2024 IEEE 30th International Conference on Parallel and Distributed Systems (ICPADS) | 979-8-3315-1596-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICPADS63350.2024.00031

LiteCrypt: Enhancing IoMT Security with Optimized HE and Lightweight Dual-Authorization

Qipeng Xie[*§], Weizheng Wang[†], Yongzhi Huang[¶], Mengyao Zheng[‖], Shuai Shang[**], Linshan Jiang[‡], Salabat Khan[*], Kaishun Wu[*¶], *Fellow, IEEE*

[*]IoT Thrust/ [¶]DSA Thrust, Information Hub, Hong Kong University of Science and Technology (Guang Zhou), Guang Zhou, China

[†]Computer Science Engineer, City University of Hong Kong, Hong Kong, China

[§]Hong Kong University of Science and Technology, Hong Kong, China

[‡]Institute of Data Science, National University of Singapore, Singapore

[‖]Harvard T.H. Chan School of Public Health, Boston, United States

[**] University of Electronic Science and Technology of China, Chengdu, Chengdu, China

***Abstract*—The integration of 5G/6G networks with intelligent healthcare systems has enabled early disease detection through patient data monitoring. However, the Internet of Medical Things (IoMT) and remote healthcare services introduce significant privacy and security risks. In this paper, we propose LiteCrypt, which addresses these challenges by introducing an optimized Homomorphic Convolutional Neural Networks (HCNN) structure for secure inference and a lightweight Threshold Signature Scheme (TSS) based dual-authorization mechanism. To enhance the practicality of Homomorphic Encryption (HE)-based secure inference in telemedicine applications, LiteCrypt presents an optimized HCNN framework that ensures efficient and adaptable operations across multiple datasets. A high-performance GPU-accelerated HE engine is developed to address the computational demands of HE operations, enabling real-time processing of encrypted patient data. Besides, LiteCrypt introduces a novel TSS-based dual-authorization protocol, requiring consent from both the patient and the hospital to access patient data, thereby mitigating unauthorized access risks. The system adapts to a flexible 2-out-of-3 authorization scheme for emergencies, ensuring timely data retrieval while maintaining security. To overcome the initial challenge of prolonged computation time due to compute-intensive operations, In LiteCrypt, we utilized the lightweight TSS protocol, based on Oblivious Transfer (OT), which is designed for resource-constrained IoMT devices, reducing computation time from 11.9 to 0.11 seconds. Empirical validation demonstrates LiteCrypt's superior performance, achieving a 233-fold increase in processing speed, a 96% reduction in encrypted message size, and a 28-fold speed increase using GPUs.**

***Index Terms*—Homomorphic Encryption, Machine Learning, GPU Acceleration, Threshold Signature.**

I. Introduction

The proliferation of Internet of Things (IoT) devices has seamlessly integrated into various aspects of our lives, spanning applications in sensing [1]–[6], dietary management [7], industrial automation [8], [9], and more. The smart healthcare system [10]–[12], enhances traditional healthcare by using monitoring technologies such as remote patient monitoring and telemedicine, supported by 5G and 6G networks. This allows real-time data collection and analysis of patient health, leading to early detection of medical issues, reduced hospitalizations, and improved overall outcomes. The key to this system is the Internet of Medical Things (IoMT) [13], [14], which enables two-way communication for remote adjustments and interventions [15], [16]. In addition, machine learning (ML) algorithms play a crucial role in analyzing patient data to predict future health trends.

IoMT devices are designed with limited resources, which means they only include basic security features, making patient data vulnerable to privacy breaches. Moreover, common ML algorithms can also pose privacy risks. To address these issues, some researchers have explored using ML based on homomorphic encryption (HE) [17] for secure predictions. In particular, several HE-based frameworks [18], [19] have been proposed. However, these approaches often require considerable memory resources. For example, Brutzkus et al. [20]'s technique requires 12 GB of RAM for a single secure prediction on the MNIST dataset. And Lou et al. [21]'s method significantly enlarges each message transmission to 55MB. These huge costs make them unsuitable for IoMT devices with limited resources.

Significant delays result from the heavy demand for computational resources. Even with frameworks designed for reduced latency [20], [21], processing times of 730 and 107 seconds are far from the needs of real-time applications. The use of GPU acceleration [22] still results in a processing time of 304.43 seconds for the same dataset. It highlights the fundamental mismatch between these optimization methods and the requirements for real-time processing, demonstrating their impracticality for immediate application scenarios in healthcare.

Beyond secure inference, the security of medical information exchange is also vital. Traditional systems rely on centralized authentication and key distribution. This approach risks single points of failure and requires significant management, possibly leading to compromised security centers and internal threats. Moreover, current Threshold Signature Schemes (TSS) struggle due to the computational load from complex Zero-Knowledge Proofs (ZKP) and commitment schemes, which makes them unsuitable for IoMT devices with limited processing power.

These challenges underscore the urgent need for an efficient,

979-8-3315-1596-6/24/$31.00 ©2024 IEEE
DOI 10.1109/ICPADS63350.2024.00031

low-memory, privacy and security focused approach to ML data analysis in IoMT-based remote healthcare. This work aims to address this need by tackling three major challenges: *(1) How to optimize the structure of Homomorphic Convolutional Neural Networks (HCNN) and develop a GPU-accelerated HE engine to improve the efficiency and adaptability of Homomorphic Encryption (HE) in real-world healthcare scenarios for real-time processing and analysis of encrypted patient data?* To effectively address critical real-time demands in remote healthcare, it is crucial to minimize computational delays. This requires the innovation of algorithms and systems that facilitate the rapid processing of encrypted data, thereby reducing latency and enhancing clinical outcomes. *(2) How to develop a lightweight TSS protocol that is suitable for deployment on IoMT devices, eliminating the need for complex Zero-Knowledge Proof (ZKP) operations and adapting to the resource limitations of these devices? (3) How to introduce a secure multi-party authorization mechanism that ensures patient data security while allowing timely data retrieval in emergency situations, mitigating single points of failure and unauthorized access risks associated with traditional single authorization mechanisms?* With the increase in unauthorized data access attempts, it is imperative to fortify medical data security. It is essential to adopt suitable cryptographic schemes that prevent cyberattacks and protect patient data privacy while incurring minimal overhead.

To address these challenges, this paper proposes the following solutions:

1) **Distribute Key Generation:** We introduce a novel key generation algorithm for remote health applications, eliminating the need for a trusted dealer while ensuring secure and reliable key generation. ❶
2) **Lightweight TSS Protocols for IoMT:** We adopt TSS protocols specifically for the low-power requirements of embedded systems and IoMT, enhancing security without compromising device capabilities. ❷
3) **Streamlined Framework for HE Applications:** Our approach introduces an optimized structural framework for HE, ensuring efficient and adaptable operations across multiple datasets. ❸
4) **GPU Acceleration Engine:** We utilize GPUs' thread-level parallelism with a custom acceleration engine to markedly increase computational efficiency, achieving significant processing speed improvements. ❸
5) **Empirical Validation and Performance Gains:** Through rigorous testing on three datasets, our methodology demonstrated a 233-fold increase in processing speed and reduced encrypted message sizes to 4% of traditional methods. Additionally, our GPU-based solution further amplified processing speeds by 28 times. ❸

II. Background and Preliminary

In this section, we delve into the technical background that underpins our research. Our work stands out in two significant ways. Firstly, we introduce the application of the Distributed Key Generation (DKG) method within the healthcare context, specifically focusing on secure and efficient requests for medical data between patients and healthcare providers. This novel application addresses the critical need for robust data security and integrity in medical communications. Secondly, our Homomorphic Encryption (HE) network design is meticulously tailored to align with subsequent GPU acceleration algorithms, ensuring both computational efficiency and scalability.

