Privacy-preserving Diverse Keyword Search and Online Pre-diagnosis in Cloud Computing

Xiangyu Wang, Jianfeng Ma, Yinbin Miao, Ximeng Liu, and Ruikang Yang

Abstract—With the development of Mobile Healthcare Monitoring Network (MHMN), patients' data collected by body sensors not only allows patients to monitor their health or make online pre-diagnosis but also enables clinicians to make proper decisions by utilizing data mining technique. However, sensitive data privacy is still a major concern. In this paper, we propose practical techniques for searching and making online pre-diagnosis over encrypted data. Firstly, we propose a new Diverse Keyword Searchable Encryption (DKSE) scheme which supports multi-dimension digital vectors range query and textual multi-keyword ranked search to gain a broad range of applications in practice. In addition, a framework called PRIDO based on the DKSE is designed to protect patients' personal data in data mining and online pre-diagnosis. According to the PRIDO framework, we achieve privacy-preserving naïve Bayesian and decision tree classifiers and discuss its potential applications in actual deployments. Security analysis proves that patients' data privacy can be well protected without loss of data confidentiality, and performance evaluation demonstrates the efficiency and accuracy in the diverse keyword search, data mining, and disease pre-diagnosis, respectively.

Index Terms—Privacy-preserving; online pre-diagnosis; searchable encryption; data mining.

1 Introduction

In recent years, Mobile Healthcare Monitoring Network (MHMN) [1] has gained much attention in both academic and industrial fields with the development of body sensor network, mobile communication technology, and cloud computing. Compared with the traditional medical system, MHMN can monitor patients' health in real-time without affecting their daily life. Moreover, massive personal medical data are stored for patient condition analysis as well as medical research. Due to the huge requirements of data storage and processing, outsourcing personal data to the cloud is a promising way to improve the processing speed and solve the excessive storage overhead [2].

In MHMN systems, patients' personal data are collected by sensors per second and uploaded to the cloud server as multi-dimension vectors, cloud server stores the personal data as well as sends monitoring information to the hospital when the real-time data is abnormal. Hospital users (i.e., doctors, etc.) may query some samples which contain certain textual keywords or digital keywords in certain ranges for disease diagnosis or medical research. For example, a certain hospital user may query all samples with textual keywords 'cancer; diabetes' and digital vector $\{$'age'$\in [30,50]$, 'blood sugar' $\in [4,8]$, 'heart rhythm' $\in [70,80]\}$. Besides, the potential value of massive medical data has attracted considerable interests recently, for example, valuable results in diagnosis model can be yield with large-scale aggregation analysis of personal medical data. The cloud server can build a diagnosis model using data mining technology over massive data, so that hospital users or pre-diagnosis users upload medical data (i.e., age, blood pressure, blood sugar, etc.) to the cloud for diagnosis.

However, how to provide accurate data query, health monitoring, data mining, and online diagnosis services without revealing personal data is still a big challenge. Former academics [3], [4] have researched the securities of storage and query of Personal Health Records (PHR) in cloud computing. Order-preserving encryption based scheme [3] can achieve digital vectors range query effectively, but it cannot support the textual keyword search. To support both digital vectors range query and textual keyword search, Li *et al.* [4] proposed a security query program based on both Searchable Encryption (SE) and Attribute-Based Encryption (ABE), but this system cannot support ranked search. Moreover, these privacy-preserving mechanisms make data monitoring and analysis impossible in practice. To this end, some researchers [5], [6], [7], [8], [9] introduce Homomorphic Encryption (HE) and make some secure outsourcing computing protocols based on HE to design secure health monitoring or online pre-diagnosis schemes. Unfortunately, these protocols bring in high communication and computation overhead, and all these schemes cannot support other features of MHMN (i.e., multi-dimension vector range query and textual keyword search, etc.).

Contribution. In this paper, we show how to efficiently achieve the desired features of MHMN without loss of data privacy. Specifically, the main contributions of this paper are listed as follows:

- *Diverse keywords search*. We propose a new **D**iverse **K**eyword **S**earchable **E**ncryption (DKSE) scheme which allows legitimate users (i.e., sensor users, hospital users, etc.) to issue both multi-dimension digi-

- *X. Wang, J. Ma, Y. Miao and R. Yang are with the School of Cyber Engineering and the Shaanxi Key Laboratory of Network and System Security, Xidian University, Xi'an 710071, China. E-mail: XYWang_Xidian@163.com.*
- *X. Liu is with the School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore, and College of Mathematics and Computer Science, Fuzhou University. Email: xmliu@smu.edu.sg.*

tal vectors range query and textual multi-keywords ranked search on encrypted cloud data.

- *Data mining and pre-online diagnosis*. Based on the DKSE, we construct a **PRI**vacy-preserving **D**ata mining and **O**nline pre-diagnosis (PRIDO) framework. In our PRIDO, the collected encrypted personal medical data can be used by the cloud to train a diagnosis model. Then, the cloud server can use the trained model to diagnose patient's diseases according to his/her personal data in a privacy-preserving way. Experimental results using real-world datasets show that the accuracy of online pre-diagnosis in our work is similar to that of the plaintext system.
- *Efficiency*. The experimental results show that our DKSE is efficiency in practice, and the running time of data mining and online pre-diagnosis in our PRIDO are comparable with the original algorithm used in plaintext.
- *Privacy*. Security analysis shows that our work is secure against the known-plaintext attack, which satisfies the privacy requirements of most high-performance searchable encryption schemes [10], [11], [12], [13].

