and $n \in \mathbb{N}$. Define the discrete Gaussian mechanism $\mathfrak{M}_{\mathfrak{t}} : \mathbb{Z}^n \to \mathbb{Z}^n$ such that, for any input $x \in \mathbb{Z}^n$, the mechanism outputs a sample drawn from the discrete Gaussian distribution $\mathcal{N}_{\mathbb{Z}^n}(x, \frac{\mathfrak{t}^2}{2\varepsilon} \cdot I_n)$, where I_n is the n-dimensional identity matrix, and $\mathfrak{t}$ is the upper bound on the CKKS decryption error.

Theorem 1 (Theorem 8 of [24]): Let $\mathfrak{G}_P$ be an indistinguishability game with black-box access to a probability ensemble P_θ. If $\mathfrak{G}_{P_\theta}$ is κ-bit secure, and $\max_\theta D(P_\theta \parallel Q_\theta) \leq 2^{-\kappa+1}$, then $\mathfrak{G}_{Q_\theta}$ is $(\kappa - 8)$-bit secure.

In general, a user privacy is guaranteed if and only if both the raw user data and the computed results always remain confidential, which can be formally defined as follows.

Definition 4 (User Privacy): Consider a user i with health data $\mathcal{D}_i$ and a server j HHM service $\mathcal{M}_j$ in a metaverse network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, let $\{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Cal}\}$ denote the components of a homomorphic encryption scheme, where $\mathsf{Gen}(\cdot)$ generates a public-private key pair (pk, sk); $\mathsf{Enc}(\cdot)$ and $\mathsf{Dec}(\cdot)$ are the encryption and decryption functions, respectively; $\mathsf{Cal}(\cdot)$ denotes computation over ciphertexts and returns encrypted results. The privacy of user i is considered preserved if the following conditions are satisfied:

1) *Ciphertext-Only Observation*: The network $\mathcal{G}$ handles only encrypted data. Specifically, user i encrypts their data as $\mathsf{Enc}(\mathcal{D}_i, pk)$, and the server computes over ciphertexts $\mathsf{Cal}(\mathcal{M}_j, \mathsf{Enc}(\mathcal{D}_i, pk))$.
2) *User-Only Decryption*: Server j sends only the encrypted result to user i. The user decrypts the output using their private key, satisfying $\mathsf{Dec}(\mathsf{Cal}(\mathcal{M}_j, \mathsf{Enc}(\mathcal{D}_i, pk)), sk) = \mathcal{M}_j(\mathcal{D}_i)$.
3) *Noise-Added Result Sharing*: If user i intends to share the result with others, they must release a perturbed version. Specifically, if a third party cannot decrypt, namely $\mathsf{Dec}(\mathsf{Cal}(\mathcal{M}_j, \mathsf{Enc}(\mathcal{D}_i, pk)), sk) \neq \mathcal{M}_j(\mathcal{D}_i)$, then the user should share the decrypted result with a noise to the third party for providing ε-zCDP, generated as $\mathsf{Dec}(\mathsf{Cal}(\mathcal{M}_j, \mathsf{Enc}(\mathcal{D}_i, pk)), sk) + \mathcal{N}(0, \sigma^2)$, where $\mathcal{N}(0, \sigma^2)$ denotes Gaussian noise with mean 0 and variance σ^2.

Note that the noise is added during the CKKS decryption stage at the user side when the user wants to share the information with a third party, and this noise addition does not affect the performance of the algorithm in the refresh library.

F. Problem Definitions

Definition 5: Given a human digital twin-assisted metaverse network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, there is a set $\mathcal{M}$ of DNN models, and a set $\mathcal{R}$ of inference requests with each request $\gamma_i \in \mathcal{R}$ requesting a service model $m_{i,q} \in \mathcal{M}$ with a given service accuracy $\mathcal{A}_i$ and delay requirements $\mathcal{T}_i$, *the maximum privacy-enhanced HHM request refreshment problem* (MRP) in $\mathcal{G}$ is to choose a subset of edge servers with models to maximize the throughput of encrypted HHM refreshment requests with the lowest total computing resource consumption of all requests for the model, while meeting the accuracy and delay requirements of admitted requests, subject to the computing capacity on each edge server.

The above-defined MRP problem is a set of requests in a snapshot scenario. However, in reality, user requests can arrive at any time point in the network monitoring period, and network bandwidth and congestion conditions fluctuate dynamically over time. Furthermore, the service delay models are not given. This leads to the following general problem - the O-MRP problem, which is defined as follows.

Definition 6: Given a human digital twin-empowered metaverse network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a finite time horizon $\mathcal{T}$, a set $\mathcal{V}$ of edge servers with each server j having computing capacity C_j and each link $e_l \in \mathcal{E}$ having dynamic bandwidth $B_l(t)$, a set $\mathcal{M}$ of DNN models for request services, and a set $\mathcal{R}(t)$ of HHM inference requests arriving randomly at each time slot $t \in \mathcal{T}$, *the online maximum privacy-enhanced HHM request refreshment problem (O-MRP) in* $\mathcal{G}$ is to admit HHM requests one by one without any future knowledge of request arrivals and encryption delay to maximize the throughput of encrypted HHM requests, using a leveled HE technique to refresh their human digital twins, subject to network resource capacities.

Theorem 2: Both defined MRP and O-MRP problems in a fabric metaverse network are NP-hard.

Proof: The NP-hardness of the MRP problem is confirmed through a polynomial reduction from the NP-hard problem - the maximum-profit Generalized Assignment Problem (GAP) that is defined as follows. Given a set $Item = \{I_1, \ldots, I_n\}$ of items, where each item $T_i \in Item$ has a profit $profit_i$ with size $size_i$, and a set $B = \{b_1, \ldots, b_m\}$ of bins, where each $b_j \in B$ has capacity cap_j. The objective of the GAP is to maximize the total profit by packing as many items as possible into the m bins, subject to computing capacity cap_j on each bin b_j with $1 \leq j \leq m$.

We consider a special MRP problem, where each HHM service demands the amount $\eta_{i,j}$ of computing resource, each edge server v_j has computing capacity C_j, and each request $\gamma_i \in \mathcal{R}$ is issued simultaneously. The system obtains a profit of 1 when a user successfully receives an encrypted result from its requested HHM service that is hosted on edge server v; otherwise, 0. The MRP problem aims to maximize the number of HHM refreshment requests admitted by securely offloading as much health data of users as possible into the selected edge servers in a metaverse network, subject to the capacity C_j on each edge server v_j. This special problem is equivalent to the GAP. Therefore, the MRP problem is NP-hard as the GAP is NP-hard. Since the MRP problem is the offline version of the O-MRP problem where user requests arrive one by one without the knowledge of future request arrivals. In other words, the MRP problem is a special case of the O-MRP when there is only one time slot, the O-MRP problem thus is NP-hard, too.□

Algorithm 1: Shallow-Deep Distiller.

Input: A set $\mathcal{M}$ of DNN model m_p, a upper limit of calculation depth L, and a compression ratio ω.
Output: Compressed and HE-compatible DNN models.

begin
  for *each original DNN model* $m_p \in \mathcal{M}$ **do**
    if m_p *has violation operations for leveled HE* **then**
      Approximation model $\hat{m}_p$ is obtained by replacing batch normalization module with linear approximation $\beta_l \cdot x + \delta_l$, activation module with a quadratic approximation $\beta_{q,2} \cdot x^2 + \beta_{q,1} \cdot x + \delta_q$, max pooling with average pooling, as stated in Theorem 3;
    else
      Approximation model is itself, $\hat{m}_p = m_p$;
    if *Layer number of model* $\hat{m}_p > L$ **then**
      Modify layer structures of model $\hat{m}_p$ to meet L in leveled HE, and distill it via teacher m_p to obtain layer-compressed model $\hat{m}_{p'}$;
    else
      Layer-compressed model is itself, $\hat{m}_{p'} = \hat{m}_p$;
    Use student-teacher model $\hat{m}_{p'}$ to deep distill to obtain a parameter-compressed model $\hat{m}_{p''}$ by reducing its channels ω times;
    Store $\hat{m}_{p'}$ and $\hat{m}_{p''}$ into model library $\mathcal{M}_p$;

IV. Wiper System

In this section, we investigate privacy-enhanced HHM service refreshment, by developing a secure resource scheduling system `Wiper`, which is depicted in Fig. 1. The system consists of two main components: a shallow-deep distiller and an agile refresher library. In the following, we first develop an algorithm for the shallow-deep distiller. We then devise algorithms for the agile refresher library, followed by a privacy module that is activated only when users wish to share their decrypted data with a third party. We finally perform theoretical analyses of the proposed algorithms.

A. Shallow-Deep Distiller

The shallow-deep distiller is mainly to 1) standardize both operation types and calculation depth of models to meet limitations in leveled HE, and 2) compress model parameters to speed up diagnosis speed and reduce computing resources required of HHM servers. Denote by $m_{p,q}$ any model that is abbreviated to m_p for simplicity. The detailed algorithm of the distiller is given as follows.