A. Notation

In this work, we use the following notations and mathematical concepts. $\mathbb{Z}$, $\mathbb{C}$ denote the ring of integers and complex numbers. We use $R = \mathbb{Z}[X]/(X^N+1)$ with N being power-of-two to denote the ring of polynomials modulo (X^N+1). Polynomials, which are elements of the ring $\mathcal{R}$, are denoted by bold, italic lowercase letters. The symbol $\boldsymbol{f} \leftarrow \mathcal{S}$ signifies that the polynomial $\boldsymbol{f}$ is chosen according to a specific distribution $\mathcal{S}$. We represent the encryption function as $[\![\cdot]\!]$ to denote $\mathsf{Enc}_{\mathtt{pk}}(\cdot)$ under the public key $\mathtt{pk}$, and the decryption function as $\mathsf{Dec}_{\mathtt{sk}}(\cdot)$ under the secret key $\mathtt{sk}$. We use the notation $[a]_q$ to signify the integer equivalent of a modulo q. Regarding homomorphic operations, for clarity, we assign a particular notation: HAdd denotes the addition of two ciphertexts, while HMult corresponds to the multiplication of two ciphertexts. Additionally, the rotation operation with a step size of i is denoted as HRot_i.

B. CKKS Scheme

Cheon-Kim-Kim-Song (CKKS) scheme [23] emerges as one of the most noteworthy, primarily owing to its support for floating-point and complex numbers in the real of Fully Homomorphic Encryption (FHE) schemes. In the following, we briefly introduce the scheme of CKKS.

Encoding: In the CKKS scheme, a single plaintext can encode up to $N/2$ distinct messages into its individual slots. This facilitates batch processing, thereby enhancing flexibility. This scheme can accommodate varying input data types, including floating-point and complex numbers. CKKS employs $\mathsf{Ecd}_\Delta : \mathbb{C}^{N/2} \to \mathcal{R}$ by a scaling factor Δ.

Encryption and Decryption: In the encryption and decryption process of CKKS, the secret key is composed of a random ring element $\boldsymbol{s} \in R$ sampled following the distribution $\mathcal{X}_k$, and is represented as $\mathtt{sk} := (1, \boldsymbol{s})$. The public key, denoted as $\mathtt{pk} := (\boldsymbol{b}, \boldsymbol{a})$, is formulated as $([-\boldsymbol{a}\cdot\boldsymbol{s}+\boldsymbol{e}]_{Q_L}, \boldsymbol{a}) \in R_Q^2$, where $\boldsymbol{a} \xleftarrow{\$} R_{Q_L}$ and $\boldsymbol{e} \leftarrow \mathcal{X}_e$. For the encryption of a plaintext $\boldsymbol{m}$, the resulting ciphertext $\mathsf{ct} := (\boldsymbol{c}_0, \boldsymbol{c}_1)$ is computed as $\mathsf{Enc}_{\mathtt{pk}}(\boldsymbol{m}) := [\boldsymbol{r}\cdot(\boldsymbol{b},\boldsymbol{a}) + (\boldsymbol{e}_0+\boldsymbol{m}, \boldsymbol{e}_1)]_{Q_L} \in R_{Q_L}$, where $\boldsymbol{r} \leftarrow \mathcal{X}_k$ and $\boldsymbol{e}_0, \boldsymbol{e}_1 \leftarrow \mathcal{X}_e$. Conversely, in level l, to decrypt the ciphertext $\boldsymbol{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1) \in R_{Q_l}^2$, the decryption result can be computed as $\boldsymbol{m}' := [\boldsymbol{c}_0 + \boldsymbol{c}_1 \cdot \boldsymbol{s}]_{Q_l}$.

Homomorphic Operators: In the context of the CKKS, homomorphic evaluation is achieved through various operations. Taking the ring elements $\boldsymbol{x}$, $\boldsymbol{y}$ as two plaintexts, with $\boldsymbol{x}$ encoding a vector $\mathbf{x}$.

- **Addition** ($\oplus$): $\mathsf{Dec}([\![\boldsymbol{x}]\!] \oplus [\![\boldsymbol{y}]\!]) = \boldsymbol{x}+\boldsymbol{y}, \mathsf{Dec}([\![\boldsymbol{x}]\!] \oplus \boldsymbol{y}) = \boldsymbol{x}+\boldsymbol{y}$.

