Computers and Electrical Engineering 000 (2017) 1–11

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

An efficient and secure ridge regression outsourcing scheme in wearable devices☆

Xinshu Ma[a], Youwen Zhu[a,b,*], Xingxin Li[a]

[a] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
[b] Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

ARTICLE INFO

Article history:
Received 14 December 2016
Revised 18 July 2017
Accepted 21 July 2017
Available online xxx

Keywords:
Cloud computing
Ridge regression
Secure outsourcing
Wearable medical devices
Privacy

ABSTRACT

Ridge regression is an important approach in many applications, such as healthcare system of smart wearable equipments. Due to the limited resources of wearable devices, outsourcing is a promising computation paradigm. Nevertheless, it also suffers from some privacy challenges, as outsourced computation probably involves some sensitive data. In this paper, we propose a ridge regression outsourcing scheme, which can securely utilize the cloud to analyse large-scale wearable device dataset and dramatically reduce the computation cost of the resource-limited clients. Technically, we use random vectors and dense matrices to perturb input dataset and regression output, such that both input privacy and output privacy can be efficiently protected. Then, we present a highly-efficient verification algorithm to robustly check the correctness of cloud's answer against a dishonest/lazy cloud server. Finally, we evaluate our scheme through theoretical analysis and extensive experiments. The results show we can achieve input/output privacy, correctness, robust checkability and practical efficiency.

1. Introduction

Ridge regression (RR) is an important approach for modeling the relationship between a dependent variable and one or more independent variables. Compared to linear regression, RR is more suitable for the case with ill-conditioned independent variables and the case with many predictors. Nowadays, regression analysis has been widely applied in many scenarios, such as air quality prediction, recommendation system and wearable devices.

As well known, wearable device is a kind of portable device with constrained resources, worn directly on the body or integrated into the user's clothing or accessory. With the rapid development of cloud computing, outsourcing computation is inevitably becoming a popular and practical computing paradigm, which enables clients with low computation power to off-load their heavy computation workloads from themselves to the cloud server and to enjoy the unlimited computing resources in a pay-per-use manner [1–3]. For example, heart rate monitor collects heart rates of users and then sends these data to the cloud server. After that, the cloud server runs diagnostic functions to identify the signs of illness and answers computation result to the monitor. Therefore, cloud computing can be utilized to enhance the utility of wearable devices. However, outsourcing computation also suffers from some security and privacy challenges [4–6]. Firstly, the most significant

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. Debiao He.

* Corresponding author at: College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China.
E-mail addresses: zhuyw@nuaa.edu.cn, zhuyouwen@gmail.com (Y. Zhu).

http://dx.doi.org/10.1016/j.compeleceng.2017.07.019
0045-7906/

one is the *privacy* issue. Data collected by wearable devices usually contains private information. Meanwhile, the regression output is also sensitive generally. That is, neither the input nor the output of RR should be learned by the cloud server. Secondly, another challenge is the *checkability* of this computing paradigm. The client should have the ability to verify the results returned from the cloud server. It is considered that semi-trusted cloud server might give an invalid result. For one thing, casual software bugs and hardware faults would lead to a wrong computation. For another, despite having more computation resources than the client, cloud server still has restricted resources and the financial incentive to laze. Besides, due to some intentional reasons, cloud server might acquire some useful information of the client's data via returning a false answer. Therefore, algorithms for secure outsourcing should have an efficient way to detect whether the server misbehaves or not. Finally, the third challenge is *efficiency*. Although the original computation problem needs to be securely transformed, the transformation must be more efficient than carrying out the computation task locally. Hence, an outsourcing computing algorithm should meet four requirements: correctness, input/output security, checkability and efficiency.

In this paper, we are motivated to propose a secure and efficient approach for outsourcing RR problem to a public cloud that addresses all the challenges above. To protect the privacy of the client's sensitive input and output, we propose a novel encryption method by adding random vectors and multiplying our constructed dense invertible matrices against the untrusted cloud server. The cloud server will then perform RR on the encrypted dataset and send the encrypted result back to the client. Meanwhile, we present an efficient decryption approach corresponding to our encryption scheme, by which the client can recover the answer of her original RR problem rapidly. To verify the returned result from cloud server, we propose a new RR answer verification algorithm, in which the client can find any deviation effectively by checking a simple equation. Recently, the work [7] and [8] also consider secure ridge regression problem. However, the cloud server in their schemes can access the secret key (for decryption), which is much different from ours. In our scheme, only the client knows decryption key, and the cloud server cannot learn anything useful about it.

Generally, our contribution in this work can be summarized as follows.

- We present a new encryption algorithm to securely outsource RR to a public cloud. We can well preserve the sensitive information of the client with the new encryption method, such that curious cloud server cannot learn anything useful about the client's private dataset.
- We can dramatically reduce the computation cost of the client. Besides, our scheme only require two rounds communication between the client and cloud server, thus our communication overhead is practically small.
- We also propose an efficient verification algorithm and decryption algorithm, by which the client can rapidly check the correctness of cloud's answer, and efficiently decrypt a correct answer to recover the original RR output.
- Additionally, we provide detailed theoretical analysis and extensive simulation experiments to evaluate our scheme, which validate the correctness, input/output privacy, checkability and practical efficiency of our solution.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 formulates the ridge regression outsourcing problem and introduces the framework of our problem. The proposed new secure outsourcing algorithm of ridge regression is presented in Section 4. We present detailed theoretical analysis in Section 5. Then, Section 6 shows experimental results of the proposed algorithm. Finally, the paper is concluded in Section 7.

