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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

Privacy Preserving Tensor-Based Multiple Clusterings on Cloud for Industrial IoT

Yaliang Zhao, Laurence T. Yang, *Senior Member*, *IEEE*, and Jiayu Sun

***Abstract*—Aiming at discovering hidden different data structures in big data from different perspectives, a tensor-based multiple clustering method has been developed recently, which can be widely used in Industrial IoT to improve production and service quality. However, due to the high computational cost and huge volume of data, outsourcing computing to relatively inexpensive cloud servers can greatly save local costs, but there is a high risk of revealing user privacy. To address the problem above, a privacy preserving tensor-based multiple clustering method on the secure hybrid cloud is proposed. The proposed scheme utilizes homomorphic cryptosystem to encrypt object tensors, then employs cloud servers to completely implement multiple clustering calculation over encrypted object tensors. Furthermore, a series of related security sub-protocols are proposed to support privacy preserving tensor-based multiple clusterings. In the proposed scheme, only encryption and removing perturbation are performed on the client, which is very lightweight for users. Experimental results show the proposed scheme is accurate and efficient when clustering objects to different groups, while no private or additional information is leaked. Moreover, when employing more cloud nodes, the scheme has high scalability, thus it is very suitable for clustering Industrial IoT big data.**

***Index Terms*—Industrial IoT, privacy preserving, multiple clusterings, cloud computing, homomorphic cryptosystem.**

I. Introduction

WITH the advent of "Industry 4.0" era, Cyber-Physical Systems (CPS), computing systems of Industrial Internet-of-Things (IoT), have been replaced by more advanced Cyber-Physical-Social Systems (CPSS), integrating cyber space, physical space and social space [1]. In CPSS, a variety of complex and massive data is collected from industrial control systems, automated production systems to smart products and user experiences, which contains abundant knowledge and tremendous value [2]. In-depth mining and exploiting these knowledge will not only make industrial automation more efficient and effective, but also enable a new level of product personalization at a lowest cost. As an emergence research field of data mining, multiple clusterings, has been extensively studied in recent years [3]. Compared to the traditional clustering, which only focuses on discovering a single grouping of objects, multiple clusterings can generate multiple different clustering results at the same time from different perspectives of the data, revealing different structures hidden in the data, and satisfying multiple analytic tasks of CPSS.

However, the existing multiple clustering researches focus on low-dimensional and single-domain data, which is difficult to apply to large-scale heterogeneous data scenarios in the real-world. Specially, a large number of perceptual devices, network communications as well as social relations, generate large-scale heterogeneous data from multiple dimensions, including text, picture, audio and video. Diverse modalities and characteristics appear in big data of different sources. Usually, applications on big data have to face enormous numbers of high dimensional records, causing high complexities of time and space. To address the above problems, a tensor-based multiple clustering (TMC) approach was proposed in the previous work [4]. But with the rapid growth of data size and volume, performing TMC in real-time requires higher computational capabilities and more storage, while cloud computing can provide a good solution with the powerful computing capabilities and huge storage resources.

Nowadays, more and more enterprises are willing to outsource their data to cloud, such as industrial production indicators, equipment operating status, meter measurement, etc., so as to save local computing and management costs. The cloud acquiring all of the computing and storage resources can process and analyze data in real-time, as well as manage them automatically through softwares. Moreover, since the cloud integrates a large number of correlative data and advanced data mining techniques, it can provide more accurate information and more intelligent services for enterprises. Therefore, it

Manuscript received December 10, 2017; revised April 04, 2018; accepted August 19, 2018. This work was supported in part by National Natural Science Foundation of China under Grant (No. 61802112), and Shenzhen Fundamental Research Program under Grant (No. JCYJ20170307172200714) (*Corresponding author: Laurence T. Yang.*)

Y. Zhao is with School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China, and School of Computer and Information Engineering, Henan University, Kaifeng 475004, China, and Shenzhen Huazhong University of Science and Technology Research Institute, Shenzhen 518057, China (e-mail: zylhenu@139.com).

L. T. Yang is with School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China, and Department of Computer Science, St. Francis Xavier University, Antigonish, Canada, and Shenzhen Huazhong University of Science and Technology Research Institute, Shenzhen 518057, China (e-mail: ltyang@ieee.org).

J. Sun is with School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China, and Shenzhen Huazhong University of Science and Technology Research Institute, Shenzhen 518057, China (e-mail: sunjy@hust.edu.cn).

Copyright (c) 2009 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

would be more efficient to carry out the TMC algorithm with cloud computing when facing CPSS big data. However, directly outsourcing data to cloud will reveal the user's sensitive or private information as the cloud service providers may be curious or malicious [5]. Once the information is leaked, it may threaten the production safety of enterprises and even people's life. For instance, if electricity consumption data of enterprises are leaked, these enterprises would be under the risks of security breach, such as the production activity detection. Hence, this study focuses on a privacy preserving TMC approach on cloud, paving the way for its widespread use in big data analysis and mining of Industrial IoT.

In recent years, cloud computing security has received wide attention from academia to industry, emerged a large number of privacy preserving approaches. One of the effective ways is to encrypt data before outsourcing them to cloud, then all calculations are performed over encrypted data on cloud, until the encrypted final result is returned to the client for decryption. In this process, the cloud does not learn anything about sensitive data and intermediate results, guaranteeing the security of user's privacy. However, implementing privacy preserving TMC over encrypted data brings some issues and challenges. This study mainly takes into account four challenges: (1) To protect user's private and all intermediate results, various security operations related to TMC are indispensable, including addition, multiplication, division, comparison, exponentiation, and so on. (2) To ensure the validity of clustering results, the distance of object tensors computed on ciphertexts are supposed to be of the same accuracy as plaintexts as possible, requiring an efficient homomorphic cryptosystem and the handling of floating point numbers. (3) The client computing cost should be reduced as much as possible during the clustering process. (4) To improve efficiency and scalability, it needs the rational use of cloud resources to meet the high cost of computing and the growing amount of data.

To address the above challenges, this paper proposes a privacy preserving TMC (PPTMC) method using homomorphic cryptosystem on a hybrid cloud model. The PPTMC method is capable of effectively clustering CPSS big data by outsourcing data and computations to cloud without disclosing any additional information. Moreover, it has high scalability when more cloud nodes are employed to its computation. To meet the requirement of TMC for all kinds of mathematical operations and fast encryption, PPTMC utilizes the formal Paillier cryptosystem. However, Paillier cryptosystem does not support exponentiation operation over encrypted data, thus the work presents a secure exponentiation operation based on a discriminant method which is suitable for PPTMC. The major contributions of this paper can be summarized as follows:

1) To support the exponentiation operation required by the calculation of distance, the work designs a secure exponentiation (SE) protocol based on a discriminant method. Then secure attribute weight ranking (SAWR) protocol and secure selective weighted tensor distance (SSWTD) protocol are developed as sub-routines of PPTMC. In addition, to implement floating point calculation, magnification and minification are utilized in the above protocols.
2) A novel complete PPTMC method over encrypted data is present based on the above security protocols. In the proposed scheme, all expensive computing tasks are offloaded on cloud without revealing or deducing any private information, not merely enhancing its efficiency and scalability, but also protecting sensitive information. Specially, once the encrypted data are uploaded to the cloud, the client no longer participates in the multiple clustering process, significantly reducing the user's computational burden.