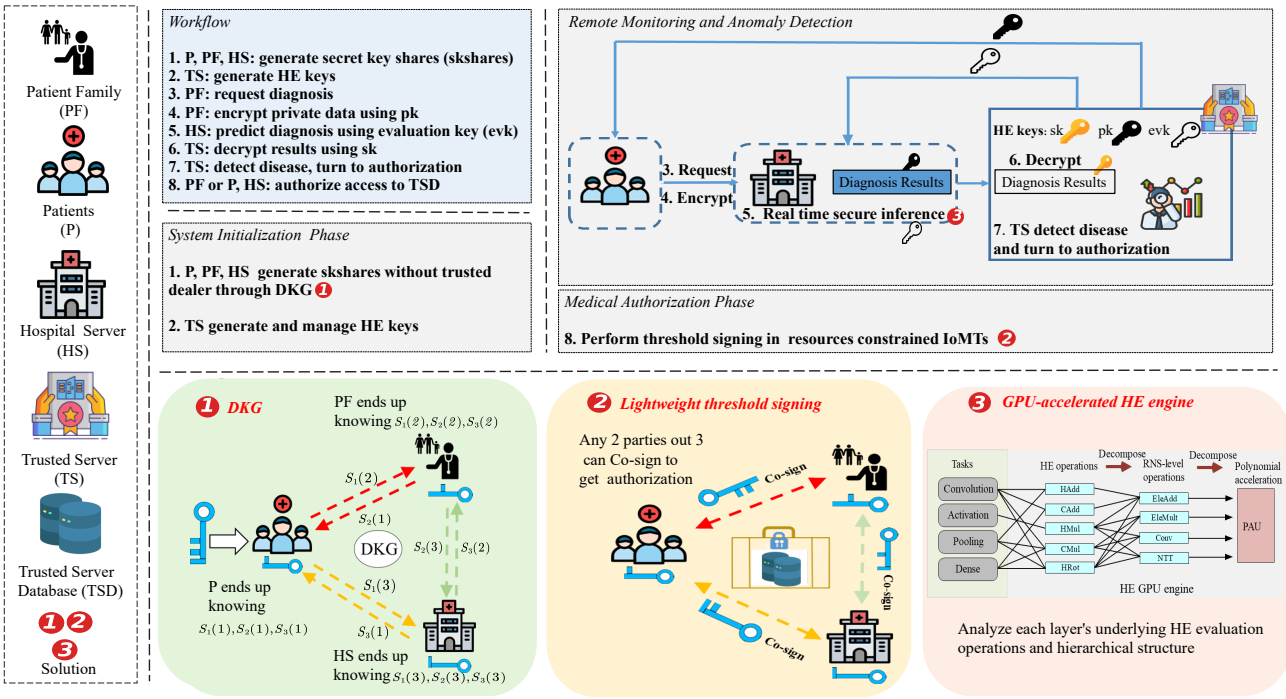


Fig. 1: Overview of LiteCrypt

- **Multiplication** ($\otimes$): $\mathsf{Dec}(\llbracket \boldsymbol{x} \rrbracket \otimes \llbracket \boldsymbol{y} \rrbracket) = \boldsymbol{x} \cdot \boldsymbol{y}, \mathsf{Dec}(\llbracket \boldsymbol{x} \rrbracket \otimes \boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{y}$.
- **Rotation** ($\mathsf{HRot}_i(\cdot)$): $\mathsf{Dec}(\mathsf{HRot}_i(\llbracket \boldsymbol{x} \rrbracket)) = \mathsf{Ecd}(\mathbf{x} \ll i)$

Residue Number System: The Residue Number System (RNS) is a numerical representation that can enhance parallelism and modularity in computations. In RNS, we can represent a large integer belonging to $\mathbb{Z}_{\mathbb{Q}}$ as a set of smaller residues in $\mathbb{Z}_{q_i}$, where $Q := \Pi_{i=0}^{L-1} q_i$ Consequently, a ring element $x \in R_Q$ is represented as: $[x]_{Q_L} = (x^{(0)}, x^{(1)}, \ldots, x^{(L-1)}) \in R_{q_0} \times R_{q_1} \times \cdots \times R_{q_{L-1}}$. This approach substantially enhances computational efficiency through the parallelization techniques, a crucial factor in managing large-scale operations.

C. HE-enabled Neural Network

In Homomorphic Convolutional Neural Networks (HCNNs), the arrangement of neurons is meticulously structured across several layers. These can be categorized into linear and nonlinear layers based on their respective functionalities:

Linear Layers: These layers are the foundational elements of HCNNs, and encompass both convolutional and fully connected layers. Their primary operation involves linear transformations, succinctly expressed as $\llbracket \boldsymbol{y} \rrbracket = w \otimes \llbracket \boldsymbol{x} \rrbracket \oplus \boldsymbol{b}$, where $\boldsymbol{w}$ and $\boldsymbol{b}$ denote the weight matrix and bias vector after encoding, respectively. Here, $\boldsymbol{x}$ signifies the encoded input data, and $\boldsymbol{y}$ stands for the output. Specifically, convolutional layers use a filter to selectively process a subset of the output of the previous layer for each neuron, while fully connected layers consider input from all neurons in the preceding layer.

Nonlinear Layers: These layers primarily handle nonlinear operations through activation functions, playing a crucial role in mitigating overfitting within the model. Homomorphic evaluations of these nonlinear functions typically require polynomial approximations. A common choice for the activation function is the Rectified Linear Unit (ReLU), defined element-wise as $y = \max(0, x)$. Normalization of image pixel values to the [0,1] range as floating-point numbers is a standard pre-processing step, aligning with the input domain of the CKKS scheme. Additionally, vector-wide encoding into a single plaintext is possible, which facilitates batch processing - a technique that substantially improves computational efficiency. Such harmonization between data input representation and the encryption scheme is beneficial for leveraging Homomorphic Encryption (HE) in neural network applications.

D. Threshold Signature

A Threshold Signature Scheme (TSS) is a cryptographic protocol that allows multiple parties to collaboratively produce a digital signature without relying on centralized private key management. Formally, a TSS can be represented by a collection of three algorithms:

Distribute Key Generation Algorithm (DKG): $DKG(1^n) \rightarrow (pk, sk_1, sk_2, \cdots, sk_n)$: pk is the public key for common usage, and sk_i, are secret shares of the private key, securely divided among n participants. This algorithm is responsible for creating the global public key pk and distributing secret shares SK_i, to each participant.

Threshold Signing Algorithm: $Sign(sk, m) \rightarrow \sigma$: This algorithm allows any group of participants holding $t+1$ secret shares or more to collectively sign a message m. Each participant uses their secret share sk_i to produce a partial signature σ_i. These partial signatures are then combined to form the final signature σ. Once a sufficient number of partial signatures are collected, a combining algorithm merges them

into a complete signature. This complete signature σ then acts as the valid signature for the message m.

Verification Algorithm: $Verify(pk, m, \sigma) \rightarrow \{true, false\}$: Anyone with access to the public key pk can verify whether the complete signature σ is a valid signature for the message m. If σ is correctly formed from at least $t + 1$ valid partial signatures, the verification algorithm returns true; otherwise, it returns false.

III. System design

In this section, we mainly discuss the network model, threat model, and the proposed scheme of LiteCrypt.

A. Network model in LiteCrypt

Patient (P): This client is responsible for collecting medical data from patients, such as electrocardiograms and daily activity videos. The data are encrypted using a public key provided by the Trusted Server (TS) to ensure privacy and security during transmission and storage. In addition, the patient-client participates in generating threshold signatures to safeguard medical records from unauthorized access by the hospital.

Patients' Family (PF): When a patient is unable to make medical decisions, the family member client allows relatives to participate in the threshold signature process on behalf of the patient, authorizing the use of medical data. This mechanism safeguards the patient's interests during emergency situations.

Hospital Server (HS): This server operates a GPU-accelerated homomorphic encryption engine and pre-trained HE-adapted machine learning models, enabling the hospital to perform efficient and secure analysis without decrypting the data. It is also involved in the threshold signature process, authorizing access to patient records.

Trusted Server (TS): Acting as a trusted third party, TS is responsible for key management, case storage, and monitoring and responding to abnormal patient behavior. Access to patient cases stored in TS database is granted only after authorization through a threshold signature. Additionally, TS is in charge of generating and distributing homomorphic encryption keys during system initialization.

B. Threat Model

In this subsection, we present the threat model considered in the design of LiteCrypt. Our threat model encompasses various types of adversaries and their potential malicious behaviors, which our system aims to mitigate. In our system, we consider the following threat model:

- **Semi-honest adversaries:** We assume that all parties involved (patients, patients' families, hospital server, and trusted server) are "honest-but-curious." That is, they will follow the protocol as specified but may attempt to infer additional information from the messages they receive.
- **External adversaries:** We consider external adversaries who may attempt to eavesdrop on or tamper with data on the communication channels. However, we assume that the adversaries do not have sufficient computational power to break the homomorphic encryption or threshold signature schemes within an acceptable time frame.
- **Insider threats:** We consider scenarios where a party within the system, such as the hospital server, may attempt to access a patient's medical records without authorization. Our scheme mitigates this threat through the threshold signature mechanism.

C. Proposed scheme

Fig. 1 shows the overview of LiteCrypt. We describe LiteCrypt as follows. Our protocol includes four stages: 1) System initialization; 2) Data Collection and Encrypted Phase; 3) Remote Monitoring and Anomaly Detection Phase; 4) Medical Authorization Phase.