2. Related work

Much attention has been paid to the problem of securely outsourcing various kinds of expensive computations recently. In this section, we review several recent works related to such outsourcing problem.

The work [9] addresses the problem of secure outsourcing scientific computations and presented some disguise techniques to guarantee the security of outsourcing. Nevertheless, the verification of the returned answer is not discussed in [9]. Recently, Chen et al. [6] propose two protocols for outsourcing linear regression problems to the cloud with security and efficiency. To preserve the privacy, the original linear regression problem is transformed into a new one before being sent to cloud server. Then the client recover the real result from cloud server's answer. Besides, a new secure outsourcing algorithm for (variable-exponent, variable-base) exponentiation modulo a prime in the two untrusted program model is proposed by Chen et al. [10]. Zhu et al. [11] propose a new efficient solution dealing with outsourcing linear regression with robust answer verification. In [3], Chen et al. investigate a new secure outsourcing algorithm of large-scale linear equations efficiently. Additionally, considering the challenge of verifiability in outsourcing, the work [12] provides a new VDB framework based on the idea of commitment binding. In 2016, Chen et al. [13] present a general Inc-VDB framework by incorporating the primitive of vector commitment and the encrypt-then-incremental MAC mode of encryption, and a concrete Inc-VDB scheme based on the computational Diffie–Hellman (CDH) assumption. In the cryptography community, two highly non-trivial protocols [14,15] are put forward to solve the problem of outsourcing any reasonable computation, which can be completed within polynomial time by a Turing machine. Barbosa et al. [16] investigate homomorphic encryption to cope with secure computation outsourcing. In 2013, Nikolaenko et al. [7] propose a privacy-preserving ridge regression scheme on hundreds of millions of records. It combines both homomorphic encryption and Yao garbled circuits protocol where each is used in a different part of the algorithm to obtain the best performance. After that, a new efficient privacy-preserving ridge regression scheme is presented in [8] utilizing Paillier encryption with excellent performance. Although the homomorphic algorithm can theoretically solve the secure outsourcing problems, it is still far from practice due to the expensive computations on homomorphic encryptions.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

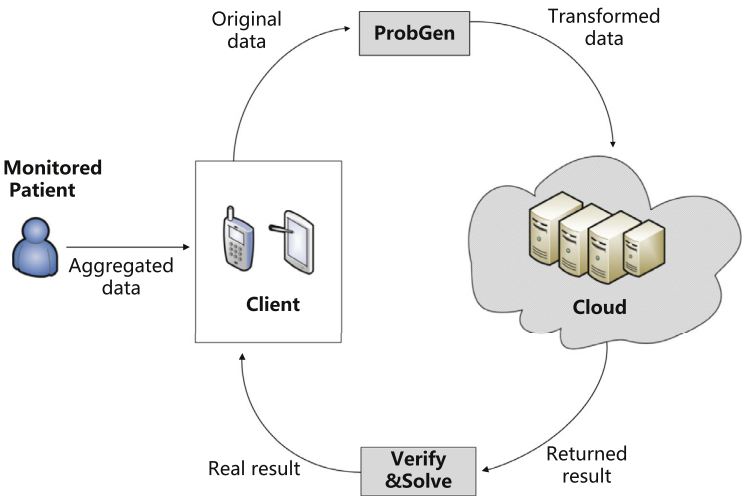

Fig. 1. Ridge regression outsourcing system model.

3. Systemd model and preliminaries

In this section, we will introduce our system model, design goals and some preliminaries.

3.1. System model

In this paper, we consider the outsourcing system involving a client and a cloud server, as described in Fig. 1. Here, the client has a dataset of aggregated private data from users who can be a smart phone or smart watch. These private data may contain sensitive personal information, such as age, weight and heart rate. The client wants to perform RR computation on the dataset, but has no enough computation capability to complete the task. Thus, the client will exploit the computation ability of cloud server to implement the RR computation. For the purpose, the client and cloud server will complete the computation as follows. Firstly, the client sends original private dataset in encrypted form to a cloud server, such that the cloud server cannot learn any sensitive information. Secondly, cloud server performs RR over these encrypted data and returns computation results to the client. Finally, the client verifies the answer that is received from the cloud server, and recover the real result of original RR problem.

3.2. Design goals

We consider the cloud server to be the potential adversary in this paper, because external attackers or employees of the cloud company could tamper the program running on the cloud server, such that the private data could be revealed. Here, we assume the cloud server to be malicious [3,6]. On one hand, the cloud would be curious, and attempt to learn some useful information of the client during the outsourcing computation. On the other hand, the cloud would be lazy and dishonest, who may return a random result instead of computing the true answer to save computing resources, or return an incorrect answer due to software bugs and hardware errors.

In such scenario, our outsourcing computation algorithm aims to guarantee the correctness of RR computation, preserve the input/output privacy of the client and decrease the computation and communication overheads of the client. Concretely, our design goals include the following four aspects.