TABLE I
TABLE OF COMMON NOTATIONS

Symbol	Definition
$\mathcal{X}$	original object tensor
$[\![\mathcal{X}]\!]$	encrypted object tensor
$[\![\mathcal{T}_a]\!]$	encrypted association tensor
$[\![\mathcal{T}_{tr}]\!]$	encrypted transition tensor
$[\![\mathcal{T}_w]\!]$	encrypted weight tensor
$[\![\mathcal{T}_{mv}]\!]$	encrypted multiview tensor
M	original matrix M
$[\![M]\!]$	encrypted matrix
v	original vector v
$[\![\mathrm{v}]\!]$	encrypted vector
a	original integer
$[\![a]\!]$	encrypted integer
σ	regularization parameter
α	probability parameter
λ	magnification factor

The rest of the paper is organized as follows. Section II introduces preliminaries used in the work. Related basic security protocols are provided in Section III, including the proposed SE, SAWR and SSWTD protocols. Section IV describes the proposed PPTMC approach. Evaluations and experiments are given in Section V. Section VI reviews the related work. Finally, the whole paper is concluded in Section VII.

II. PRELIMINARIES

This section reviews preliminaries used in the proposed scheme including TMC clustering algorithm, Paillier cryptosystem and security model. Some common notations used in the proposed scheme are shown in Table I.

A. TMC clustering algorithm

In the previous work [4], a tensor-based multiple clustering method is proposed. The goal of TMC is to reveal latent different data structures in big data from different perspectives. Specially, based on the integration of multi-source information, it allows the user to select different feature combinations according to different applications, resulting in different clustering results, so as to meet the requirement of big data multiple analysis tasks. The main idea of TMC is as follows: (1) Data objects tensorization transforms heterogeneous data to a unified object tensor model. (2) Weight tensor construction, that refers to, employing the multilinear attribute weight ranking algorithm to obtain the weight tensor, which can effectively improve the quality of clustering. (3) Selective weighted tensor distance (SWTD), that is, the weight factors and selection coefficients are introduced in tensor distance, indicating the importance of each attribute combination, as well as providing flexible selection of desired different attribute combinations

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

upon applications. (4) Any clustering algorithm with distances as input can be chosen to cluster object and produce multiple clustering results. Here, the proposed method chooses the relatively efficient clustering by fast search and find of density peaks (CFS) published in Science magazine [6].

Specially, the definition of SWTD is described as follows. Given an object tensor $\mathcal{X} \in \Re^{I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}}$, x represents the vector unfolding of $\mathcal{X}$. The SWTD of $\mathcal{X}$ and $\mathcal{Y}$ is calculated as

$$d_{SWTD} = \sqrt{\sum_{l,m=1}^{I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}} g_{lm} w_l (x_l - y_l) w_m (x_m - y_m)} = \sqrt{(\mathrm{w} * (\mathrm{x} - \mathrm{y}))^{\mathrm{T}} \boldsymbol{G} (\mathrm{w} * (\mathrm{x} - \mathrm{y}))}, \tag{1}$$

where w_l, w_m are weight factors of the location $\mathcal{X}_{i_1 i_2 \ldots i_k}$ (corresponding to x_l) and $\mathcal{X}_{i_1' i_2' \ldots i_k'}$ (corresponding to x_m), respectively.

Here, the metric matrix G is defined as

$$g_{lm} = \frac{1}{2\pi\sigma^2} \exp\left\{ \frac{-\|p_l - p_m\|_2^2}{2\sigma^2} \right\}, \tag{2}$$

where the location distance $\|p_l - p_m\|_2$ is given by

$$\|p_l - p_m\|_2 = \sqrt{v_1 (i_1 - i_1')^2 + v_2 (i_2 - i_2')^2 + \cdots + v_k (i_k - i_k')^2}, \tag{3}$$

where $v_j (1 \leq j \leq k)$ is the element of feature space combination vector $\mathrm{v} \in \{0,1\}^k$. When selecting the *j*-th feature space, the value of v_j is 1, otherwise it is 0.

B. Paillier cryptosystem

The Paillier cryptosystem [7] is proposed as an additive homomorphic encryption technique with probabilistic asymmetry and semantic security. Given any ciphertexts $[\![a]\!], [\![b]\!]$ and constant $m \in Z_N$, it has two homomorphic properties: (1) homomorphic addition $[\![a+b]\!] = [\![a]\!] * [\![b]\!] \bmod N^2$; (2) homomorphic multiplication $[\![m * a]\!] = [\![a]\!]^m \bmod N^2$.

C. Security Model

In secure multi-party computation, a semi-honest model [8] assumes that any participant party faithfully performs the protocols with its correct inputs, but during the execution of the protocols, it always tries to deduce the confidential information about other parties. Here, a hybrid cloud [9] is a secure two-party computation model, including a public cloud and a private cloud. The hybrid cloud has both high security and high computing power of the two parties. It is assumed that usually public clouds are provided by renowned IT enterprises like Amazon and Apple, and private clouds are provided by credible departments or organizations. Taking into account reputation, commercial interests as well as legal risks, they will not collude with each other and maliciously leak user's privacy. Hence, the model is practical.

In terms of the relationship of the two clouds, every security protocol on this model needs the two clouds to work together. The main idea of these protocols is the public cloud perturbs the ciphertexts by homomorphic addition and sends them to the private cloud; then the private cloud decrypts the perturbed ciphertexts, does some specific operation on them, encrypts the results again and sends them back to the public cloud; finally, the public cloud removes the perturbation from the encrypted results by homomorphic addition. The private cloud can not get the plaintexts but only the completely perturbed plaintexts. The public cloud can not get the plaintexts but only the ciphertexts which it is unable to decrypt.

TABLE II
TABLE OF EXISTING SECURITY PROTOCOL

Protocol	Definition
Secure Multiplication	$SM([\![a]\!], [\![b]\!]) \rightarrow [\![a \cdot b]\!]$
Secure Comparison	$SC([\![a]\!], [\![b]\!]) \rightarrow [\![a \geq b]\!]$
Secure Division 1	$SD1([\![a]\!], b) \rightarrow [\![qu_1]\!]$
Secure Division 2	$SD2([\![a]\!], [\![b]\!]) \rightarrow [\![qu_2]\!]$
Secure Higher-order CFS	$SHOCFS([\![\mathcal{X}_1]\!], [\![\mathcal{X}_2]\!], \ldots, [\![\mathcal{X}_n]\!]) \rightarrow [\![cl]\!]$

III. BASIC SECURITY PRIMITIVES

First, this section introduces existing security protocols involved in PPTMC. Second, this section proposes a series of generic protocols as sub-routines of PPTMC. Here, all protocols obey the rules of the above security model, where C_1 and C_2 represent the public cloud and the private cloud, respectively. Initially, the private cloud generates a public key *pk* and a private key *sk* by using the Paillier cryptosystem, and publishes *pk* to the client and the public cloud.

A. Existing Security Protocol

The existing security protocols involved in PPTMC are listed in Table II, including secure multiplication (SM) [10], secure comparison (SC) [11], secure division 1 (SD1) [12], secure division 2 (SD2) [13], and secure higher-order CFS (SHOCFS) [14].

B. The Proposed Protocols

As sub-routines, some universal protocols are proposed in order to achieve secure tensor-based multiple cluterings. Moreover, to implement floating point calculation, magnification and minification are utilized in the protocols.