System Initialization Phase: Patients, hospital sever, and family members execute the Distributed Key Generation (DKG) algorithm over a secure channel to generate (2,3) share of the threshold signature key. Then, TS generates an evaluation key **`evk`** and securely distributes it to HS, together with the public key of CKKS **`pk`** to the patient's client.

Remote Monitoring and Anomaly Detection Phase: Patient clients gather medical data from patients, such as electrocardiograms [24] and daily activity videos, encrypt the collected data with the **`pk`**, and upload it to the hospital server. Following this, the hospital server performs real-time behavior analysis or anomaly event detection on encrypted patient data. Then, upon detecting an anomaly (e.g., a fall), the hospital server outputs the result (still encrypted) and sends it to the TS. Then, TS decrypts the detection result using **`sk`**. If an anomaly is detected, TS will immediately notify the patient.

Medical Authorization Phase: Upon notification, the patient initiates a medical request, selecting the hospital as a participant in the threshold signature. Second, if the patient fails to respond in a timely manner, the hospital takes the initiative to forward the medical request to the designated patient family automatically. The patient's family, upon notification, selects the hospital to engage in Threshold Signing and communicates this decision to the hospital. Subsequently, the hospital, having received authorization from either the patient or the family member, proceeds with the Threshold Signature Generation, successfully creating a complete signature for the medical request. Finally, this complete threshold signature is dispatched to the TS. Then TS employs a verification algorithm to ascertain the signature's legitimacy. Upon validation, the hospital is granted the necessary authorization to access the patient's electronic medical records.

IV. Implementation Details and Optimizations

In this section, we delve into the implementation details and optimization methods that form the core of our approach. These details are pivotal to achieving the efficiency and lightweight characteristics of our system, revealing the innovative strategies that allow us to adhere to stringent performance and resource constraints.

A. HCNN structure

In pursuing a resource-optimized approach to model deployment, we advocate for a singular, comprehensive model instead of a collection of specialized models. We introduce a compact and cohesive structure of the HCNN detailed in Table I. Our methodology is tailored to homomorphic evaluation requirements, featuring input transformation and simplification, efficient feature management, and classification processes within a unified framework, thereby enhancing resource efficiency.

During the inference stage of our HCNN, various layer types are orchestrated to extract and process features from the input data. The initial convolutional layer applies a suite of eight 3×3 filters on the batched inputs, generating a set of ciphertexts that embody the preliminary features. The filter weights and biases are integrated into plaintexts via stacked mapping and combined with encrypted input vectors, transitioning the input from a dimensionality of 1024×9 to an output of 1024×8. Zero-padding is employed on the images to expand their dimensions to 32×32, which, in conjunction with the modest filter size, ensures thorough feature extraction capable of identifying fine-grained patterns.

Subsequent to the convolution, an activation function is engaged, endowing the model with the capacity to capture more intricate data relationships. The activated output is then streamlined through an average pooling process with a kernel size and stride of 4, reducing the feature map dimensions to 64×8. This pooling step abstracts the feature representation and curbs overfitting while concurrently diminishing computational overhead by decreasing dimensional complexity.

The process concludes with the feature maps undergoing refinement through a sequence of dense layers, which perform linear transformations to integrate features within a broader context. With 128 and 10 filters in these dense layers, the architecture effectively narrows the feature dimensions while maintaining the complexity required for robust analysis, achieving an optimized balance for private inference tasks.

B. Hierarchical Decomposition and Acceleration of Layers

To expedite the computation of HCNN layers, we dissect the HE evaluation operations that are fundamental to each layer and their hierarchical construction. Our bespoke GPU acceleration engine is utilized to fine-tune these operations for unparalleled performance. The main reason is that our structure design improves computational efficiency by enabling more parallel processing within the specially designed Homomorphic Encryption (HE) network on the GPU. The methodology of our engine implementation and the acceleration of HE operations within the layers are depicted in Fig. 1.

GPU Acceleration Engine: We follow the RNS level operations that are innately amenable to batch processing on GPUs owing to the parallel processing capabilities for individual residues, which are pivotal to HE operations. We leveraged this attribute to devise a Polynomial Acceleration Unit (PAU), the linchpin of our GPU acceleration engine. The PAU's role is to bolster the efficiency of polynomial arithmetic within the RNS context, achieving this through effective residue batching, thus facilitating more streamlined computations. The principal operations of the PAU include four parts: EleAdd, EleMult, Conv, and NTT. For EleAdd, we execute modular addition, denoting an element-wise addition operation on polynomials. This operation highlights the straightforward yet vital aspect of polynomial arithmetic—addition at the elemental level. EleMult embodies the Hadamard (element-wise) product of two polynomials, indicating a core function of the PAU's ability to manage complex polynomial operations. Conv designates the base conversion operation, illustrating the PAU's adeptness at enabling flexible arithmetic operations across various numerical bases. NTT denotes the number-theoretic transform, a pivotal tool in polynomial arithmetic for converting polynomials across domains to quicken polynomial multiplications.

For each operation, we dedicate a thread block to process each residue, with these blocks composed of 128 threads. We batch these thread blocks within each kernel, allowing concurrent processing of RNS residues. For the element-wise operations EleAdd, EleMult, and Conv, each thread addresses a single coefficient of the residues, executing the relevant operation.

For NTT operations, a hierarchical method is employed, following cutting-edge techniques as outlined in [25], [26]. Our implementation is specifically tailored to $N = 2^{14}$, encompassing 14 levels of processing. This process is divided into two distinct kernels, each taking responsibility for different levels of the NTT. The first kernel handles the initial 8 levels, while the second manages the subsequent 6 levels. In each kernel invocation, individual threads are responsible for manipulating eight coefficients apiece, which are retained within the register file for rapid access and processing during the transformation phase. Subsequently, the shared memory of the GPU serves as the conduit for inter-thread communication, thereby enabling seamless data interchange.

Reconstruction of HE Operations: Utilizing this unit, we construct homomorphic evaluation operations. We show the above details in Algorithm 1. Each HE operation is reconstructable via RNS level operations. Specifically, the operations HAdd and CAdd are comprised of EleAdd. On the contrary, the operations for ciphertext-ciphertext multiplication HMult, plaintext-ciphertext multiplication CMult, and rotation HRot are more intricate, involving EleMult, EleAdd, Conv, and NTT, respectively.

Employing a hierarchical decomposition strategy enables us to streamline the computation within HCNNs, simplifying the execution of complex operations and heightening efficiency. Furthermore, our GPU acceleration engine ensures that these operations are executed swiftly, enhancing the overall prowess of our HCNN framework.

C. Distributed Key Generation

For a threshold Signature scheme with t out of n parties, each party (P, PS, HS) generates a polynomial of degree t, $S_1(x) \dots S_n(x)$ and exchanges information through the

TABLE I: Overview of the HCNN model stages, input and output dimensions, and the associated operations for each stage.

