This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3079106, IEEE Internet of Things Journal

Toward Privacy-Preserving Healthcare Monitoring based on Time-Series Activities over Cloud

Yandong Zheng, Rongxing Lu, *Fellow, IEEE*, Songnian Zhang, Yunguo Guan, Jun Shao, and Hui Zhu, *Senior Member, IEEE*

***Abstract*—The thriving of the Internet of Things (IoT) has become the enabler of smart eHealthcare, which greatly benefits patients by providing various data-driven healthcare monitoring services. Among those promising services, the time-series activities based healthcare monitoring service is highly regarded due to its popularity. Meanwhile, with the rapidly growing volume of healthcare data, an emerging trend is to outsource the time-series activities based healthcare monitoring models and the corresponding services to a cloud, which however inevitably entails privacy concerns. Although many existing works have put forth some solutions for privacy-preserving time-series activities based healthcare monitoring, they are not applicable to the outsourced scenario with a single-server setting. To address the challenge, in this paper, we propose an efficient and privacy-preserving forward algorithm (PPFA) and further apply PPFA to construct a remote healthcare monitoring scheme over the cloud. To the best of our knowledge, our PPFA is the first privacy-preserving forward algorithm over cloud while without any accuracy loss. In addition, our remote healthcare monitoring scheme is also the first privacy-preserving HMM-based healthcare monitoring scheme in the single-server setting. Detailed security analysis shows that our PPFA and healthcare monitoring scheme are indeed privacy-preserving. In addition, extensive simulations are conducted, and the results also demonstrate their efficiencies.**

***Index Terms*—Privacy, time-series activities, eHealthcare, healthcare monitoring, hidden Markov model.**

I. Introduction

DRIVEN by the recent technological advances in the Internet of Things (IoT) [1], [2] and wireless communications, the wireless body area network (WBAN) has become increasingly flourishing and received considerable attention from both industry and academia [3]. In eHealthcare, each patient that is equipped with medical devices can be regarded as a WBAN, and the data collected by the WBAN can assist doctors located at healthcare centers to remotely monitor the patient's healthcare state [4]. With the dramatic increase of healthcare data, an emerging trend among healthcare centers is to outsource the healthcare monitoring models to a powerful cloud and employ the cloud server to offer the healthcare monitoring service to patients [5], [6]. Due to the privacy issue of models, healthcare centers usually demand to encrypt the models before outsourcing them to the cloud server. However, a consensus has emerged that encrypted models will inevitably affect the query functionalities [7], [8]. In the eHealthcare scenario, it will hinder the cloud server from efficiently offering the healthcare monitoring service to patients.

To achieve healthcare monitoring over encrypted models, many schemes have been proposed, and most of them target different topics, e.g., human activities recognition [9], [10] and disease predication [11]. In this work, we monitor patients' healthcare states through time-series activities based classification, i.e., classify patients to be normal or abnormal based on the sequences of their daily living activities. Specifically, when a patient has collected a time-series activities sequence $\mathbf{y}$ (e.g., $\mathbf{y} = \{\text{Waking up-Toileting-Breakfast-Resting}\}$), the patient will send the encrypted sequence $\mathrm{E}(\mathbf{y})$ to the cloud server. Then, the cloud server uses an encrypted classification model (that is outsourced by the healthcare center) to do classification and returns the encrypted classification result (i.e., normal or abnormal) to the patient.

In the design of the healthcare monitoring scheme, the classification model is one of the most critical factors. The work in [12], [13] has demonstrated that long short-term memory (LSTM) neural network and hidden Markov model (HMM) based classification models perform quite well in time-series activities based healthcare monitoring. At the same time, due to the simplicity and effectiveness, HMM becomes more popular than the LSTM neural network in eHealthcare. Naturally, privacy-preserving LSTM network and HMM based classification models can be used for the construction of a privacy-preserving healthcare monitoring scheme. However, existing such models are not applicable to our outsourced scenario with a single-server setting. On the one hand, some privacy-preserving LSTM neural network based classification models [14], [15] were constructed under the setting of two non-collusive servers. These schemes are impractical because the non-collusive assumption between two servers seems strong, and the economic costs of two servers are relatively high. On the other hand, too little work has been devoted to designing privacy-preserving HMM based classification models in the outsourced scenario. Although some existing privacy-preserving forward algorithms [16]–[21] can be used to construct HMM-based classification models, they only consider the scenario of two-party computation, (i.e., one party holds the model, and the other one holds the time-series activities sequence) and are not applicable to our outsourced scenario.

In this paper, we aim at designing a healthcare monitoring

Y. Zheng, R. Lu, S. Zhang and Y. Guan are with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: yzheng8@unb.ca, rlu1@unb.ca, songnian.zhang@unb.ca, yguan4@unb.ca).

J. Shao is with School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, 310018, China (e-mail: chn.junshao@gmail.com).

H. Zhu is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, 710071, China (e-mail: zhuhui@xidian.edu.cn).

scheme under a single-server setting. Due to its popularity, we employ HMM as our classification model. The key idea of our scheme is to train a normal HMM $\mathbf{\Lambda}_1$ and an abnormal HMM $\mathbf{\Lambda}_2$ by respectively using normal and abnormal time-series activities data. Then, given a time-series activities sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$, we can classify for it by comparing which model it matches better. Specifically, we can use the forward algorithm of HMM to compute the probabilities of $\mathbf{y}$ under the models $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$, i.e., $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$. If $\Pr(\mathbf{y}|\mathbf{\Lambda}_1) > \Pr(\mathbf{y}|\mathbf{\Lambda}_2)$, $\mathbf{y}$ is classified to be normal. Otherwise, $\mathbf{y}$ is classified to be abnormal. We can observe that the forward algorithm is the core of the HMM based classification model, and the security of the whole healthcare monitoring scheme depends on the security of the forward algorithm. However, it is challenging to design a privacy-preserving forward algorithm under a single-server setting. This is the reason why the existing works [16]–[21] only focus on designing privacy-preserving forward algorithms in the scenario of two party computation.

Before introducing the challenges of designing privacy-preserving forward algorithm, we first briefly describe the forward algorithm. Suppose that the healthcare center has an HMM $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$ (see details in Subsection III-B), and a patient has a time-series sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ with the length of T. The forward algorithm computes $\Pr(\mathbf{y}|\mathbf{\Lambda})$ as follows.

• ***Step 1:*** Compute $\mathbf{\Psi} = \mathbf{\Pi}\mathbf{B}_{y_1}\prod_{i=2}^{T}\mathbf{A}\mathbf{B}_{y_i}$, where $\mathbf{B}_{y_i}$ is a matrix related to y_i among $\{\mathbf{B}_j\}_{j=1}^{l}$.

• ***Step 2:*** Compute $\Pr(\mathbf{y}|\mathbf{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{\Psi}_{i,j}$.

When designing the privacy-preserving forward algorithm, we will face two challenges.

(1) **How to compute $\{\mathbf{B}_{y_i}\}_{i=1}^{T}$ in *Step 1*?** Since $\{\mathbf{B}_j\}_{j=1}^{l}$ and $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ are respectively held by the healthcare center and the patient, a straightforward method to privately compute $\{\mathbf{B}_{y_i}\}_{i=1}^{T}$ is to use a fully homomorphic encryption to encrypt $\{\mathbf{B}_j\}_{j=1}^{l}$ into $\{\mathrm{E}(\mathbf{B}_j)\}_{j=1}^{l}$ and encrypt y_i into an l-dimensional vector $\mathbf{c} = (c_1, c_2, \cdots, c_l) = (\mathrm{E}(0), \cdots, \mathrm{E}(1), \cdots, \mathrm{E}(0))$, where the y_i-th element of $\mathbf{c}$ is $\mathrm{E}(1)$, and other elements are $\mathrm{E}(0)$. Then, one can compute $\mathrm{E}(\mathbf{B}_{y_i}) = \sum_{j=1}^{l}(\mathrm{E}(\mathbf{B}_j) * c_j)$ with the homomorphic properties. Although the straightforward method is workable, it needs to employ a fully homomorphic encryption technique. However, existing fully homomorphic encryption schemes [22], [23] are based on computationally expensive cryptographic primitives and have performance issues.

(2) **How to guarantee the accuracy of $\Pr(\mathbf{y}|\mathbf{\Lambda})$?** Since HMM is a probabilistic model, all involved numbers are small real numbers. However, most of the existing cryptographic primitives perform computations over integers and cannot provide a reasonable accuracy for the forward algorithm. Thus, it is also challenging to guarantee the accuracy of the computation result $\Pr(\mathbf{y}|\mathbf{\Lambda})$.

To address the above challenges, in this paper, we propose an efficient and privacy-preserving forward algorithm (PPFA) and further use it as a building block to construct a privacy-preserving healthcare monitoring scheme under a single-server setting. To the best of our knowledge, our PPFA is the first privacy-preserving forward algorithm in the cloud while without any accuracy loss. Meanwhile, our healthcare monitoring scheme is also the first one to achieve privacy-preserving HMM-based healthcare monitoring in the cloud with a single-server setting. Specifically, our contributions are three folds as follows.

• First, to privately compute $\mathbf{B}_{y_i}$ in the single-server setting, we propose the concept of mutually orthogonal matrices and introduce an approach to construct a set of mutually orthogonal matrices. In our definition, a set of mutually orthogonal matrices $\{\mathbf{D}_i\}_{i=1}^{l}$ satisfy

$$\mathbf{D}_i^T\mathbf{D}_j = \begin{cases}\mathbf{I} & i = j\\ \mathbf{O} & i \neq j\end{cases}$$

for $1 \leq i, j \leq l$, where $\mathbf{I}$ and $\mathbf{O}$ respectively denote an identity matrix and a zero matrix. In this case, $\mathbf{B}_{y_i}$ can be computed as $\mathbf{B}_{y_i} = \mathbf{D}_{y_i}^T\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{B}_j)$.

• Second, we propose an efficient and privacy-preserving forward algorithm, i.e., PPFA, by using a set of mutually orthogonal matrices and a lightweight matrix encryption technique. In the PPFA, we use a set of mutually orthogonal matrices to compute $\{\mathbf{B}_{y_i}\}_{i=1}^{T}$ and further use the matrix encryption to preserve the privacy of the computation $\mathbf{\Psi} = \mathbf{\Pi}\mathbf{B}_{y_1}\prod_{i=2}^{T}\mathbf{A}\mathbf{B}_{y_i}$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{\Psi}_{i,j}$. We have three considerations when we choose the matrix encryption as our encryption technique.

(1) The core operations in $\mathbf{\Psi}$ are matrix multiplications. Since the matrix encryption can perform efficient matrix multiplications, it exactly fits well with the computational requirement of the forward algorithm.
(2) The matrix encryption is lightweight, which can greatly optimize the efficiency of our scheme.
(3) Since the matrix encryption performs computations over real domain, it can guarantee the accuracy of the forward algorithm.

• Third, we propose an efficient and privacy-preserving healthcare monitoring scheme based on PPFA. Moreover, we formally prove the security of the PPFA and analyze the security of our healthcare monitoring scheme. The results show that both PPFA and our healthcare monitoring scheme are privacy-preserving. In addition, we conduct experiments to validate their efficiencies, and the results indicate that they are indeed efficient.