1) ***SE Protocol****:* As there is no homomorphic exponentiation operations in the Paillier cryptosystem, it can not directly support the secure computation of the SWTD between objects. At present, the Taylor series expansion is usually used to convert exponentiation operation into a polynomial function, involving only addition and multiplication operations [15]. However, it has two main limitations to the proposed method. First, the low efficiency of Taylor expansion can not to meet the requirement of secure computation. Second, it needs to preserve the floating point precision by amplification, while the Paillier cryptosystem is limited by the size of the plaintexts, so that it is difficult to achieve an acceptable range of precision.

Observing Eqs. (2) and (3), the numerators of exponent are 0, -1, -2, -3, ... according to the size of the object tensor, and there is a fact that smaller exponent corresponds to smaller exponential result. So the potential exponents in SWTD are

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

a series of discrete values, and part of the results of them are so close to 0 that can be replaced by 0. Aiming at these characteristics, this work presents an exponentiation operation method based on discriminant that is summarized in Algorithm 1, the details of which are described as follows.

In line 2, depending on the size of tensor object, C_1 selects the top *k* maxima from all possible values of *x*, but discards the remaining values of which the exponential results are so small that it is difficult to preserve their precision by magnification. In general, selecting the top three maxima can achieve a high enough accuracy.

From line 3 to 5, C_1 computes exponential results of the top *k* maxima over plaintexts and encrypts them. At the same time, C_1 encrypts the top *k* maxima as criteria of discriminant.

From line 7 to 10, the algorithm iteratively computes the discriminated results. Utilizing the SC protocol, C_1 with the encrypted input $[\![\lambda_d x]\!]$ and the encrypted criteria, and C_2 with *sk* securely compare them one by one in line 8. In line 9, applying the SM protocol, both C_1 and C_2 securely compute the discriminated results.

Finally, C_1 locally uses homomorphic addition to securely compute the exponential result in line 11.

Algorithm 1 SE Protocol

Input: C_1 has $[\![\lambda_d x]\!]$, C_2 has *sk*.
Output: Encrypted exponential result $[\![\lambda_e e^x]\!]$ only to C_1.

1: C_1:
2: Select the top *k* maxima $m_1, m_2, \ldots, m_k$ from all possible values of x.
3: **for** *i*=1 to *k* **do**
4: Compute $[\![\lambda_e e^{m_i}]\!]$, $[\![\lambda_d m_i]\!]$.
5: **end for**
6: C_1, C_2:
7: **for** *i*=1 to *k* **do**
8: $[\![c_i]\!] \leftarrow SC([\![\lambda_d x]\!], [\![\lambda_d m_i]\!])$.
9: $[\![\lambda_e s_i]\!] \leftarrow SM([\![c_i]\!], [\![\lambda_e e^{m_i}]\!])$.
10: **end for**
11: C_1: $[\![\lambda_e e^x]\!] \leftarrow \prod_{i=1}^{k} [\![\lambda_e s_i]\!]$.

Theorem 1: The proposed SE protocol is secure on the semi-honest model.

Proof: The plaintexts appearing in line 2 are potentially possible values that can be made public. Here, except for the size of tensor, they do not represent any privacy information about the data. Since the size is not actually a category of privacy, line 2 is secure. Line 4, 11 use the Paillier cryptosystem which is semantically secure. Moreover, Line 2, 4, 11 do not interact between C_1 and C_2. Line 8 and 9 apply the SC protocol and SM protocol, respectively, which both have formal proofs that ensure the security of them under the semi-honest model. In conclusion, the proposed SE protocol is secure.

Note that, the SE protocol is suitable for exponentation operations with discrete exponent inputs, including the one that required by SWTD. Hence, the proposed SE protocol has a certain versatility.

*2) **SAWR Protocol**:* Based on the multilinear attribute weight ranking algorithm proposed in the previous work [4], this scheme proposes the SAWR protocol depicted in Algorithm 2. C_1 with the encrypted *K*th-order transition tensor $\left[\!\left[\lambda_w \mathcal{T}_{tr}^{(l)}\right]\!\right]$, and C_2 with *sk* securely compute the encrypted attribute weight ranking vectors $[\![\lambda_w \mathrm{w}_l]\!]$, which are only known to C_1, for $1 \leq l \leq k$.

At the beginning of the SAWR protocol, C_1 encrypts the probability parameter α. In the following steps, the attribute ranking vector for each feature space is securely computed.

In order to ensure that the Z-eigenvector converges linearly to the unique solution for any initial vector, C_1 randomly initializes and encrypts vector w_0 and u whose sum equals 1, respectively.

From line 7 to 23, utilizing the secure *i*-mode product definition proposed in [13] and the SM protocol, C_1 and C_2 jointly and iteratively compute the ciphertexts $\left[\!\left[\lambda_w (w_j)_t\right]\!\right]$, but they learn nothing about user's privacy information during the whole process. Specially, repeatedly performing multiplications can cause quick accumulation of the magnification factors, resulting in overflow of the ciphertexts, hence the SD1 protocol needs to be used to remove the extra magnification factors.

Finally, considering the extended dimensionality of the transition tensor, C_1 takes the first I_{f_l} elements of $[\![\lambda_w \mathrm{w}_j]\!]$ corresponding to each feature space as encrypted ranking vector $[\![\lambda_w \mathrm{w}_l]\!]$ locally in line 24.

*3) **SSWTD Protocol**:* In the SSWTD protocol, C_1 holds encrypted object tensors $[\![\lambda_o \mathcal{X}]\!]$ and $[\![\lambda_o \mathcal{Y}]\!]$, metric matrix $[\![\lambda_e G]\!]$, weighted tensor $[\![\lambda_w \mathcal{T}_w]\!]$, and C_2 possesses the private key *sk*. Here $\mathcal{X}$ and $\mathcal{Y}$ denote *K*th-order tensors $\Re^{I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}}$, where $I_{f_1}, I_{f_2}, \ldots, I_{f_k}$ represent the dimensions of *K* feature spaces, respectively. The SSWTD protocol focuses on securely calculating the encypted SWTD $[\![\lambda_o d_{SWTD}]\!]$. Notice that the square of the distances among objects is another form of the distances without root operations, thus the former is chosen here. This protocol does not reveal any relevant information about $\mathcal{X}$, $\mathcal{Y}$, *G* or $\mathcal{T}_w$ to C_1 or C_2. The calculation of SWTD follows the Eq. (1).

Algorithm 3 shows the main steps participating in SSWTD. Briefly, from line 1 to 15, C_1 and C_2 federatively compute $[\![\lambda_o d_{lm}]\!]$ using the SM protocol, and adopting the SD1 protocol to prevent overflow after every multiplication, for $1 \leq l, m \leq I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}$. Note that the outputs of the SM and SD1 protocols are known only to C_1. Finally, C_1 computes $[\![\lambda_o d_{SWTD}]\!]$ locally by applying homomorphic properties on $[\![\lambda_o d_{lm}]\!]$ in line 16.

IV. PPTMC Approach

The PPTMC algorithm on cloud is described in this section. In order to securely supply different high-quality clustering services for Industrial IoT, the proposed method aims at implementing the TMC algorithm by collaborating cloud computing power without divulging any confidential information. Along this goal, this work proposes a completely secure protocol, implementing a privacy preserving equivalence for each step of the original TMC algorithm proposed in [4].

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

Algorithm 2 SAWR Protocol

Input: C_1 has encrypted *K*th-order transition tensors $\left[\left[\lambda_w \mathcal{T}_{tr}^{(1)}\right]\right], \left[\left[\lambda_w \mathcal{T}_{tr}^{(2)}\right]\right], \ldots, \left[\left[\lambda_w \mathcal{T}_{tr}^{(k)}\right]\right]$, C_2 has *sk*.
Output: Encrypted attribute weight ranking vectors $[\![\lambda_w \mathrm{w}_1]\!], [\![\lambda_w \mathrm{w}_2]\!], \ldots, [\![\lambda_w \mathrm{w}_k]\!]$ only to C_1.