Compared with the preliminary version [14] of this paper, this journal version further supports textual multi-keyword ranked search. This version also discusses how to achieve more classifiers apart from naïve Bayesian classifier [15] based on the proposed DKSE. Moreover, we give formal security definitions and prove the security of our work. Besides, to evaluate the newly added textual multi-keywords ranked search and classifier, we improve the experiments with the new real-data set to get close to the real situation. Finally, we make a thorough comparison with the most recent works and describe the related work to better evaluate our work.

Organization. The remainder of this paper is organized as follows: In Section 2, we introduce some basic knowledge used in our work such as the BM25 ranking model, naïve Bayesian classifier and decision tree classifier. In Section 3, we present the system model and clarify the privacy requirements. We propose our new searchable encryption scheme in Section 4. Next, we construct the framework of data mining and online pre-diagnosis based on DKSE in Section 5 and discuss how to achieve naïve Bayesian classifier and decision tree classifier. Then, we analyze the privacy and performance of the proposed work in Section 6 and Section 7, respectively. Finally, we give the related work in Section 8 and conclude our paper in Section 9.

2 Preliminaries

In this section, we give a brief review of the BM25 ranking model [16], naïve Bayesian classifier [15] and decision tree classifier [17] used in our work.

2.1 BM25 Ranking Model

BM25 [16] is a famous ranking model based on the probabilistic ranking principle [16]. Consider an unstructured document $\bar{d}$ belonging to a collection F. We regard the document as a keywords vector $\bar{d} = (\tilde{d}_1, ..., \tilde{d}_j, ...\tilde{d}_m)$, where $\tilde{d}_j$ denotes the keyword frequency of the j-th keyword in $\bar{d}$ and m is the total number of keywords in the vocabulary. In order to score such a document against a query, most ranking functions define a keyword weighting function $w_j(\bar{d})$, which exploits keyword frequency as well as other factors such as the document's length and collection statistics.

For one-time retrieval, and ignoring any repetition of keywords in the query, BM25 keyword weighting function can be simplified as

$$w_j(\bar{d}) = \frac{(k_1+1)\tilde{d}_j}{k_1((1-b)+b\frac{dl}{avgdl})+\tilde{d}_j} \log \frac{N_{doc}-df_j+0.5}{df_j+0.5},$$

where df_j denotes the document frequency of keyword d_j, dl denotes the document length, $avgdl$ denotes the average document length across the collection F, N_{doc} denotes the number of documents in the collection, and k_1 and b are free parameters. The document BM25 score is then obtained by adding the document keyword weights of keywords matching the query $Q = \{q_1, q_2, ..., q_m\}$ as

$$BM(\bar{d}, Q) = \sum_{j=1}^{n} w_j(\bar{d}) \cdot q_j.$$

2.2 Naïve Bayesian Classifier

Naïve **B**ayesian (NB) classifier [15] is a classic classifier, which can be used in text classification, medical diagnosis, and other practical applications. Here, we briefly review basic algorithms of the naïve Bayesian classifier as follows. Suppose that there are $NClass$ classes denoted as C_1, C_2, ... , C_{NClass}. Each sample in predicting has h attributes A_1, ... , A_h, which is expressed as h-dimensional vector $\vec{Y} = \{Y_1, ..., Y_h\}$. The classifier needs to predict which kind of class the sample $\vec{Y}$ most likely belongs to. The possibility of $\vec{Y}$ belongs to $C_i (0 < i \leq h)$ can be calculated by Bayes's theorem shown in Eq. 1.

$$P(C_i|\vec{Y}) = \frac{P(\vec{Y}|C_i)P(C_i)}{P(\vec{Y})}. \tag{1}$$

We can see that $P(\vec{Y})$ is the same for all classes, only $P(\vec{Y}|C_i)P(C_i)$ needs to be maximized. In order to calculate $P(\vec{Y}|C_i)P(C_i)$, we can calculate the conditional probability shown in Eq. 2 on condition that values of all are independence of each other. If a feature attribute is discrete, the probabilities $P(Y_1|C_i)$, $P(Y_2|C_i)$, ... , $P(Y_h|C_i)$ can be easily obtained from the training set.

$$P(\vec{Y}|C_i) \approx \prod_{k=1}^{h} P(Y_k|C_i). \tag{2}$$

If a feature attribute is continuous, the feature attribute values in $\vec{Y}$ is regarded as follow the Gaussian distribution shown in Eq. 3. Therefore, the conditional probability estimates for each continuous feature attribute in each class can be calculated by Eq. 4.