- *Correctness.* The most basic demand of the outsourcing computation task is the correctness of the result. That is, the client could get a correct answer while both the cloud server and the client strictly obey the protocol.
- *Input/Output security.* The algorithm should protect the privacy of the client's data including input and output, which means that the curious cloud server cannot learn anything useful from the interactions with the client.
- *Checkability.* Corresponding to the validity of the computation, the algorithm should enable the client to check the correctness of the returned result. More precisely, any false answer given by the malicious cloud server will be detected by the client and cannot be accepted.
- *Efficiency.* The overheads of computation and communication of the client must be as low as possible, and theoretically less than those of conducting the original RR problem locally. Thus, we will compare the overheads of the client with and without outsourcing respectively, and the cost of cloud server will be listed as well to represent how much workload the cloud server shares.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

3.3. Ridge regression

In statistics, *linear regression* is an approach for modeling the relationship between a scalar dependent variable and one or more explanatory variables (or independent variables). Given a set of m input variables $x_i \in \mathbb{R}^n$, and output variables $y_i \in \mathbb{R}$ $(1 \le i \le m)$. Linear regression assumes that the relationship between dependent variable y_i and explanatory variable x_i is linear, thus there exists a function $f : \mathbb{R}^n \to \mathbb{R}$. Namely, $y_i \simeq f(x_i)$ for $1 \le i \le m$. Therefore, given a parameter vector $\beta \in \mathbb{R}^n$, these m equations are stacked together and written in vector form as

$$y \simeq X\beta, \tag{1}$$

where

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}. \tag{2}$$

Ridge regression (RR) was developed by Hoerl and Kennard to mainly deal with the case when there were ill-conditioned X or too many predictors. The potential instability in the linear regression estimator

$$\beta = (X^T X)^{-1} X^T y, \tag{3}$$

could be improved by adding a constant value λ to the diagonal entries of the matrix $X'X$ before it taking inverse. Thus, the following RR estimator is created

$$\beta = (X^T X + \lambda I)^{-1} X^T y. \tag{4}$$

To compute the vector β, we minimize the following quadratic function:

$$F(\beta_j) = \sum_{i=1}^{m} (y_i - \sum_{j=1}^{n} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{n} \beta_j^2 \tag{5}$$

Therefore, ridge regression puts further constraints on the parameters β_j. Instead of just minimizing the residue sum of squares, we also have a penalty term on β. This penalty term is equal to λ, a pre-chosen positive constant, times the squared norm of β.

The minimizer of the Eq. (5) can be computed by solving the following linear system:

$$A\beta = b, \tag{6}$$

where $A = X^T X + \lambda I$ and $b = X^T y$. While using this model, the estimation is given by the following equation

$$\beta = A^{-1} b. \tag{7}$$

In Section 4.1, we will present our encryption algorithm based on this linear system.

3.4. Framework

The work [14] proposed a formal definition for securely outsourcing computation problem. Similarly, our secure outsourcing scheme contains five algorithms (**KeyGen, RREnc, RRSolve, ResultVerify, ResultDec**) as follows.

- **KeyGen** $(m, n) \longmapsto \{K\}$: We input two parameters m and n representing the dimension of variables like X. Then the algorithm will generate the random secret key denoted by K in order to encrypt the original RR problem as well as decrypt the answer received from cloud server. Note that each key can only encrypt one RR problem.
- **RREnc** $(\Phi, K) \longmapsto \{\Phi'\}$: After generating secret keys K, the original RR problem Φ will be encrypted into a new problem Φ', such that sensitive information of the original problem cannot be learned by the cloud server.
- **RRSolve** $(X', \Phi') \longmapsto \{E, \beta'\}$: In our scheme, client only requires two rounds communication with the cloud server. Once receiving the data X', cloud server computes the result $E = X'^T X'$ and sends it back instantly. Once receiving the encrypted problem Φ', cloud server will solve the RR problem and answer the client with result β'.
- **ResultVerify** $(\beta') \longmapsto (1/0)$: On receiving the encrypted answer β', the client performs **ResultVerify** algorithm to check whether the answer is correct. If β' is valid, the client will output 1 and perform the next algorithm **ResultDec**. Otherwise, the client will output 0 and consider that cloud server misbehaved.
- **ResultDec** $(\beta') \longmapsto \{\beta\}$: Once the output of the cloud β' is valid, client will decrypt the result β' with secret key K to obtain the answer to the original RR problem β.

4. Our secure scheme for ridge regression outsourcing

In this section, we propose our privacy-preserving and efficient RR outsourcing scheme. First we show how the original RR problem is encrypted. After the original problem is perturbed, we propose our answer verification algorithm to guarantee that the received answer is valid. Then, we introduce the decryption method. Finally, formal description of our proposed scheme will be given.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

4.1. Our encryption method

Here, we utilize the RR model in Eqs. (4) and(6). Clearly, we can ensure the following equation to hold,

$$\beta = (X^T X + \lambda I)^{-1} X^T y = A^{-1} b. \tag{8}$$

Therefore the primary task for outsourcing is to compute Eq. (8), where $A = X^T X + \lambda I$ and $b = X^T y$. We encrypt A and b into A' and b', then send the encrypted items to the cloud server. Our main idea is to protect X and y employing matrix addition and multiplication operations, simultaneously decreasing the client's time complexity to $O(mn)$ as much as possible. The detailed encryption techniques are as follows.

We initially consider A to be a single variable, directly encrypting it by $A' = MAN$, where M and N are two random $n \times n$ matrices. However, this method contributes to $O(mn^2)$ execution time because the matrix multiplication $X^T X$ in $M(X^T X + \lambda I)N$ is unavoidable. Thus, to reduce the computing overhead, we take X as our smallest encryption item but still regard A as a single variable. Namely, by encrypting X progressively, we eventually obtain A' in the form of MAN, while M and N are the denotions of several matrices' calculation results.

To protect X, we introduce a novel technique and let $X' = PXQ$, wherein P and Q are $m \times m$ and $n \times n$ matrices respectively. Here, P is a diagonal matrix with its diagonal element randomly set to $\pm k (k \neq 0)$, satisfying $P^T P = k^2 I$. It is of great importance that another matrix Q is a random dense invertible matrix, rather than a diagonal one, which is quite different from the encryption algorithm proposed in [3]. Surprisingly, we can complete this step with $O(mn)$ time, mainly due to the generation rules of matrix Q.