1: C_1:
2: Set a probability $0 \leq \alpha < 1$ and compute $[\![\lambda_w \alpha]\!]$.
3: **for** *l*=1 to *k* **do**
4: Pick an initial vector $\mathrm{w}_0 \in \Re^m$ and $\|\mathrm{w}_0\|_2 = 1$, compute $[\![\lambda_w \mathrm{w}_0]\!]$.
5: Set a random vector $\mathrm{u} \in \Re^m$ and $\|\mathrm{u}\|_2 = 1$, compute $[\![\lambda_w \mathrm{u}]\!]$.
6: $[\![AR]\!] \leftarrow \left[\left[\lambda_w \mathcal{T}_{tr}^{(l)}\right]\right]$.
7: C_1, C_2:
8: **for** *j*=1 to *c* **do**
9: **for** *i*=1 to *l*-1 **do**
10: $[\![AR]\!] \leftarrow \left[\left[AR \times_i (\lambda_w \mathrm{w}_{j-1})\right]\right]$.
11: $[\![AR]\!] \leftarrow SD1([\![AR]\!], \lambda_w)$.
12: **end for**
13: **for** *i*=*l*+1 to *k* **do**
14: $[\![AR]\!] \leftarrow \left[\left[AR \times_i (\lambda_w \mathrm{w}_{j-1})\right]\right]$.
15: $[\![AR]\!] \leftarrow SD1([\![AR]\!], \lambda_w)$.
16: **end for**
17: $[\![\lambda_w \mathrm{w}_j]\!] \leftarrow [\![AR]\!]$.
18: **for** *t*=1 to *m* **do**
19: $\left[\left[{\lambda_w}^2 (w_j)_t\right]\right] \leftarrow SM([\![\lambda_w \alpha]\!], [[\lambda_w (w_j)_t]])$.
20: $\left[\left[{\lambda_w}^2 (w_j)_t\right]\right] \leftarrow \left[\left[{\lambda_w}^2 (w_j)_t\right]\right] * SM([\![\lambda_w]\!] * [\![\lambda_w \alpha]\!]^{N-1}, [\![\lambda_w u_t]\!])$.
21: $\left[\left[\lambda_w (w_j)_t\right]\right] \leftarrow SD1(\left[\left[{\lambda_w}^2 (w_j)_t\right]\right], \lambda_w)$.
22: **end for**
23: **end for**
24: C_1: Take the first I_{f_l} elements of $[\![\lambda_w \mathrm{w}_j]\!]$ as the encrypted ranking vector $[\![\lambda_w \mathrm{w}_l]\!]$.
25: **end for**

Algorithm 3 SSWTD Protocol

Input: C_1 has encrypted object tensors $[\![\lambda_o \mathcal{X}]\!], [\![\lambda_o \mathcal{Y}]\!]$, metric matrix $[\![\lambda_e G]\!]$, weighted tensor $[\![\lambda_w \mathcal{T}_w]\!]$, C_2 has *sk*.
Output: Encrypted selective weighted tensor distance $[\![\lambda_o d_{SWTD}]\!]$ only to C_1.

1: C_1, C_2:
2: **for** *l*=1 to $I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}$ **do**
3: **for** *m*=1 to $I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}$ **do**
4: $[\![\lambda_e \lambda_w d_{lm}]\!] \leftarrow SM([\![\lambda_e g_{lm}]\!], [\![\lambda_w w_l]\!])$.
5: $[\![\lambda_e d_{lm}]\!] \leftarrow SD1([\![d_{lm}]\!], \lambda_w)$.
6: $[\![\lambda_o (x_l - y_l)]\!] \leftarrow [\![\lambda_o x_l]\!] * [\![\lambda_o y_l]\!]^{N-1}$.
7: $[\![\lambda_e \lambda_o d_{lm}]\!] \leftarrow SM([\![\lambda_e d_{lm}]\!], [\![\lambda_o (x_l - y_l)]\!])$.
8: $[\![\lambda_o d_{lm}]\!] \leftarrow SD1([\![\lambda_e \lambda_o d_{lm}]\!], \lambda_e)$.
9: $[\![\lambda_o \lambda_w d_{lm}]\!] \leftarrow SM([\![\lambda_o d_{lm}]\!], [\![\lambda_w w_m]\!])$.
10: $[\![\lambda_o d_{lm}]\!] \leftarrow SD1([\![\lambda_o \lambda_w d_{lm}]\!], \lambda_w)$.
11: $[\![\lambda_o (x_m - y_m)]\!] \leftarrow [\![\lambda_o x_m]\!] * [\![\lambda_o y_m]\!]^{N-1}$.
12: $\left[\left[{\lambda_o}^2 d_{lm}\right]\right] \leftarrow SM([\![\lambda_o d_{lm}]\!], [\![\lambda_o (x_m - y_m)]\!])$.
13: $[\![\lambda_o d_{lm}]\!] \leftarrow SD1(\left[\left[{\lambda_o}^2 d_{lm}\right]\right], \lambda_o)$.
14: **end for**
15: **end for**
16: C_1: $[\![\lambda_o d_{SWTD}]\!] \leftarrow \prod_{l,m=1}^{I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}} [\![\lambda_o d_{lm}]\!]$.

Initially, the client sends the encrypted object tensors and the feature space combination vectors to C_1. Upon receiving, C_1 with private inputs ($[\![\lambda_o \mathcal{X}_i]\!]$, $[\![\mathrm{v}_j]\!]$), for $1 \leq i \leq n, 1 \leq j \leq b$, and C_2 with the private key *sk* are jointly involved in the PPTMC protocol. The outputs are the encrypted multiple clustering results $[\![cl_j]\!]$, which are only known to C_1, for $1 \leq j \leq b$.

The main steps involved in the proposed PPTMC protocol are discribed in Algorithm 4. In line 1, C_1 locally computes the encrypted association tensor $[\![\lambda_o \mathcal{T}_a]\!]$ by using homomorphic addition on all encrypted object tensors.

From line 3 to 6, C_1 with the help of C_2 securely computes the encrypted transition tensors $\left[\left[\lambda_o \mathcal{T}_{tr}^{(l)}\right]\right]$ with the SD2 protocol, which are known only to C_1, for $1 \leq l \leq k$.

After that, by utilizing the SAWR protocol in line 7, C_1 with the input $\left[\left[\lambda_w \mathcal{T}_{tr}^{(l)}\right]\right]$ and C_2 collaboratively compute the encrypted attribute weight ranking vectors $[\![\lambda_w \mathrm{w}_l]\!]$, for $1 \leq l \leq k$. The outputs of this step are known only to C_1.

From line 8 to 11, by using the SM protocol in an iterative manner, C_1 with the input $\left[\left[\lambda_w \mathcal{T}_{tr}^{(l)}\right]\right]$ and C_2 federatively compute the encrypted weighted tensor based on Eq. (4) over plaintexts in [4]. Same as the SAWR protocol, the SD1 protocol is utilized to avoid overflow after each multiplication. The output is $[\![\lambda_w \mathcal{T}_w]\!]$, which is known only to C_1.

$$\mathcal{T}_w = \mathrm{w}_1 \circ \mathrm{w}_2 \circ \cdots \circ \mathrm{w}_k \tag{4}$$

From line 13 to 20, by applying the homomorphic properties and SE protocol, C_1 with the help of C_2 securely computes the encrypted metric matrix $[\![\lambda_e G_q]\!]$ based on Eqs. (2), (3), for $1 \leq q \leq b$. Moreover, the SD1 protocol is used to prevent overflow after each multiplication.