Stage	Input Size	Output Size	Description
Zero-Padding	$28 \times 28 \times 1$	$32 \times 32 \times 1$	Enlarges the image to a 32×32 grid
Batching	$32 \times 32 \times 1$	1024×9	Rearranges image for convolution
Convolution	1024×9	1024×8	Applies 8 3×3 filters for feature detection
Activation	1024×8	1024×8	Polynomial approximation to ReLu for activation mapping
Pooling	1024×8	64×8	Reduces dimensions with 4×4 pooling, stride 4
Dense	512×1	128×1	Integrates features into a 128-unit layer
Activation	128×1	128×1	Polynomial approximation to ReLu for activation mapping
Final Dense	128×1	10×1	Compresses features into 10 output units
Output Mapping	10×1	10×1	Final classification vector representing the output labels

Algorithm 1: Implemented Homomorphic Evaluation Operations

function HADD(ct, ct')
  $ct \leftarrow (c_0, c_1), ct' \leftarrow (c_0', c_1')$
  $d_0 \leftarrow c_0 + c_0', d_1 \leftarrow c_1 + c_1'$
  return $ct^A \leftarrow (d_0, d_1)$
end function
function HMULT(ct_0, ct_1, swk)
  $ct \leftarrow (c_0, c_1), ct' \leftarrow (c_0', c_1')$
  $d_0 \leftarrow c_0 \cdot c_0', d_2 \leftarrow c_1 \cdot c_1', d_1 \leftarrow c_1 \cdot c_0' + c_0 \cdot c_1'$
  $(d_0', d_1') \leftarrow$ KEYSWITCH(d_0, d_2, swk)
  return $ct^M \leftarrow (d_0' + d_1', d_1)$
end function
function HROT(ct, sn, swk)
  $ct \leftarrow (c_0, c_1)$
  $d_0 \leftarrow$ FROBENIUSMAP$(c_0, sn), d_1 \leftarrow$ FROBENIUSMAP(c_1, sn)
  $(d_0', d_1') \leftarrow$ KEYSWITCH(d_0, d_1, swk)
  return $ct^R \leftarrow (d_0', d_1 + d_1')$
end function
function RESCALE(ct)
  $ct \leftarrow (c_0, c_1) \in R_{q_j}, j \in [0, l]$
  $d_0 \leftarrow \left[q_l^{-1} \cdot (c_0^{(j)} - c_0^{(l)})\right]_{q_l}, d_1 \leftarrow \left[q_l^{-1} \cdot (c_1^{(j)} - c_1^{(l)})\right]_{q_l}$
  return $ct' \leftarrow (d_0, d_1)$
end function
function KEYSWITCH(ct, swk)
  $ct \leftarrow (c_0, c_1) \in R_{q_l}, swk \leftarrow \{(b_i, a_i)\}_{i \in [0,l]}$
  $(c_0', c_1') \leftarrow (0, 0)$
  for $i \in [0, l]$ **do**
    $\boldsymbol{c}^* \leftarrow \mathsf{ModUp}_{q_i \rightarrow Q_l P}(\boldsymbol{c}_1^{(i)})$;
    $(\boldsymbol{c}_0^*, \boldsymbol{c}_1^*) \leftarrow (\langle \boldsymbol{c}^*, \boldsymbol{b}_i \rangle, \langle \boldsymbol{c}^*, \boldsymbol{a}_i \rangle)$;
    $(\boldsymbol{c}_0', \boldsymbol{c}_1') \leftarrow (\boldsymbol{c}_0' + \boldsymbol{c}_0^*, \boldsymbol{c}_1' + \boldsymbol{c}_1^*)$;
  $c_0'' \leftarrow$ MODDOWN$(q_p \rightarrow q_i, c_0')$
  $c_1'' \leftarrow$ MODDOWN$(q_p \rightarrow q_i, c_1')$
  return $ct'' \leftarrow (c_0'', c_1'')$
end function

secure channel. Notably, the key remains unknown to any party beyond those directly involved, and the scheme operates without the need for intervention from trusted third parties, distinguishing it from Shamir's Secret Sharing Key Shares [27].

Here, $S_i(x)$ represents the secret polynomial generated by the individual n parties. In the next step, the parties generate shares of their individual polynomials for each party $S_i(x = 1, \ldots, n) = a_0 x^0 + a_1 x^1 + \ldots + a_t x^t$, which are then exchanged over the pairwise authentic and private channel.

Upon receiving all their n shares from the n individual parties, each party performs arithmetic addition of the shares to generate a sharing of the shared secret polynomial $S(j = 1, \ldots, n) = \sum_{i=1}^{n} S_i(j)$. Each participant must broadcast their individual share of the Elliptic curve point (i.e., the sharing of the public key), calculated as follows: $Q_i = S(i) \cdot G$.

Once the broadcast is complete, each participant can verify each individual sharing of the public key by interpolating in the exponent. $Q_j = ExpInt(Q_1, Q_2, \ldots, Q_{t+1}; j)$. Then we can get $Q = \lambda_1 \cdot Q_1 + \lambda_2 \cdot Q_2 + \ldots + \lambda_{t+1} \cdot Q_{t+1}$ Here, ('+' denotes EC addition), and $\lambda_i = \prod_{1<m<t+1, m \neq i} \frac{m-j}{m-i}$

If the check passes, the group public key is generated by interpolating in the exponent as follows: $Q = ExpInt(Q_1, Q_2, \ldots, Q_{t+1}; 0)$.

D. Threshold Signing

To achieve a reliable signature technology, we utilize zero-knowledge proof (ZKP) operations to ensure the security and correctness of the threshold signing process. Zero-knowledge proofs allow one party to prove to another that a statement is true without revealing any information beyond the validity of the statement itself. This property is crucial in maintaining the confidentiality and integrity of the cryptographic protocol, particularly in threshold signature schemes where multiple parties collaborate to generate a signature.

However, generating and verifying ZKPs introduces significant computational overhead. The process involves complex cryptographic operations that demand substantial computational resources and time. This overhead makes ZKP-based approaches less suitable for resource-constrained environments, such as Internet of Medical Things (IoMT) devices with limited processing power and energy availability.

In contrast, our proposed lightweight Threshold Signature Scheme (TSS) protocol achieves the same security guarantees without relying on computationally expensive ZKP operations. Instead, our protocol is based on Oblivious Transfer (OT), a fundamental primitive in secure multiparty computation. Oblivious Transfer allows a sender to transfer one of potentially many pieces of information to a receiver without knowing which piece was transferred. This mechanism is inherently efficient and requires significantly fewer computational resources than ZKP.

V. EVALUATION

A. Experiment Setup

We used WiFi in the experiments as the main communication protocol between the IoT device simulated as the patient and family and the desktop computers simulated as the hospital server. The setups use a Jeston Nano as the IoT device. We connected an AC1200 Wireless Dual-Band USB Adapter produced by Edimax to the Jeston Nano as shown in Fig. 2b. Using this adapter, we shared the Ethernet connection of the desktop as a WiFi hotspot which Jeston Nano connected to. We used a desktop computer for the hospital server with 12th Gen Intel(R) Core(TM) i7-12700F, 16GB memory and NVIDIA 4080. The desktop computer was connected to the Internet through a gigabit Ethernet connection inside the campus. They ran our TSS code, and the desktop computer also ran our GPU engine HCNN model. For the TSS protocol experiment, we also simulated the entire lifecycle of our TSS protocol using a desktop computer. Three distinct processes represented the participating parties in the scheme, and inter-process communication was facilitated via sockets.