To this end, we generate two $n \times n$ matrices separately, denoted by Q_1 and Q_2, which satisfy $Q = Q_1 + Q_2$. Q_1 is a diagonal matrix and q_i is the ith diagonal element (from the top) of Q_1. We set $q_i \neq 0$ for each $i \in [1, n]$. However, Q_2 is a dense matrix which could be represented by $Q_2 = UV$. For U, we randomly generate an $n \times 1$ matrix such that $U = (u_1, u_2, \ldots, u_n)^T$ with $u_i \neq 0$ for each $i \in [1, n]$. For V, we randomly generate $(n-1)$ nonzero numbers such that $V = (1, v_2, v_3, \ldots, v_n)$ and for each $i \in [2, n]$, we have $v_i \neq 0$. Therefore, $Q_2 = [U, v_2 U, v_3 U, \ldots, v_n U]$, that is to say, the first column of Q_2 equals to U, and the ith column of Q_2 equals to the ith element in V multiplying U for each $i \in [2, n]$.

So far, we already have $X' = PXQ$. Considering that $X'^T X'$ requires expensive computation, we send X' to the cloud server to compute $X'^T X'$. Thus, the client obtains $E = X'^T X'$ without calculating it by itself. Now we have $E = X'^T X' = (PXQ)^T (PXQ) = k^2 Q^T (X^T X) Q$. The expression enclosed in parentheses is much similar to A, thus the left side of $X^T X$ could be M and the right side could be N as we mentioned above. We set $F = k^2 Q^T (\lambda I) Q$, adding it to E, then we can obtain $E + F = k^2 Q^T (X^T X + \lambda I) Q$, which can be denoted by A'. If we directly send A' to the cloud server, one problem is that cloud server could know the value of F using $A' - E$. To prevent the cloud server from learning anything about these secret keys, we let $E' = HEJ$ and redefine the F by $F = k^2 H Q^T \lambda I Q J$. Eventually, we have the following equations,

$$A' = E' + F = k^2 H Q^T (X^T X + \lambda I) Q J = MAN. \tag{9}$$

Evidently, $M = k^2 H Q^T$ and $N = QJ$.

We then discuss our scheme to perturb b. If we add a random $n \times 1$ vector to β in Eq. (6), then we have

$$A(\beta + r) = A\beta + Ar = b + Ar. \tag{10}$$

Hence, we use M and Ar to encrypt b, and there will be $b' = M(b + Ar)$ in our encryption scheme. This yields

$$\beta' = A'^{-1} b' = (MAN)^{-1} M(b + Ar). \tag{11}$$

If M and N are nonsingular, based on Eqs. (8) and(11), we have

$$\beta' = N^{-1} A^{-1} M^{-1} M(b + Ar) = N^{-1}(\beta + r). \tag{12}$$

Thus, while the cloud server computes β' and sends it back to the client, the client can recover the answer to the original RR problem with $\beta = N\beta' - r$.

Clearly, it is of great importance that M and N must be invertible. The explanation on how to ensure the invertibility of M and N is shown in Section 5.1.

4.2. Our answer verification algorithm

Our efficient and secure scheme for answer verification is introduced in this subsection. The purpose of verification is to guarantee that β', the answer from the cloud server, is indeed equal to $A'^{-1} b'$. Due to outsourcing computation involving multiple matrix multiplication and inversion, it is of great importance to efficiently check the answer. In our scheme, the answer β' is correct if and only if

$$A'\beta' = b', \tag{13}$$

which is equivalent to

$$A'\beta' - b' = 0. \tag{14}$$

Note that the left side of Eq. (14) can be computed within $O(mn)$ time.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

4.3. Details of our proposed scheme

Detailed introduction of our proposed scheme to securely and efficiently outsource RR computation is as follows.

- **KeyGen**$(m, n) \longmapsto \{K\}$. The client firstly generates a random number $k > 0$, a random $n \times 1$ vector r, and $9n - 6$ nonzero random numbers, including the elements of three matrices with the same construction, i.e, Q, H, J. Obviously, each of these three matrices requires $3n - 2$ random numbers. Taking Q, for example, the client needs to generate $d_{q1}, d_{q2}, \ldots, d_{qn} \cup u_{q2}, u_{q3}, \ldots, u_{qn} \cup v_{q2}, v_{q3}, \ldots, v_{qn}$. Then the client randomly generates three nonzero numbers u_{q1}, u_{h1}, u_{j1}, which satisfy $u_{q1} \neq -d_{q1}(1 + \sum_{i=2}^{n} \frac{u_{qi}v_{qi}}{d_{qi}})$, $u_{h1} \neq -d_{h1}(1 + \sum_{i=2}^{n} \frac{u_{hi}v_{hi}}{d_{hi}})$ and $u_{j1} \neq -d_{j1}(1 + \sum_{i=2}^{n} \frac{u_{ji}v_{ji}}{d_{ji}})$ respectively. Note that the reason of the generation rules will be discussed in Section 5.1. These random numbers are utilized to set matrices P, Q, H, J in the following manner. P is a random $m \times m$ diagonal matrix with its diagonal elements set to k or $-k$. Due to Q, H, J have the same construction, we only illustrate the detailed generation step of Q. Here, Q_1 is an $n \times n$ diagonal matrix with its diagonal elements in i-th row set to d_{qi} for each $i \in [1, n]$. $U_Q = (u_{q1}, u_{q2}, \ldots, u_{qn})^T$, and $V_Q = (1, v_{q2}, v_{q3}, \ldots, v_{qn})$. Then, the client's secret key is $K = \{r, P, Q_1, U_Q, V_Q, H_1, U_H, V_H, J_1, U_J, V_J\}$, which will be privately kept by the client. Note that each key is used to encrypt only one RR problem.
- **RREnc** $(\Phi, K) \longmapsto \{\Phi'\}$. The client perturbs the original input in two phases. First, the client encrypts X by the following equation

$$X' = PXQ_1 + (PXU_Q)V_Q \tag{15}$$

and submits it to the cloud server. Second, after receiving the result $E = X'^T X'$, the client utilizes the method shown in the following.