From line 21 to 25, adopting the SSWTD protocol, C_1 with the encrypted input object tensors $[\![\lambda_o \mathcal{X}_j]\!], [\![\lambda_o \mathcal{X}_h]\!]$, metric matrix $[\![\lambda_e G_q]\!]$ and weighted tensor $[\![\lambda_w \mathcal{T}_w]\!]$, and C_2 jointly compute the encrypted SWTD $\left[\left[\lambda_o (S_V{}^{(q)})_{j,h}\right]\right]$, for $1 \leq j \leq n, j+1 \leq h \leq n, 1 \leq q \leq b$.

In line 27, C_1 locally builds the encrypted multiview tensor $[\![\mathcal{T}_{mv}]\!]$ by stacking the encrypted view matrices $\left[\left[S_V^{(1)}\right]\right]$, $\left[\left[S_V^{(2)}\right]\right], \ldots, \left[\left[S_V^{(b)}\right]\right]$.

In line 28, by using the encrypted multiview tensor as the input of the SCFS that is a part of the SHOCFS protocol, C_1 and C_2 cooperatively compute multiple clusterings $[\![cl_1]\!], [\![cl_2]\!], \ldots, [\![cl_b]\!]$, which are only known to C_1.

In the end, by adding a random number *r* to the final encrypted clustering results $[\![cl_1]\!], [\![cl_2]\!], \ldots, [\![cl_b]\!]$, C_1 locally generates the perturbed ciphertext results $[\![cl_1 + r]\!], [\![cl_2 + r]\!], \ldots, [\![cl_b + r]\!]$, then sends them to C_2 and *r* to the client, respectively. C_2 with the private key *sk* decrypts the perturbed ciphertext results to get $cl_1 + r, cl_2 + r, \ldots, cl_b + r$ and then sends them to the client. As C_2 only sees the perturbed results rather than the final plaintext results, no privacy information is exposed to C_2.

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

Upon receiving, the client computes the plaintext multiple clustering results $cl_1, cl_2, \ldots, cl_b$ by reducing *r*, which can provide different clustering services for Industrial IoT, such as device classification, material recognition, electricity prediction, etc.

V. EVALUATION AND EXPERIMENTS

In this section, the performance of PPTMC is evaluated. First of all, the security analysis of PPTMC is proved. Then, the complexity of PPTMC is theoretical evaluated about its computation and communication costs. Next, the datasets and evaluation metrics are introduced. In the end, simulated experimental results as well as corresponding analyses are provided. All experiments were implemented on a simulated cloud platform being consisted of the laboratory computers with Simgrid tool, and each PC is with 3.20GHz Intel Core i5 3470 CPU (four cores) and 16-GB RAM.

A. Security Analysis

This section provides a proof of security assurance of PPTMC under the semi-honest model. Note that users are not involved in any computation of PPTMC after outsourcing encrypted object tensors to the cloud. The parties here refer to the cloud C_1 and C_2, who follow the protocol correctly, but try to get as much extra information as they can. Due to the formal Paillier cryptosystem guarantee, C_1 only obtains the ciphertexts of intermediate results and final results. Meanwhile, since C_2 has the private key *sk*, the protocol allows it to decrypt intermediate results, but it only gets the completely perturbed plaintexts. Besides, in each step computation, the proposed protocols utilize homomorphic properties or some basic subprotocols of which the security have been given the formal proofs, including SM, SC, SD1, etc., the proposed PPTMC method is completely secure according to the composition theorem [8]. Therefore, during the execution of the whole protocol, no user data is leaked to C_1 and C_2.

B. Complexity Analysis

This section contains theoretical analyses about the computation cost as well as communication cost in regard to the proposed method. Suppose an object tensor has *m* elements, including m_0 zero elements and m_1 nonzero elements, and there are *n* object tensors in a dataset.

Computation Cost: To release the burden of the client, in the proposed PPTMC method, it is required that the client encrypts all object tensor prior to outscouring them to C_1, then all computations of multiple clusterings do not need user participation any more. Hence, the computation cost about the client is determined by the encryption time of a single element and the total number of nonzero elements of all object tensors. Therefore, the client's computational complexity is $O(m_1 n)$.

Upon the PPTMC protocol, the cloud's computation cost T_c contains the cost of secure computing transition tensor $T_{c_{tr}}$, the cost of secure constructing weight tensor $T_{c_{sawr}}$, the cost of computing SSWTD matrix $T_{c_{sswtd}}$ and the cost of SCFS $T_{c_{scfs}}$, which is defined as the equation:

$$T_c = T_{c_{tr}} + T_{c_{sawr}} + T_{c_{sswtd}} + T_{c_{scfs}}. \tag{5}$$

Algorithm 4 PPTMC Clustering Protocol

Input: C_1 has the encrypted object tensors $[\![\lambda_o \mathcal{X}_1]\!], [\![\lambda_o \mathcal{X}_2]\!], \ldots, [\![\lambda_o \mathcal{X}_n]\!]$ and the feature space combination vectors $[\![\mathrm{v}_1]\!], [\![\mathrm{v}_2]\!], \ldots, [\![\mathrm{v}_b]\!]$, C_2 has *sk*.
Output: Encrypted multiple clusterings $[\![cl_1]\!], [\![cl_2]\!], \ldots, [\![cl_b]\!]$ only to C_1.