1) Dataset: We deploy our proposed network on four datasets: MNIST [28], PneumoniaMNIST, BloodMNIST, BloodMNIST [29]. PneumoniaMNIST includes 5,856 pediatric chest X-ray images specifically intended to classify pneumonia against normal instances. The BloodMNIST dataset, derived from a collection of 780 breast ultrasound images, is designed to classify breast cancer. BloodMNIST is based on a dataset of individual normal cells obtained from individuals without any infection, hematologic, or oncologic disease and who were free of any pharmacologic treatment at the time of blood collection. The primary objective of this dataset is to recognize different types of normal peripheral blood cells. All datasets consist of 28×28 grayscale images.

2) HE Parameters Setting: The total depth of multiplication is 7 in our network, and, as a consequence, the modulus chain is instantiated with $|Q_L| = 340$ and $L = 7$. For our requirements, this setting provides a substantial security level of 128-bits. To streamline computations, we keep ciphertexts in a double-CRT representation, where each residue under the RNS representation is in the Number Theoretic Transformation (NTT) domain. For the choice of dimension N,we do an experiment in HE Hyperparameter Analysis. The key-switching, which is an intergal of our implementation, is performed with a special modulus $|P| = 60$. This setting allows for a larger decomposition number of ciphertext while simultaneously allowing a reduction in other HE parameters, such as the ring dimension N. Notably, this provides a balance between complexity and parameter size.It upholds the same security level, thereby making it a more efficient choice than the full RNS variant [30], particularly for circuits where the evaluation depth is not excessively deep.

B. Overall performance

We have designed a robust privacy-preserving remote real-time medical monitoring system that allows patients to upload encrypted sensitive information to hospital servers. The information is then analyzed by our HCNN model accelerated by a GPU engine on the hospital server while still encrypted. The encrypted medical analysis results are then transmitted to the TS. The TS then decrypts the results and takes appropriate medical actions based on the analysis. This process is particularly critical for immediate intervention in emergency medical situations, such as cardiac events or epileptic seizures.

Our HCNN model was evaluated on multiple datasets to detect patient anomalies in real-time, with specific results shown in Table II. The model achieved a 99.14% accuracy rate on the MNIST dataset, 90% on the PneumoniaMNIST dataset, and over 82% accuracy on the Blood-MNIST dataset.

When a condition requiring immediate medical intervention is detected, the TS prompts the patient for authorization. Once the patient grants permission, the patient collaborates with the hospital to complete a 2 out 3 threshold signing process, ensuring that the hospital can access the patient's medical records within the TS. If the patient is unresponsive, the TS will wait for one minute before automatically notifying the patient's emergency contact for authentication.

Evaluations conducted in various experimental settings have demonstrated the successful implementation of the aforementioned system features. In summary, the method we propose not only effectively enhances patient safety through continuous monitoring but also ensures the timeliness, personalization, and security of medical interventions.

C. Overhead measurement

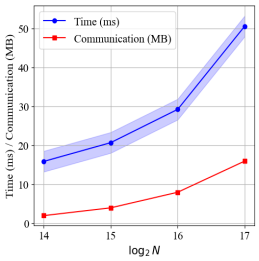
The model performed excellently in real-time performance, with an average response time of 0.016 seconds, meeting the time requirements for emergency medical responses as shown in Table. II. As for the communication overhead, An encrypted input tensor of the network has 2MB from the PC to the HS, as shown in Fig.2a.

D. HE Hyperparameter Analysis

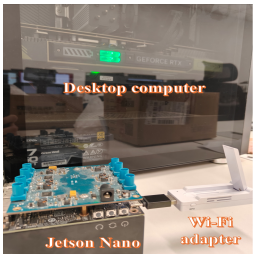
We provided an analysis of the impacts in ring dimension N, which refers to the number of slots packed in a single ciphertext, commonly ranging from 2^{14} to 2^{17}. As shown in Fig. 2a, we first investigate the N on the execution time and communication overhead in MNIST datasets. we observed a substantial increase in both execution time and communication overhead as N grows. Given that 2^{14} slots are sufficient to encapsulate the encrypted information, selecting a ring dimension N greater than 2^{14} would lead to unnecessary redundancy, as evidenced by the increase in execution time and communication overhead without a corresponding benefit in data capacity or computational parallelism.

E. Comparison with baseline

1) HCNN results: Table II provides a performance comparison of our proposed method with other established techniques, evaluated on MNIST, PneumoniaMNIST, BreastMNIST, and BloodMNIST. Across all datasets, our method not only achieves competitive accuracy, but also significantly



(a)



(b)

Fig. 2: (a) FHE ring dimension N Analysis and (b) Experiment settings.

TABLE II: Preliminary evaluations on accuracy and latency.

Method	Dataset	Accuracy(%)	Latency(s)	Platform
CryptoNets [31]	MNIST	98.95	205	CPU
Lola [20]	MNIST	98.95	2.2	CPU
A*FV [22]	MNIST	98.95	5.16	GPU
LiteCrypt	MNIST	99.14	0.46	CPU
LiteCrypt	MNIST	99.14	0.016	GPU
LiteCrypt	Pneumonia MNIST	90.06	0.016	GPU
LiteCrypt	Breast MNIST	82.69	0.016	GPU
LiteCrypt	Blood MNIST	81.76	0.016	GPU

reduces latency compared to other works. In the context of MNIST, our approach attains the highest accuracy of 99.14%, outperforming CryptoNets, and LoLa. More impressively, it reduces the execution time from 205 seconds in CryptoNets to 0.46 seconds on CPU and a mere 0.016 seconds on GPU. When it comes to the larger dataset, our solution still maintains superior performance in terms of latency, staying consistent at 0.46 seconds on CPU and 0.016 seconds on GPU, despite a slight drop in accuracy compared to benchmark [29]. This demonstrates that our method successfully provides a real-time solution for secure neural network inference across various datasets and platforms, breaking through the limitations of existing works.

For comparison, we also refer to the work of A*FV [22] running on an NVIDIA Tesla V100 GPU. According to their reported results, AFV required 5.16 seconds for inference on the MNIST dataset. The significant gap in inference times (0.032s for LiteCrypt on vs 5.16s for A*FV) still suggests a notable efficiency advantage of LiteCrypt. This can be attributed to several key technical innovations in our framework, such as the optimized HE operations, efficient ciphertext packing, and fine-grained GPU acceleration.

2) Threshold Signature Scheme result: We compare our approach with two baseline approaches, [32] and [33], which are the traditional authentication protocols without any protection and the existing threshold signature protocols, separately.

The comparison results are displayed in Table III. Compared to [32], our proposed method offers an enhanced level of security while maintaining a computational time within the same order of magnitude.

Compared to [33], we observe a significant reduction in the computation time from 11.9 seconds to merely 0.11 seconds. This substantial decrease not only attests to the lower computational complexity of our methodology but also indicates its suitability for deployment on resource-constrained IoMT devices. The signing phase in our approach costs more than the other two baselines for communication overhead. However, if we consider the additional protection, 160kb in 13 seconds is acceptable in our IoMT application. This is because [33] employs zero-knowledge proof (ZKP) operations to ensure the security and correctness of the threshold signing process. However, the generating and verifying ZKPs introduce significant computational overhead, making it less suitable for resource-constrained IoMT devices. In contrast, our lightweight TSS protocol achieves the same security guarantees without needing computationally expensive ZKP operations because our proposed TSS protocol is based on Oblivious Transfer (OT), a fundamental primitive in secure multiparty computation. For the IoT setting, the sign time is 34.96s, which is close to the baseline of relevant work [34].