The remainder of this paper is organized as follows. In Section II, we introduce our system model, security model, and design goals. Then, we describe some preliminaries in Section III. In Section IV, we present our scheme, followed by security analysis and performance evaluation in Section V and Section VI, respectively. In Section VII, we present some related work. Finally, we draw our conclusion in Section VIII.

II. Models and Design Goals

In this section, we introduce our system model, security model, and identify our design goals.

A. System Model

In the system model, we consider a time-series activities based healthcare monitoring model in the cloud, which in-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3079106, IEEE Internet of Things Journal

volves three kinds of entities, including a healthcare center, a cloud server, and multiple patients, as shown in Fig. 1.

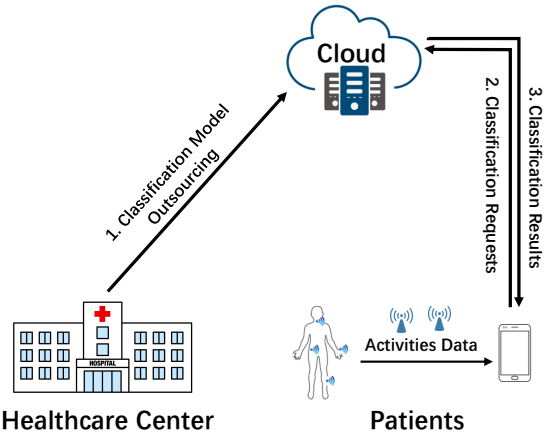


Fig. 1. System model under consideration

• Healthcare Center: The healthcare center is responsible for detecting patients' physical states by classifying their time-series activities data to be normal or abnormal. Specifically, when each patient registers in the system, the healthcare center will collect a time-series activities dataset from the patient and identify the class label of each time-series sequence. Through the collected dataset, the healthcare center trains a personalized classification model, which can be used for the time-series activities based healthcare monitoring for the patient. Since the healthcare center needs to do healthcare monitoring for a large number of patients, to ease the computing pressure, it will outsource all patients' classification models to a powerful cloud and employ the cloud server to do classification for each patient. To preserve the privacy of classification models, it encrypts these models before outsourcing them. As shown in Fig. 1, in this work, we focus on using the classification model to do classification in the outsourced scenario and leave the research on training a classification model for the patient in our future work.

• Cloud Server: The cloud server has plenty of computing resources and is willing to provide them to others. In particular, it firstly stores the encrypted classification models received from the healthcare center, and then offers the classification service to patients according to the encrypted classification models and encrypted time-series activities of patients.

• Patients: There exist many patients in our system. Each of them has a personalized classification model trained by the healthcare center, and the corresponding ciphertext is stored in the cloud server. To monitor the physical state, each patient is equipped with many medical IoT devices, which collect his/her time-series activities data and send them to his/her smart phones in real time. The smart phones periodically encrypt the collected data and report them to the cloud server for classification. Finally, the patients obtain the health states from the cloud server.

B. Security Model

In our security model, the healthcare center is considered to be *trusted* because it initializes the entire system. The patients are assumed to be *honest*, namely, they will honestly follow the scheme to launch classification requests to the cloud server. While for the cloud server, it is considered to be *honest-but-curious*. That is, it will honestly follow the protocol to offer the classification service to patients but may be curious about some private information, including the plaintexts of the classification models, classification requests, and classification results. In addition, we assume that there is no collusion between the cloud server and patients. The non-collusive assumption is reasonable because the penalty of collusion for the involving cloud server and patients is high, including losing the trust of the healthcare center and being prosecuted. Note that there may be other active attacks in the system, such as data pollution attack. Since this work focuses more on privacy preservation, those attacks are beyond the scope of this paper and will be discussed in our future work.

C. Design Goals

In this work, our goal is to design an efficient and privacy-preserving time-series activities based healthcare monitoring scheme over cloud, and the following objectives should be achieved.

• *Privacy preservation:* The privacy of the classification models, classification requests, and classification results should be preserved.

• *Efficiency:* The computational cost of the classification algorithm should be minimized as much as possible.

III. Preliminaries

In this section, we review the concept of HMM and a variant of forward algorithm of HMM, named VFA-HMM, which will serve as the building blocks of our proposed scheme.

A. HMM

HMM is a statistical Markov model and contains two kinds of states, i.e., (i) n hidden states $\mathcal{S} = \{S_1, S_2, \cdots, S_n\}$; and (ii) l observation states $\mathcal{O} = \{O_1, O_2, \cdots, O_l\}$. The states in the observed time-series sequences are from the set of observation states $\{O_1, O_2, \cdots, O_l\}$. HMM assumes that each observed time-series sequence corresponds to a hidden time-series sequence, whose states are from the set of hidden states $\{S_1, S_2, \cdots, S_n\}$. Specifically, suppose that $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ is an observed time-series sequence, and $y_i \in \mathcal{O}$ for $i = 1, 2, \cdots, T$. Then, $\mathbf{y}$ corresponds to a hidden time-series sequence $\mathbf{x} = \{x_1, x_2, \cdots, x_T\}$, where $x_i \in \mathcal{S}$. Meanwhile, x_i is called as the hidden state of y_i, and y_i is called as the emission state of x_i. $\{\mathbf{x}, \mathbf{y}\}$ is an HMM if it satisfies the following two conditions.

(1) $\mathbf{x} = \{x_1, x_2, \cdots, x_T\}$ is a Markov process. That is, the hidden state x_i only depends on x_{i-1} and is independent of other previous hidden states.

(2) $\Pr(y_i|x_1, x_2, \cdots, x_i) = \Pr(y_i|x_i)$. That is, the observation state y_i only depends on the hidden state x_i and is independent of other hidden states.

Based on these conditions, HMM defines three kinds of probabilities, i.e., initial hidden states probability, transition probability matrix among hidden states, and emission probability matrix from hidden states to observation states. Specifically, an HMM model $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{B})$ is defined as follows.

• Initial hidden state probability $\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_n)$. Each π_i denotes the probability of the initial hidden state at S_i, i.e., $\pi_i = \Pr(x_1 = S_i)$.

• Transition probability matrix among hidden states $\mathbf{A} = (a_{i,j})$. $\mathbf{A}$ is an $n \times n$ matrix, and each $a_{i,j}$ denotes the transition probability from the hidden state S_i to the hidden state S_j, i.e., $a_{i,j} = \Pr(x_{k+1} = S_j | x_k = S_i)$.

• Emission probability matrix from hidden states to observation states $\mathbf{B} = (b_{i,j})$. $\mathbf{B}$ is an $n \times l$ matrix, and each $b_{i,j}$ denotes the emission probability from the hidden state S_i to the observation state O_j, i.e., $b_{i,j} = \Pr(y_k = O_j | x_k = S_i)$.

B. A Variant of Forward Algorithm of HMM

Let $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ denote a time-series sequence of length T and $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{B})$ denote an HMM. The forward algorithm of HMM is used for evaluating the probability of $\mathbf{y}$ under the model $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{B})$, i.e., $\Pr(\mathbf{y}|\boldsymbol{\lambda})$. Based on the traditional HMM and forward algorithm, we have proposed a variant of HMM and a variant of forward algorithm of HMM (VFA-HMM) in [24], which can also be employed to compute $\Pr(\mathbf{y}|\boldsymbol{\lambda})$. The details are described as follows.

A variant of HMM. Given an HMM $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{B})$, we can construct a variant of HMM, denoted by $\boldsymbol{\Lambda} = (\boldsymbol{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$, as the following steps.

• ***Step-1:*** According to $\boldsymbol{\pi}$, construct an $n \times n$ matrix $\boldsymbol{\Pi} = (\pi_{i,j})$. Each $\pi_{i,j}$ is a random real number and satisfies that the sum of values in the j-th column is π_j. That is, $\sum_{i=1}^{n} \pi_{i,j} = \pi_j$ for $j = 1, 2, \cdots, n$.

• ***Step-2:*** According to $\mathbf{B}$, construct l diagonal matrices $\{\mathbf{B}_j\}_{j=1}^{l}$. Each $\mathbf{B}_j$ is constructed based on the j-th column of $\mathbf{B}$, i.e., $\mathbf{B}_j = \mathrm{diag}(b_{1,j}, b_{2,j}, \cdots, b_{n,j})$.

Then, the variant of HMM is $\boldsymbol{\Lambda} = (\boldsymbol{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$.

VFA-HMM. Given a time-series sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ and a variant of HMM $\boldsymbol{\Lambda} = (\boldsymbol{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$, VFA-HMM computes $\Pr(\mathbf{y}|\boldsymbol{\Lambda})$ as follows.

• ***Step-1:*** Compute $\boldsymbol{\Psi} = \boldsymbol{\Pi}\mathbf{B}_{y_1} \prod_{i=2}^{T} \mathbf{A}\mathbf{B}_{y_i}$, where $\mathbf{B}_{y_i}$ is the diagonal matrix corresponding to the observation state y_i for $i = 1, 2, \cdots, T$.

• ***Step-2:*** Compute $\Pr(\mathbf{y}|\boldsymbol{\Lambda}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{\Psi}_{i,j}$.

IV. Our Proposed Scheme

In this section, we introduce an efficient and privacy-preserving time-series activities based healthcare monitoring scheme. Before delving into the details, we first define the concept of mutually orthogonal matrices and introduce an approach to construct a set of mutually orthogonal matrices. Then, we present a privacy-preserving forward algorithm, i.e., PPFA. Both mutually orthogonal matrices and PPFA are important building blocks of our privacy-preserving healthcare monitoring scheme over cloud.

A. Mutually Orthogonal Matrices

In this subsection, we propose the concept of mutually orthogonal matrices and introduce how to construct a set of mutually orthogonal matrices.

Definition 1 (Mutually Orthogonal Matrices): Given a set of matrices $\{\mathbf{D}_i\}_{i=1}^{l}$, they are called mutually orthogonal matrices *iff* they satisfy that

$$\mathbf{D}_i^T \mathbf{D}_j = \begin{cases} \mathbf{I} & i = j \\ \mathbf{O} & i \neq j \end{cases}$$

for $1 \leq i, j \leq l$, where $\mathbf{I}$ and $\mathbf{O}$ respectively denote an identity matrix and a zero matrix.

In the PPFA (introduced in Subsection IV-B), we need to generate l mutually orthogonal matrices $\{\mathbf{D}_i\}_{i=1}^{l}$, and they satisfy that each $\mathbf{D}_i^T \mathbf{D}_j$ is an $n \times n$ square matrix for any $1 \leq i, j \leq l$. In the following, we introduce how to generate such kinds of mutually orthogonal matrices.

• ***Step-1:*** Generate an $(n*l) \times (n*l)$ orthogonal matrix $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_{n*l}]$, where $\mathbf{w}_j$ is the j-column of $\mathbf{W}$. Since $\mathbf{W}$ is an orthogonal matrix, the vectors $\{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_{n*l}\}$ satisfy that

$$\mathbf{w}_i^T \mathbf{w}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j, \end{cases}$$

for $1 \leq i, j \leq n * l$.