C_1:
Compute the association tensor $[\![\lambda_o \mathcal{T}_a]\!]$ with elements $[\![\lambda_o t^a_{i_1 i_2 \ldots i_k}]\!] = \prod_{d=1}^{n} \left[\!\left[\lambda_o t^{ob(d)}_{i_1 i_2 \ldots i_k}\right]\!\right]$.
C_1, C_2:
Set $z = max\{I_{f_1}, I_{f_2}, \ldots, I_{f_k}\}$.
for *l*=1 to *k* **do**
  Compute the transition tensor $\left[\!\left[\lambda_o \mathcal{T}^{(l)}_{tr}\right]\!\right]$ with elements $\left[\!\left[\lambda_w t^{tr(l)}_{i_1 \ldots i_l \ldots i_k}\right]\!\right] \leftarrow SD2([\![\lambda_o t^a_{i_1 \ldots i_l \ldots i_k}]\!]^{\lambda_w}, \prod_{i_l=1}^{z} [\![\lambda_o t^a_{i_1 \ldots i_l \ldots i_k}]\!])$.
end for
$[\![\lambda_w \mathrm{w}_1]\!], [\![\lambda_w \mathrm{w}_2]\!], \ldots, [\![\lambda_w \mathrm{w}_k]\!] \leftarrow SAWR(\left[\!\left[\lambda_w \mathcal{T}^{(1)}_{tr}\right]\!\right], \left[\!\left[\lambda_w \mathcal{T}^{(2)}_{tr}\right]\!\right], \ldots, \left[\!\left[\lambda_w \mathcal{T}^{(k)}_{tr}\right]\!\right])$.
Initialize the weighted tensor $[\![\lambda_w \mathcal{T}_w]\!]$ with elements $[\![\lambda_w t^w_{i_1 i_2 \ldots i_k}]\!] \leftarrow [\![\lambda_w]\!]$.
for *l*=1 to *k* **do**
  Update the weighted tensor $[\![\lambda_w \mathcal{T}_w]\!]$ with elements $[\![\lambda^2_w t^w_{i_1 \ldots i_l \ldots i_k}]\!] \leftarrow SM([\![\lambda_w t^w_{i_1 \ldots i_l \ldots i_k}]\!], [\![\lambda_w (w_l)_{i_l}]\!])$.
  $[\![\lambda_w t^w_{i_1 \ldots i_l \ldots i_k}]\!] \leftarrow SD1([\![\lambda^2_w t^w_{i_1 \ldots i_l \ldots i_k}]\!], \lambda_w)$.
end for
for *q*=1 to *b* **do**
  for *l*=1 to $I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}$ **do**
    for *m*=1 to $I_{f_1} \times I_{f_2} \times \cdots \times I_{f_k}$ **do**
      $\left[\!\left[\|p_l - p_m\|_2^2\right]\!\right] \leftarrow \prod_{t=1}^{k} [\![(v_q)_t]\!]^{(i_t - i_t^{'})^2}$.
      $\left[\!\left[\frac{\lambda_d \|p_l - p_m\|_2^2}{2\sigma^2}\right]\!\right] \leftarrow SD1(\left[\!\left[\|p_l - p_m\|_2^2\right]\!\right]^{\lambda_d \lambda_\sigma}, \lambda_\sigma 2\sigma^2)$.
      $[\![\lambda_\sigma \lambda_e (g_q)_{lm}]\!] \leftarrow SE(\left[\!\left[\frac{\lambda_d \|p_l - p_m\|_2^2}{2\sigma^2}\right]\!\right]^{N-1}) \frac{\lambda_\sigma}{2\pi\sigma^2}$.
      $[\![\lambda_e (g_q)_{lm}]\!] \leftarrow SD1([\![\lambda_\sigma \lambda_e (g_q)_{lm}]\!], \lambda_\sigma)$.
    end for
  end for
  for *j*=1 to *n* **do**
    for *h*=*j*+1 to *n* **do**
      $\left[\!\left[\lambda_o (S_V{}^{(q)})_{j,h}\right]\!\right] \leftarrow SSWTD([\![\lambda_o \mathcal{X}_j]\!], [\![\lambda_o \mathcal{X}_h]\!], [\![\lambda_e G_q]\!], [\![\lambda_w \mathcal{T}_w]\!])$.
    end for
  end for
end for
Build the encrypted multiview tensor $[\![\mathcal{T}_{mv}]\!]$ with $\left[\!\left[S^{(1)}_V\right]\!\right]$, $\left[\!\left[S^{(2)}_V\right]\!\right]$, …, $\left[\!\left[S^{(b)}_V\right]\!\right]$.
Generate multiple clusterings $[\![cl_1]\!], [\![cl_2]\!], \ldots, [\![cl_b]\!]$ by using SCFS to $[\![\mathcal{T}_{mv}]\!]$ in parallel.

The computation of the transition tensor is based on an association tensor where all the original tensors accumulate

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

together, time complexity $T_{c_{tr}}$ is $O(m_0 n + m_1 n)$, because it has to add up every object tensor, and carry out a division on every nonzero element. The time complexity of secure constructing of weight tensor $T_{c_{sawr}}$ is $O(m)$, because every element of the transition tensor is involved in a series of secure multiplications, but the number of times of the secure multiplications is a constant. The cost of computing SSWTD matrix $T_{c_{sswtd}}$ is $O((m^2 - m_0^2)n^2)$, because for one single distance, it has to carry out four secure multiplications and two homomorphic subtractions for every g_{lm}, and the number of g_{lm} is m^2, where the number of distances is $n(n-1)/2$. However, the reason of $T_{c_{sswtd}}$ not being $O(m^2 n^2)$ is that zero elements are not encrypted, and if one subtraction is about two zeros, the corresponding computations can be omitted. So the cost of computing SSWTD matrix $T_{c_{sswtd}}$ is $O((m^2 - m_0^2)n^2)$. The time complexity of SCFS algorithm $T_{c_{scfs}}$ is proved to be $O(n^2)$ in [14]. In summary, the total time complexity T_c is $O((m^2 - m_0^2)n^2)$.

Communication Cost: Given the Paillier encryption key size *s*, the client uploads $(nm_1 + bk)s$ messages to C_1 prior to carrying out the PPTMC on the cloud. After the whole algorithm is executed, the client downloads $(bn+1)s$ messages from the clouds, including the perturbed multiple clustering results from C_2 and the random number from C_1.

C. Datasets and Evaluation Metrics

In experiments and simulations of this section, the proposed method is applied to two real-world datasets. The first dataset is about the smart grid [14], including electricity consumption of more than one thousand enterprises in Yangzhong High-tech Development Zone, Jiangsu, China in 2015, economy data and meteorology data. In the grid dataset, each object tensor has four dimensions: date, the PPI (producer price index, an economic index number), weather (cloudy, sunny, overcast, rain, snow) and average temperature of that day. A company corresponds to an object tensor. The element in an object tensor is the electricity consumption of the company on its coordinate (date, weather, temperature and PPI). So the size of each object tensor is $24 \times 24 \times 5 \times 11$, corresponding to twenty-four different dates, twenty-four different temperatures, five different weathers and eleven different PPI numbers, respectively. The second dataset is from a smart bike maintenance system in New York City [16]. In the dataset used in this experiment, there are 473620 bike-sharing records, as well as following information: start time, stop time, origin station, destination station, and so on. In the bike dataset, each object tensor has $7 \times 4 \times 28 \times 14$ elements, and the dimensions separately corresponding transition pattern, weather, temperature, and wind speed. A record corresponds to an object tensor. Comparing to the second dataset, the object tensors of the grid dataset have larger dimensions, more nonzero elements and usually larger elements.

In the evaluation of accuracy, two widely used metrics, E_* and *RI* [17], are applied to evaluate the clustering accuracy of the PPTMC algorithm.

E_* is utilized to measure the quality of clustering centers, which computes the distance of the produced clustering centers by one algorithm and the actual ones, and is calculated as

$$E_* = \sqrt{\sum_{i=1}^{c} \left\| v_{ideal}^i - v_*^i \right\|^2}, \tag{6}$$

where, v_{ideal}^i denotes the *i*th actual cluster center and v_*^i represents the *i*th cluster center generated by the specific algorithm *. The lower the E_* value, the more accurate the generated clustering centers.

Rand Index (*RI*) is utilized to evaluate the quality of clustering result by mearsuring whether the result produced by an clustering algorithm is consistent with the true clustering result, which is defined as

$$RI = \frac{TP + TN}{TP + FP + TN + FN}, \tag{7}$$

where *TP* represents two similar objects are parted into one cluster correctly; *TN* indicates two dissimilar objects are grouped into two clusters correctly; *FP* denotes two dissimilar objects are parted into one cluster incorrectly; *FN* implies two similar objects are grouped into two clusters incorrectly. The higher the *RI* value, the more accurate the produced clustering result.

D. Encryption Time

In the proposed scheme, the client needs to encrypt the data with the Paillier cryptosystem before outsourcing them to cloud for clustering. The encryption is carried out on the client, so it is essential to evaluate the burden on the client during the encryption. For assessing the impact of dataset size on encryption time, 40, 80, 120, 160 and 200 objects are encrypted on client in the two datasets, respectively. Fig. 1 (a) shows the encryption time, which changes from 1.486s to 6.764s (grid dataset) and from 0.618s to 3.415s (bike dataset), respectively, linearly increasing with the number of objects. In addition, the encryption time of bike dataset is always less than that of grid dataset, indicating the different sizes of dimensions, sparseness as well as the values of elements have an important impact on encryption time.