$$g(x, \eta, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\eta)^2}{2\sigma^2}}. \tag{3}$$

$$P(Y_k|C_i) = g(Y_k, \eta_{C_i}, \sigma_{C_i}). \tag{4}$$

2.3 Decision Tree Classifier

Decision Tree (DT) [18] is also a classic classifier, which is a tree structure consisting of *decision node* and *leaves*. A decision node specifies a *test* over one of the attributes. Each possible outcome of the test presents a child node or a leave. The test on a continuous attribute has two possible outcomes, $A_i \leq t$ and $A_i > t$, where t is a value determined at the node. Repeat the test until reaching a leaf, the class specified at the leaf is the class predicted by the decision tree. In this paper, we use the classic decision construction algorithm C4.5 algorithm to construct the decision tree with a divide and conquers strategy. In C4.5, each node in a tree is associated with a set of cases. Also, cases are assigned weights to take into account unknown attribute values. Let T be the set of cases associated at the node. The weighted frequency $freq(C_j, T)$ is computed of cacses in T whose class is C_i for $i \in [1, NClass]$. If all cases in T belong to the same class C_j, then the node is a leaf, with associated class C_j (respectively, the most frequent class). The classification error of the leaf is the weighted sum of the cases in T whose class is not C_j. If T contains cases belonging to two or more classes, then the information gain of each attribute is calculated. For discrete attributes, the information gain is relative to the splitting of cases in T into sets with distinct attribute values. If A_i is discrete, and $T_1, T_2, ..., T_s$ are the subsets of T consisting of cases with distinct known value for attribute A_i, the information gain can be calculated as

$$Gain = Info(T) - \sum_{i=1}^{s} \frac{|T_i|}{|T|} \times Info(T_i), \tag{5}$$

where

$$Info(T) = -\sum_{j=1}^{Nclass} (\frac{freq(C_j, T)}{|T|} \times \log_2 \frac{freq(C_j, T)}{|T|})$$

is the entropy function. The attribute of maximum *Gain* will be selected as the best one to split the dataset into p partitions.

3 Problem Formulation

In this section, we introduce the system model, threat model, and privacy requirements, respectively.

3.1 System Model

In this system, we focus on how to meet the actual needs of MHMN without leaking patients' private information, such as multi-dimension digital vectors range query and textual multi-keyword ranked search, data mining, and online pre-diagnosis. The system model of our work involves four main entities, namely Sensors User (SU), Cloud Server (CS), Hospital User (HU), and Pre-diagnosis User (PU), which are demonstrated in Fig. 1.

- SU collects personal data in real-time with sensors and sends the encrypted data to CS. In addition, SU shares the data encryption key with HU so that HU can query and obtain data from CS.
- CS which has unlimited storage space and computation abilities, can provide data storage and search services for SU and HU. In addition, CS can perform calculations over stored cloud data and provide computation services for PU.
- HU has access to query data stored in CS. Moreover, HU can provide training trapdoor for CS to train classifiers and send encrypted medical data to CS for clinical decision support.
- PU authorized by HU can obtain online pre-diagnosis results through sending encrypted personal data to CS.

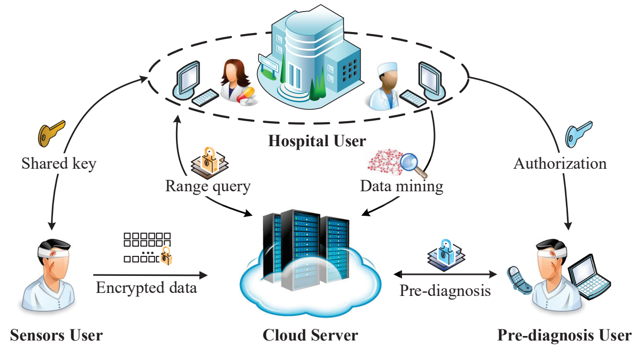


Fig. 1. System model.

3.2 Threat Model

In our threat model, we assume that SU and CS are trustable, which can provide data collecting and encrypt, respectively. The CS is considered to be an *honest-but-curious* third-party which is interested in SU's historical personal data and the classifier trained from SU's data but honestly follows the protocols established in the system. PU is also considered as an *honest-but-curious* entity that is interested in SU's historical personal data. HU is considered as a trusted entity that has access to SU's historical data stored in CS, and HU is the owner of the classifier trained from SU's historical data. Besides, authorized parties cannot sell or leak their secret keys to unauthorized ones, the entities in the system do not collude with each other. Moreover, an external adversary is interested in all data transmitted in the system by eavesdropping.

Based on what information the CS or adversary knows, we consider two attack models considered in [19] with different attack capabilities as follows.

1) *Level-1: Ciphertext-only Attack (COA) [20]*. The adversary is only able to know the encrypted personal data or search queries.
2) *Level-2: Known-plaintext Attack (KPA) [20]*. Apart from the ciphertext, the adversary is supposed to gain a set of tuples in dataset or queries and he/she knows the corresponding encrypted values of those tuples.

These two models are widely used in searchable encryption works [10], [11], [12], [13], which satisfy the requirements of data privacy in high-performance scenarios. The KPA attack is more powerful than the COA attack. If an encryption scheme resists the KPA attack, it resists the COA attack as well.

TABLE 1
Notation descriptions

Notations	Descriptions
I_j	Search index of P_j
F_j	Document corresponding to I_j
$E_K(\cdot)$	Encrypt by AES with secret key K
T_s	Trapdoor used to search
Q_R	Query range set
Ref_t	Training trapdoor
D_{tj}	Transformed personal data
$SK = \{S, M_1, M_2\}$	Secret key
$P_j = \{d_1, ..., d_i, ..., d_n\}$	Personal data
$R_t = \{r_1, ..., r_i, ..., r_n\}$	Divide parameter set
$\mathcal{S} = \{s_1, ..., s_i, ..., s_n, \gamma\}$	Randomization parameter set
$Q = \{q_1, ..., q_i, ..., q_n\}$	Search query
$\mathcal{W} = \{W_{w_1}, W_{w_2}, ..., W_{w_m}\}$	Textual keywords dictionary
$Ref = \{c_1, ..., c_i, ..., c_n\}$	Training reference vector
$P_U = \{p_{U_1}, p_{U_2}, ..., p_{U_n}\}$	Personal data of PU (HU)
$D_{tU} = \{a'_{U_1}, ..., a'_{U_n}\}$	Tansformed data of PU (HU)