$$\begin{cases} A' = (H_1E + U_HV_HE)(J_1 + U_JV_J) \\ \quad\quad +\lambda k^2(H_1 + U_HV_H)(Q_1 + U_QV_Q)^T(Q_1 + U_QV_Q)(J_1 + U_JV_J) \\ b' = k^2(H_1 + U_HV_H)(Q_1 + U_QV_Q)^T[X^Ty + X^T(Xr) + \lambda r] \end{cases}$$

Note that $(H_1E + U_HV_HE)(J_1 + U_JV_J)$ is utilized to encrypt E into $E' = HEJ = (H_1E + U_HV_HE)(J_1 + U_JV_J)$, and $\lambda k^2(H_1 + U_HV_H)(Q_1 + U_QV_Q)^T(Q_1 + U_QV_Q)(J_1 + U_JV_J)$ aims at computing the value of F.
- **RRSolve** $(X', \Phi') \longmapsto \{E, \beta'\}$. In our scheme, the client only requires two rounds communicating with the cloud server. Once receiving the data X', the cloud server computes the result $E = X'^T X'$ and sends it back instantly. Once receiving the encrypted problem $\Phi' = \{A', b'\}$, the cloud server will solve the RR problem and answer the client with the result β'.
- **ResultVerify** $(\beta') \longmapsto (1/0)$. While receiving the encrypted answer β', the client performs $A'\beta' - b'$ to check whether the answer is correct. If β' is valid, the client will output 1 and perform the next algorithm **ResultDec**. Otherwise, the client will output 0 and consider that cloud server misbehaved.
- **ResultDec** $(\beta') \longmapsto \{\beta\}$. If β' is checked to be valid, the client obtains the answer to the original RR problem via

$$\beta = (Q_1 + U_QV_Q)(J_1 + U_QV_Q)\beta' - r. \tag{16}$$

5. Theoretical analysis

In this section, we present a comprehensive theoretical analysis about our proposed scheme in four aspects: correctness, input/output security, answer checkability and the computation/communication overheads.

5.1. Correctness analysis

At first, we will give the explanation on the invertibility of M and N. In Section 4.1, we already discussed that $M = k^2HQ^T$ and $N = QJ$. Therefore, our task is to prove that the equivalent equation HQ^T and QJ are invertible. Because the three matrices H, Q, J are identical essentially, we only need to just prove Q is nonsingular and the other two will have the same property.

To deal with this problem, we utilize det (Q), i.e, the determinant of Q. Before computing det (Q), we introduce two significant properties about the matrix determinant in Lemmas 1 and 2.

Lemma 1 [17]**.** *For any square matrix M, adding a multiple of one column to another does not change the value of its determinant.*

Lemma 2 [18]**.** *Suppose that C, D, F, G are matrices of dimension of $a \times a$, $a \times b$, $b \times a$, and $b \times b$, respectively. The square matrix G is invertible. Then,*

$$\begin{bmatrix} C & D \\ F & G \end{bmatrix} = det(G) * det(C - DG^{-1}F). \tag{17}$$

Based on the two lemmas above, Zhu et al. [11] have proved the following Theorem 1.

Theorem 1 [11]. *Suppose $Q = Q_1 + Q_2$. Here, Q_1 is a diagonal matrix, and d_i represents the diagonal element in its ith row (from the top). $Q_2 = U_Q V_Q$ where $U_Q = (u_{q1}, u_{q2}, \ldots, u_{qn})^T$, $V_Q = (1, v_{q2}, \ldots, v_{qn})$. For each $i \in [1, n]$, it has $d_i \neq 0$, $u_{qi} \neq 0$ and $v_{qi} \neq 0$. Then,*

$$det(Q) = \left(1 + \frac{u_{q1}}{d_1} + \sum_{i=2}^{n} \frac{u_{qi} v_{qi}}{d_i}\right) \prod_{j=1}^{n} d_j. \tag{18}$$

Therefore, $det(Q) = (1 + U_Q Q_1^{-1} V_Q) * det(Q_1)$.

We have set $d_j \neq 0$ for each $j \in [1, n]$. Hence,the matrix is invertible if and only if $1 + \frac{u_{q1}}{d_1} + \sum_{i=2}^{n} \frac{u_{qi} v_{qi}}{d_i} \neq 0$ based on Theorem 1. Thus, we can ensure the invertibility of Q, by generating $3n - 2$ nonzero random numbers as illustrated in Section 4.3, especially randomly setting nonzero u_{q1} to satisfy $u_{q1} \neq -d_{q1}(1 + \sum_{i=2}^{n} \frac{u_{qi} v_{qi}}{d_{qi}})$.

Since Q, H, J are all invertible, $M = k^2 H Q^T$ and $N = QJ$ are also invertible, by which our proposed RR outsourcing scheme can be correctly supported.