First, approximation models $\hat{m}_p$ are derived by replacing violation modules in DNN models with approximation modules compatible with leveled HE. Nonlinear modules, essential for model fitting, are primarily of three types: batch normalization, activation, and pooling operations, which often conflict with leveled HE due to violation operations beyond multiplication

and addition. By analyzing the calculation principle of nonlinear modules in $Theorem$ 3, we propose approximation modules composed of learning parameters including $\beta_l, \delta_l, \beta_{q,2}, \beta_{q,1}$, and δ_q to replace violation modules. These hyperparameters break the value size limitations of model parameters constrained by Taylor formulas to reduce approximation errors and enhance model scalability.

Second, a layer-compressed model $\hat{m}_{p'}$ is derived by distilling the original or approximation models helped by teacher model m_p. Note that the main difference between m_p and $\hat{m}_{p'}$ is the type of nonlinear modules and the number of model layers, that is, m_p has more layers and illegal nonlinear modules including Relu, Tanh, or Sigmoid activations. This shallow distiller ensures that compressed models meet the computation depth limits of leveled HE while maintaining the performance of approximation modules with learned parameters close to the violation modules.

Finally, a parameter-compressed model $\hat{m}_{p''}$ is obtained by distilling with the assistance of the corresponding student-teacher model $m_{p'}$, where the channel number of each layer in $\hat{m}_{p'}$ is ω times that of $\hat{m}_{p''}$. This deep distiller accelerates diagnosis and reduces parameters by compressing the number of model channels to a given compression ratio of ω times. Note that these layer-compressed $\hat{m}_{p'}$ and parameter-compressed models $\hat{m}_{p''}$ with different accuracies are placed in the corresponding model library $\mathcal{M}_p$ for users to choose by their personalized requirements during the avatar refreshing stage. The detailed algorithm for the shallow-deep distiller is given in `Algorithm` 1.

B. Agile Refresher Library

Having adopted all models to the operation depth and type constraints on leveled HE through model distillation, the agile refresher library selects appropriate HHM services running on edge servers for users in the fabric metaverse network, and offloads encrypted data to these servers for processing, ensuring real-time updates of health data for digital avatars.

1) Offline Refresher: For offline refreshments where user refreshment requests arrive at one time and all network information remains stable and given the reward function, the offline refreshers, including the integer linear programming (ILP) refresher, and an Approximate Refresher are proposed, each customized for different scale refresh scenarios.

ILP Refresher: We propose an ILP solution for the offline refresher when the problem size is small, and the problem optimization objective is to

$$\text{Maximize} \quad \sum_{\gamma_i \in \mathcal{R}} \sum_{v_j \in \mathcal{V}} x_{i,j} \tag{10}$$

subject to: (7), (8), and (9),

$$\sum_{v_j \in \mathcal{V}} x_{i,j} \leq 1, \quad \forall \gamma_i \in \mathcal{R} \tag{11}$$

$$\sum_{v_j \in \mathcal{V}} T_{i,j} \cdot x_{i,j} \leq \mathcal{T}_i, \quad \forall \gamma_i \in \mathcal{R} \tag{12}$$

$$\sum_{v_j \in \mathcal{V}} \eta_{i,j} \cdot x_{i,j} \geq 0, \quad \forall \gamma_i \in \mathcal{R} \tag{13}$$

$$\sum_{\gamma_i \in \mathcal{R}} \eta_{i,j} \cdot x_{i,j} \leq 1, \quad \forall v_j \in \mathcal{V} \tag{14}$$

$$x_{i,j} \in \{0,1\}, \quad \forall \gamma_i \in \mathcal{R} \text{ and } v_j \in \mathcal{V} \tag{15}$$

Formula (10) aims to maximize the throughput of refreshment requests by minimizing computing resource, where $x_{i,j}$ is a binary variable, indicating whether the encrypted data of a request γ_i is transmitted to server v_j. Inequality (11) imposes a position constraint where the encrypted data of each user is passed to at most one edge server. Inequality (13) simplifies inequality (12) by the formula (9), which represents the delay constraint where $\eta_{i,j}$ is negative only when the total end-to-end delay exceeds the delay requirement for user i. Inequality (14) sets a computation resource limit, stating that the percent of resources allocated to all users for data processing at server v_j cannot exceed its capacity C_j, with this percent being 1.

Approximate Refresher: To alleviate the prohibitive time complexity of the ILP solution when the problem size is large, we develop an approximation algorithm as follows.

First, we introduce a threshold $\mathcal{K}$ for the percentage of allocated resources for each request γ_i. Specifically, using the information on servers and the allocated ratio matrix H, we identify the index i' of the request at the median in the sorted request set $\hat{\mathcal{R}}$ and the index j' of the server with the minimum residual computing resource. By setting threshold $\mathcal{K} = \eta_{i',j'}$, each server is divided into $1/\mathcal{K}$ resource blocks to ensure a lower bound on the request throughput, which is later shown in $Theorem$ 4.

Next, servers in $\mathcal{V}$ are sorted by computing resources as $\hat{\mathcal{V}}$, and the sorted requests are matched to the appropriate servers for processing data. If no suitable server is found, the request is pended for the second allocation round. Finally, an appropriate server for each pended request is found in the order they are waiting, ignoring threshold control. If no suitable server is found, the request is refused due to limited resources. By controlling the threshold and sorting requests and servers, the algorithm prioritizes deploying requests with lower computing demands on servers with fewer resources to enhance the throughput.

2) Online Refresher: For online refreshment problems, where user refreshment requests arrive sequentially without any future arrival information, network bandwidth dynamically fluctuates, and the delay of a refreshing avatar is unknown, for which we formulate a Markov Decision Process (MDP), and design a residual diffusion reinforcement learning (RDRL) algorithm.

- *State space* $\mathbb{S}$*:* The environment state observed by an agent at each time slot t is $s_t = \{\mathbb{C}, \mathbb{B}, \hat{\gamma}\}$, where $\mathbb{C} = \{C_1, \hat{C}_1 \ldots, C_{|\mathcal{V}|}, \hat{C}_{|\mathcal{V}|}\}$, C_j is total capacity, and $\hat{C}_j$ is available resources of edge server v_j, respectively. $\mathbb{B} = \{B_1, \ldots, B_{|\mathcal{E}|}\}$ is the set of bandwidth on all links, and $\hat{\gamma} = \{F_{i,q}, \mathcal{D}_i, \mathcal{L}_i, \mathcal{T}_i\}$ is service information of user i.
- *Action space* $\mathbb{A}$*:* Based on observed s_t, agent selects an action $a_t = \{\tilde{x}_{i,1}, \ldots, \tilde{x}_{i,j}, \ldots, \tilde{x}_{i,|\mathcal{V}|}\}$, where $\tilde{x}_{i,j}$ is probability of refreshing a digital avatar of user i by an edge server v_j. From action space $\mathbb{A}$, the one with the highest value will be selected and executed.

Algorithm 2: Approximate Refresher.

Input: A set $\mathcal{R}$ of HHM service requests, and a fabric network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Output: A refresh scheme for digital avatars of users.

begin
  Obtain allocated ratio matrix $H = \{\eta_{i,j}\}$ for requests in $\mathcal{R}$ by formula (9), and sort $\mathcal{R}$ in descending order by $\min\{\eta_{i,j} \cdot C_j \mid v_j \in \mathcal{V}\}$ of request γ_i to get $\hat{\mathcal{R}}$;
  Obtain a threshold $\mathcal{K} = \eta_{i',j'}$ by the index i' of request at median in $\hat{\mathcal{R}}$ and the index j' of server in $\mathcal{V}$;
  Sort $\mathcal{V}$ in ascending order based on resources as $\hat{\mathcal{V}}$;
  for *each request* γ_i *in* $\hat{\mathcal{R}}$ **do**
    Flag = False;
    for *each server* v_j *in* $\hat{\mathcal{V}}$ **do**
      if $\eta_{i,j} \le \mathcal{K}$ *and* $\eta_{i,j} \cdot C_j \le \hat{C}_j$ **then**
        Receive the request and update available resources of v_j, Flag = True, and break;
    if *not Flag* **then**
      Include the request into pending set $\mathcal{R}_p$;
  for *each request* γ_i *in pending set* $\mathcal{R}_p$ **do**
    Find a suitable server; if none, reject the request;

- *State transition:* After a_t is executed, the state will transit from s_t to s_{t+1}, following the conditional probability $p(s_{t+1}|s_t, a_t)$ that is unknown to the agent.
- *Reward function:* The reward r_t acts as a feedback signal to evaluate the value of action a_t for achieving the final goal at state s_t,

$$r_t = \begin{cases} \psi_r \cdot e^{\frac{1}{\eta_{i,j} \cdot C_j}}, & \text{if } \eta_{i,j} \cdot C_j \le \hat{C}_j \text{ and } \eta_{i,j} > 0 \\ -\psi_p, & \text{if } \eta_{i,j} \cdot C_j \le \hat{C}_j \text{ and } \eta_{i,j} \le 0 \\ -\psi_p \cdot \sum\limits_{i \in \mathcal{R}(t)} \rho_{i,j}, & \text{if } \eta_{i,j} \cdot C_j > \hat{C}_j \end{cases}$$

where ψ_r and ψ_p are the reward and penalty coefficients. A server crashes and disrupts all services when its capacity is violated, where $\rho_{i,j}$ is the completion degree of a service required by user i on the server v_j. $\mathcal{R}(t)$ is a request set that has not yet been completed until slot t.