3.3 Privacy Requirements

In our system model, SU's personal data contain confirmed SU's diseases and some sensitive personal information. These data can be used to train the classifier. During the data querying and data mining processes, SU's data cannot be directly exposed to untrusted parties; otherwise, SU will not provide its own data to the other parties due to personal data privacy leakage. HU authorizes CS to train the classifier by using SU's personal data and then authorizes the CS to make an online diagnosis. The classifier is of great value and cannot be leaked to other untrusted parties during training and online diagnosis. In addition, HU or PU will upload personal data to CS during online diagnosis, PU's data and the diagnosis results are highly sensitive and cannot be directly exposed or leaked to untrusted parties. According to the above threat models, to ensure the data privacy of each entity in the system, the following privacy requirements should be satisfied in our work.

1) *Index (trapdoor) confidentiality.* Proposed schemes should guarantee that the CS or adversary cannot obtain the content of indexes (trapdoors) from the encrypted indexes.
2) *Keyword privacy.* Proposed schemes should generate secure trapdoors to avoid the keywords leakage.
3) *Trapdoor unlinkability.* In each search process, indexes and trapdoors (i.e., search queries) are exposed to the CS. The trapdoors should be randomized so that the same query is different in different search processes. Furthermore, the CS cannot infer the relationship between these trapdoors.
4) *Privacy in data mining and online pre-diagnosis.* In the data mining process, CS uses stored personal data to train the classifier. In this process, SU's personal data privacy should be guaranteed. In addition, since the trained classifier is owned by HU, the classifier should not be used by CS or any unauthorized users to get the correct diagnosis results. In the online pre-diagnosis process, the authorized PU sends his or her personal data to CS for online diagnosis. PU's personal data should be protected.

4 Privacy-preserving Diverse Keyword Search

In this section, we first propose a **D**iverse **K**eyword **S**earchable **E**ncryption (DKSE) scheme to meet the practical needs of digital vector range query and textual multi-keyword ranked search. In order to achieve DKSE, we improve the **A**symmetric **S**calar-product-**P**reserving **E**ncryption (ASPE) [19] which can encrypt two vectors and compute their scalar product confidentially as our foundation. The detail of ASPE will be described together with our DKSE as follows. The key notations used in this paper are listed in TABLE 1. In DKSE, SU's personal data are collected by sensors per second as multi-dimension vectors, and encrypted by SU before uploading to CS. CS provides diverse keyword search services for HU. DKSE contains five algorithms, namely **KeyGen**, **DataEnc**, **TrapGen**, **QueryRangeGen** and **Search** as follows.

KeyGen(n, m, U): Given the dimensions of SU's personal data n, the size of keyword dictionary m, and noise parameter U, SU first randomly chooses a $(3n+m+U)$-dimension boolean vector S and two $(3n+m+U)\times(3n+m+U)$ invertible matrices $M_1, M_2 \in \mathbb{Z}$ as secret keys $SK = \{S, M_1, M_2\}$. Then, SU randomly chooses a symmetric encryption (e.g. AES) key K used to encrypt SU's document that contains sensitive information. Finally, SU sends $\{K, SK\}$ to HU via a secure channel.

DataEnc(SK, P_j, F_j, W, K): Given a personal data $P_j = \{d_1, ..., d_i, ..., d_n\}(1 \leq i \leq n) \in \mathbb{Z}$, document F_j and keyword dictionary $\mathcal{W} = \{W_{w_1}, ..., W_{w_i}, ..., W_{w_m}\}(1 \leq i \leq m)$. SU generates encrypted search index I_j according to P_j, F_j, W as follows.

- First, SU extends the elements in P_j to $3n + m$ dimensions, the element in the $(3i-2)$-th dimension is $d_i(1 \leq i \leq n)$, the elements in both the $(3i-1)$-th dimension and $3i$-th dimension are $1(1 \leq i \leq n)$, and the elements from $(3n+1)$-th to $(3n+m)$-th dimensions are set as BM25 keyword weighting of each textual keyword w_k generated according to F_j.
- Then, SU sets U random numbers $\delta_\xi(1 \leq \xi \leq U) \in \mathbb{Z}$ in the last U dimensions as noise[1] which protect the privacy better from stronger threat like scale-analysis attack [10], [21]. We can describe the vector as
$$D_j = \{d_1, 1, 1, ..., d_i, 1, 1, ..., d_n, 1, 1, w_1(F_j), \\ ..., w_k(F_j), ..., w_m(F_j), \delta_1, \delta_2, ..., \delta_U\}.$$
- Finally, SU splits the $(3n+m+U)$-dimension vector D_j into two $(3n+m+U)$-dimension vectors according to S. If $S[i]$ is 1, then SU randomly generates $D_{j,1}[i]$ and $D_{j,2}[i]$, where satisfying $D_{j,1}[i] + D_{j,2}[i] = D_j[i]$; if $S[i]$ is 0, then SU sets $D_{j,1}[i] \leftarrow D_{j,2}[i] \leftarrow D_j[i]$. Next, the split index pair I_j is encrypted as $\{M_1^T D_{j,1}, M_2^T D_{j,2}\}$, and the F_j is encrypted as $E_K(F_j)$ by AES with K. SU sends $\{I_j, E_K(F_j)\}$ to the CS.