Next, we will prove our scheme is correct.

Theorem 2. *Our proposed outsourcing scheme is correct.*

Proof: According to Section 4.1, we have $A' = MAN$, $b' = M(b + Ar)$ and $A'\beta' = b'$. Since we have proved M and N are invertible, thus we have

$$\beta' = A'^{-1} b' = N^{-1} A^{-1} M^{-1} M(b + Ar) = N^{-1}(A^{-1} b + r)$$

We decrypt $\beta = N\beta' - r$ during **ResultDec**, thus,

$$\beta = N[N^{-1}(A^{-1} b + r)] - r = (A^{-1} b + r) - r = A^{-1} b.$$

It is evident that the expression of β is exactly identical to the RR estimation in Eq. (7). Consequently, in our scheme the client can correctly obtain the answer to the original RR problem, if both the client and cloud server operate follow our algorithms. This completes the proof.

5.2. Input/output security analysis

In our scheme, the cloud server receives X', A' and b' from the client. Besides, each secret key K is private to the client. In the following, we will analyse the privacy of X, A, and b.

5.2.1. Input security

Security of X. Our proposed method transforms the original input X into $X' = PXQ_1 + (PXU_Q)V_Q$. Thus, X is protected not only by a random diagonal matrix P, but also by a dense invertible matrix $(Q_1 + U_Q V_Q)$. Let x'_{ij} and x_{ij} represent the ith row, jth column element of X' and X respectively. Then we have $x'_{i,j} = \pm k(d_j x_{ij} + v_{qj} \sum_{t=1}^{n} u_{qt} x_{ij})$, for each $i \in [1, m]$, $j \in [1, n]$. Clearly, x'_{ij} is a linear combination of n elements in ith row of X. Considering the random additive and multiplicative factors, x_{ij} cannot be learned from x'_{ij}. Hence, the input matrix X can be properly protected from the cloud server.

Security of A. We protect A employing $A' = MAN$, similar to the encryption method of X. Here, we also have $A' = E' + F = k^2 H Q^T (X^T X + \lambda I) QJ$. On one hand, it is difficult for the attackers to learn A from MAN without knowing anything about the coefficient matrices M and N. On the other hand, the cloud server cannot obtain the F although it knows E, because we encrypt E again with $E' = HEJ$. Thus we preserve the A properly as well.

Security of b. The vector b is perturbed by $b' = M(b + Ar)$, where $M = k^2 H Q^T$. First, we protect each element of b with random additive approach, and then multiply them with coefficient matrix M. Thus, no attackers can obtain useful things of b from b', because it is of big difficulty to strip the random Ar from b'. Therefore, we can guarantee that all elements of b are well preserved.

5.2.2. Output security

Theorem 3. *Our scheme cannot learn useful information about β, which is the private output of the client.*

Proof: In our solution, the cloud server can compute $\beta' = N^{-1}(\beta + r)$, as shown in Eq. (12). Since N, r are private to the client, the cloud server cannot obtain the answer to the original problem. Therefore, this proof is completed.

5.3. Checkability analysis

In our proposed algorithm, the client verifies the answer β' by checking the equation $A'\beta' - b' = 0$. While there is some deviation in β', i.e, the received answer β' is not correct, it must result in $A'\beta' - r \neq 0$. Therefore, the client can detect such misbehavior of the cloud server via our answer verification algorithm.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

Table 1
Performance of secure ridge regression outsourcing scheme.

#	Benchmark	Directed time	Our method		Speedup(C)	Efficiency(S)
	Dimension	$t_d(ms)$	$t_s(ms)$	$t_c(ms)$	t_d/t_c	t_s/t_d
1	400×50	0.9000	0.3000	0.4000	1.8000	0.4444
2	800×200	8.8000	4.2000	3.8000	2.3158	0.4773
3	1000×400	21.100	11.400	12.600	1.6746	0.5403
4	1500×600	62.700	34.600	30.900	2.0291	0.5518
5	2000×800	113.00	60.300	55.500	2.0360	0.5336
6	3000×1000	229.90	111.30	108.90	2.1111	0.4841
7	4000×1000	289.40	144.00	135.60	2.1342	0.4976
8	5000×1200	474.80	262.20	242.20	1.9620	0.5522

5.4. Computation/communication overheads analysis

5.4.1. Computation complexity of client

During **KeyGen** algorithm, the client generates $(10n-5)$ random numbers and computes $-d_{j1} * (1 + \sum_{i=2}^{n} \frac{u_{ji} v_{ji}}{d_{ji}})$ three times to produce the secret key K. Thus, **KeyGen** takes $O(n)$ times.

Next, to encrypt the original data, the client needs to compute $X' = PXQ_1 + (PXU_Q)V_Q$, $A' = (H_1E + U_H V_H E)(J_1 + U_J V_J) + \lambda k^2 (H_1 + U_H V_H)(Q_1 + U_Q V_Q)^T (Q_1 + U_Q V_Q)(J_1 + U_J V_J)$, and $b' = k^2 (H_1 + U_H V_H)(Q_1 + U_Q V_Q)^T [X^T y + X^T (Xr) + \lambda r]$. Fortunately, utilizing the diagonal matrices and dense matrices makes these computations completed within $O(mn)$ time.

For verifying the answer of the cloud server, the client will compute $(A'\beta' - b)$, which requires $O(n^2)$ time. As n is usually less than m, the execution time still within $O(mn)$.

At last, the decryption algorithm $\beta = N\beta' - r$ needs $O(n^2)$ time, since we use $Q[J_1\beta' + U_J(V_J\beta')] - r$ to reduce the computation complexity efficiently.