The online refresher is described as follows. For simplicity, the state s_t observed by an agent at time t is denoted as $\mathbf{s}$, and the action a_t performed is denoted as $\mathbf{a}$ in the following.

First, a denoising diffusion model [15] and a residual convolutional model are combined to form a three-layer residual network ϵ_θ, which supports an actor policy $\pi_\theta(\mathbf{a} \mid \mathbf{s})$, as shown in Fig. 3. The final action $\mathbf{a}_0$ is obtained by removing the noise through τ steps from an initial random action $\mathbf{a}_\tau \sim \mathcal{N}(0, I)$ that is sampled by a Gaussian (normal) distribution,

$$\pi_\theta(\mathbf{a} \mid \mathbf{s}) = \mathcal{N}(\mathbf{a}_\tau; 0, I) \cdot \prod_{k=1}^{\tau} p_\theta(\mathbf{a}_{k-1} \mid \mathbf{a}_k, \mathbf{s}), \tag{16}$$

$$p_\theta(\mathbf{a}_{k-1} \mid \mathbf{a}_k, \mathbf{s}) = \mathcal{N}\left(\mathbf{a}_{k-1}; \mu_\theta(\mathbf{a}_k, k, \mathbf{s}), \tilde{\beta}_k I\right), \tag{17}$$

$$\mu_\theta(\mathbf{a}_k, k, \mathbf{s}) = \frac{1}{\sqrt{\alpha_k}} \left(\mathbf{a}_k - \frac{\beta_k}{\sqrt{1-\bar{\alpha}_k}} \cdot \epsilon_\theta(\mathbf{a}_k, k, \mathbf{s})\right), \tag{18}$$

Algorithm 3: Residual Diffusion Reinforcement Learning Based Online Algorithm for Refresher.

Input: A three-layer residual policy network with parameter θ, a multilayer perceptron critic network with parameter φ, a target policy network $\hat{\theta}$, and a target critic network $\hat{\varphi}$.
Output: A well-train policy network with parameter θ^*.

begin
  for *each training step* **do**
    for *each collected transition* **do**
      Observe state s_t and sample a random action $\mathbf{a}_\tau \sim \mathcal{N}(0, I)$;
      for *the denoising step from* $k = \tau$ *to* 1 **do**
        Predict noise distribution $\epsilon_\theta(\mathbf{a}_k, k, \mathbf{s})$ using a network in Fig. 3, calculate the mean μ_θ of probability distribution $p_\theta(\mathbf{a}_{k-1} \mid \mathbf{a}_k, \mathbf{s})$, get a distribution of $\mathbf{a}_{k-1}$ by formula (21);
      Select action $a_t = \mathbf{a}_0$ based on probability distribution $Sigmoid(\pi_\theta(\mathbf{a} \mid \mathbf{s}))$ calculated by the formula (22);
      Execute the action a_t in the fabric metaverse network, and observe the next state s_{t+1} and reward r_t, and store the transition (s_t, a_t, s_{t+1}, r_t) in the replay buffer $\mathbf{D}$;
    Sample a batch of data $\hat{\mathcal{B}} = \{(s_t, a_t, s_{t+1}, r_t)\}$ from replay buffer $\mathbf{D}$;
    Update the policy parameters θ by using gradient descent to minimize formula (23), and update the critic parameters φ by using gradient descent to minimize formula (24), and update parameters $\hat{\theta}$ and $\hat{\varphi}$ of target networks by formulas (25) and (26);

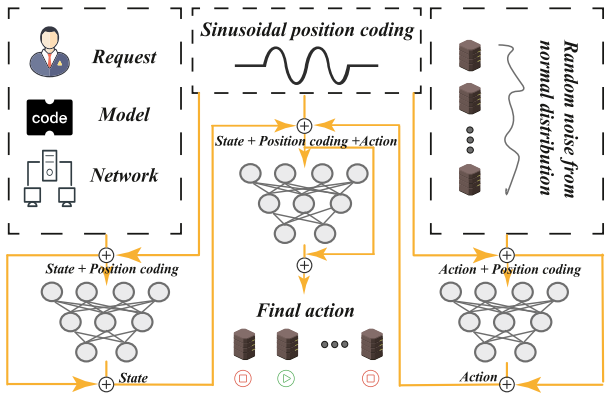


Fig. 3. The three-layer residual diffusion policy network.

where $k \in [1, \tau]$ and $\mu_\theta(\mathbf{a}_k, k, \mathbf{s})$ is the mean value of the denoising noise distribution $p_\theta(\mathbf{a}_{k-1} \mid \mathbf{a}_k, \mathbf{s})$ from $\mathbf{a}_k$ to $\mathbf{a}_{k-1}$. The remaining parameters are described as follows:

$$\beta_k = 1 - \alpha_k = 1 - e^{-\frac{\beta_{min}}{\tau} - \frac{(2k-1)\cdot(\beta_{max}-\beta_{min})}{2\cdot\tau^2}}, \tag{19}$$

$$\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_k \quad \text{and} \quad \tilde{\beta}_k = \frac{1-\bar{\alpha}_{k-1}}{1-\bar{\alpha}_k}\beta_k, \tag{20}$$

where $\beta_{min} = 0.1$ and $\beta_{max} = 10$. In addition, to address the issue of gradients not being back-propagated through random variables during sampling, we apply the reparameterization

method from [10] to reformulate the sampling process, which decouples randomness from distribution parameters,

$$\mathbf{a}_{k-1} = \mu_\theta(\mathbf{a}_k, k, \mathbf{s}) + \frac{\tilde{\beta}_k^2}{4} \cdot \epsilon, \epsilon \sim \mathcal{N}(0, I), \tag{21}$$

Thus, by applying softmax function to probability distribution $\pi_\theta(\mathbf{a} \mid \mathbf{s})$ of all servers, we compute the likelihood of choosing each edge server v_j, yielding final action $\mathbf{a}_0 = a_t$ for user i,

$$\mathbf{a}_0 = \arg\max_j Sigmoid(\pi_\theta(\mathbf{a} \mid \mathbf{s})) = \arg\max_j \{\tilde{x}_{i,j}\}, \tag{22}$$

where $Sigmoid = \frac{1}{1+e^{-x}}$.

Second, the agent executes the action a_t for request γ_i, receives reward r_t, and observes the next environment state s_{t+1}. In addition, the transition (s_t, a_t, s_{t+1}, r_t) is stored in the replay buffer $\mathbf{D}$ for the agent to learn from the previous experience periodically. Finally, the relevant neural networks are updated through a batch $\hat{\mathcal{B}}$ of transitions extracted from the replay buffer $\mathbf{D}$. The policy network π_θ minimizes the total loss $\mathcal{L}_\theta$ by gradient descent to improve the policy as follows:

$$\begin{aligned}
\mathcal{L}_\epsilon &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I),(\mathbf{s},\mathbf{a})\sim\hat{\mathcal{B}}}[||\epsilon - \epsilon_\theta(\mathbf{a}\sqrt{\bar{\alpha}_k} - \epsilon\sqrt{1-\bar{\alpha}_k}, k, \mathbf{s})||_1], \\
\mathcal{L}_q &= -\mathbb{E}_{\mathbf{s}\sim\hat{\mathcal{B}}}[Sigmoid(\pi_\theta)^\top \mathcal{Q}_\varphi(\mathbf{s}) + \alpha_l \cdot \mathcal{H}(Sigmoid(\pi_\theta))], \\
\mathcal{L}_\theta &= \varrho_l \cdot \mathcal{L}_q + (1-\varrho_l) \cdot \mathcal{L}_\epsilon,
\end{aligned} \tag{23}$$

where α_l and ϱ_l are scale factors. $\mathcal{L}_\epsilon$ is the policy-regularization loss, $\mathcal{L}_q$ is policy-improvement loss, and $\mathcal{H}(Sigmoid(\pi_\theta))$ is probability distribution entropy. The critic network $\mathcal{Q}_\varphi$ minimizes the total loss $\mathcal{L}_\varphi$ by gradient descent to improve the accuracy of state assessment,

$$\begin{aligned}
\mathcal{L}_\varphi = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\hat{\mathcal{B}}}\Bigg[\sum_{n=1,2} ||\mathcal{Q}_\varphi^n(s_t,a_t) \\
- \left(r_t + \gamma_r \cdot Sigmoid(\pi_{\hat{\theta}(s_{t+1})})^\top \mathcal{Q}_{\hat{\varphi}(s_{t+1})}\right)||_2\Bigg],
\end{aligned} \tag{24}$$

where r_t is the reward at slot t, γ_r is the discount factor, $\mathcal{Q}\varphi(s_t, a_t)$ is Q value for action a_t at state s_t, and $\mathcal{Q}\hat{\varphi}(s_{t+1})$ is Q value for all actions at next state s_{t+1}.