TABLE III: Performance evaluation on different authentication approaches.

	DKG (Ours)	**Sign (Ours)**	**DKG (IoT)**	**Sign (IoT)**	**[32]**	**[33]**
Computation Time	0.01s	0.1s	0.12s	4.52s	10.8ms	11.9s
Communication Time	0.02s	13s	0.08s	30.42s	17ms	4.7s
Communication Overhead	0.2kb	160kb	0.2kb	160kb	0.41kb	3.8kb

VI. Related Work

A. Threshold Signature

With the development of blockchain technology in recent years, virtual currencies have received increasing attention. Threshold DSA/ECDSA signatures have been widely studied as one of the basic cryptographic techniques for virtual currencies.

Gennaro et al. [35] gave a formal definition of interactive threshold signature, and designed a multi-party secure computation method based on joint random secret sharing scheme and joint zero secret sharing scheme to compute product and inverse, thresholded DSA, and constructed a fully distributed thresholded digital signature standard. However, this scheme consumes a large amount of communication, so it is difficult to apply to practical scenarios. To address the polynomial expansion problem of the threshold digital signature standard, Gennaro et al. [36] constructed an optimized threshold digital signature standard based on the threshold Paillier homomorphic encryption scheme [37] and the trapdoor commitment technique [38]. Gennaro and Goldfeder et al. [33] designed a simple and efficient threshold ECDSA scheme using a special multiparty secure computation technique SPDZ protocol.

B. HCNN

Recent advancements in the field of private predictions have garnered notable attention in academia, leading to significant

contributions. Early work in this domain embraced a batch-encrypted processing methodology. A pioneering step towards privacy-preserving encrypted prediction services was CryptoNets [18]. Despite utilizing the YASHE [39] homomorphic encryption (HE) scheme, which is now deemed insecure, CryptoNets achieved a noteworthy 98.95% accuracy on the MNIST dataset. Progressing from this foundation, Boemer et al. [19] developed nGraph-HE, a HE-based enhancement to the Intel nGraph deep learning compiler, refining the techniques of CryptoNets and incorporating alternative HE schemes, specifically BFV [40] and CKKS [41]. Further accelerating execution, Badawi et al. [22] introduced a GPU implementation.

These collective efforts have underscored a design philosophy prioritizing high throughput, where network nodes are individually encrypted as distinct ciphertexts, advocating for the batch processing of images. This strategy, while reducing amortized time, leads to substantial memory demands when scaling to larger networks and may not be efficient for smaller sets of images. In contrast, Brutzkus et al. [42] proposed LoLa, a low-latency privacy-preserving inference method. LoLa innovates with a data batching technique that encrypts complete layers and transforms the data format during computation, enabling efficient low-latency inference on individual images. Building on LoLa, Lou et al. [21] identified and addressed the excessive homomorphic rotations by designing a homomorphic discrete Fourier transform algorithm that transitions ciphertexts into the spectral domain, significantly diminishing the need for rotations.

VII. Security Analysis

In this section, we analyze LiteCrypt's security properties in terms of patient data privacy, data integrity, trustworthiness, and access control. We demonstrate how our system effectively addresses the threats outlined in the threat model and provides strong security guarantees.

- **Patient data privacy:** The patient's medical data is encrypted using the public key pk provided by the trusted server. Under the CKKS-based homomorphic encryption, an unauthorized party cannot infer the plaintext data m from the ciphertext ct, i.e., $\mathcal{A}(ct) \neq m$. This ensures the privacy of patient data during storage and computation.
- **Data integrity:** Our scheme employs threshold signatures to verify the authenticity of medical requests. A valid signature σ requires partial signatures σ_i from at least $t + 1$ participants, preventing any single party from forging a medical request. Formally, for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the following probability is negligible: $Pr[Verify(pk, m, \sigma^*) = true \wedge \sigma^* \notin Sign(sk, m)] \leq negl(n)$
- **Trustworthiness:** During the anomaly detection phase, the hospital server performs computations in the ciphertext domain without decrypting patient data. The detection results remain encrypted and require decryption by the trusted server using the secret key sk. This ensures the trustworthiness of the analysis process, as the hospital can only access plaintext detection results upon authorization.
- **Access control:** Our dual authorization mechanism prevents unauthorized access to patient data. Access requests must be jointly signed by any 2 out of 3 participants (i.e., patient, patient's family, and hospital). This (2,3) threshold ensures the integrity of authorization while providing flexibility in emergency situations. For any PPT adversary $\mathcal{A}$, the following probability is negligible: $Pr[Verify(pk, m, \sigma^*) = true \wedge |\sigma_i| < 2] \leq negl(n)$

VIII. Conclusion

In this study, we introduced LiteCrypt, a novel framework for secure and efficient remote medical monitoring using real-time and low-memory threshold signature and homomorphic encryption. Our approach employs a streamlined homomorphic encryption framework, GPU acceleration, and lightweight TSS protocols to address the challenges of privacy preservation, latency reduction, and memory efficiency in IoMTs. Extensive empirical validation demonstrated remarkable performance gains, with a 233× speedup and 96% reduction in encrypted message sizes compared to existing solutions. LiteCrypt paves the way for privacy-preserving healthcare systems and opens exciting directions for future research on secure remote medical monitoring.

Acknowledgment

This work is supported by Guangdong Provincial Key Lab of Integrated Communication, Sensing and Computation for Ubiquitous Internet of Things (No.2023B1212010007), China NSFC Grant U2001207, the Project of DEGP (No.2023KCXTD042, 2021ZDZX1068).

References

[1] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, "Csi-based indoor localization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1300–1309, 2012.

[2] Y. Huang, S. Cai, L. Wang, and K. Wu, "Oinput: A bone-conductive qwerty keyboard recognition for wearable device," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 946–953.

[3] K. Wu, Y. Huang, L. Wang, and C. Kaixin, "Auxiliary sensing method and system based on sensory substitution," Apr. 19 2022, uS Patent 11,310,620.

[4] K. Chen, L. Wang, Y. Huang, K. Wu, and L. Wang, "Lit: Fine-grained toothbrushing monitoring with commercial led toothbrush," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, 2023, pp. 1–16.

[5] Q. Dai, Y. Huang, L. Wang, R. Ruby, and K. Wu, "mm-humidity: Fine-grained humidity sensing with millimeter wave signals," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 204–211.

[6] B. Yang, S. Jiang, L. Xu, K. Liu, H. Li, G. Xing, H. Chen, X. Jiang, and Z. Yan, "Drhouse: An llm-empowered diagnostic reasoning system through harnessing outcomes from sensor data and expert knowledge," *arXiv preprint arXiv:2405.12541*, 2024.

[7] Y. Huang, K. Chen, Y. Huang, L. Wang, and K. Wu, "Vi-liquid: unknown liquid identification with your smartphone vibration," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021, pp. 174–187.