However, the encryption operation can be pre-performed offline, and it only needs to be performed once during the entire method. In addition, after downloading cluster results from the cloud, the client does not perform decryption operations in the proposed scheme, instead, it only needs to remove perturbation from the plaintext results, the time of which can be ignored. Moreover, Fig. 1 (b) shows a set of encryption time of the grid dataset using the BGV encryption scheme for comparison, and the encryption time for the client of PPTMC is between 1/40 and 1/30 of BGV. The BGV technique is used in a similar privacy preserving higher-order CFS method [17]. In summary, the proposed method is very lightweight for the client.

E. Execution Time

In this section, speedup in latency is used as the speedup ratio, which is defined as the quotient of the execution time in parallel and the execution time in serial, and the ratio of the PPTMC in cloud platform is simulated to evaluate its

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

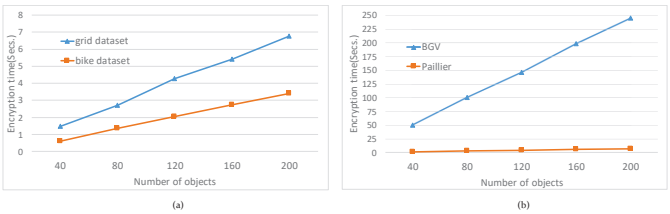


Fig. 1. Encryption time of two datasets and two encryption schemes.

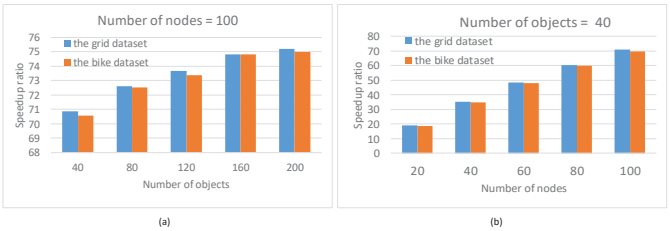


Fig. 2. Speedup ratio on the two datasets.

scalability and efficiency. The execution time in serial is from experiments using just one node on both C_1 and C_2, while the execution time in parallel is from experiments using many nodes on both clouds. The simulations are operated on the grid dataset and bike dataset, with different number of objects and nodes, respectively. And the simulation results are shown in Fig. 2.

In Fig. 2 (a), the speedup ratio keeps increasing slowly from 70.85 to 75.19 for 40, 80, 120, 160 and 200 objects with 100 nodes, indicating the PPTMC method is very scalable to be carried out in parallel. Fig. 2 (b) shows how the speedup ratio changes for 40 objects. As the number of nodes increased from 20 to 100, the speedup ratio increases almost linearly from 18.93 to 70.85 and 18.79 to 69.56 with grid dataset and bike dataset, respectively. Such result reveals that PPTMC has high scalability for Industrial IoT big data when employing more nodes on cloud. Besides, experiments with the bigger size of dataset usually generate higher speedup ratios, but the differences are not significant.

F. Clustering Accuracy

The clustering accuracy of PPTMC is measured in regard to E_* and *RI* in this section. Specially, considering that the proposed scheme aims at implementing completely privacy preserving TMC algorithm, the results of original TMC algorithm is used as a benchmark for clustering accuracy. Meanwhile, to evaluate robustness of the proposed approach and how magnification factor influence the accuracy of the method, PPTMC was performed with different magnification factors on the two datasets with 200 objects and the results are shown in Table III.

From the experimental results, the E_* and *RI* are gradually close to 0 and 1, respectively, when the magnification factor increases below 10^5. That means the results are almost consistence with the plaintext algorithm, but there are some little errors. In fact, these errors are mainly *FN* errors. After the magnification factor reaches 10^5, the values of E_* and *RI* are

TABLE III
TABLE OF THE EVALUATION OF CLUSTERING ACCURACY

Dataset	Evaluation criteria	10^3	10^4	10^5	10^6	10^7
Grid	E_*	0.027	0.01	0	0	0
	RI	0.901	0.988	1	1	1
Bike	E_*	0.031	0.015	0	0	0
	RI	0.915	0.976	1	1	1

both 0 and 1, respectively. Such result demonstrates that the clustering results of PPTMC are accorded with the original TMC algorithm completely, which is ensured by the formal Paillier cryptosystem and magnification factor for floating point precision. Consequently, the PPTMC can achieve perfect performances about clustering accuracy and robustness.

VI. RELATED WORK

In recent years, some privacy preserving approaches for clustering algorithms have been developed, including two kinds of technologies in popular: randomization method and encryption method. The former uses the data distortion technique to meet privacy preserving for clustering analysis [18, 19]. The latter was proposed by using cryptographic method for privacy preserving clustering [20]. Because the encryption method not only can provide formal guarantees of privacy, but also is superior to randomization method as far as accuracy. This work focuses on the encryption method.

There are mainly two scenarios of privacy preserving clustering using encryption technology, one is the distributed clustering based on secure multiparty computation (SMC), another is the outsourced clustering based on cloud computing. In distributed clustering, each party carries out calculation tasks separately on its own data and shares part of its data to cluster, containing two popular homomorphic schemes: homomorphic public-key cryptosystems and secret additive sharing schemes [21]. [22, 23] utilize the Paillier cryptosystem to realize secure k-means clustering on distributed dataset, while [24] shows an additive secret sharing scheme requiring non-colluding parities for k-means clustering. However, since k-means algorithm has the iterative nature, the above protocols do not achieve a complete preservation of privacy. For outsourced clustering, the data owner outsources all its data to cloud for clustering calculation, while its cost is expected to be reduced to minimum by using the cloud to carry out as much as the calculations. Liu et al. [25] present an outsourced k-means clustering method using homomorphic encryption, but it requires the client to participate in comparing encrypted distances. Zhang et al. [17] proposed PPHOCFS extended on the traditional CFS algorithm by utilizing the BGV encryption scheme, but due to some limitations of operations, such as division and comparison, it only implements the distance calculation on ciphertexts, while CFS is still executed on plaintexts.

The proposed PPTMC method is the first to achieve the privacy preserving multiple clustering algorithm, and there are two main differences in contrast to the above schemes. First, PPTMC can realize a completely private protocol over encrypted data under the semi-honest model using the secure

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

two-party computing, while the protected information contains intermediate results, distances, clustering centers, objects in each cluster, number of clusters, as well as number of objects in each cluster. Moreover, current schemes focus on designing privacy preserving methods on un-encrypted data with multiple colluding or non-colluding parties in data sharing, or over encrypted data requiring user's involvement in outsourced calculations. Different from them, the aim of PPTMC is to provide the privacy preserving multiple clustering algorithm over encrypted data, to improve it's efficiency by cooperating the power of clouding computing, as well as to make the client lightweight as much as possible.

VII. Conclusion

Aiming at securely and efficiently providing different clustering services according to different applications in Industrial IoT, this paper proposes a PPTMC method as well as the related SE, SAWR and SSWTD protocols. In the proposed scheme, all computational tasks are implemented on cloud whereas any confidential information is not exposed or inferred, not only enhancing efficiency, but also preserving users' privacy. The method adopts the Paillier cryptosystem on hybrid cloud model to carry out PPTMC in private. Finally, the work theoretically analyzes PPTMC with respect to security, computation and communication cost, implements and evaluates it on two real-world datasets.