In terms of target network $\pi_{\hat{\theta}}$ and $\mathcal{Q}_{\hat{\varphi}}$, they are updated periodically by a soft update mechanism, which is defined as follows.

$$\pi_{\hat{\theta}_{step+1}} = \varrho_s \cdot \pi_{\theta_{step}} + (1-\varrho_s) \cdot \mathcal{Q}_{\hat{\varphi}_{step+1}}, \tag{25}$$

$$\mathcal{Q}_{\hat{\varphi}_{step+1}} = \varrho_s \cdot \mathcal{Q}_{\varphi_{step}} + (1-\varrho_s) \cdot \mathcal{Q}_{\hat{\varphi}_{step+1}}, \tag{26}$$

where ϱ_s is a scaling factor of updating the target network.

C. Privacy Module

To prevent privacy leakage caused by private key recovery when the query dimension $n = 1$ and Rényi divergence order $\widetilde{\alpha} = 1$, each user is assigned a given privacy budget $\hat{\varepsilon}_i = \mathfrak{q} \cdot \varepsilon_i$, which restricts the decrypted data users can disclose for diagnosis up to $\mathfrak{q}$ times. For the nth refresh request issued by the same user i, the user samples independent noise from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with $\sigma = 8\mathfrak{t}\sqrt{\mathfrak{q}}2^{\kappa/2}$ (as shown in Theorem 6) and a privacy budget ε_i, and adds the noise to the decrypted query result to ensure differential privacy prior to the result sharing, where $\mathfrak{q}$ is the number of queries, $\mathfrak{t}$ represents the upper bound on the CKKS decryption error, and κ is a system-dependent security parameter. With the increase on the number of refreshments, the remaining privacy budget decreases. Once the available budget is exhausted (i.e., approaches zero), the user must update his private key to restore the total privacy budget to its maximum value.

D. Algorithm Analysis

The rest is to analyze the feasibility of replacing the violation modules with the approximate one in `Algorithm 1`. Also, the approximation ratio and time complexity of `Algorithm 2` is analyzed.

Theorem 3: Given a set of DNN models containing violation modules incompatible with leveled HE, the shallow-deep distiller, `Algorithm 1`, replaces violation modules with approximate modules is reasonable.

Proof: We first identify potential violation on each module. We then propose their corresponding replacement counterparts to avoid the violations.

As for the batch normalization operations, the output y_i for information x_i is

$$y_i = \frac{\beta_{bn}}{\sqrt{\sigma_{bn}^2 + \epsilon_{bn}}} \cdot x_i + \delta_{bn} - \frac{\beta_{bn} \cdot \mu_{bn}}{\sqrt{\sigma_{bn}^2 + \epsilon_{bn}}},$$

where $\mu_{bn} = 1/N \cdot \sum_{1\le i\le N} x_i$, $\sigma_{bn}^2 = 1/N \cdot \sum_{1\le i\le N}(x_i - \mu_{bn})^2$, and $\epsilon_{bn} \approx 10^{-9}$ to avoid 0 during calculation. Thus, approximating the batch normalization function using a linear function module with learnable parameters β_l and δ_l is reasonable,

$$\beta_l \simeq \frac{\beta_{bn}}{\sqrt{\sigma_{bn}^2 + \epsilon_{bn}}},$$

$$\delta_l \simeq \delta_{bn} - \frac{\beta_{bn} \cdot \mu_{bn}}{\sqrt{\sigma_{bn}^2 + \epsilon_{bn}}},$$

where the near-optimal β_l and δ_l are obtained through gradient descent.

In terms of the activation operations, the Taylor expansion formula is the core of approximate modules. For example, $Sigmoid = \frac{1}{1+e^{-x}}$ and $Tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ are widely used as activation functions in deep neural network-based services, and their Taylor expansions are as follows.

$$Sigmoid' = \frac{1}{2} + \frac{x}{4} - \frac{x^3}{48} + \mathcal{O}(x^5),$$

$$Tanh' = x - \frac{x^3}{3} + \mathcal{O}(x^5).$$

Essentially, nonlinear activation functions can be expressed as polynomials over a given value range of independent variables, like the interval around 0. To overcome range limitation and uniformly replace all nonlinear functions with the simplest

option, quadratic functions with learnable parameters, such as

$$y_i = \beta_{q,2} \cdot x_i^2 + \beta_{q,1} \cdot x_i + \delta_q,$$

are the preferred choice. These hyperparameters $\beta_{q,2}$, $\beta_{q,1}$, and δ_q are also optimized by gradient descent and have fewer computational burdens for leveled HE.

Regarding the pooling module, replacing maximum pooling with average pooling is a more common approach in machine learning. This is the proof. □

Theorem 4: Given a human digital twin-assisted metaverse network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a set $\mathcal{M}$ of DNN models, a set $\mathcal{R}$ of HHM refreshment requests from different users, there is an Approximate Refresher, `Algorithm 2`, with bounded approximation ratio $\min\{2, \mathcal{K} \cdot \Xi + 1\}$ to offload encrypted health data to selected edge server with HHM services for maximizing the throughput of refreshing digital avatars of users while meeting different requirements on accuracy and delay in $\mathcal{O}(|\mathcal{R}|(|\mathcal{V}| + \log|\mathcal{R}|))$, where $\mathcal{K}$ is a threshold for the percent of allocated computing resources, and $\Xi = \mathcal{R}^{opt}/|\mathcal{V}| \leq \mathcal{R}/|\mathcal{V}|$ represents the upper bound of service throughout in an ideal scenario.

Proof: We analyze the approximation ratio of the approximate refresher based on two distinct cases derived from the relationship between $|\mathcal{R}|/2$ and $|\mathcal{V}|/\mathcal{K}$.

Case 1. Assuming a small number of HHM requests arrive at fabric metaverse network, that is, $\frac{|\mathcal{R}|}{2} \leq \frac{|\mathcal{V}|}{\mathcal{K}}$. In this scenario, the lower bound $\mathcal{R}_{app}^{-}$ of service throughput is $\frac{|\mathcal{R}|}{2}$, namely $\mathcal{R}_{app} \geq \frac{|\mathcal{R}|}{2}$. This is because all servers can handle at least half of arrival requests after partitioning their computing resources into $|\mathcal{V}|/\mathcal{K}$ blocks based on the computing demand of the median request at $\hat{\mathcal{R}}$. Therefore,

$$\frac{\mathcal{R}_{opt}}{\mathcal{R}_{app}} \leq \frac{2 \cdot \mathcal{R}_{opt}}{\mathcal{R}} \leq \frac{2 \cdot \mathcal{R}}{\mathcal{R}} = 2,$$

this establishes that the approximation ratio in Case 1 is bounded by 2.

Case 2. Suppose a large number of HHM requests arrive at the network at the same time, namely $\frac{|\mathcal{R}|}{2} > \frac{|\mathcal{V}|}{\mathcal{K}}$. The lower bound $\mathcal{R}_{app}^{-}$ of service throughout for Approximate Refresher is $\frac{|\mathcal{V}|}{\mathcal{K}}$, this is $\mathcal{R}_{app} \geq \frac{|\mathcal{V}|}{\mathcal{K}}$. To analyze the approximation ratio, two special constants are introduced,

$$CX = \max_{j \in \mathcal{V}} C_j,$$

$$CN = \min_{i \in \mathcal{R}, j \in \mathcal{V}} \eta_{i,j} \cdot C_j.$$

where CX is the maximum amount of computing resources for an edge server, and CM represents the minimum amount of demand computing resources for a request. Thus,

$$\begin{aligned} \frac{\mathcal{R}_{opt}}{\mathcal{R}_{app}} &= \frac{\mathcal{R}_a^{opt} + \mathcal{R}_r^{opt}}{\mathcal{R}_o^{app} + \mathcal{R}_r^{app}} \\ &= \frac{\mathcal{R}_a^{opt} + \mathcal{R}_r^{app} - \mathcal{R}_r^{app} + \mathcal{R}_r^{opt}}{\mathcal{R}_o^{app} + \mathcal{R}_r^{app}} \\ &= 1 + \frac{\mathcal{R}_r^{opt} - \mathcal{R}_r^{app}}{\mathcal{R}_o^{app} + \mathcal{R}_r^{app}} \\ &\leq 1 + \frac{\mathcal{R}_r^{opt} - \mathcal{R}_r^{app}}{|\mathcal{V}|/\mathcal{K}} \leq 1 + \frac{\mathcal{R}^{opt}}{|\mathcal{V}|/\mathcal{K}} \\ &\leq 1 + \frac{|\mathcal{V}| \cdot \frac{CX}{CN}}{|\mathcal{V}|/\mathcal{K}} = 1 + \mathcal{K} \cdot \Xi \end{aligned} \tag{27}$$

Formula (27) indicates that the $\mathcal{R}_o^{app}$ and $\mathcal{R}_r^{app}$ form the set $\mathcal{R}^{app}$ of requests received by Approximate Refresher, where $\mathcal{R}_o^{app} = \mathcal{R}_a^{opt}$ identical refresh decisions between approximation and optimization, and $\mathcal{R}_r^{app} \leq \mathcal{R}_r^{opt}$ represents the remaining differing requests. The formula (28) utilizing the ratio $\Xi = \frac{CX}{CN}$ represents the upper bound of service throughout in an ideal scenario. Thus, in Case 2, the approximation ratio is bounded above by $1 + \mathcal{K} \cdot \Xi$. Combining both cases, we conclude that the approximation ratio of the Approximate Refresher is $\min\{2, \mathcal{K} \cdot \Xi + 1\}$.