[8] Y. Huang, K. Chen, J. Zhao, L. Wang, and K. Wu, "Beverage deterioration monitoring based on surface tension dynamics and absorption spectrum analysis," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 3722–3740, 2023.

[9] Y. Huang, K. Chen, L. Wang, Y. Dong, Q. Huang, and K. Wu, "Lili: liquor quality monitoring based on light signals," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021, pp. 256–268.

[10] Q. Xie, H. Yang, L. Jiang, Z. Zhao, S. Jiang, S. Shen, S. Khan, Z. Liu, and K. Wu, "Cnn-guardian: Secure neural network inference acceleration on edge gpu," in *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, 2023, pp. 524–525.

[11] Q. Xie, Z. Zhao, L. Jiang, S. Jiang, S. Khan, W. Wang, and K. Wu, "Threshold cryptography-based authentication protocol for remote healthcare," in *2024 23rd ACM/IEEE IPSN*. IEEE, 2024, pp. 303–304.

[12] H. Wu, K. Liu, S. Jiang, Z. Zhao, Z. Yan, and G. Xing, "Demo abstract: Caringfm: An interactive in-home healthcare system empowered by large foundation models," in *2024 23rd ACM/IEEE IPSN*. IEEE, 2024, pp. 255–256.

[13] W. Wang, Q. Chen, Z. Yin, G. Srivastava, T. R. Gadekallu, F. Alsolami, and C. Su, "Blockchain and puf-based lightweight authentication protocol for wireless medical sensor networks," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8883–8891, 2021.

[14] Z. Lian, Q. Zeng, W. Wang, T. R. Gadekallu, and C. Su, "Blockchain-based two-stage federated learning with non-iid data in iomt system," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 4, pp. 1701–1710, 2022.

[15] M. Kim, X. Jiang, K. Lauter, E. Ismayilzada, and S. Shams, "Secure human action recognition by encrypted neural network inference," *Nature communications*, vol. 13, no. 1, p. 4799, 2022.

[16] B. D. Deebak, F. H. Memon, S. A. Khowaja, K. Dev, W. Wang, N. M. F. Qureshi, and C. Su, "A lightweight blockchain-based remote mutual authentication for ai-empowered iot sustainable computing systems," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 6652–6660, 2022.

[17] Q. Xie, S. Jiang, L. Jiang, Y. Huang, Z. Zhao, S. Khan, W. Dai, Z. Liu, and K. Wu, "Efficiency optimization techniques in privacy-preserving federated learning with homomorphic encryption: A brief survey," *IEEE Internet of Things Journal*, vol. 4, no. 11, pp. 33 081–33 096, 2024.

[18] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International conference on machine learning*. PMLR, 2016, pp. 201–210.

[19] F. Boemer, A. Costache, R. Cammarota, and C. Wierzynski, "ngraph-he2: A high-throughput framework for neural network inference on encrypted data," in *Proceedings of the 7th ACM workshop on encrypted computing & applied homomorphic cryptography*, 2019, pp. 45–56.

[20] A. Brutzkus, O. Elisha, and R. Gilad-Bachrach, "Low latency privacy preserving inference," *ArXiv*, vol. abs/1812.10659, 2018.

[21] Q. Lou, W.-j. Lu, C. Hong, and L. Jiang, "Falcon: Fast spectral inference on encrypted data," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2364–2374, 2020.

[22] A. A. Badawi, C. Jin, J. Lin, C. F. Mun, S. J. Jie, B. Tan, X. Nan, K. M. M. Aung, and V. R. Chandrasekhar, "Towards the alexnet moment for homomorphic encryption: Hcnn, the first homomorphic cnn on encrypted data with gpus," *IEEE TETIC*, vol. 9, pp. 1330–1343, 2021.

[23] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology–ASIACRYPT 2017: Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer, 2017, pp. 409–437.

[24] P. Wang, T. Kausar, H. Li, S. Jiang, Y. Rong, and Y. Lu, "Eeg decoding of auditory spatial attention based on visibility graph and machine learning," in *Proceedings of the International Conference on Computer Vision and Deep Learning*, 2024, pp. 1–5.

[25] W. Jung, S. Kim, J. H. Ahn, J. H. Cheon, and Y. Lee, "Over 100x faster bootstrapping in fully homomorphic encryption through memory-centric optimization with gpus," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 114–148, 2021.

[26] S. Shen, H. Yang, Y. Liu, Z. Liu, and Y. Zhao, "Carm: Cuda-accelerated rns multiplication in word-wise homomorphic encryption schemes for internet of things," *IEEE Transactions on Computers*, 2022.

[27] K. Yu, L. Tan, C. Yang, K.-K. R. Choo, A. K. Bashir, J. J. Rodrigues, and T. Sato, "A blockchain-based shamir's threshold cryptography scheme for data protection in industrial internet of things settings," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8154–8167, 2021.

[28] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[29] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni, "Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification," *Scientific Data*, vol. 10, no. 1, p. 41, 2023.

[30] K. Han and D. Ki, "Better bootstrapping for approximate homomorphic encryption," in *Cryptographers' Track at the RSA Conference*. Springer, 2020, pp. 364–390.

[31] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. E. Lauter, M. Naehrig, and J. R. Wernsing, "Cryptonets: applying neural networks to encrypted data with high throughput and accuracy," in *TCML*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:217485587

[32] M. Fotouhi, M. Bayat, A. K. Das, H. A. N. Far, S. M. Pournaghi, and M.-A. Doostari, "A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care iot," *Computer Networks*, vol. 177, p. 107333, 2020.

[33] R. Gennaro and S. Goldfeder, "Fast multiparty threshold ecdsa with fast trustless setup," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1179–1194.

[34] A. Kurt, K. Akkaya, S. Yilmaz, and S. Mercan, "Lngate 2: Secure bidirectional iot micro-payments using bitcoin's lightning network and threshold cryptography," *IEEE TMC*, 2023.

[35] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust threshold dss signatures," in *Advances in Cryptology—EUROCRYPT'96: Saragossa, Spain, May 12–16, 1996 Proceedings 15*. Springer, 1996, pp. 354–371.

[36] R. Gennaro, S. Goldfeder, and A. Narayanan, "Threshold-optimal dsa/ecdsa signatures and an application to bitcoin wallet security," in *Applied Cryptography and Network Security: 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings 14*. Springer, 2016, pp. 156–174.

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

[38] I. Damgard and J. Groth, "Non-interactive and reusable non-malleable commitment schemes," in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 2003, pp. 426–437.

[39] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Cryptography and Coding: 14th IMA International Conference, IMACC 2013, Oxford, UK, December 17-19, 2013. Proceedings 14*. Springer, 2013, pp. 45–64.

[40] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, 2012.

[41] H. Yang, S. Shen, S. Jiang, L. Zhou, W. Dai, and Y. Zhao, "Xnet: A real-time unified secure inference framework using homomorphic encryption," *Cryptology ePrint Archive*, 2023.

[42] A. Brutzkus, R. Gilad-Bachrach, and O. Elisha, "Low latency privacy preserving inference," in *International Conference on Machine Learning*. PMLR, 2019, pp. 812–821.