• ***Step-2:*** Split $\mathbf{W}$ into l matrices $\{\mathbf{D}_i\}_{i=1}^{l}$, and each $\mathbf{D}_i$ is in the form of

$$\mathbf{D}_i = [\mathbf{w}_{(i-1)*n+1}, \mathbf{w}_{(i-1)*n+2}, \cdots, \mathbf{w}_{i*n}],$$

for $1 \leq i \leq l$. Then, $\{\mathbf{D}_i\}_{i=1}^{l}$ is a set of mutually orthogonal matrices, and each $\mathbf{D}_i^T \mathbf{D}_j$ is an $n \times n$ matrix.

Correctness. The above approach is correct *iff* $\{\mathbf{D}_i\}_{i=1}^{l}$ are l mutually orthogonal matrices. According to Definition 1, $\{\mathbf{D}_i\}_{i=1}^{l}$ is a set of mutually orthogonal matrices *iff* each $\mathbf{D}_i^T \mathbf{D}_j$ satisfies that

$$\mathbf{D}_i^T \mathbf{D}_j = \begin{cases} \mathbf{I}_{n \times n} & i = j \\ \mathbf{O}_{n \times n} & i \neq j, \end{cases} \tag{1}$$

where $\mathbf{I}_{n \times n}$ is an $n \times n$ identity matrix, and $\mathbf{O}_{n \times n}$ is an $n \times n$ zero matrix. In the following, we prove that each $\mathbf{D}_i^T \mathbf{D}_j$ satisfies Eq. (1).

Proof. We prove in two cases, i.e., $i = j$, and $i \neq j$.

Case 1: When $i = j$, we have

$$\begin{aligned} \mathbf{D}_i^T \mathbf{D}_j &= \mathbf{D}_i^T \mathbf{D}_i \\ &= \begin{bmatrix} \mathbf{q}_{(i-1)*n+1}^T \\ \mathbf{q}_{(i-1)*n+2}^T \\ \vdots \\ \mathbf{q}_{i*n}^T \end{bmatrix} [\mathbf{q}_{(i-1)*n+1}, \mathbf{q}_{(i-1)*n+2}, \cdots, \mathbf{q}_{i*n}] \\ &= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \mathbf{I}_{n \times n}. \end{aligned}$$

Case 2: When $i \neq j$, we have

$$\mathbf{D}_i^T\mathbf{D}_j = \begin{bmatrix} \mathbf{q}_{(i-1)*n+1}^T \\ \mathbf{q}_{(i-1)*n+2}^T \\ \vdots \\ \mathbf{q}_{i*n}^T \end{bmatrix} [\mathbf{q}_{(j-1)*n+1}, \mathbf{q}_{(j-1)*n+2}, \cdots, \mathbf{q}_{j*n}]$$
$$= \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} = \mathbf{O}_{n\times n},$$

Thus, $\{\mathbf{D}_i\}_{i=1}^l$ is a set of mutually orthogonal matrices. □

The mutually orthogonal matrices can be used to achieve one matrix retrieval from a set of matrices. Let $\{\mathbf{X}_i\}_{i=1}^l$ be l matrices with the size of $n \times n$. Let $\{\mathbf{D}_i\}_{i=1}^l$ be l mutually orthogonal matrices with the size of $(n*l)\times n$, and they satisfy Eq. (1). We can deduce that

$$\mathbf{D}_i^T \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{X}_j) = \sum_{j=1}^{l}(\mathbf{D}_i^T\mathbf{D}_j\mathbf{X}_j) = \mathbf{D}_i^T\mathbf{D}_i\mathbf{X}_i = \mathbf{X}_i.$$

That is, $\mathbf{D}_i^T \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{X}_j) = \mathbf{X}_i$. We can observe that if $\{\mathbf{D}_i\}_{i=1}^l$ and $\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{X}_j)$ are respectively distributed to a patient and a server, the patient can use $\mathbf{D}_i^T$ to retrieve the matrix $\mathbf{X}_i$ from the server. Note that this idea will be used in the design of our PPFA in Subsection IV-B.

B. The Details of PPFA

PPFA is a privacy-preserving forward algorithm that can privately compute $\Pr(\mathbf{y}|\mathbf{\Lambda})$ through the ciphertext of $\mathbf{\Lambda}$ and the query token of $\mathbf{y}$, where $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^l)$ is a variant of HMM, and $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ is a time-series sequence. In the following, we first introduce the key idea of the PPFA, and then introduce its detailed construction.

Key idea of PPFA. In the PPFA, we use the VFA-HMM to compute $\Pr(\mathbf{y}|\mathbf{\Lambda})$. According to the VFA-HMM in Subsection III-B, computing $\Pr(\mathbf{y}|\mathbf{\Lambda})$ has two steps, i.e., (i) compute $\mathbf{\Psi} = \mathbf{\Pi}\mathbf{B}_{y_1}\prod_{i=2}^{T}\mathbf{A}\mathbf{B}_{y_i}$, where $\mathbf{B}_{y_i}$ is the diagonal matrix corresponding to the observation state y_i for $i = 1, 2, \cdots, T$; and (ii) compute $\Pr(\mathbf{y}|\mathbf{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{\Psi}_{i,j}$. From the VFA-HMM, we can observe that the design of PPFA involves three challenges.

(1) How to retrieve $\{\mathbf{\Pi}\mathbf{B}_{y_1}, \{\mathbf{A}\mathbf{B}_{y_i}\}_{i=2}^T\}$ from $\{\mathbf{\Pi}\mathbf{B}_i\}_{i=1}^l$ and $\{\mathbf{A}\mathbf{B}_i\}_{i=1}^l$? Specifically, to compute $\mathbf{\Psi}$, we need to retrieve the matrices $\{\mathbf{\Pi}\mathbf{B}_{y_1}, \{\mathbf{A}\mathbf{B}_{y_i}\}_{i=2}^T\}$ from all matrices $\{\mathbf{\Pi}\mathbf{B}_i\}_{i=1}^l$ and $\{\mathbf{A}\mathbf{B}_i\}_{i=1}^l$ based on $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$.
(2) How to preserve the privacy of all computations?
(3) How to guarantee the accuracy of the computation result $\Pr(\mathbf{y}|\mathbf{\Lambda})$? Since all data in the VFA-HMM are small probabilities, it is challenging to guarantee the accuracy of $\Pr(\mathbf{y}|\mathbf{\Lambda})$.

To address these challenges, we have two countermeasures as follows.

• **Countermeasure I.** We adopt a set of mutually orthogonal matrices to retrieve $\{\mathbf{\Pi}\mathbf{B}_{y_1}, \{\mathbf{A}\mathbf{B}_{y_i}\}_{i=2}^T\}$ from $\{\mathbf{\Pi}\mathbf{B}_i\}_{i=1}^l$ and $\{\mathbf{A}\mathbf{B}_i\}_{i=1}^l$. Let $\{\mathbf{D}_i\}_{i=1}^l$ be l mutually orthogonal matrices. We have

$$\begin{cases} \mathbf{D}_{y_i}^T\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{\Pi}\mathbf{B}_j) = \mathbf{\Pi}\mathbf{B}_{y_i} \\ \mathbf{D}_{y_i}^T\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{A}\mathbf{B}_j) = \mathbf{A}\mathbf{B}_{y_i}. \end{cases}$$

Furthermore, we have

$$\mathbf{\Psi} = \mathbf{\Pi}\mathbf{B}_{y_1}\prod_{i=2}^{T}\mathbf{A}\mathbf{B}_{y_i}$$
$$= (\mathbf{D}_{y_1}^T\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{\Pi}\mathbf{B}_j)) * \prod_{i=2}^{l}(\mathbf{D}_{y_i}^T\sum_{j=1}^{l}(\mathbf{D}_j\mathbf{A}\mathbf{B}_j)).$$

Without lose of generality, let $\widetilde{\mathbf{\Pi}} = \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{\Pi}\mathbf{B}_j)$ and $\widetilde{\mathbf{A}} = \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{A}\mathbf{B}_j)$. Then, we have

$$\mathbf{\Psi} = (\mathbf{D}_{y_1}^T\widetilde{\mathbf{\Pi}}) * (\prod_{i=2}^{l}(\mathbf{D}_{y_i}^T\widetilde{\mathbf{A}})).$$

Next, given two n-dimensional vectors $\boldsymbol{\alpha} = \boldsymbol{\beta} = [1, 1, \cdots, 1]^T$, and we can deduce that

$$\Pr(\mathbf{y}|\mathbf{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{\Psi}_{i,j} = \boldsymbol{\alpha}^T\mathbf{\Psi}\boldsymbol{\beta}.$$

• **Countermeasure II.** Since all data in the VFA-HMM are in the form of matrices, we consider applying the matrix encryption technique to preserve the privacy of the PPFA. Besides, we can gain two additional benefits from the matrix encryption, i.e., (i) the matrix encryption is computationally efficient; and (ii) the matrix encryption can guarantee the accuracy of the computation result $\Pr(\mathbf{y}|\mathbf{\Lambda})$ because it performs computations over real domain.

Detailed PPFA. Based on the above idea, we introduce the detailed PPFA that has five algorithms, i.e., key generation, encryption, token generation, retrieval, and decryption as follows.

• $\mathsf{VFA.KeyGen}(n, l, w, T_{max})$: The input of the key generation algorithm is n, l, w, and T_{max}, where n is the number of hidden states, l is the number of observations, w is a parameter for the randomness of ciphertexts and tokens, and T_{max} is the maximal length of time-series activities sequences in the system. Then, on input (n, l, w, T_{max}), the key generation algorithm generates the secret key as the following steps.

(1) Generate l mutually orthogonal matrices $\{\mathbf{D}_i\}_{i=1}^l$, where $\mathbf{D}_i \in \mathbb{R}^{(n*l)\times n}$ for $1 \leq i \leq l$, and $\mathbb{R}$ denotes the real domain.

(2) Generate $2T_{max}$ invertible matrices $\{\mathbf{M}_{2i-1}, \mathbf{M}_{2i}\}_{i=1}^{T_{max}}$, where $\mathbf{M}_{2i-1} \in \mathbb{R}^{(n*l+w)\times(n*l+w)}$ and $\mathbf{M}_{2i} \in \mathbb{R}^{(n+w)\times(n+w)}$. Meanwhile, generate T_{max} $w \times w$ matrices $\mathbf{R}_{\mathbf{\Pi}}$ and $\{\mathbf{R}_{\mathbf{A},i}\}_{i=2}^{T_{max}}$.

The key generation algorithm outputs the secret key $sk = \{\{\mathbf{D}_i\}_{i=1}^l, \{\mathbf{M}_{2i-1}, \mathbf{M}_{2i}\}_{i=1}^{T_{max}}, \mathbf{R}_{\mathbf{\Pi}}, \{\mathbf{R}_{\mathbf{A},i}\}_{i=2}^{T_{max}}\}$.

• $\mathsf{VFA.Enc}(\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^l), sk)$: We use sk to encrypt the model $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^l)$ as follows.