The time complexity of the Approximate Refresher is analyzed as follows. It incurs the complexity of $\mathcal{O}(|\mathcal{R}||\mathcal{V}|)$ to calculate the allocated ratio matrix H and two user-server matching processes. The time consumed on user and server sorting is $\mathcal{O}(|\mathcal{R}| \cdot \log|\mathcal{R}|)$ and $\mathcal{O}(|\mathcal{V}| \cdot \log|\mathcal{V}|)$, respectively. Since the number $|\mathcal{R}|$ of user requests is generally larger than the number $|\mathcal{V}|$ of servers, the time complexity of the Approximate Refresher is $\mathcal{O}(|\mathcal{R}|(|\mathcal{V}| + \log|\mathcal{R}|))$. The theorem then follows. □

E. Privacy and Security Analysis of Algorithms

In this section, we analyze the security and privacy of `Wiper`, enhanced with a dedicated privacy module.

Theorem 5: For any $\varepsilon > 0$ and $n \in \mathbb{N}$, the privacy module with noise sampled from the Gaussian mechanism $\mathfrak{M}_{\mathfrak{t}}$ ensures ε-zCDP. In addition, the expected cumulative ℓ_2-norm error is $\mathbb{E}[\|X\|_2] = \mathcal{O}(\mathfrak{t}\sqrt{n\mathfrak{q}\widetilde{\alpha}})$, where $\mathfrak{t}$ is the CKKS error bound (also sensitivity), n is the query dimension, $\mathfrak{q}$ is the number of queries, and $\widetilde{\alpha}$ is the Rényi divergence order.

Proof: The proof can be found in Appendix A, available online. □

Theorem 6: Given a metaverse network $\mathcal{G}$, involving a server provider, users, and an adversary, the privacy module provides both ε-zero-concentrated differential privacy and $IND - CPA^D$ security [18], by incorporating Gaussian noise sampled from $\mathcal{N}(0, \sigma^2)$, where $\sigma = 8\mathfrak{t}\sqrt{n\mathfrak{q}\widetilde{\alpha}}2^{\kappa/2}$, where $\mathfrak{q}$ is the number of queries, $\mathfrak{t}$ is the upper bound on the CKKS decryption error, κ is a system-dependent security parameter, and $\widetilde{\alpha}$ is the Rényi divergence order.

Proof: The proof can be found in Appendix B, available online. □

V. Performance Evaluation

In this section, we first provided the experimental parameter settings and several baseline algorithms. We then conducted extensive experiments to validate the effectiveness and efficiency of model inferences based on leveled HE and model distillation on edge servers. Finally, we analyzed the impacts of key parameters on algorithm performances in the simulated fabric metaverse network.

TABLE II
MODEL INFORMATION CONFORMING TO LEVELED HE

Model	SPANet	EEGNet	SERNet	LDDNet
Number of input data	256	70	180	40×862
Maximum Flops	0.16×10^6	1.18×10^6	3.02×10^6	13.59×10^6
Highest Accuracy	99.99%	88.45%	88.31%	82.61%

TABLE III
THE COEFFICIENTS ABOUT REFRESHMENT DELAYS

β_{em}	δ_{em}	β_{ec}	δ_{ec}	$\beta_{ed}, \delta_{ed}, \beta_{et}$	δ_{et}	β_{dc}
0.0014	-106.37	0.0008	21.957	$0.1 \sim 0.5$	$1 \sim 2$	$0.01 \sim 0.05$

TABLE IV
THE ACCURACY COMPARISON OF MODEL DISTILLATION

Model	Tea	Stu_HS	StuT_HS	Vanilla	Stu_H	StuT_H
LDDNet	88.04%	**82.61%**	**82.61%**	73.37%	77.72%	69.57%
SERNet	92.48%	**88.31%**	**86.69%**	80.56%	82.99%	78.82%
EEGNet	95.22%	**88.45%**	**81.76%**	79.42%	87.03%	78.96%

A. Experimental Environment Settings

We consider four types of HHM services in the fabric metaverse: SPANet for sitting posture analysis, EEGNet for emotional intensity prediction, SERNet for emotional type determination, and LDDNet for lung disease detection. Each service includes a DNN model library $\mathcal{M}_p$ with five models of similar structures but different parameters. The largest models, adapted to leveled HE and server memory, are detailed in Table II. To generalize and diversify the model library $\mathcal{M}_p$ for adapting various models in real scenarios, we keep the model with the highest accuracy and randomly simulate the accuracy of four additional models from 0.5 to the highest accuracy, where the flops of a model are linearly correlated with accuracy [28]. The memory size for floating-point and integer data units is 4B.

We simulate the fabric metaverse network, using the Abilene network [29] as the topology, covering a 5,500 km by 5,500 km area with 11 edge servers. Servers have computing resources between 40,000 and 80,000 MHz, and link bandwidths fluctuate between 4, 8, 16, and 32 MB/s [28]. The standard computing resource C_{sta} is 5,000MHz, and the network congestion coefficient ζ is set to 5,000. There are 1,000 users with fabric sensors sending HHM refreshment requests to refresh their digital avatars. Requests follow a Poisson process with an average arrival rate k_p of 0.001 [10], resulting in 1,000 requests within a time horizon $\mathcal{T} = 1 \times 10^6$ seconds. For each request γ_i, the index of $\mathcal{M}_i$ is randomly selected from 0 to 3, $\mathcal{A}_i$ is chosen from the accuracy list of $\mathcal{M}_i$, and $\mathcal{T}_i$ is drawn from $\mathcal{N}((1-\varsigma) \cdot T_{sta}, 0.15 \cdot T_{sta})$, where T_{sta} is the standard refresh delay and $\varsigma = 0$. Refreshment delay coefficients are listed in Table III.

We adopt the Pyfhel library [14] to implement the CKKS scheme with 128-bit security, polynomial modulus degree 2^{13}, maximum homomorphic depth L of 7, and ciphertext coefficient modulus $[22, 22, 22, 22, 22, 22, 22]$. Each DNN model (Table IV) includes layers: convolution, approximate BN, approximate ReLU, average pooling, fully connected, another approximate ReLU, and final fully connected. Most of the diffusion reinforcement learning parameters follow those in [10], except that $\varrho_l = 0.8$, $\tau = 5$, $\psi_r = 2$, and $\psi_p = 12$. The above values are default values for each trial unless otherwise specified.

B. Algorithm Performance Evaluation

1) Baseline Algorithms: To evaluate the performance of the proposed algorithms, we adopt the following seven comparison algorithms.

APP: The Approximate Refresher (APP), proposed in this paper, combines a control threshold $\mathcal{K}$ and request and server sorting to provide a solution with a provable approximation ratio for the MRP problem.

ILP [23]: The ILP refresher finds the optimal solution in exponential time, using ILP solver library-$Docplex$ in Python. The optimal solution will serve as the upper-bound baseline for the MRP problem.

RDRL: The online refresher (RDRL), proposed in this paper, combines a denoising diffusion model and a residual model to provide a near-optimal solution for the O-MRP problem.

DSAC [10]: It is based on the denoising diffusion model that is designed for scheduling and resource allocation in wireless networks [10]. We modified it for the O-MRP problem.

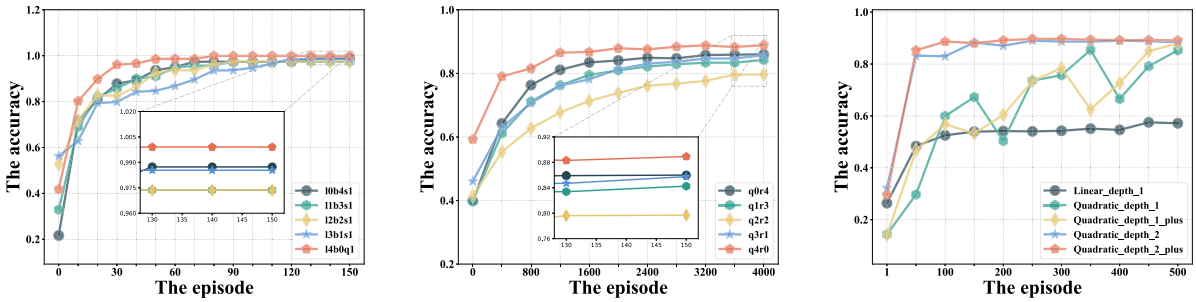
DQN [35]: It is based on the deep Q-learning network model for resource allocation in wireless networks [35]. We modified it for the O-MRP problem.

Prophet [10]: It assumes that both the delay and reward functions are given prior to HHM service refreshment, serving as the upper bound of the solution for the O-MRP problem.

Random: It randomly assigns users to any server for refreshing their digital avatars, or not at all, serving as the lower-bound baseline for both MRP and O-MRP problems.