[1] The noise is drawn from a Laplacian distribution, where $\delta \leftarrow Laplace(0, b)$. The scale parameter b is considered as a trade-off parameter between query accuracy and privacy. We will show a detailed evaluation of how the added noise affects the query accuracy in experiments.

TrapGen$(SK, Q, R_t, \mathcal{S})$: HU appoints a search query $Q = \{q_1, ..., q_i, ..., q_{n+m}\}(1 \leq i \leq n+m) \in \mathbb{Z}$, a divide parameter set $R_t = \{r_1, ..., r_i, ..., r_n\}(1 \leq i \leq n) \in \mathbb{Z}$, and a randomization parameter set $\mathcal{S} = \{s_1, ..., s_i, ..., s_n, \gamma\}(1 \leq i \leq n) \in \mathbb{Z}$, we denote that $r_1 \gg r_2 \gg r_3 ... \gg r_n$ and $s_i \ll r_{i-1}$ to make the values suitable to our search method. HU generates a $(3n+U)$-dimension trapdoor T_s, and then sends T_s to CS for range query. The specific trapdoor generation is shown in **Algorithm** 1.

Algorithm 1: Trapdoor generation

```
Input: Secret keys $SK$, query $Q = \{q_1, ..., q_i, ..., q_{n+m}\}$,
       divide parameter set $R_t = \{r_1, ..., r_i, ..., r_n\}$,
       Randomization parameter set
       $\mathcal{S} = \{s_1, ..., s_i, ..., s_n, \gamma\}$.
Output: Trapdoor $T_s$.
for $1 \leq i \leq n$ do
    $Q'[3i-2] \leftarrow \gamma \cdot r_i$;
    $Q'[3i-1] \leftarrow \gamma \cdot r_i \cdot q_i$;
    $Q'[3i] \leftarrow \gamma \cdot r_i \cdot s_i$;
for $n+1 \leq i \leq n+m$ do $Q'[i] \leftarrow \gamma \cdot q_i$ ;
Randomly choose $V$ and $1$ positions from the last $U$
    dimensions of $Q'$ and set them to $\gamma$, 1, respectively;
for $1 \leq i \leq 3n+U$ do
    if $S[i]$ is 1 then Set $Q'_1[i] \leftarrow Q'_2[i] \leftarrow Q'[i]$ ;
    else
        Randomly generate $Q_1[i]$, $Q_2[i]$, where satisfy
            $Q'_1[i] + Q'_2[i] = Q'[i]$;
return the trapdoor $T_s = \{M_1^{-1}Q'_1, M_2^{-1}Q'_2\}$.
```

QueryRangeGen$(Q_R, R_t, \mathcal{S})$: Given the divide parameter set R_t and the randomization parameter set $\mathcal{S}$, HU appoints a query range set $Q_R = \{q_{r1}, q_{r2}, ..., q_{r2i-1}, q_{r2i}, ..., q_{r2n-1}, q_{r2n}\}(1 \leq i \leq 2n)$ which contains the range of the values be queried, for example, given a query value $q_1 = 100$, if a search user wants to search values between 80 to 140, then $q_{r1} = q_1 - 80 = 20$, $q_{r2} = 140 - q_1 = 40$. The search query range parameter Q'_R is generated as

$$Q'_R =$$
$$\{\gamma r_1(2q_1 - q_{r1} + s_1) + \delta'_1, \gamma r_1(2q_1 + q_{r2} + s_1) + \delta'_2, ...,$$
$$\gamma r_n(2q_n - q_{r2n-1} + s_n) + \delta'_n, \gamma r_n(2q_n + q_{r2n} + s_n) + \delta'_n\}.$$

The $(2i-1)$-th dimension of Q_R is set as $r_i \cdot (2q_i - q_{r2i-1} + s_i) + \delta$, the $2i$-th dimension of Q'_R is $r_i \cdot (2q_i + q_{r2i} + s_i) + \delta$, where $\delta \leftarrow Laplace(0, b)$ denotes a noise drawn for each dimension of Q_R. Then, HU sends Q'_R and R_t to CS for search query.

Search(I_j, T_s, Q'_R, R_t): After gaining the trapdoor T_s, search query range Q'_R and the divide parameter set R_t, CS first calculates the search result as

$$\begin{aligned} Scores_j &= I_j \cdot T_s \\ &= \gamma(\sum_{i=1}^{n} r_i(d_i + q_i + s_i) + \sum_{k=1}^{m} w_k(F_j)q_{n+k} + \sum \delta_{\xi}^{(V)}) + \delta' \\ &= \gamma(\sum_{i=1}^{n} r_i(d_i + q_i + s_i) + BM(F_j, Q) + \sum \delta_{\xi}^{(V)}) + \delta', \end{aligned}$$

where $\sum \delta_{\xi}^{(V)}$ represents the sum of any V values in last U positions of I_j. Then, CS obtains ranked documents list using **Algorithm** 2.