(1) We compute

$$\widetilde{\mathbf{\Pi}} = \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{\Pi}\mathbf{B}_j);\ \widetilde{\mathbf{A}} = \sum_{j=1}^{l}(\mathbf{D}_j\mathbf{A}\mathbf{B}_j).$$

(2) We use the matrices $\mathbf{R_\Pi}$ and $\{\mathbf{R_{A},}_i\}_{i=2}^{T_{max}}$ to extend $\widetilde{\mathbf{\Pi}}$ and $\widetilde{\mathbf{A}}$ as

$$\widetilde{\mathbf{\Pi}}' = \begin{bmatrix} \widetilde{\mathbf{\Pi}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_\Pi} \end{bmatrix};\ \widetilde{\mathbf{A}}'_i = \begin{bmatrix} \widetilde{\mathbf{A}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{A},}_i \end{bmatrix} \text{for } i = 2, 3, \cdots, T_{max}.$$

(3) We use the matrices $\{\mathbf{M}_{2i-1}, \mathbf{M}_{2i}\}_{i=1}^{T_{max}}$ to encrypt $\widetilde{\mathbf{\Pi}}'$ and $\{\widetilde{\mathbf{A}}'_i\}_{i=2}^{T_{max}}$ as

$$\begin{cases} \mathbf{C}_1 = \mathbf{M}_1 \widetilde{\mathbf{\Pi}}' \mathbf{M}_2 \\ \mathbf{C}_i = \mathbf{M}_{2i-1} \widetilde{\mathbf{A}}'_i \mathbf{M}_{2i} \quad \text{for } i = 2, 3, \cdots, T_{max}. \end{cases} \tag{2}$$

The algorithm outputs the ciphertexts $\{\mathbf{C}_i\}_{i=1}^{T_{max}}$.

Remark: In the encryption algorithm, we respectively add T_{max} $w \times w$ random matrices $\mathbf{R_\Pi}$ and $\{\mathbf{R_{A},}_i\}_{i=2}^{T_{max}}$ into $\widetilde{\mathbf{\Pi}}'$ and $\{\widetilde{\mathbf{A}}'_i\}_{i=2}^{T_{max}}$. These random matrices can make the ciphertexts $\{\mathbf{C}_i\}_{i=1}^{T_{max}}$ probabilistic rather than deterministic.

- $\mathsf{VFA.TokenGen}(\mathbf{y} = \{y_1, y_2, \cdots, y_T\}, sk)$: Let $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ be a time-series activities sequence, where $T \leq T_{max}$. We generate a query token for it as follows.

(1) We choose T non-zero random numbers $\{r_{y_i}\}_{i=1}^{T}$ and T non-zero random matrices $\{\mathbf{R_{D},}_{y_i}\}_{i=1}^{T}$, where $r_{y_i} \in \mathbb{R}$ and $\mathbf{R_{D},}_{y_i} \in \mathbb{R}^{w \times w}$. Then, we use r_{y_i} and $\mathbf{R_{D},}_{y_i}$ to extend $\mathbf{D}_{y_i}$ to a new matrix $\widetilde{\mathbf{D}}_{y_i}$ as

$$\widetilde{\mathbf{D}}_{y_i} = \begin{bmatrix} r_{y_i} * \mathbf{D}_{y_i} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{D},}_{y_i} \end{bmatrix}.$$

(2) We choose two n-dimensional vectors $\boldsymbol{\alpha} = \boldsymbol{\beta} = [1, 1, \cdots, 1]^T$. Then, we generate two w-dimensional vectors $\mathbf{r}_\alpha, \mathbf{r}_\beta \in \mathbb{R}^w$. Furthermore, we construct two $(n+w)$-dimensional vectors $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ as

$$\begin{cases} \boldsymbol{\alpha}' = r'_\alpha * [\boldsymbol{\alpha}^T, \mathbf{r}_\alpha^T]^T \\ \boldsymbol{\beta}' = r'_\beta * [\boldsymbol{\beta}^T, \mathbf{r}_\beta^T]^T, \end{cases}$$

where r'_α and r'_β are two non-zero random numbers. Then, we generate the following query token.

$$\begin{cases} \mathbf{Q}_1 = \boldsymbol{\alpha}'^T \widetilde{\mathbf{D}}_{y_1}^T \mathbf{M}_1^{-1} \\ \mathbf{Q}_i = \mathbf{M}_{2i-2}^{-1} \widetilde{\mathbf{D}}_{y_i}^T \mathbf{M}_{2i-1}^{-1} \quad \text{for } i = 2, 3, \cdots, T \\ \mathbf{V} = \mathbf{M}_{2T}^{-1} * \boldsymbol{\beta}'. \end{cases} \tag{3}$$

The algorithm outputs the query token $\{\{\mathbf{Q}_i\}_{i=1}^{T}, \mathbf{V}\}$ and chosen numbers $\mathcal{R} = \{\{r_{y_i}, \mathbf{R_{D},}_{y_i}\}_{i=1}^{T}, r'_\alpha, r'_\beta, \mathbf{r}_\alpha, \mathbf{r}_\beta\}$.

Remark: In the token generation algorithm, we also add some random numbers and matrices into the query token $\{\{\mathbf{Q}_i\}_{i=1}^{T}, \mathbf{V}\}$, which makes them probabilistic rather than deterministic.

- $\mathsf{VFA.Retrieval}(\{\mathbf{C}_i\}_{i=1}^{T_{max}}, \{\mathbf{Q}_i\}_{i=1}^{T}, \mathbf{V})$: On input $\{\mathbf{C}_i\}_{i=1}^{T_{max}}$, $\{\mathbf{Q}_i\}_{i=1}^{T}$ and $\mathbf{V}$, the retrieval algorithm outputs the value of $z = (\prod_{i=1}^{T}(\mathbf{Q}_i \mathbf{C}_i)) * \mathbf{V}$.
- $\mathsf{VFA.Dec}(z, sk, \mathcal{R})$: On input $\{z, sk\}$ and $\mathcal{R} = \{\{r_{y_i}, \mathbf{R_{D},}_{y_i}\}_{i=1}^{T}, r'_\alpha, r'_\beta, \mathbf{r}_\alpha, \mathbf{r}_\beta\}$, the algorithm computes $\Pr(\mathbf{y}|\mathbf{\Lambda})$ as

$$\Pr(\mathbf{y}|\mathbf{\Lambda}) = \frac{1}{\prod_{i=1}^{T} r_{y_i}} \left(\frac{z}{r'_\alpha * r'_\beta} - \mathbf{r}_\alpha^T * (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi} \prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) * \mathbf{r}_\beta\right).$$

Correctness. The PPFA is correct *iff* $\mathsf{VFA.Dec}(\cdot)$ is correct, i.e.,

$$\begin{aligned} &\Pr(\mathbf{y}|\mathbf{\Lambda}) \\ &= \frac{1}{\prod_{i=1}^{T} r_{y_i}} \left(\frac{z}{r'_\alpha * r'_\beta} - \mathbf{r}_\alpha^T * (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi} \prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) * \mathbf{r}_\beta\right) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{\Psi}_{i,j}. \end{aligned}$$

Proof. First, we have

$$\begin{aligned} z &= (\prod_{i=1}^{T} (\mathbf{Q}_i \mathbf{C}_i)) * \mathbf{V} = (\mathbf{Q}_1 \mathbf{C}_1 \prod_{i=2}^{T} (\mathbf{Q}_i \mathbf{C}_i)) * \mathbf{V} \\ &= (\boldsymbol{\alpha}'^T \widetilde{\mathbf{D}}_{y_1}^T \mathbf{M}_1^{-1} \mathbf{M}_1 \widetilde{\mathbf{\Pi}}' \mathbf{M}_2) \\ &\quad * (\prod_{i=2}^{T} (\mathbf{M}_{2i-2}^{-1} \widetilde{\mathbf{D}}_{y_i}^T \mathbf{M}_{2i-1}^{-1} \mathbf{M}_{2i-1} \widetilde{\mathbf{A}}'_i \mathbf{M}_{2i})) * (\mathbf{M}_{2T}^{-1} * \boldsymbol{\beta}') \\ &= \boldsymbol{\alpha}'^T \widetilde{\mathbf{D}}_{y_1}^T \widetilde{\mathbf{\Pi}}' (\prod_{i=2}^{T} (\widetilde{\mathbf{D}}_{y_i}^T \widetilde{\mathbf{A}}'_i)) \boldsymbol{\beta}'. \end{aligned}$$

Meanwhile, we have

$$\begin{aligned} \widetilde{\mathbf{D}}_{y_1}^T \widetilde{\mathbf{\Pi}}' &= \begin{bmatrix} r_{y_1} * \mathbf{D}_{y_1} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{D},}_{y_1} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{\Pi}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_\Pi} \end{bmatrix} \\ &= \begin{bmatrix} r_{y_1} * \mathbf{D}_{y_1} \widetilde{\mathbf{\Pi}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{D},}_{y_1} \mathbf{R_\Pi} \end{bmatrix} \\ \widetilde{\mathbf{D}}_{y_i}^T \widetilde{\mathbf{A}}' &= \begin{bmatrix} r_{y_i} * \mathbf{D}_{y_i} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{D},}_{y_i} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{A}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{A},}_{y_i} \end{bmatrix} \\ &= \begin{bmatrix} r_{y_i} * \mathbf{D}_{y_i} \widetilde{\mathbf{A}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i} \end{bmatrix} \quad \text{for } i = 2, 3, \cdots, T. \end{aligned}$$

Then, we can deduce that

$$\begin{aligned} &\widetilde{\mathbf{D}}_{y_1}^T \widetilde{\mathbf{\Pi}}' (\prod_{i=2}^{T} (\widetilde{\mathbf{D}}_{y_i}^T \widetilde{\mathbf{A}}')) \\ &= \begin{bmatrix} (r_{y_1} \mathbf{D}_{y_1}^T \widetilde{\mathbf{\Pi}}) (\prod_{i=2}^{l} (r_{y_i} \mathbf{D}_{y_i}^T \widetilde{\mathbf{A}})) & \mathbf{O} \\ \mathbf{O} & (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi}) (\prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{\Psi} \prod_{i=1}^{T} r_{y_i} & \mathbf{O} \\ \mathbf{O} & (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi}) (\prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) \end{bmatrix}. \end{aligned}$$

Furthermore, we have

$$\begin{aligned} z &= \boldsymbol{\alpha}'^T \widetilde{\mathbf{D}}_{y_1}^T \widetilde{\mathbf{\Pi}}' (\prod_{i=2}^{T} (\widetilde{\mathbf{D}}_{y_i}^T \widetilde{\mathbf{A}}')) \boldsymbol{\beta}' \\ &= \boldsymbol{\alpha}'^T \begin{bmatrix} \mathbf{\Psi} \prod_{i=1}^{T} r_{y_i} & \mathbf{O} \\ \mathbf{O} & (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi}) (\prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) \end{bmatrix} \boldsymbol{\beta}' \\ &= r'_\alpha * r'_\beta ((\prod_{i=1}^{T} r_{y_i}) \boldsymbol{\alpha}^T \mathbf{\Psi} \boldsymbol{\beta} + \mathbf{r}_\alpha^T (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi}) (\prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) \mathbf{r}_\beta). \end{aligned}$$

In this case, we can deduce that

$$\begin{aligned} &\Pr(\mathbf{y}|\mathbf{\Lambda}) \\ &= \frac{1}{\prod_{i=1}^{T} r_{y_i}} \left(\frac{z}{r'_\alpha * r'_\beta} - \mathbf{r}_\alpha^T (\mathbf{R_{D},}_{y_1} \mathbf{R_\Pi} \prod_{i=2}^{T} (\mathbf{R_{D},}_{y_i} \mathbf{R_{A},}_{y_i})) \mathbf{r}_\beta\right) \\ &= \frac{1}{\prod_{i=1}^{T} r_{y_i}} (\prod_{i=1}^{T} r_{y_i}) \boldsymbol{\alpha}^T \mathbf{\Psi} \boldsymbol{\beta} \\ &= \boldsymbol{\alpha}^T \mathbf{\Psi} \boldsymbol{\beta} = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{\Psi}_{i,j}. \end{aligned}$$

Thus, the correctness of PPFA follows. □

C. Our Proposed Healthcare Monitoring Scheme

In this subsection, we propose an efficient and privacy-preserving time-series activities based healthcare monitoring scheme. In our scheme, we achieve the time-series activities based healthcare monitoring through two variant of HMMs, i.e., $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$. The models $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ are respectively trained by normal time-series activities sequences and abnormal time-series activities sequences. Given a time-series activities sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$ that needs to be classified, we first compute $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$. If $\Pr(\mathbf{y}|\mathbf{\Lambda}_1) > \Pr(\mathbf{y}|\mathbf{\Lambda}_2)$, $\mathbf{y}$ is classified to be normal. Otherwise, $\mathbf{y}$ is classified to be abnormal. Specifically, our scheme contains three phases, i.e., setup, classification model outsourcing, and classification query.