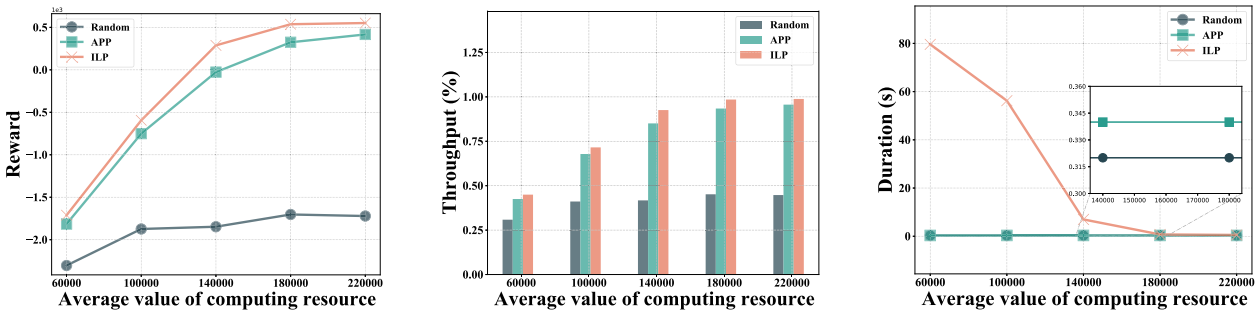
2) Impact of Key Parameters on Different Distillers: In the following, we first analyze the impact of replacing violation modules with approximations on model accuracy, as illustrated in Fig. 4, where original models violate not only the number of model layers, but also the type of modules. Here, the batch normalization (BN) function in SPANet is replaced by a linear function $y_i = \beta_l \cdot x_i + \delta_l$, and the Relu function in EEGNet is replaced by the quadratic function $y_i = \beta_{q,2} \cdot x_i^2 + \beta_{q,1} \cdot x_i + \delta_q$.

Approximation functions: Fig. 4(a) and (b) show that the model accuracy decreases first and then increases as the number of approximation modules increases. Here, $l0b4s1$ has no linear approximations, $l4b0q1$ is fully replaced with 4 linear and 1 quadratic approximation, $q0r4$ has no quadratic approximations, and $q4r0$ is fully replaced with quadratic approximation. In Fig. 4(c), we examine the nonlinear capability of quadratic approximation after replacing Relu functions in the EEGNet. $Quadratic\_depth\_1\_plus$, with multiple hyperparameters per layer and $\beta_{q,2} = 1$, has stronger nonlinear fitting ability (higher accuracy) than $Linear\_depth\_1$ with $\beta_{q,2} = 0$ and $Quadratic\_depth\_1$ with only 1 hyperparameter, and less operation complexity (calculation depth) than $Quadratic\_depth\_2$ and $Quadratic\_depth\_2\_plus$ with $\beta_{q,2} \neq 1$. Therefore, $y_i = x_i^2 + \beta_{q,1} \cdot x_i + \delta_q$ is adopted as a quadratic approximation in the subsequent experiments.



(a) Linear approximation (b) Quadratic approximation (c) Computation depth and nonlinear

Fig. 4. The impact of approximation modules on accuracy.



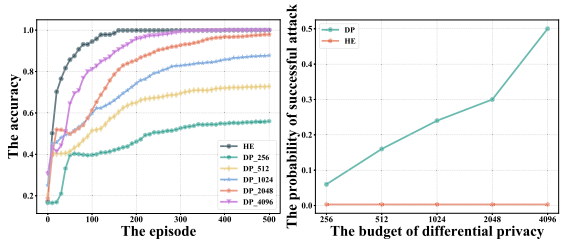
(a) The total reward (b) The HHM service throughput (c) The running time

Fig. 5. The performance of different offline algorithms by varying average computing resources from 60,000 to 220,000.

Next, we investigate the impact of model distillation methods applied to ensure compliance with HE constraints on model accuracy, as shown in Table IV. Here, the teacher model (Tea) has the largest number of parameters and includes the violation module, student-teacher models (Stu) are of medium size with standard parameters, and student models ($StuT$) are smaller with fewer parameters. Although the shallow-deep distiller can enable homomorphic encryption for DNN inference at the expense of model accuracy reduction and interpretability to some extent, a well-designed module replacement and distillation method can mitigate the loss on accuracy.

Model distillation: According to Table IV, the shallow-deep distiller, denoted as Stu_HS + $StuT\_HS$, using both hard (real) and soft (logits) labels, exhibits the least accuracy loss compared to other methods and its accuracy is closer to that of the teacher model Tea. Specifically, this distiller achieves an accuracy improvement ranging from 2.34% to 9.24% over the $Vanilla$ algorithm [37] that directly distills Tea into $StuT$ using hard and soft labels. Moreover, it outperforms a two-stage distillation approach that first distills Tea into Stu_H and then into $StuT\_H$ using only hard labels, and achieves an accuracy gain of 2.88% to 13.04% . Note the SPANet model structure is relatively simple and does not require distillation to meet the constraints of leveled HE.

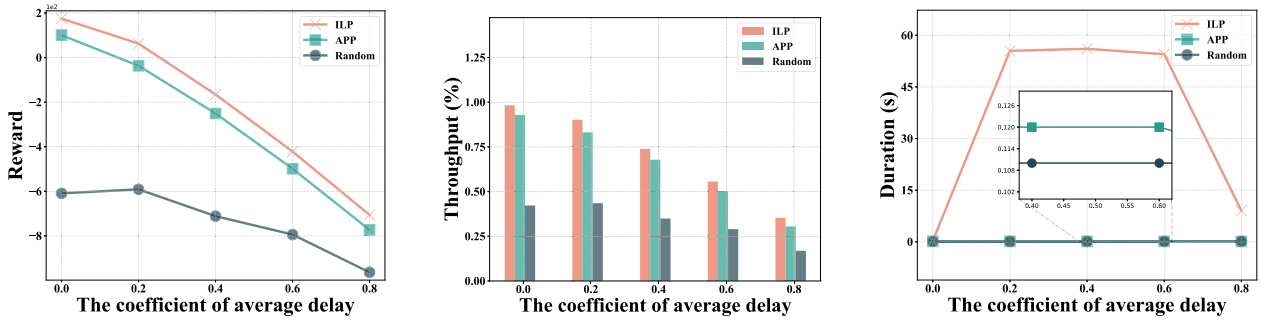
Finally, we studied the impact on the SPANet model under our threat model with homomorphic encryption (HE) and differential privacy (DP) approaches, by varying the privacy budget from 256 to 4096. A key threat considered is the attribute inference attack, where adversaries exploit refreshment data to infer the probability of a user in a specific group.



(a) The service accuracy (b) Successful attack probability

Fig. 6. Homomorphic encryption versus differential privacy.

Privacy comparison: Under attribute inference attacks, HE secures both input data and computation results, forcing the server to rely on random guessing and resulting in a low success rate. In contrast, DP adds the Laplacian noise only to the raw data: insufficient noise risks privacy leakage, while excessive noise degrades model accuracy. Specifically, with the increase on the privacy budget, the noise decreases, leading to a higher success on inference attacks as shown in Fig. 6(b). When the budget is too small, the excessive noise degrades accuracy, causing DP schemes to perform worse than HE schemes, as shown in Fig. 6(a). In addition, DP noise must be tailored to different data types in multimodal scenarios, increasing the system complexity, whereas homomorphic encryption provides a universal solution by simply encrypting the data. For example,



(a) The total reward (b) The HHM service throughput (c) The running time

Fig. 7. The performance of offline algorithms by varying the coefficient ς in delay requirements from 0 to 0.8.

pressure sensor data is allowed to inject noises into it, while EEG signals need to add the targeted noise to waveform features like peaks and valleys [22], [33].

3) Impact of Key Parameters on Offline Refreshers: We increased the computing resource capacity C_j on edge server j from $[40,000, 80,000]$ to $[200,000, 240,000]$ MHz and set the number of arrival requests per time slot to 300 to study its impact on the performance of offline algorithms.

Computing resource: As shown in Fig. 5(a), the increase of computing resource capacity on edge servers results in a larger reward for APP, though the reward remains slightly lower than the one delivered by the ILP, and significantly larger than the one by $Random$. Fig. 5(b) indicates that the throughput bottleneck shifts from computing resource to the number of arrival requests after the amount of resources exceeds 180,000 MHz, causing the total throughput to grow initially and then level off. With more available computing resources, the running time of the ILP can decrease a bit but is still significantly higher than any of the other algorithms.

We then study the impact of coefficient ς on algorithm performance by increasing its scale in the Normal distribution $\mathcal{N}((1-\varsigma)\cdot T_{sta}, 0.15\cdot T_{sta})$ of delay requirements from 0 to 0.8 and setting the arrival rate of requests per time slot of 100.

Delay requirements: As ς increases, delay requirements become stricter, leading to a decrease in both reward and throughput for all algorithms, as shown in Fig. 7(a) and (b). An interesting pattern is observed in Fig. 7(c): the running time of the ILP exhibits a double inflection point trend, initially rising, then flattening, and finally falling. Before the first inflection point, the problem is relatively easy to solve, leading to shorter running times. After the second inflection point, the problem becomes very difficult as most requests fail to meet the delay requirements, resulting in low running times. Between these inflection points, the problem is moderately challenging, requiring a reasonable allocation method to optimize the goal, which increases the running time. In contrast, delay requirements have little effect on the running time of the APP and $Random$.