Overall, the computation complexity of the client is just within $O(mn)$ time.

5.4.2. Communication overheads

In our scheme, the client sends X', A', b' to cloud server, and cloud returns E and β' to the client, which requires be implemented twice. The dimension of X', A', b' and β' are just the same as those of X, A, b and β respectively. Besides, the dimension of E is $n \times n$. Therefore, if each number is b_0 bits, our whole communication overheads are $(mn + 2n^2 + 2n)b_0$ bits.

6. Performance evaluation

In this section, we describe our experimental evaluation of our proposed algorithm. All programs are implemented in Matlab with a version of R2014b, simulating a client and a cloud server locally. Namely, both sides are conducted on an Intel(R) Core(TM) i7-4790 3.60 GHz computer with 4 GB RAM running Windows 10. Since the objective of experiment is to evaluate the running time of our scheme, we implement the schemes on synthetic database whose dimensions vary from 400×50 to 5000×1200. The experiment results in Table 1 are the average time of 1000 runs.

As illustrated in Section 4.3, the client needs to deal with four algorithms including **KeyGen, RREnc, Resultverify** and **ResultDec**. Here, we use **RREnc1** and **RREnc2** to denote two phases of **RREnc**. Specifically, **RREnc1** denotes the step to encrypt X into X', and **RREnc2** represents the step to compute A' and b'. Since computation steps of E', F and b' are irrelevant to each other, we compute the three values in parallel by leveraging a parallel tool "parpool" in Matlab R2014b.

Table 1 shows detailed performance of two methods: the directed method and our method. The directed method represents the client computing ridge regression locally. Besides, We let t_d denote the directed time; let t_s denote the time consumed in the cloud server in our algorithm; let t_c denote the running time of the client to conduct **KeyGen, RREnc1, RREnc2, ResultVerify** and **ResultDec** steps in our scheme. For each kind of dimension, we implement 1000 tests in order to measure the average execution time of the client and the cloud server respectively. We can see from the table that the average execution time of all three parts increase rapidly with the increase of m and n. Among these three parts, t_c and t_s are much faster than the directed time t_d. Here, we utilize t_d/t_c to illustrate the speedup of the client between our scheme and the directed method. Note that this value should be theoretically greater than 1 which means that there exists a considerable performance increase. The value varying between 1.8 to 2.4 indicates that our scheme indeed helps in saving the computing time. The other ratio, we have considered, is the cloud server's efficiency, and it is denoted by t_s/t_d. Apparently this index represents how much computing workload is shared by the server. The value of t_s/t_d around 0.5 reveals that the server helps reduce half of the burden of computing.

Fig. 2 shows the average execution time of the directed method and our method. We can see that the directed method needs the most execution time under all kinds of dimensions. In our method, the execution time in the client and in the cloud server are almost equivalent when $m \leq 3500$. Then, running time of the client is shorter than that of the cloud server while $m > 3500$. Generally, our method has better performance in large-scale datasets than small ones.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

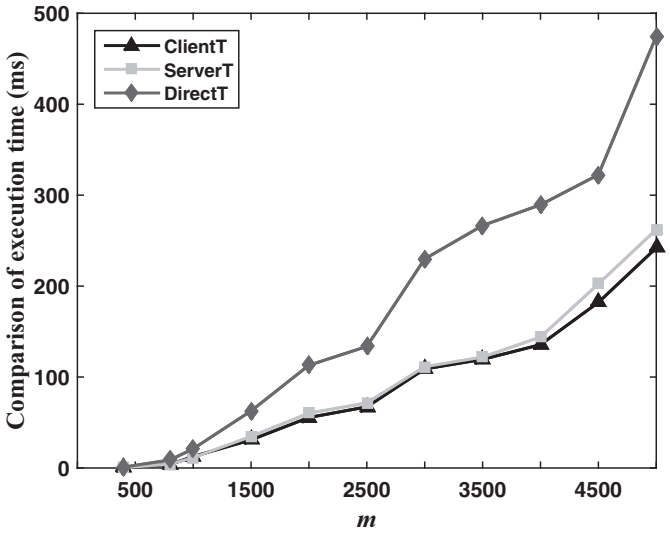

Fig. 2. Average execution time (ms) under direct method and our method.

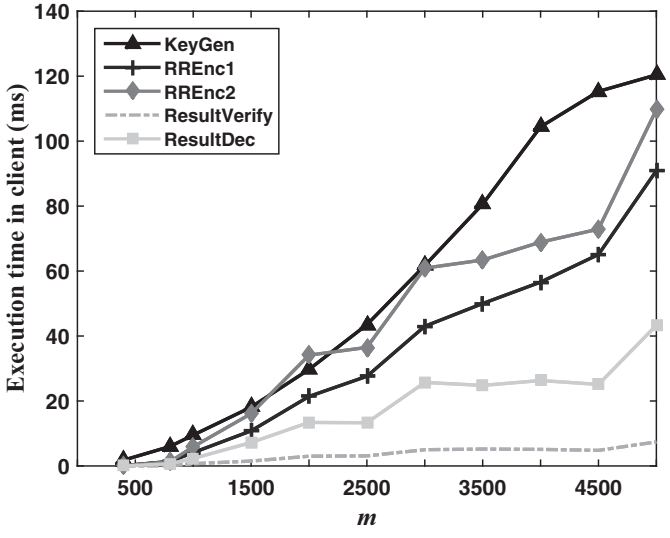

Fig. 3. Execution time (ms) of five stages in client.