• ***Setup:*** The healthcare center is responsible for setting up the whole system. As described above, it has a classification model with two variant of HMMs, i.e., $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$. Suppose that $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ have n hidden states and l observation states, and T_{max} is the maximal length of the patient's time-series activities sequences. Then, the healthcare center sets the parameter w and calls the $\mathsf{VFA.KeyGen}(\cdot)$ algorithm to generate two secret keys as

$$\begin{cases} sk_1 \leftarrow \mathsf{VFA.KeyGen}(n, l, w, T_{max}) \\ sk_2 \leftarrow \mathsf{VFA.KeyGen}(n, l, w, T_{max}), \end{cases}$$

where sk_1 and sk_2 are respectively used for encrypting $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$. Furthermore, when the patient registers in the system, the healthcare center authorizes them with the secret keys $\{sk_1, sk_2\}$.

• ***Classification model outsourcing:*** In this phase, the healthcare center encrypts the classification model $(\mathbf{\Lambda}_1, \mathbf{\Lambda}_2)$, and outsources it to the cloud server. Specifically, it first calls the $\mathsf{VFA.Enc}(\cdot)$ algorithm to encrypt $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ as

$$\begin{cases} \{\mathbf{C}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T_{max}} \leftarrow \mathsf{VFA.Enc}(\mathbf{\Lambda}_1, sk_1) \\ \{\mathbf{C}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T_{max}} \leftarrow \mathsf{VFA.Enc}(\mathbf{\Lambda}_2, sk_2). \end{cases}$$

Then, it outsources the encrypted classification model $(\{\mathbf{C}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T_{max}}, \{\mathbf{C}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T_{max}})$ to the cloud server.

• ***Classification query:*** In the classification query phase, a patient can obtain the classification result with the help of the cloud server as follows.

(1) Patient $\rightarrow$ Server: The patient calls the $\mathsf{VFA.TokenGen}(\cdot)$ algorithm to generate two tokes

$$\begin{cases} \{\{\mathbf{Q}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_1}\}, \mathcal{R}_{\mathbf{\Lambda}_1} \leftarrow \mathsf{VFA.TokenGen}(\mathbf{y}, sk_1) \\ \{\{\mathbf{Q}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_2}\}, \mathcal{R}_{\mathbf{\Lambda}_2} \leftarrow \mathsf{VFA.TokenGen}(\mathbf{y}, sk_2). \end{cases}$$

Then, the patient sends $(\{\{\mathbf{Q}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_1}\}, \{\{\mathbf{Q}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_2}\})$ to the cloud server via a secure channel and locally keeps $(\mathcal{R}_{\mathbf{\Lambda}_1}, \mathcal{R}_{\mathbf{\Lambda}_2})$.

(2) Server $\rightarrow$ Patient: On receiving $(\{\{\mathbf{Q}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_1}\}, \{\{\mathbf{Q}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_2}\})$, the cloud server calls the $\mathsf{VFA.Retrieval}(\cdot)$ algorithm to obtain

$$\begin{cases} z_1 \leftarrow \mathsf{VFA.Retrieval}(\{\mathbf{C}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T_{max}}, \{\mathbf{Q}_{\mathbf{\Lambda}_1,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_1}) \\ z_2 \leftarrow \mathsf{VFA.Retrieval}(\{\mathbf{C}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T_{max}}, \{\mathbf{Q}_{\mathbf{\Lambda}_2,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{\Lambda}_2}). \end{cases}$$

Then, the cloud server returns (z_1, z_2) to the patient via a secure channel.

(3) Patient: On receiving (z_1, z_2), the patient calls the $\mathsf{VFA.Dec}(\cdot)$ algorithm to obtain

$$\begin{cases} \Pr(\mathbf{y}|\mathbf{\Lambda}_1) \leftarrow \mathsf{VFA.Dec}(z_1, sk_1, \mathcal{R}_{\mathbf{\Lambda}_1}) \\ \Pr(\mathbf{y}|\mathbf{\Lambda}_2) \leftarrow \mathsf{VFA.Dec}(z_2, sk_2, \mathcal{R}_{\mathbf{\Lambda}_2}). \end{cases}$$

If $\Pr(\mathbf{y}|\mathbf{\Lambda}_1) > \Pr(\mathbf{y}|\mathbf{\Lambda}_2)$, $\mathbf{y}$ is classified to be normal. Otherwise, $\mathbf{y}$ is classified to be abnormal.

V. Security Analysis

In this section, we analyze the security of the proposed healthcare monitoring scheme. Since the healthcare monitoring scheme is constructed based on the PPFA, we first prove the security of the PPFA, and then analyze the security of the healthcare monitoring scheme.

A. Security of PPFA

PPFA is essentially a searchable encryption scheme. In this scheme, a variant of HMM $\mathbf{\Lambda}$ is encrypted into ciphertexts $\{\mathbf{C}_i\}_{i=1}^{T_{max}}$. Given a time-series activities sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$, we can generate a query token that can be used to compute $\Pr(\mathbf{y}|\mathbf{\Lambda})$. Same as the searchable encryption scheme in [25], we prove the security of the PPFA in the real and ideal experiments setting. The main idea of the real and ideal experiments setting is to show that a simulator in the ideal experiment and without any secret information can simulate the ciphertexts and tokens in the real experiment such that any probabilistic polynomial time (PPT) adversary cannot distinguish the real and ideal experiments. Both real and ideal experiments have four phases, including setup phase, token generation phase 1, challenge phase, and token generation phase 2, which respectively correspond to the $\mathsf{VFA.KeyGen}(\cdot)$ algorithm, the $\mathsf{VFA.TokenGen}(\cdot)$ algorithm, the $\mathsf{VFA.Enc}(\cdot)$ algorithm, and the $\mathsf{VFA.TokenGen}(\cdot)$ algorithm. It is worth noting that the real and ideal experiments contain two token generation phases, where the first one is before the challenge phase and the second one is after the challenge phase. That is, an adversary can adaptively obtain query tokens. The reason why we allow the adversary to have such an attack capability is that the patients in our healthcare monitoring scheme are allowed to launch classification queries multiple times, so the adversary (i.e., the cloud server) has a chance to adaptively obtain query tokens. Specifically, the real and ideal experiments can be defined as follows.

Real experiment. The real experiment has two participants, i.e., a PPT adversary $\mathcal{A}$ and a challenger. They interact as the following four phases.

• ***Setup:*** In the setup phase, the adversary $\mathcal{A}$ generates a variant of HMM $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$. The model $\mathbf{\Lambda}$ contains n hidden states and l observation states. The maximal length of time-series activities sequences in the system is T_{max}. Meanwhile, $\mathcal{A}$ sets a parameter w. Then, $\mathcal{A}$ sends $\mathbf{\Lambda}$ and (n, l, w, T_{max}) to the challenger. On receiving them, the challenger generates a secret key $sk \leftarrow \mathsf{VFA.KeyGen}(n, l, w, T_{max})$. Furthermore, it uses sk to encrypt $\mathbf{\Lambda}$ into ciphertexts as $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}} \leftarrow \mathsf{VFA.Enc}(\mathbf{\Lambda}, sk)$.

• ***Token generation phase 1:*** In the token generation phase 1, the adversary $\mathcal{A}$ generates p_1 time-series activities sequences $\{\mathbf{y}_j\}_{j=1}^{p_1}$, and the length of $\mathbf{y}_j$ is T_j, where $T_j \leq T_{max}$. Then, $\mathcal{A}$ sends $\{\mathbf{y}_j\}_{j=1}^{p_1}$ to the challenger. On receiving $\{\mathbf{y}_j\}_{j=1}^{p_1}$, the challenger generates query tokens for them. Specifically, for each $\mathbf{y}_j$, the challenger generates a query token as $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{y}_j}\} \leftarrow \mathsf{VFA.TokenGen}(\mathbf{y}_i, sk)$. Then, the challenger returns these tokens $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_1}$ to $\mathcal{A}$.

• ***Challenge phase:*** The challenger returns the ciphertexts $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ to $\mathcal{A}$.

• ***Token generation phase 2:*** In the token generation phase 2, $\mathcal{A}$ generates $(p_2 - p_1)$ time-series activities sequences $\{\mathbf{y}_j\}_{j=p_1+1}^{p_2}$, and the length of $\mathbf{y}_j$ is T_j, where $T_j \leq T_{max}$. Then, same as ***Token generation phase 1***, $\mathcal{A}$ gets the query tokens of $\{\mathbf{y}_j\}_{j=p_1+1}^{p_2}$, i.e., $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{y}_j}\}_{j=p_1+1}^{p_2}$, from the challenger.

Ideal experiment. The ideal experiment has two participants, i.e., a PPT adversary $\mathcal{A}$ and a simulator, and they interact as follows.

• ***Setup:*** In the setup phase, $\mathcal{A}$ sends a variant of HMM $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$ and the corresponding parameters (n, l, w, T_{max}) to the simulator. On receiving them, the simulator generates T_{max} $(n * l + w) \times (n + w)$ non-zeros random matrices $\{\mathbf{C}'_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ as the ciphertexts.

• ***Token generation phase 1:*** In the token generation phase 1, $\mathcal{A}$ sends p_1 time-series activities sequences $\{\mathbf{y}_j\}_{j=1}^{p_1}$ to the simulator. On receiving $\{\mathbf{y}_j\}_{j=1}^{p_1}$, the simulator generates p_1 random query tokens. Specifically, for $\mathbf{y}_j$, the simulator generates an $(n * l + w)$-dimensional vector $\mathbf{Q}'_{\mathbf{y}_j,1}$, $(T-1)$ $(n+w) \times (n * l + w)$ matrices $\{\mathbf{Q}'_{\mathbf{y}_j,i}\}_{i=2}^{T}$, and an $(n+w)$-dimensional vector $\mathbf{V}'_{\mathbf{y}_j}$ as the query token of $\mathbf{y}_j$. Finally, the simulator sends all query tokens $\{\{\mathbf{Q}'_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}'_{\mathbf{y}_j}\}_{j=1}^{p_1}$ to $\mathcal{A}$.