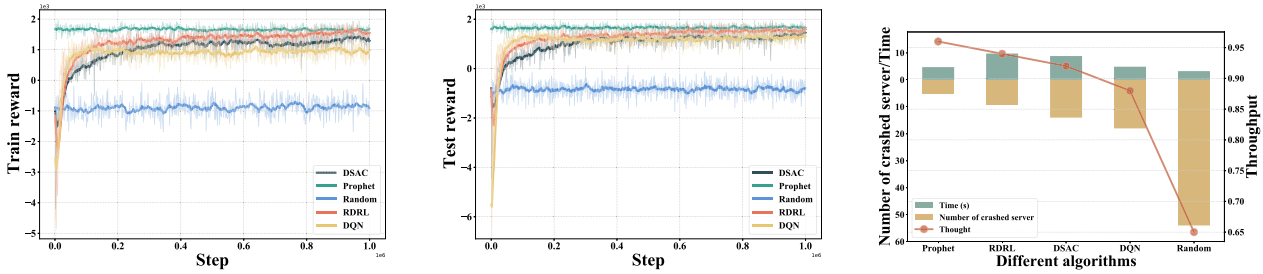
4) Impact of Key Parameters on Online Refreshers: In the following, we first trained all online refreshers with the random seed set to 0 and the training step set to 1×10^6. We then examined the effects of key parameters, including the random seed, congestion coefficient, bandwidth, and the number of HHM requests, on the performance of various algorithms.

Refresher training: As shown in Fig. 8(a) and (b), with an increase in the number of training steps, the reward obtained by each of the comparison algorithms initially raises and then stabilizes. Among them, $Prophet$ ranks first due to its prior information on reward and delay. Following in order are $RDRL$, $DSAC$, and DQN, with $Random$ ranking last. Notably, $RDRL$ with diffusion and residual modules continues to improve even after DQN with neural network converges, and it learns more information than $DSAC$ with only the diffusion module. Additionally, $RDRL$ is closest to the optimal algorithm $Prophet$ in terms of throughput and the number of crashed servers, ranking second. However, these excellent performances come at the cost of running time, which is the longest, as shown in Fig. 8(c).

Random seed: We evaluated the ability of online refreshers to handle out-of-distribution environment states by varying the value of the random seed from 5 to 65. As shown in Fig. 9(a) and (b), in the presence of out-of-distribution data, the throughput and reward of learning-based algorithms decline to some extent. However, $RDRL$ consistently outperforms $DSAC$ and DQN, even though DQN occasionally surpasses $DSAC$. Therefore, the algorithm with diffusion and residual modules, $RDRL$, demonstrates greater stability.

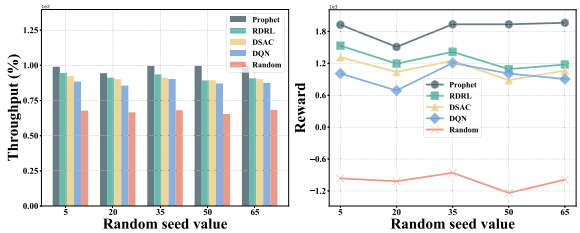
Network congestion coefficient: We then studied the impact of the network congestion coefficient ζ on the proposed algorithm by varying it from 3,000 to 7,000. As the network congestion coefficient increases, the propagation delay of encrypted data gradually rises, leading to reduced latency in the remaining encryption calculations in Fig. 10(a). Consequently, the throughput and reward of all algorithms gradually decline, where the $RDRL$ performs closely behind the optimal algorithm and outperforms the others.

Network bandwidth: We third investigated the impact of the bandwidth on the proposed algorithm by varying it from 0.016 to 16 MB. Fig. 11 illustrates a significant performance gap between $RDRL$ and the optimal algorithm before 1.6 MB, attributed to bandwidth values considered out-of-distribution data. As bandwidth increases, this gap gradually narrows, and the performance advantages of $RDRL$ are gradually highlighted



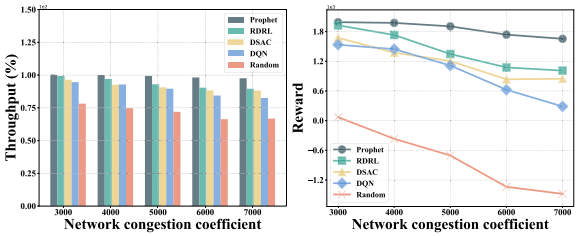
(a) The training reward (b) The test reward (c) The performance comparison on throughput, reward, and running time

Fig. 8. The performance of the online refresher in the training process.



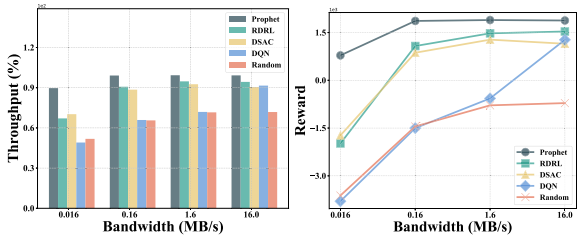
(a) The HHM service throughput (b) The total reward

Fig. 9. The performance of different online algorithms by varying the value of random seed from 5 to 65.



(a) The HHM service throughput (b) The total reward

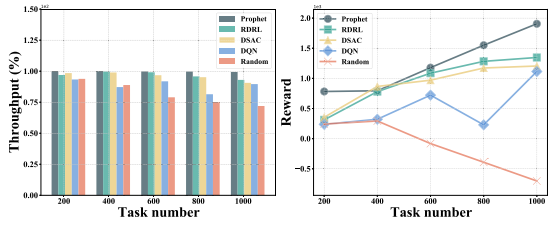
Fig. 10. The performance of different online algorithms by varying network congestion coefficient from 3,000 to 7,000.



(a) The HHM service throughput (b) The total reward

Fig. 11. The performance of different online algorithms by varying the network bandwidth from 0.016 to 16.

compared to $DSAC$. Throughout this progression, the DQN algorithm shows rapid improvement with higher bandwidth. However, it becomes evident that its ability to handle out-of-distribution data is limited.



(a) The HHM service throughput (b) The total reward

Fig. 12. The performance of different online algorithms by varying average request number from 200 to 1,000.

Numbers of HHM requests: Finally, we increased the number of requests over the given time horizon $\mathcal{T}$ from 200 to 1,000, by adjusting parameter λ_p in the Poisson process for request generations. Fig. 12(a) shows that the throughput of all algorithms initially remains high and then slightly decreases. This trend occurs because the number of arriving requests gradually approaches the upper limit of what all servers can concurrently process, although it has not yet exceeded that limit. Furthermore, as the number of requests increases, most algorithms show an increasing trend in reward, while reward decreases in $Random$. This is because $Random$ fails to effectively manage the increasing number of requests, resulting in a reduction in total reward, as depicted in Fig. 12(b).

VI. Conclusion

In this paper, we investigated the privacy-enhanced HHM service refreshment maximization problem in the fabric metaverse. To this end, we proposed the `Wiper` system comprising a shallow-deep distiller and an agile refresher library to address the challenges that integrate data encryption, model compression, and personalized health refresh of digital avatars. The distiller replaces the violation modules with approximate ones, and compresses model layers and parameters to reduce refreshment delays. The library integrates an exact, an approximate, and a novel residual diffusion reinforcement learning algorithms to maximize the throughput of encrypted HHM services across different scenarios, respectively. Compared with existing methods, experimental results showed significant performance improvements in both accuracy and throughput.

ACKNOWLEDGMENT

The authors appreciate three anonymous referees and the Associate Editor for their constructive comments and invaluable suggestions, which help us to improve the quality and presentation of the paper greatly.