Algorithm 2: Digital and textual keywords search

```
Input: Search range $Q'_R$, scores set
       $\{Scores_1, Scores_2, ..., Scores_N\}$, the divide
       parameter set $R_t$.
Output: The top-$k$ list of the documents.
for $1 \leq j \leq N$ do
    for $1 \leq i \leq n$ do
        if $Q'_R[2i-1] \leq Scores_j \leq Q'_R[2i]$ then
            $Scores_j \leftarrow Scores_j \bmod R_t[i]$;
        else break ;
Obtain the top-$k$ documents with the highest scores
    from the ranked list added with the document;
return the ranked list.
```

Remark. In fact, the DKSE can perform only textual multi-keyword ranked search. Given the scores set $\{Scores_1, Scores_2, ..., Scores_N\}$, the divide parameter set $R_t = \{r_1, ..., r_i, ..., r_n\}(1 \leq i \leq n)$. For each search result score $Scores_j$, from r_1 to r_n, let $Scores_j$ modulate these numbers in order, so that $Scores_j = \gamma(BM(F_j, Q) + \sum \delta_{\xi}^{(V)}) + \delta'$. Then, CS can return top-k list of the documents with the highest scores.

Example. Assume that there is a personal data with two digital attributes and two keywords {"male", "cancer"}, where digital attributes are $\{1100, 1100\}$ and the BM25 keywords weighting are $\{3, 5\}$. For convenience, we set the divide parameter set $R_t = \{10^8, 10^4\}$, the randomization parameter set $\mathcal{S} = \{2000, 4000, 1000\}$ and $U = 2$. Thus, we can generate $D_j = \{1100, 1, 1, 1100, 1, 1, 3, 5, 1, 2\}$. If a certain user wants search a personal data satisfies $\{900 - 1300, 800 - 1200\}$ with keywords {"male","cancer"}, he or she generates the search query $Q = \{1000, 1000, 1, 1\}$, and sets the query range as $Q_R = \{100, 300, 200, 200\}$. Hence, we have $Q' = \{10^{11}, 10^{14}, 2 \times 10^{14}, 10^7, 10^{10}, 4 \times 10^{10}, 3 \times 10^3, 5 \times 10^3, 10^3, 1\}$ and $Q'_R = \{3.9 \times 10^{14} + 2, 4.3 \times 10^{14} + 7, 5.8 \times 10^{10} + 3, 6.2 \times 10^{10} + 6\}$. In search process, CS first get the search result $Scores_j = 4.1 \times 10^{14} + 6.1 \times 10^{10} + 9 \times 10^3 + 2$. Since $3.9 \times 10^{14} + 2 < Scores_j < 4.3 \times 10^{14} + 7$, the CS computes $Scores_j \leftarrow Scores_j \bmod 10^8 = 6.1 \times 10^{10} + 9 \times 10^3 + 2$. Next, the CS computes $Scores_j \leftarrow Scores_j \bmod 10^4 = 9 \times 10^3 + 2$ since $5.8 \times 10^{10} + 3 < Scores_j < 6.2 \times 10^{10} + 6$. Finally, the CS can obtain the BM25 ranking score $Scores_j$ of D_j, where query keywords are "male", and "cancer". If there are N personal data, CS can return top-k list of the personal data containing "male" and "cancer" with the highest scores.

5 Privacy-preserving Data Mining and Online Pre-diagnosis

In this section, we first propose a **PRI**vacy-preserving **D**ata mining and **O**nline pre-diagnosis (PRIDO) framework based on DKSE to enable data mining and online pre-diagnosis over outsourced cloud server without revealing privacy of user's personal data. Then, we show how to achieve naïve Bayesian and decision tree based on the PRIDO and discuss how to support more classifiers.

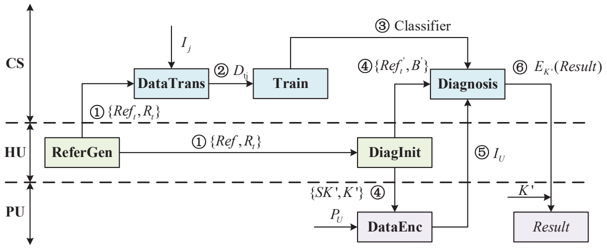


Fig. 2. Overview of PRIDO.

5.1 Proposed PRIDO Framework

The Overview of PRIDO is shown in Fig. 2. To train a classifier, the HU first sends a training reference vector to CS (Step ①), and then CS transforms SU's encrypted data stored in CS(Step ②) and trains a classifier (Step ③). Since the PU and the SU are not mutually trusted, that is, the PU does not want to disclose his/her personal data to the SU, and the SU does not want to leak his/her historical data to the PU. In order to solve this problem, we reset the secret key for each user before making online pre-diagnosis to support multi-user multi-key online pre-diagnosis (Step ④). To make an online pre-diagnosis, HU or PU sends encrypted personal data to CS (Step ⑤) and obtains the results (Step ⑥). The PRIDO contains five algorithms, namely **ReferGen**, **DataTrans**, **Train**, **DiagInit**, and **Diagnosis**. We introduce them separately as follows.

ReferGen$(SK, Ref, R_t, \mathcal{S})$: Given a random n-dimension vector $Ref = \{c_1, ..., c_i, ..., c_n\} (1 \leq i \leq n) \in \mathbb{Z}$ as training reference vector, a divide parameter set $R_t = \{r_1, ..., r_i, ..., r_n\} (1 \leq i \leq n) \in \mathbb{Z}$, and a randomization parameter set $\mathcal{S} = \{s_1, ..., s_i, ..., s_n, \gamma\} (1 \leq i \leq n) \in \mathbb{Z}$. HU first generates training reference trapdoor Ref_t in the similar way of **TrapGen**$(SK, Ref, R_t, \mathcal{S})$, the only different is that the dimensions from $(3n+1)$-th to $(3n+m)$-th are set to 0 and V random positions in the last U dimensions are set as r_i. Then, HU sends Ref_t, R_t to the CS.