Evaluation and experimental results show that: (1) PPTMC provides a complete privacy preserving protocol over encrypted data through the security analysis; (2) by using the formal homomorphic cryptosystem and controlling the floating point precision, PPTMC can achieve 100% clustering accuracy compared with plaintext TMC method; (3) without getting involved in any multiple clustering calculations, the client only needs to perform fast encryption that its encryption time is between 1/40 and 1/30 of BGV, and remove perturbation with the returned clustering results from cloud, which is very lightweight for users; (4) with the number of nodes increasing from 20 to 100, the speedup ratio increases almost linearly from 18.93 to 70.85 and 18.79 to 69.56 with grid dataset and bike dataset, respectively, which indicates PPTMC has high scalability when using more cloud servers, and this is very important for the analysis of Industrial IoT big data.

Future work focuses on improving the efficiency of the PPTMC by seeking more efficient methods, and estimating it on more real Industrial IoT applications, as well as studying the incremental PPTMC method for streaming data.

References

[1] X. Wang, L. T. Yang, H. Liu, and M. J. Deen, "A big data-as-a-service framework: state-of-the-art and perspectives," *IEEE Trans. Big Data.*, vol. 4, no. 3, pp. 325–340, 2018.

[2] C. W. Tsai, C. F. Lai, M. C. Chiang, and L. T. Yang, "Data mining for internet of things: a survey," *IEEE Commun Surv. Tutor.*, vol. 16, no. 1, pp. 77–97, 2014.

[3] E. Müller, I. Assent, S. Günnemann, T. Seidl, and J. Dy, "Multiclust special issue on discovering, summarizing and using multiple clusterings," *Mach. Learn.*, vol. 98, no. 1-2, pp. 1–5, 2015.

[4] Y. Zhao, L. T. Yang, and R. Zhang, "A tensor-based multiple clustering approach with its applications in automation systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 283–291, 2018.

[5] M. Ma, D. He, N. Kumar, K. K. R. Choo, and J. Chen, "Certificateless searchable public key encryption scheme for industrial internet of things," *IEEE Trans. Ind. Informat.*, DOI:10.1109/TII.2017.2703922, 2017.

[6] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Sci.*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[7] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. 17th Int. Conf. Theor. Appl. Crypto. Tech.*, vol. 99, Prague, Czech Republic, May. 2-6, 1999, pp. 223–238.

[8] O. Goldreich, "Foundations of cryptography," *Knowl. Inf. Syst., Cambridge Univ. Press,*, vol. 2, 2009.

[9] X. Huang and X. Du, "Achieving big data privacy via hybrid cloud," in *Proc. 33rd Annu. IEEE Int. Conf. Comput. Commun.*, Toronto, Canada, April. 27-May. 2, 2014, pp. 512–517.

[10] B. K. Samanthula, Y. Elmehdwi, and W. Jiang, "k-nearest neighbor classification over semantically secure encrypted relational data," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1261–1273, 2015.

[11] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, "Machine learning classification over encrypted data," in *Proc. of 22nd Annu. Netw. Distrib. Syst. Secur. Symp.*, San Diego, USA, Feb. 8-11, 2015.

[12] T. Veugen, "Encrypted integer division and secure comparison," *Int. J. Appl. Crypt.*, vol. 3, no. 2, pp. 166–180, 2014.

[13] J. Feng, L. T. Yang, Q. Zhu, and K. K. R. Choo, "Privacy-preserving tensor decomposition over encrypted data in a federated cloud environment," *IEEE Trans. Dependable Secure Comput., Accepted*, 2017.

[14] Y. Zhao, L. T. Yang, and J. Sun, "A secure high-order CFS algorithm on clouds for industrial internet-of-things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3766–3774, 2018.

[15] Q. Zhang, L. T. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1351–1362, 2016.

[16] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proc. 23rd ACM SIGSPATIAL Int. Conf. Adv. Geogr. 606 Inf. Syst.* ACM, Nov. 3-6, 2015, pp. 1–10.

[17] Q. Zhang, L. T. Yang, Z. Chen, and B. Fanyu, "P-PHOCFS: privacy preserving high-order CFS algorithm on the cloud for clustering multimedia data," *ACM Trans. Multimedia Comput. Commun, Appli.*, vol. 12, no. 4s, pp. 66:1–15, 2016.

[18] S. R. M. Oliveira and O. R. Zaïane, "Privacy preserving clustering by data transformation," in *Proc. 18th Braz. Sympo. Databases.*, Amazonas, Brazil, Dec. 6-8, 2003,

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TII.2018.2871174, IEEE Transactions on Industrial Informatics

pp. 304–318.

[19] M. N. Lakshmi and D. K. S. Rani, "Privacy preserving clustering by hybrid data transformation approach," *Int. J. Emerg. Technol. Adv. Eng., ISSN 2250-2459, ISO 9001:2008 Certified J.*, vol. 3, no. 8, pp. 696–700, 2013.

[20] Y. Lindell and B. Pinkas, "Privacy-preserving data mining," in *Proc. 20th Annu. Int. Crypto. Conf.*, California, USA, Aug. 20-24, 2000, pp. 36–54.

[21] F. Meskine and S. N. Bahloul, "Privacy preserving k-means clustering: a survey research," *Int. Arab. J. Inf. Technol.*, vol. 9, no. 2, pp. 194–200, 2012.

[22] P. Bunn and R. Ostrovsky, "Secure two-party k-means clustering," in *Proc. 14th ACM conf. Comput. Commun. Secur.*, Alexandria, USA, Oct. 29-Nov. 2, 2007, pp. 486–497.

[23] J. Sakuma and S. Kobayashi, "Large-scale k-means clustering with user-centric privacy-preservation," *Knowl. Inf. Syst.*, vol. 25, no. 2, pp. 253–279, 2010.

[24] M. C. Doganay, T. B. Pedersen, and A. Levi, "Distributed privacy preserving k-means clustering with additive secret sharing," in *Proc. Int. Workshop. Privacy. Anonymity. Inf. Soc.*, Nantes, France, Mar. 29, 2008, pp. 3–11.

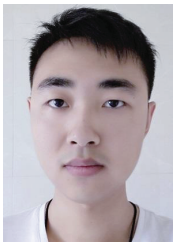
[25] D. Liu, E. Bertino, and X. Yi, "Privacy of outsourced k-means clustering," in *Proc. ACM Symp. Inf. Comput. Commun. Secur.*, 2014, pp. 123–134.

Yaliang Zhao received the B.E. degree in computer science and the M.S. degree in applied mathematics from the School of Computer and Information Engineering, Henan University, Kaifeng, China. She is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China.

She is currently an associate professor in the School of Computer and Information Engineering, Henan University, Kaifeng, China. Her research interests include big data security, cloud computing security, and internet of things.

Laurence T. Yang (M'97-SM'15) received the B.E. degree in computer science and technology from Tsinghua University, China and the PhD degree in computer science from University of Victoria, Canada.

He is a professor in the School of Computer Science and Technology, Huazhong University of Science and Technology, China, and in the Department of Computer Science, St. Francis Xavier University, Canada. His research interests include parallel and distributed computing, embedded and ubiquitous computing, big data. His research has been supported by National Sciences and Engineering Research Council and Canada Foundation for Innovation.



Jiayu Sun received the B.E. degree in internet of things engineering from the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. He is currently working toward the M.S. degree with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China.

His research interests include big data security, cloud computing security, and internet of things.

1551-3203 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.