REFERENCES

[1] M. Bun and T. Steinke, "Concentrated differential privacy: Simplifications, extensions, and lower bounds," in *Proc. Theory Cryptography Conf.*, Springer, 2016, pp. 635–658.
[2] J. Chen et al., "A revolution of personalized healthcare: Enabling human digital twin with mobile AIGC," *IEEE Netw.*, vol. 38, no. 6, pp. 234–242, Nov. 2024.
[3] J. Chen, C. Yi, S. D. Okegbile, J. Cai, and X. S. Shen, "Networking architecture and key supporting technologies for human digital twin in personalized healthcare: A comprehensive survey," *IEEE Commun. Surv. Tuts.*, vol. 26, no. 1, pp. 706–746, First Quarter 2024.
[4] J. Chen et al., "Digital twin empowered wireless healthcare monitoring for smart home," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3662–3676, Nov. 2023.
[5] M. Chen et al., "Imperceptible, designable, and scalable braided electronic cord," *Nature Commun.*, vol. 13, no. 1, 2022, Art. no. 7097.
[6] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. 23rd Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Springer, 2017, pp. 409–437.
[7] H.-J. Chien, H. Khalili, A. Hass, and N. Sehatbakhsh, "Enc2: Privacy-preserving inference for tiny IoTs via encoding and encryption," in *Proc. 29th Annu. Int. Conf. Mobile Comput. Netw.*, 2023, pp. 1–16.
[8] S. Disabato, A. Falcetta, A. Mongelluzzo, and M. Roveri, "A privacy-preserving distributed architecture for deep-learning-as-a-service," in *Proc. 2020 Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.
[9] T. V. T. Doan, M.-L. Messai, G. Gavin, and J. Darmont, "A survey on implementations of homomorphic encryption schemes," *J. Supercomputing*, vol. 79, no. 13, pp. 15098–15139, 2023.
[10] H. Du et al., "Diffusion-based reinforcement learning for edge-enabled AI-generated content services," *IEEE Trans. Mobile Comput.*, vol. 23, no. 9, pp. 8902–8918, Sep. 2024.
[11] S. Gupta, R. Garg, N. Gupta, W. S. Alnumay, U. Ghosh, and P. K. Sharma, "Energy-efficient dynamic homomorphic security scheme for fog computing in IoT networks," *J. Inf. Secur. Appl.*, vol. 58, 2021, Art. no. 102768.
[12] F. Hu and B. Chen, "Channel coding scheme for relay edge computing wireless networks via homomorphic encryption and NOMA," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1180–1192, Dec. 2020.
[13] Z. Huang, W. J. Lu, C. Hong, and J. Ding, "Cheetah: Lean and fast secure Two-Party deep neural network inference," in *Proc. 31st USENIX Secur. Symp.*, 2022, pp. 809–826.
[14] A. Ibarrondo and A. Viand, "Pyfhel: Python for homomorphic encryption libraries," in *Proc. 9th Workshop Encrypted Comput. Appl. Homomorphic Cryptography*, 2021, pp. 11–16.
[15] B. Kang, X. Ma, C. Du, T. Pang, and S. Yan, "Efficient diffusion policies for offline reinforcement learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, pp. 67195–67212.
[16] A. Kim et al., "General bootstrapping approach for RLWE-based homomorphic encryption," *IEEE Trans. Comput.*, vol. 73, no. 1, pp. 86–96, Jan. 2024.
[17] M. Kim, X. Jiang, K. Lauter, E. Ismayilzada, and S. Shams, "Secure human action recognition by encrypted neural network inference," *Nature Commun.*, vol. 13, no. 1, 2022, Art. no. 4799.
[18] B. Li, D. Micciancio, M. Schultz-Wu, and J. Sorrell, "Securing approximate homomorphic encryption using differential privacy," in *Proc. Annu. Int. Cryptol. Conf.*, Springer, 2022, pp. 560–589.
[19] J. Liu and M. Chen, "FaGeL: Fabric LLMs agent empowered embodied intelligence evolution with autonomous human-machine collaboration," 2024. [Online]. Available: https://arxiv.org/abs/2412.20297
[20] Y. Li, W. Liang, Z. Xu, W. Xu, and X. Jia, "Budget-constrained digital twin synchronization and its application on fidelity-aware queries in edge computing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 165–182, Jan. 2025.
[21] X. Liang, W. Liang, Z. Xu, Y. Zhang, and X. Jia, "Multiple service model refreshments in digital twin-empowered edge computing," *IEEE Trans. Serv. Comput.*, vol. 17, no. 5, pp. 2672–2686, Sep./Oct. 2024.
[22] C. Lin, Z. Song, H. Song, Y. Zhou, Y. Wang, and G. Wu, "Differential privacy preserving in big data analytics for connected health," *J. Med. Syst.*, vol. 40, pp. 1–9, 2016.
[23] C. U. Manual, "IBM ILOG CPLEX optimization studio," *Version*, vol. 12, no. 1, pp. 1987–2018, 1987.
[24] D. Micciancio and M. Walter, "On the bit security of cryptographic primitives," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, Springer, 2018, pp. 3–28.
[25] C.-H. Nguyen et al., "Encrypted data caching and learning framework for robust federated learning-based mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2705–2720, Jun. 2024.
[26] S. D. Okegbile, J. Cai, H. Zheng, J. Chen, and C. Yi, "Differentially private federated multi-task learning framework for enhancing human-to-virtual connectivity in human digital twin," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3533–3547, Nov. 2023.
[27] Z. Peng, Y. Xu, R. Kang, Y. Liu, and Y. Zhang, "Cooperative multiagent deep reinforcement learning for computation offloading in digital twin cyber–physical–social system," *IEEE Trans. Computat. Social Syst.*, early access, Mar. 17, 2025, doi: 10.1109/TCSS.2025.3548476.
[28] Y. Qiu et al., "Spotlighter: Backup age-guaranteed immersive virtual vehicle service provisioning in edge-enabled vehicular metaverse," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 13375–13391, Dec. 2024.
[29] Y. Qiu et al., "Reliable or green? Continual individualized inference provisioning in fabric metaverse via multi-exit acceleration," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11449–11465, Dec. 2024.
[30] Y. Qiu, J. Liang, V. C. M. Leung, X. Wu, and X. Deng, "Online reliability-enhanced virtual network services provisioning in fault-prone mobile edge cloud," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7299–7313, Sep. 2022.
[31] Y. Qiu, J. Liang, V. C. Leung, and M. Chen, "Online security-aware and reliability-guaranteed AI service chains provisioning in edge intelligence cloud," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 5933–5948, May 2024.
[32] D. Reis, J. Takeshita, T. Jung, M. Niemier, and X. S. Hu, "Computing-in-memory for performance and energy-efficient homomorphic encryption," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 11, pp. 2300–2313, Nov. 2020.
[33] H. Xu, W. Li, D. Takabi, D. Seo, and Z. Cai, "Privacy-preserving multimodal sentiment analysis," *IEEE Internet Things J.*, vol. 12, no. 11, pp. 15467–15478, Jun. 2025.
[34] Y. Xu et al., "Enhancing privacy in cyber–physical systems: An efficient blockchain-assisted data-sharing scheme with deniability," *J. Syst. Archit.*, vol. 150, 2024, Art. no. 103132.
[35] L. Yu, S. Jiang, J. Zheng, F. Yan, and S. Zhao, "A DQN-based joint computing offloading and resource allocation algorithm for MEC networks," in *Proc. 2023 IEEE Int. Conf. Commun.*, 2023, pp. 2553–2558.
[36] Y. Zhang, W. Liang, Z. Xu, and X. Jia, "Mobility-aware service provisioning in edge computing via digital twin replica placements," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11295–11311, Dec. 2024.
[37] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, "Decoupled knowledge distillation," in *Proc. 2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11943–11952.
[38] Y. Zheng, R. Lu, Y. Guan, S. Zhang, and J. Shao, "Towards private similarity query based healthcare monitoring over digital twin cloud platform," in *Proc. IEEE/ACM 29th Int. Symp. Qual. Serv.*, 2021, pp. 1–10.
[39] H. Zhou et al., "Toward human motion digital twin: A motion capture system for human-centric applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 619–630, 2025.

Yu Qiu received the BSc degree in pharmaceutical engineering from the Tianjin University of Technology, Tianjin, China, in 2020, and the ME degree in computer technology from Guangxi University, Nanning, China, in 2023. He is currently working toward the PhD degree in artificial intelligence with the South China University of Technology, Guangzhou, China. His research interests include digital health, metaverse, edge intelligence, network function virtualization, and optimization theory.

Min Chen (Fellow, IEEE) has been a full professor with the School of Computer Science and Engineering, South China University of Technology. He is also the director of Embedded and Pervasive Computing (EPIC) Lab, Huazhong University of Science and Technology. He is the founding chair of IEEE Computer Society Special Technical Communities on Big Data. He was an assistant professor with the School of Computer Science and Engineering, Seoul National University before he joined HUST. He is the chair of IEEE Globecom 2022 eHealth Symposium. His Google Scholar Citations reached 51,250+ with an h-index of 101. His top paper was cited 5,150+ times. He was selected as highly cited researcher from 2018 to 2024. He got IEEE Communications Society Fred W. Ellersick Prize in 2017, the IEEE Jack Neubauer Memorial Award in 2019, and IEEE ComSoc APB Oustanding Paper Award in 2022. He is a fellow of the IET.

Weifa Liang (Senior Member, IEEE) received the BSc degree from Wuhan University, China, in 1984, the ME degree from the University of Science and Technology of China, in 1989, and the PhD degree from the Australian National University, in 1998, all in computer science. He is a full professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he held the positions from lecturer to full professor with the Australian National University more than 20 years. His research interests include wireless ad hoc and sensor networks, edge and cloud computing, machine learning, Internet of Things and digital twins, design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He currently serves as an editor of the *IEEE Transactions on Communications*.

Lejun Ai received the BSc degree from the South China University of Technology, Guangzhou, China, in 2024. His current research interests include fabric computing, automatic sleep staging, and electroencephalogram.

Dusit Niyato (Fellow, IEEE) received the BEng degree from the King Mongkuts Institute of Technology Ladkrabang (KMITL), Thailand, in 1999, and the PhD degree in electrical and computer engineering from the University of Manitoba, Canada, in 2008. He is a professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include the areas of sustainability, edge intelligence, decentralized machine learning, and incentive mechanism design.

Gang Wei received the BSc, MSc, and PhD degrees from Tsinghua University and South China University of Technology, in 1984, 1987, and 1990, respectively. He is currently a professor with the Department of Electronic Engineering, South China University of Technology. His research interests include personal communications, emotion care, AI music, and audio generation.