DataTrans(I_j, Ref_t, R_t): Given the training reference trapdoor Ref_t and the divide parameter set R_t. Encrypted personal data I_j which stored in CS are transformed to D_{tj} with Eq. 6, where $\sum \delta_{\xi_i}^{(V)}$ is the sum of the noise number multiplied by r_i.

$$\begin{aligned} D_{tj} = \{ & \gamma \sum_{i=1}^{n} r_i(d_i + c_i + s_i + \sum \delta_{\xi_i}^{(V)}) + \delta', ..., \\ & \gamma \sum_{i=x}^{n} r_i(d_i + c_i + s_i + \sum \delta_{\xi_i}^{(V)}) + \delta', ..., \\ & \gamma r_n(d_n + c_n + s_n + \sum \delta_{\xi_n}^{(V)}) + \delta' \}. \end{aligned} \tag{6}$$

Algorithm 3: Data transform

```
Input: Encrypted data $I_j$, the training reference
       trapdoor $Ref_t$ and the divide parameter set
       $R_t = \{r_1, ..., r_i, ..., r_n\} (1 \leq i \leq n)$.
Output: Transformed data $D_{tj}$.
$Scores = I_j \cdot Ref_t$;
for $1 \leq i < n$ do
    $D_{tj}[i] \leftarrow Scores$;
    $Scores \leftarrow Scores \bmod r_i$;
$D_{tj}[n] \leftarrow Scores$;
return $D_{tj}$
```

Algorithm 3 shows the specific data transform. We can see that the real values of SU are hidden by γ, $\delta_i^{\xi}, \delta', s_i$ after transformation. We denote D_t as the set of D_{tj}.

Train(D_t): After obtain the transformed dataset D_t, the CS trains the corresponding classifier according to different classification algorithms and stores the trained classifier $\mathcal{C}$.

DiagInit$(R_{ef}, R_t, \mathcal{S})$: HU keeps the training reference vector Ref, the divide parameter set R_t, and the randomization parameter set $\mathcal{S}$ used in **ReferGen**. Before providing online pre-diagnosis server for PU, HU first initializes the system as follow steps:

- *Step 1*: HU generates new encryption keys as
$$\{S_K^{'}, K^{'}\} \leftarrow \textbf{KeyGen}(n, U). \tag{7}$$
- *Step 2*: HU generates new training reference trapdoor $Ref_t^{'}$ according to $SK^{'}$, R_{ef}, R_t, and $\mathcal{S}$ as
$$Ref_t^{'} \leftarrow \textbf{ReferGen(}SK^{'}, Ref, R_t, \mathcal{S}\textbf{)}. \tag{8}$$
- *Step 3*: HU sends $\{K^{'}, SK^{'}\}$ to a registered PU for online pre-diagnosis, and sends $Ref_t^{'}$, $\{E_{K'}(y_1), ..., E_{K'}(y_l)\}$ to CS.

Diagnosis$(Ref_t^{\prime}, R_t)$: Given the training reference trapdoor $Ref_t^{\prime}$ and the divide parameter set R_t, HU or registered PU can make an online pre-diagnosis by uploading encrypted personal data.

- *Step 1*: PU or HU encrypts personal data $P_U = \{p_{U_1}, p_{U_2}, ..., p_{U_n}\}$ based on **DataEnc** with $SK^{'}$, and then sends encrypted personal data I_U to CS.
- *Step 2*: CS transforms I_U to D_{tU} according to $Ref_t^{'}$ and R_t as $D_{tU} \leftarrow$ **DataTrans**$(I_U, Ref_t^{'}, R_t)$.
- *Step 3*: CS obtains the diagnosis result $E_{K'}(y_k)$ according the trained classifier $\mathcal{C}$ and the transformed personal data D_{tU}.
- *Step 4*: CS sends $E_{K'}(y_k)$ to PU or HU, and the PU or HU gets diagnosis result by decrypting $E_{K'}(y_k)$.

5.2 Two Examples for Naïve Bayesian Classifier and Decision Tree

Here we give two examples, privacy-preserving naïve Bayesian classifier and decision tree based on the PRIDO framework. For convenience, we only describe different parts of the two classifier algorithms.

5.2.1 *Naïve Bayesian classifier*

Train(D_t): CS uses D_t to train naïve Bayesian classifier. Assume that D_t contains l classes $\{E_{K'}(y_1), ..., E_{K'}(y_l)\}$ which are encrypted by AES. For each attribute A_i in D_t, CS calculates the mean $\Delta^{(A_i)} = \{\eta_{E_{K'}(y_1)}^{(A_i)}, ..., \eta_{E_{K'}(y_l)}^{(A_i)}\}$ and standard deviation $\Phi^{(A_i)} = \{\sigma_{E_{K'}(y_1)}^{(A_i)}, ..., \sigma_{E_{K'}(y_l)}^{(A_i)}\}$, respectively. CS keeps all of these parameters as classifier $\mathcal{C}$ secretly.