• ***Challenge phase:*** The simulator returns the ciphertexts $\{\mathbf{C}'_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ to $\mathcal{A}$.

• ***Token generation phase 2:*** In the token generation phase 2, $\mathcal{A}$ generates $(p_2 - p_1)$ time-series activities sequences $\{\mathbf{y}_j\}_{j=p_1+1}^{p_2}$, and the length of $\mathbf{y}_j$ is T_j, where $T_j \leq T_{max}$. Then, same as ***Token generation phase 1*** of the ideal experiment, $\mathcal{A}$ gets the query tokens of $\{\mathbf{y}_j\}_{j=p_1+1}^{p_2}$, i.e., $\{\{\mathbf{Q}'_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}'_{\mathbf{y}_j}\}_{j=p_1+1}^{p_2}$, from the simulator.

In the real experiment, the view of $\mathcal{A}$ is $\mathrm{View}_{\mathcal{A},\mathrm{Real}} = \{\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}, \{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}\}$. In the ideal experiment, the view of $\mathcal{A}$ is $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}} = \{\{\mathbf{C}'_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}, \{\{\mathbf{Q}'_{\mathbf{y}_j,i}\}_{i=1}^{T}, \mathbf{V}'_{\mathbf{y}_j}\}_{j=1}^{p_2}\}$. Based on $\mathcal{A}$'s views in the real and ideal experiments, we formally define the security of the privacy-preserving VFA-HMM.

Definition 2 (Security of PPFA): PPFA is selectively secure *iff* for any PPT adversary $\mathcal{A}$ issuing one time model encryption and a polynomial number of query token generations, there exists a simulator such that the advantage that $\mathcal{A}$ can distinguish the views of real and ideal experiments is negligible. That is, $|\Pr[\mathrm{View}_{\mathcal{A},\mathrm{Real}} = 1] - \Pr[\mathrm{View}_{\mathcal{A},\mathrm{Ideal}} = 1]|$ is negligible.

Theorem 1: PPFA is selectively secure.

Proof. According to Definition 2, PPFA is selectively secure *iff* $\mathcal{A}$ cannot distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ and $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$. Since all ciphertexts and tokens in the ideal experiment are random generated, distinguishing $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$ is equivalent to distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from random ciphertexts and tokens. Since $\mathrm{View}_{\mathcal{A},\mathrm{Real}} = \{\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}, \{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}\}$, we respectively show that $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$, $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$, and the intermediate results computed by them are indistinguishable from random ciphertexts and tokens.

• $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ **are indistinguishable from random ciphertexts.** According to Eq. (2), we have $\mathbf{C}_1 = \mathbf{M}_1\widetilde{\mathbf{\Pi}}'\mathbf{M}_2$ and $\mathbf{C}_i = \mathbf{M}_{2i-1}\widetilde{\mathbf{A}}'_i\mathbf{M}_{2i}$ for $i = 2, 3, \cdots, T_{max}$, where

$$\widetilde{\mathbf{\Pi}}' = \begin{bmatrix} \widetilde{\mathbf{\Pi}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathbf{\Pi}} \end{bmatrix}; \ \widetilde{\mathbf{A}}'_i = \begin{bmatrix} \widetilde{\mathbf{A}} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathbf{A},i} \end{bmatrix} \text{for } i = 2, 3, \cdots, T_{max}.$$

Since $\mathbf{R}_{\mathbf{\Pi}}$, $\{\mathbf{R}_{\mathbf{A},i}\}_{i=2}^{T_{max}}$ are random matrices and $\{\mathbf{M}_{2i-1}, \mathbf{M}_{2i}\}_{i=1}^{T_{max}}$ are unknown, $\mathbf{C}_1$ and $\{\mathbf{C}_i\}_{i=2}^{T_{max}}$ are indistinguishable from random matrices. That is, $\{\mathbf{C}_i\}_{i=1}^{T_{max}}$ are indistinguishable from random matrices.

• $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$ **are indistinguishable from random tokens.** For different $\mathbf{y}_j$'s, the token $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}$, $\mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$ are different. Without loss of generality, we use the same symbol to denote each query token, i.e., $\{\{\mathbf{Q}_i\}_{i=1}^{T}, \mathbf{V}\}$. According to Eq. (3), we have $\mathbf{Q}_1 = \boldsymbol{\alpha}'^T\widetilde{\mathbf{D}}_{y_1}^T\mathbf{M}_1^{-1}$, $\mathbf{Q}_i = \mathbf{M}_{2i-2}^{-1}\widetilde{\mathbf{D}}_{y_i}^T\mathbf{M}_{2i-1}^{-1}$ for $i = 2, 3, \cdots, T$, and $\mathbf{V} = \mathbf{M}_{2T}^{-1} * \boldsymbol{\beta}'$, where

$$\begin{cases} \boldsymbol{\alpha}' = r'_\alpha * [\boldsymbol{\alpha}^T, \mathbf{r}_\alpha^T]^T \\ \boldsymbol{\beta}' = r'_\beta * [\boldsymbol{\beta}^T, \mathbf{r}_\beta^T]^T \\ \widetilde{\mathbf{D}}_{y_i} = \begin{bmatrix} r_{y_i} * \mathbf{D}_{y_i} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathbf{D},y_i} \end{bmatrix} \text{ for } i = 1, 3, \cdots, T. \end{cases}$$

For the vector $\mathbf{Q}_1$, since r'_α and r_{y_i} are two random numbers, $\mathbf{R}_{\mathbf{D},y_1}$ is a random matrix, and $\mathbf{D}_{y_1}$, $\mathbf{M}_1^{-1}$ are unknown, we can deduce that $\mathbf{Q}_1$ is indistinguishable from a random vector. For $\mathbf{Q}_i$, since $\{r_{y_i}, \mathbf{R}_{\mathbf{D},y_i}\}_{i=1}^{T}$ are random chosen and $\{\mathbf{M}_{2i-1}, \mathbf{M}_{2i}\}_{i=1}^{T}$ are unknown, we can deduce that $\{\mathbf{Q}_i\}_{i=2}^{T}$ are indistinguishable from random matrices. For the vector $\mathbf{V}$, since r'_β is a random number, $\mathbf{r}_\beta$ is a random vector, and $\mathbf{M}_{2T}$ is unknown, we can deduce that $\mathbf{V}$ is indistinguishable from a random vector. Therefore, $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$ are indistinguishable from random tokens.

• **The intermediate results computed from** $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ **and** $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$ **are indistinguishable from random numbers.** Since each ciphertext or token in $\{\mathbf{C}_{\mathbf{\Lambda},j}\}_{j=1}^{T_{max}}$ and $\{\{\mathbf{Q}_{\mathbf{y}_j,i}\}_{i=1}^{T_j}, \mathbf{V}_{\mathbf{y}_j}\}_{j=1}^{p_2}$ contains random numbers, random vector, random matrices, or unknown matrices. When the adversary $\mathcal{A}$ uses them to generate some intermediate results, these random numbers still exist in the intermediate results. Thus, the intermediate results are also indistinguishable from random numbers.

Therefore, $\mathcal{A}$ cannot distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from random ciphertexts and tokens. That is, $\mathcal{A}$ cannot distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$. Hence, PPFA is selectively secure. □

B. Security of Our Healthcare Monitoring Scheme

In this subsection, we analyze the security of our healthcare monitoring scheme. We will show that the classification model,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3079106, IEEE Internet of Things Journal

the classification query requests, and the classification results are privacy-preserving.

• **The classification model $(\mathbf{\Lambda}_1, \mathbf{\Lambda}_2)$ is privacy-preserving.** Since the classification model is a private asset of the healthcare center, it should be kept secret from the cloud server and patients. In our classification scheme, the model $(\mathbf{\Lambda}_1, \mathbf{\Lambda}_2)$ are respectively encrypted by the PPFA. Then, the security of the PPFA can guarantee that the cloud server has no idea on the plaintext model. For the patients, when they launch a classification request, they can receive two values z_1 and z_2. Although they can use z_1 and z_2 to recover the probabilities $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$, it is hard for them to obtain the classification model $(\mathbf{\Lambda}_1, \mathbf{\Lambda}_2)$ from the probabilities $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$. Therefore, the classification model $(\mathbf{\Lambda}_1, \mathbf{\Lambda}_2)$ is privacy-preserving.

• **The classification query requests are privacy-preserving.** The query requests contain patients' private time-series activities sequences, which should be kept secret from the cloud server. In our classification scheme, patients use PPFA to generate tokens for the queried time-series sequences. Then, the security of the PPFA can guarantee that the cloud server has no idea on the queried time-series activities sequences. Thus, the classification query requests are privacy-preserving.

• **The classification results are privacy-preserving.** The classification results are the private information of patients, which should be kept secret from the cloud server. In our scheme, the classification results are determined by $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$. For the cloud server, it can only obtain the values z_1 and z_2 that are related to $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$. However, since z_1 and z_2 also contain other random numbers, it is hard for the cloud server to recover $\Pr(\mathbf{y}|\mathbf{\Lambda}_1)$ and $\Pr(\mathbf{y}|\mathbf{\Lambda}_2)$ from z_1 and z_2. Thus, the cloud server has no idea on the classification results, so the classification results are privacy-preserving.

VI. Performance Evaluation

In this section, we compare our proposed healthcare monitoring scheme with existing related schemes and evaluate the computational cost of our healthcare monitoring scheme.

Comparison with existing schemes. Our healthcare monitoring scheme is the first privacy-preserving healthcare monitoring scheme in the single-serve setting. For other existing schemes, they are constructed either in the scenario of two-party computation [16]–[21] or in the two-server setting [12], [13], which are not applicable to our scenario. Moreover, as discussed in Section I, we can use a fully homomorphic encryption scheme to construct a healthcare monitoring scheme in the single-serve setting. However, existing fully homomorphic encryption schemes are computationally inefficient, so the fully homomorphic encryption based healthcare monitoring scheme will be theoretically much less efficient than our healthcare monitoring scheme.

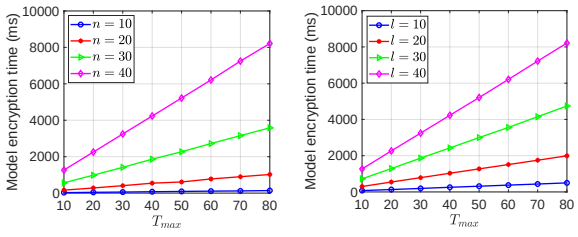
Next, we will evaluate the performance of our healthcare monitoring scheme. Since our healthcare monitoring scheme is constructed based on the PPFA, its computational cost completely depends on that of PPFA. Thus, we focus on evaluating the computational cost of PPFA.

Experimental setting. We implemented PPFA in Java and run experiments on a machine with an Intel(R) Core(TM) i7-3770 CPU @3.40GHz, 16GB RAM and Windows 10 operating system. Since the computational cost of the PPFA is not affected by the distribution and skewness of the dataset, we perform the evaluation on a synthetic dataset. That is, the HMM model and time-series activities sequences are generated randomly. In the HMM model, the number of hidden states ranges from 10 to 40. The number of observations ranges from 10 to 40. The maximal length of the time-series activities sequences ranges from 10 to 80.