Fig. 3 presents that the execution time of five stages in the client. As we can see, the execution time of all five stages goes up rapidly with the increase of dimension. Evidently, the slop of **KeyGen, RREnc1** and **RREnc2** are greatest which suggests that the required time of these three steps are more sensitive to the change of dimension. Besides, **KeyGen, RREnc1** and **RREnc2** also have the longest execution time. By contrast, **ResultVerify** and **ResultDec** consume the shorter time. It suggests that we should try to reduce the execution time of **KeyGen, RREnc1** and **RREnc2** to improve the performance in the future work.

7. Conclusions

In this paper, we proposed an algorithm for secure and efficient ridge regression outsourcing on encrypted wearable data. Our proposed scheme can well preserve both input privacy and output privacy in an efficient manner, and support the client to fast verify the correctness of final results. Theoretical analysis and extensive experiments are leveraged to confirm the correctness, input/output privacy, and checkability of our solution. For the future work, we will devote to a highly-efficient ridge regression outsourcing scheme with provable security.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

Acknowledgments

This work is partly supported by the National Key Research and Development Program of China (No. 2017YFB0802304), the Natural Science Foundation of China (No. 61602240), the Natural Science Foundation of Jiangsu Province of China (No. BK20150760), the Research Fund of Guangxi Key Laboratory of Trusted Software (No. kx201611), and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] Sundareswaran S, Squicciarini A, Lin D. Ensuring distributed accountability for data sharing in the cloud. IEEE Trans Depend Secure Comput 2012;9:556–68.
[2] Tang Y, Lee PPC, Lui JCS, Perlman R. Secure overlay cloud storage with access control and assured deletion. IEEE Trans Depend Secure Comput 2012;9:903–16.
[3] Chen X, Huang X, Li J, Ma J, Lou W, Wong DS. New algorithms for secure outsourcing of large-scale systems of linear equations. IEEE Trans Inf Forensics Secur 2015;10(1):69–78.
[4] Ren K, Wang C, Wang Q. Security challenges for the public cloud. IEEE Internet Comput 2012;16:69–73.
[5] Wang C, Cao N, Ren K, Lou W. Enabling secure and efficient ranked keyword search over outsourced cloud data. IEEE Trans Parallel Distrib Syst 2012;23:1467–79. doi:10.1109/TPDS.2011.282.
[6] Chen F, Xiang T, Lei X, Chen J. Highly efficient linear regression outsourcing to a cloud. IEEE Trans Cloud Comput 2014;2:499–508.
[7] Nikolaenko V, Weinsberg U, Ioannidis S, Joye M, Boneh D, Taft N. Privacy-preserving ridge regression on hundreds of millions of records. In: IEEE Symposium on security and privacy; 2013. p. 334–48.
[8] Hu S, Wang Q, Wang J, Chow SSM, Zou Q. Securing fast learning! ridge regression over encrypted big data. In: 2016 IEEE Trustcom/BigDataSE/ISPA; 2016. p. 19–26.
[9] Atallah MJ, Pantazopoulos K, Rice JR, Spafford EE. Secure outsourcing of scientific computations. Trends Softw Eng 2002;54:215–72.
[10] Chen X, Li J, Ma J, Tang Q, Lou W. New algorithms for secure outsourcing of modular exponentiations. IEEE Trans Parallel Distrib Syst 2014;25(9):2386–96.
[11] Zhu Y, Wang Z, Qian C, Wang J. On efficiently harnessing cloud to securely solve linear regression and other matrix operations. In: 2016 IEEE/ACM 24th international symposium on quality of Sservice (IWQoS), 2; 2016. p. 1–2. doi:10.1109/IWQoS.2016.7590402.
[12] Chen X, Li J, Huang X, Ma J, Lou W. New publicly verifiable databases with efficient updates. IEEE Trans Depend Secure Comput 2015;12(5):546–56.
[13] Chen X, Li J, Weng J, Ma J, Lou W. Verifiable computation over large database with incremental updates. IEEE Trans Comput 2016;65(10):3184–95.
[14] Rosario Gennaro BP, Craig Gentry. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: Proceedings of the international cryptology conference (CRYPTO); 2010. p. 465–82.
[15] Chung K-M, Kalai Y, Vadhan S. Improved delegation of computation using fully homomorphic encryption. Springer Berlin Heidelberg; 2010. p. 483–501.
[16] Barbosa M, Farshim P. Delegatable homomorphic encryption with applications to secure outsourcing of computation. Springer Berlin Heidelberg; 2012. p. 296–312.
[17] Leon S. Linear algebra with applications (8th edition). Person Prentice Hall; 2009. p. 94–5. ISBN 978-3-642-33167-1.
[18] Brookes M. The matrix reference manual. Imperial College London; 2011. p. 256–70.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019

Xinshu Ma received her B.S. degree in Nanjing University of Aeronautics and Astronautics in 2017. She is currently pursuing the M.S. degree at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. Her research interests include cloud computing and applied cryptography.

Youwen Zhu received his PhD degree in computer science from University of Science and Technology of China in 2012. He is currently an associate professor at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include information security and privacy in cloud computing.

Xingxin Li received the B.S. degree in 2014 and master degree in 2017, both from Nanjing University of Aeronautics and Astronautics. He is currently pursuing the PhD. degree at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include cloud computing and data privacy.

Please cite this article as: X. Ma et al., An efficient and secure ridge regression outsourcing scheme in wearable devices, Computers and Electrical Engineering (2017), http://dx.doi.org/10.1016/j.compeleceng.2017.07.019