We evaluate the computational cost of the PPFA from the aspects of encryption, token generation, retrieval, and decryption algorithms. As described in Subsection IV-B, the computational cost of the PPFA is affected by five parameters including (i) n: the number of hidden states; (ii) l: the number of observations; (iii) T_{max}: the maximal length of the time-series activities sequences; (iv) T: the length of a time-series activities sequence; and (v) w: the number of artificial dimensions. In our experiments, we set $w = 2$, which is enough to provide the randomness of the ciphertexts and tokens in the PPFA. Then, we mainly consider how n, l, T, and T_{max} affect the computational cost of the PPFA. Each experiment is conducted 100 times, and the average result is reported. Detailed experimental results are described below.

A. Computational Cost of Encryption Algorithm

The encryption algorithm is used to encrypt an HMM model $\mathbf{\Lambda} = (\mathbf{\Pi}, \mathbf{A}, \{\mathbf{B}_j\}_{j=1}^{l})$. The corresponding computational cost mainly comes from two parts, i.e., (i) compute $\widetilde{\mathbf{\Pi}} = \sum_{j=1}^{l}(\mathbf{D}_j \mathbf{\Pi} \mathbf{B}_j)$ and $\widetilde{\mathbf{A}} = \sum_{j=1}^{l}(\mathbf{D}_j \mathbf{A} \mathbf{B}_j)$; and (ii) compute $\mathbf{C}_1 = \mathbf{M}_1 \widetilde{\mathbf{\Pi}}' \mathbf{M}_2$ and $\{\mathbf{C}_i = \mathbf{M}_{2i-1} \widetilde{\mathbf{A}}_i' \mathbf{M}_{2i}\}_{i=2}^{T_{max}}$. For the first part, the computational cost is $O(2n^3 l^2)$. For the second part, the computational cost is $O(T_{max}(n^3 l^2 + n^3 l))$. Thus, the total computational cost is about $O(T_{max}(n^3 l^2 + n^3 l) + 2n^3 l^2)$, which is closely related to the parameters $\{n, l, T_{max}\}$. In the following, we show how $\{n, l, T_{max}\}$ affect the computational cost of the encryption algorithm.



(a) n and T_{max} (b) l and T_{max}

Fig. 2. Computational cost of model encryption algorithm

• n and T_{max}: In Fig. 2(a), we plot the computational cost of the model encryption algorithm varying with n and T_{max}. In this experiment, the parameters are set as $l = 40$, n is in the range of $\{10, 20, 30, 40\}$, and T_{max} ranges from 10 to 80. From Fig. 2(a), we can see that the computational cost of the model encryption algorithm linearly increases with T_{max} and shows a cubic growth trend with n. Overall, the model

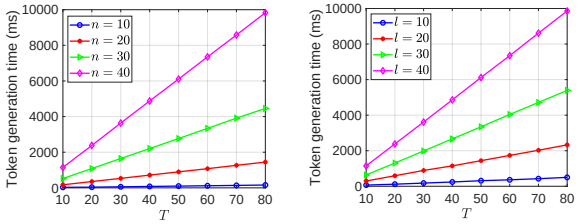
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3079106, IEEE Internet of Things Journal

encryption algorithm is efficient. For example, when $n = 40$, $l = 40$, and $T_{max} = 80$, encrypting an HMM model only takes 8213.5 ms.

• l and T_{max}: In Fig. 2(b), we plot the computational cost of the model encryption algorithm varying with l and T_{max}. In this experiment, the parameters are set as $n = 40$, l is in the range of $\{10, 20, 30, 40\}$, and T_{max} ranges from 10 to 80. From Fig. 2(b), we can see that the computational cost of the model encryption algorithm linearly increases with T_{max} and quadratically increases with l.

B. Computational Cost of Token Generation Algorithm

The token generation algorithm is used to generate a token for a time-series activities sequence $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$. The computational cost is mainly from computing $\mathbf{Q}_1 = \boldsymbol{\alpha}'^T \widetilde{\mathbf{D}}_{y_1}^T \mathbf{M}_1^{-1}$, $\{\mathbf{Q}_i = \mathbf{M}_{2i-2}^{-1} \widetilde{\mathbf{D}}_{y_i}^T \mathbf{M}_{2i-1}^{-1}\}_{i=2}^T$, and $\mathbf{V} = \mathbf{M}_{2T}^{-1} * \boldsymbol{\beta}'$. Specifically, computing $\mathbf{Q}_1$ requires $O(n^2 l + n^2 l^2)$ computational cost. Computing $\{\mathbf{Q}_i = \mathbf{M}_{2i-2}^{-1} \widetilde{\mathbf{D}}_{y_i}^T \mathbf{M}_{2i-1}^{-1}\}_{i=2}^T$ requires $O((T-1)(n^3 l^2 + n^3 l))$ computational cost. Computing $\mathbf{V}$ requires $O(n^2)$ computational cost. Thus, the total computational cost is $O((T-1)(n^3 l^2 + n^3 l) + n^2 l + n^2 l^2 + n^2)$. Thus, the computational cost of the token generation algorithm is affected by the parameters $\{n, l, T\}$. Detailed experimental results are shown as follows.



(a) n and T (b) l and T

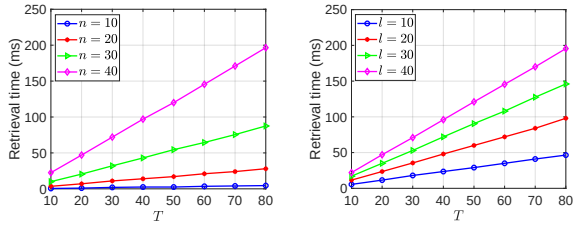
Fig. 3. Computational cost of token generation algorithm

• n and T: In Fig. 3(a), we plot the computational cost of the token generation algorithm varying with n and T. In this experiment, the parameters are set as $l = 40$, n is in the range of $\{10, 20, 30, 40\}$, and T ranges from 10 to 80. From Fig. 3(a), we can see that the computational cost of the token generation algorithm linearly increases with T and cubically increases with n. Overall, the token generation algorithm is efficient. For example, when $n = 40$, and $l = 40$, generating a query token for an 80-dimensional time-series activities sequence takes 9830.5 ms.

• l and T: In Fig. 3(b), we plot the computational cost of the token generation algorithm varying with l and T. In this experiment, the parameters are set as $n = 40$, l is in the range of $\{10, 20, 30, 40\}$, and T ranges from 10 to 80. From Fig. 3(b), we can see that the computational cost of the token generation algorithm linearly increases with T and quadratically increases with l.

C. Computational Cost of Retrieval Algorithm

The retrieval algorithm is used to compute $z = (\prod_{i=1}^T (\mathbf{Q}_i \mathbf{C}_i)) * \mathbf{V}$, and its computational cost is $O((T-1)n^3 l + n^2 l + n^2)$. Thus, the computational cost is mainly affected by the parameters $\{n, l, T\}$, and the detailed experimental results are shown as follows.



(a) n and T (b) l and T

Fig. 4. Computational cost of retrieval algorithm

• n and T: In Fig. 4(a), we plot the computational cost of the retrieval algorithm varying with n and T. In this experiment, the parameters are set as $l = 40$, n is in the range of $\{10, 20, 30, 40\}$, and T ranges from 10 to 80. From Fig. 4(a), we can see that the computational cost of the retrieval algorithm linearly increases with T and cubically increases with n. Overall, the retrieval algorithm is efficient. For example, when $n = 40$, $l = 40$, and $T = 80$, the retrieval algorithm takes 196.5 ms.

• l and T: In Fig. 4(b), we plot the computational cost of the retrieval algorithm varying with l and T. In this experiment, the parameters are set as $n = 40$, l is in the range of $\{10, 20, 30, 40\}$, and T ranges from 10 to 80. From Fig. 4(b), we can see that the computational cost of the retrieval algorithm linearly increases with T and l.

D. Computational Cost of Decryption Algorithm

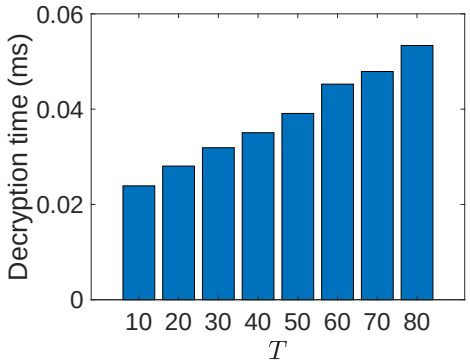


Fig. 5. Computational cost of decryption algorithm

The decryption algorithm is to compute $\Pr(\mathbf{y}|\boldsymbol{\Lambda}) = \frac{1}{\prod_{i=1}^T r_{y_i}} (\frac{z}{r'_\alpha * r'_\beta} - \mathbf{r}_\alpha^T * (\mathbf{R}_{\mathbf{D},y_1} \mathbf{R}_{\mathbf{\Pi}} \prod_{i=2}^T (\mathbf{R}_{\mathbf{D},y_i} \mathbf{R}_{\mathbf{A},y_i})) * \mathbf{r}_\beta)$. Since $\{\mathbf{r}_\alpha, \mathbf{r}_\beta\}$ are w-dimensional and $\{\mathbf{R}_{\mathbf{D},y_i}, \mathbf{R}_{\mathbf{\Pi}}, \mathbf{R}_{\mathbf{A},y_i}\}$ are $w \times w$, the computational cost of the decryption algorithm is about $O(Tw^3 + 2w^2)$. Since w is fixed, the computational cost is only affected by the parameter T. In Fig. 5, we plot the computational cost of the decryption algorithm varying with T, where T ranges from 10 to 80. From this figure, we can see that the experimental result validates that the computational cost of the decryption algorithm linearly increases with T

and also demonstrates that the decryption algorithm is indeed efficient. For example, when $T = 80$, the decryption algorithm only takes 0.05335 ms.

VII. Related Work

Since HMM and LSTM neural network can perform well in time-series activities based healthcare monitoring, we review some existing works on privacy-preserving HMM and privacy-preserving LSTM that are close to our work.

HMM is a good probability model for the discrete time-series activities data [13], [26]–[28] and is an effective method to realize time-series activities based healthcare monitoring for patients. In the HMM-based healthcare monitoring model, the efficiency and security completely depend on that of the forward algorithm. To make the forward algorithm privacy-preserving, many solutions were proposed in the literature. However, they were designed in the scenario of two party computation and are not applicable to the outsourced scenario.

Specifically, Smaragdis et al. [16] employed a public-key homomorphic encryption scheme to design a client-server based two party computation protocol. In this protocol, the client and the server can privately compute the probability of a time-series sequence under an HMM model through multi-round intersections. Since many involved protocols in this algorithm require plaintext information, this algorithm is not applicable to the outsourced scenario. Meanwhile, this algorithm has the issue of accuracy because the proposed algorithm can only deal with integers. To improve the accuracy, Pathak et al. [17], [18] proposed two forward algorithms by using the homomorphic encryption technique and oblivious transfer protocol. By representing small probability values into fixed-point numbers, these algorithms can achieve a reasonable accuracy when the HMM is small. However, same as the scheme in [16], these algorithms are not applicable to the outsourced scenario.

To obtain a reasonable accuracy, Franz et al. [20] proposed a privacy-preserving forward algorithm based on a homomorphic encryption scheme in [29]. However, this scheme is also a two-party computation protocol and is not applicable to the outsourced scenario. To improve both the computational efficiency and accuracy, Polat et al. [19] employed the blinding and permutation techniques to propose a privacy-preserving forward algorithm. This algorithm is highly efficient and without any accuracy loss because all computations were performed over the real domain. However, this algorithm cannot support the outsourced scenario. To outsource a part of forward algorithm computation to the cloud, Ziegeldorf et al. [21] employed garbled circuits, additive secret sharing, and oblivious transfer to design a scheme. Although the scheme can outsource a majority of two parties' computation to the cloud, the party (i.e., healthcare center in our scheme) with the HMM model must be online to assist the cloud server to do the forward algorithm computation. In this case, this scheme cannot support the outsourced scenario in our work.

In [24], we proposed a variant of forward algorithm, in which all computations are reduced to the matrix multiplication. Then, we used the proposed algorithm to construct a privacy-preserving forward algorithm, in which the computational cost of the forward algorithm is outsourced to two non-collusive cloud servers. However, this scheme discloses the plaintext HMM model and time-series data to the cloud servers, so it is not applicable to the outsourced scenario. Besides, some secure floating point primitives [30]–[33] can be employed to achieve privacy-preserving forward algorithm computation. However, same as existing homomorphic encryption scheme based algorithms, such algorithms can not be applied to the outsourced scenario.

LSTM neural network is also an effective healthcare monitoring model for time-series activities data. However, existing privacy-preserving LSTM neural network schemes [14], [15] are not practical because they were constructed in the two-server setting. Specifically, Ma et al. [14] leveraged the secret sharing technique to design a privacy-preserving LSTM neural network scheme. Wang et al. [15] employed the secret sharing technique to construct security activation functions and further proposed a privacy-preserving bidirectional LSTM neural network based on the security activation functions.

Different from the above works, our privacy-preserving healthcare monitoring scheme is designed in the outsourced scenario with a single server. It can preserve the privacy of the classification models and time-series activities data, and it does not incur any accuracy loss.

VIII. Conclusion

In this paper, we have proposed an efficient and privacy-preserving forward algorithm and further used it to construct a privacy-preserving healthcare monitoring scheme. Specifically, we first proposed the concept of mutually orthogonal matrices and introduced an approach to construct a set of mutually orthogonal matrices. Then, we designed an efficient and privacy-preserving forward algorithm, i.e., PPFA, by leveraging a set of mutually orthogonal matrices and the lightweight matrix encryption technique. Finally, we proposed a privacy-preserving healthcare monitoring scheme based on PPFA. In addition, we analyzed the security of our PPFA and healthcare monitoring scheme, and conducted experiments to validate their efficiency. In our future work, we plan to design a more efficient healthcare monitoring scheme in the outsourced scenario while guaranteeing the same security level.

Acknowledgments

This research was supported in part by NSERC Discovery Grants (04009), Natural Science Foundation of Shaanxi Province (2019ZDLGY12-02), ZJNSF (LZ18F020003), and NSFC (U1709217, 61972304).

References

[1] R. Lu, "A new communication-efficient privacy-preserving range query scheme in fog-enhanced iot," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2497–2505, 2019.

[2] X. Zhang, R. Lu, J. Shao, H. Zhu, and A. A. Ghorbani, "Secure and efficient probabilistic skyline computation for worker selection in MCS," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11 524–11 535, 2020.

[3] M. E. S. Saeed, Q. Liu, G. Y. Tian, B. Gao, and F. Li, "Remote authentication schemes for wireless body area networks based on the internet of things," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4926–4944, 2018.

[4] J. Hua, H. Zhu, F. Wang, X. Liu, R. Lu, H. Li, and Y. Zhang, "CINEMA: efficient and privacy-preserving online medical primary diagnosis with skyline query," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1450–1461, 2019.

[5] Y. Zheng, R. Lu, and J. Shao, "Achieving efficient and privacy-preserving k-nn query for outsourced ehealthcare data," *J. Medical Syst.*, vol. 43, no. 5, pp. 123:1–123:13, 2019.

[6] S. Zhang, S. Ray, R. Lu, Y. Zheng, and J. Shao, "Preserving location privacy for outsourced most-frequent item query in mobile crowdsensing," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[7] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Efficient and privacy-preserving similarity range query over encrypted time series data," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2021.

[8] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Achieving efficient and privacy-preserving set containment search over encrypted data," *IEEE Transactions on Services Computing*, no. 01, pp. 1–1, 2021.

[9] L. Lyu, X. He, Y. W. Law, and M. Palaniswami, "Privacy-preserving collaborative deep learning with application to human activity recognition," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, 2017, pp. 1219–1228.

[10] Y. Shen, C. Luo, D. Yin, H. Wen, D. Rus, and W. Hu, "Privacy-preserving sparse representation classification in cloud-enabled mobile applications," *Comput. Networks*, vol. 133, pp. 59–72, 2018.

[11] F. Wang, H. Zhu, R. Lu, Y. Zheng, and H. Li, "Achieve efficient and privacy-preserving disease risk assessment over multi-outsourced vertical datasets," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2020.

[12] S.-C. Poh, Y.-F. Tan, X. Guo, S.-N. Cheong, C.-P. Ooi, and W.-H. Tan, "Lstm and hmm comparison for home activity anomaly detection," in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. IEEE, 2019, pp. 1564–1568.

[13] A. R. M. Forkan, I. Khalil, Z. Tari, S. Foufou, and A. Bouras, "A context-aware approach for long-term behavioural change detection and abnormality prediction in ambient assisted living," *Pattern Recognit.*, vol. 48, no. 3, pp. 628–641, 2015.

[14] Z. Ma, Y. Liu, X. Liu, J. Ma, and F. Li, "Privacy-preserving outsourced speech recognition for smart iot devices," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8406–8420, 2019.

[15] Q. Wang, C. Feng, Y. Xu, H. Zhong, and V. S. Sheng, "A novel privacy-preserving speech recognition framework using bidirectional LSTM," *J. Cloud Comput.*, vol. 9, p. 36, 2020.

[16] P. Smaragdis and M. V. S. Shashanka, "A framework for secure speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 15, no. 4, pp. 1404–1413, 2007.

[17] M. A. Pathak, S. Rane, W. Sun, and B. Raj, "Privacy preserving probabilistic inference with hidden markov models," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011, May 22-27, 2011, Prague Congress Center, Prague, Czech Republic*, 2011, pp. 5868–5871.

[18] M. A. Pathak, B. Raj, S. Rane, and P. Smaragdis, "Privacy-preserving speech processing: Cryptographic and string-matching frameworks show promise," *IEEE Signal Process. Mag.*, vol. 30, no. 2, pp. 62–74, 2013.

[19] H. Polat, W. Du, S. Renckes, and Y. Oysal, "Private predictions on hidden markov models," *Artif. Intell. Rev.*, vol. 34, no. 1, pp. 53–72, 2010.

[20] M. Franz, B. Deiseroth, K. Hamacher, S. Jha, S. Katzenbeisser, and H. Schröder, "Towards secure bioinformatics services (short paper)," in *Financial Cryptography and Data Security - 15th International Conference, FC 2011, Gros Islet, St. Lucia, February 28 - March 4, 2011, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 7035, 2011, pp. 276–283.

[21] J. H. Ziegeldorf, J. Metzke, J. Rüth, M. Henze, and K. Wehrle, "Privacy-preserving HMM forward computation," in *Proceedings of the Seventh ACM Conference on Data and Application Security and Privacy, CODASPY 2017, Scottsdale, AZ, USA, March 22-24, 2017*, 2017, pp. 83–94.

[22] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, vol. 2012, p. 144, 2012.

[23] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[24] Y. Zheng, R. Lu, and M. S. I. Mamun, "Privacy-preserving computation offloading for time-series activities classification in ehealthcare," in *2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, June 7-11, 2020*, 2020, pp. 1–6.

[25] S. Lai, S. Patranabis, A. Sakzad, J. K. Liu, D. Mukhopadhyay, R. Steinfeld, S. Sun, D. Liu, and C. Zuo, "Result pattern hiding searchable encryption for conjunctive queries," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, 2018, pp. 745–762.

[26] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 33:1–33:33, 2014.

[27] M. Abreu, M. Barandas, R. Leonardo, and H. Gamboa, "Detailed human activity recognition based on multiple HMM," in *Proceedings of the 12th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2019) - Volume 4: BIOSIGNALS, Prague, Czech Republic, February 22-24, 2019*, 2019, pp. 171–178.

[28] L. Lu, C. Qing-Ling, and Z. Yi-Ju, "Activity recognition in smart homes," *Multimedia Tools and Applications*, vol. 76, no. 22, pp. 24 203–24 220, 2017.

[29] M. Franz, B. Deiseroth, K. Hamacher, S. Jha, S. Katzenbeisser, and H. Schröder, "Secure computations on non-integer values," in *2010 IEEE International Workshop on Information Forensics and Security, WIFS 2010, Seattle, WA, USA, December 12-15, 2010*, 2010, pp. 1–6.

[30] M. Aliasgari and M. Blanton, "Secure computation of hidden markov models," in *SECRYPT 2013 - Proceedings of the 10th International Conference on Security and Cryptography, Reykjavík, Iceland, 29-31 July, 2013*, P. Samarati, Ed., 2013, pp. 242–253.

[31] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele, "Secure computation on floating point numbers," in *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*, 2013.

[32] L. Kamm and J. Willemson, "Secure floating point arithmetic and private satellite collision analysis," *Int. J. Inf. Sec.*, vol. 14, no. 6, pp. 531–548, 2015.

[33] D. Demmler, G. Dessouky, F. Koushanfar, A. Sadeghi, T. Schneider, and S. Zeitouni, "Automated synthesis of optimized circuits for secure computation," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, 2015, pp. 1504–1517.

Yandong Zheng received her M.S. degree from the Department of Computer Science, Beihang University, China, in 2017 and she is currently pursuing her Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada. Her research interest includes cloud computing security, big data privacy and applied privacy.

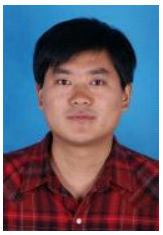
Rongxing Lu (S'09-M'11-SM'15-F'21) is an associate professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor General's Gold Medal", when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. Also, Dr. Lu received his first PhD degree at Shanghai Jiao Tong University, China, in 2006. Dr. Lu is an IEEE Fellow. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise (with H-index 75 from Google Scholar as of May 2021), and was the recipient of 9 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Songnian Zhang received his M.S. degree from Xidian University, China, in 2016 and he is currently pursuing his Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada. His research interest includes cloud computing security, big data query and query privacy.

Yunguo Guan is a PhD student of the Faculty of Computer Science, University of New Brunswick, Canada. His research interests include applied cryptography and game theory.

Jun Shao received the Ph.D. degree from the Department of Computer and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Post-Doctoral Fellow with the School of Information Sciences and Technology, Pennsylvania State University, Pennsylvania, PA, USA, from 2008 to 2010. He is currently a Professor with the School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China. His current research interests include network security and applied cryptography.



Hui Zhu (M'13-SM'19) received the B.Sc. degree from Xidian University, Xian, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, China, in 2005, and the Ph.D. degree from Xidian University, in 2009.

He was a Research Fellow with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, in 2013. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His current research interests include applied cryptography, data security, and privacy.