Diagnosis$(Ref_t^{\prime}, R_t)$:

- *Step 1*: PU or HU encrypts personal data $P_U = \{p_{U_1}, p_{U_2}, ..., p_{U_n}\}$ based on **DataEnc** with SK', and then sends encrypted data I_U to CS.
- *Step 2*: CS transforms I_U to D_{tU} according to Ref'_t and R_t as $D_{tU} \leftarrow$ **DataTrans**(I_U, Ref'_t, R_t).
- *Step 3*: For each continuous attribute A_i, assume that the attributes values of D_{tU} are $D_{tU} = \{a'_{U_1}, ..., a'_{U_n}\}$, CS calculates each probability of D_{tU} attribute values which belongs to y_i as

$$P(E_K(y_i)|D_{tU}) = \frac{P(E_K(y_i))}{P(D_{tU})} \cdot \prod_{j=1}^{n} g(a'_{U_j}, \eta^{(A_j)}_{E_K(y_i)}, \sigma^{(A_j)}_{E_K(y_i)}). \tag{9}$$

  In this way, CS can easily find the $P(E_{K'}(y_k)|D_{tU})$ which is the max value in $\{P(E_{K'}(y_1)|D_{tU}), ..., P(E_{K'}(y_l)|D_{tU})\}$ $(1 \leq k \leq l)$, and the patient who uploads D_{tU} to CS is diagnosed with disease y_k possibly.
- *Step 4*: CS sends $E_{K'}(y_k)$ to PU or HU, and the PU or HU gets diagnosis result by decrypting $E_{K'}(y_k)$.

5.2.2 Decision tree

Train(D_t): CS uses D_t to construct a decision tree. The attribute of maximum $Gain$ as shown in Eq. 5 will be selected as the best one to split the dataset into l partitions. The process will iterate until there are no more partitions. Each leaf node corresponds to one encrypted class $E_{K'}(y_k)$.

Diagnosis(Ref'_t, R'_t):

- *Step 1*: PU or HU encrypts personal data $P_U = \{p_{U_1}, p_{U_2}, ..., p_{U_n}\}$ based on **DataEnc** with SK', and then sends encrypted data I_U to CS.
- *Step 2*: CS transforms I_U to D_{tU} according to Ref'_t and R_t as $D_{tU} \leftarrow$ **DataTrans**(I_U, Ref'_t, R'_t).
- *Step 3*: CS uses D_{tU} to perform the *test* on each decision node one by one until reaching the leaf node and obtaining the result $E_{K'}(y_k)$.
- *Step 4*: CS sends $E_{K'}(y_k)$ to PU or HU, and the PU or HU gets diagnosis result by decrypting $E_{K'}(y_k)$.

Discussion. As described in Section 2, in naïve Bayesian classifier, values of different attributes are independence of each other. Therefore, for continuous attributes, as long as the same attribute takes the same transformation parameters, the accuracy of the prediction results can be guaranteed. In the online pre-diagnosis process of our framework, despite the secret key is rebuild, PU's (or HU's) personal data can be transformed into the same form since the Ref, R_t, and $\mathcal{S}$ are unchanged. Therefore, all classifiers that satisfy the independence assumption can be implemented based on our framework.

6 Privacy Analysis

In this section, we analyze our proposed work to check whether it can satisfy the privacy requirements described in Section 3.3. Since ASPE has been proved to be weak against **K**nown-**P**laintext **A**ttack (KPA) [21], [22], our framework is proposed based on ASPE with Noise (ASPEN), we first briefly review its security proof, and then analyze the privacy of our work.

6.1 Security of ASPEN

To prove the security of the ASPEN, without loss of generality, we assume that the query Q and the data object P are n-dimensional.

Definition 6.1 (Asymmetric Scalar-product-Preserving Encryption with Noise (ASPEN))**.** *Let $E(p, SK)$ be the encrypted value of a digital vector p, where E is an encryption function and SK is a secret key. E is ASPEN if and only if there exits a computational procedure f such that $\forall p_1, p_2, SK, f(E(p_1, SK), E(p_2, SK)) = Scal(p_1, p_2) + \delta$, where $Scal(p_1, p_2)$ is the scalar product of p_1, p_2, and δ is a random number. We summarize the procedures of ASPEN used in our work as follows.*

- ***Key**: Two $(n+U) \times (n+U)$ invertible matrices M_1, M_2; a $n + U$-bits randomly generated boolean vector $S \in \{0, 1\}^{n+U}$.*
- ***Index encryption**: Each index vector p is firstly extended to $(n + U)$-dimension, where the last U dimensions are set as random number $\delta_1, \delta_2, ..., \delta_U$. Then, each index vector is split into two parts $\{p', p''\}$ according to S, if $S[i]$ is 1, then randomly generate $p'[i]$ and $p''[i]$, where satisfy $p'[i] + p''[i] = p[i]$; if $S[i]$ is 0, then set $p'[i] \leftarrow p''[i] \leftarrow p[i]$. The encrypted value of p is the pair $\hat{p} = \{M_1^T p', M_2^T p''\}$.*
- ***Query encryption**: Each index vector p is firstly extended to $(n + U)$-dimension, where the first n dimensions of p are multiplied by γ and a random position in the last U dimensions is set as γ. Then, each query vector q is split into two parts $\{q', q''\}$ according to S, if $S[i]$ is 1, then set $q'[i] \leftarrow q''[i] \leftarrow q[i]$; if $S[i]$ is 0, then randomly generate $q'[i]$ and $q''[i]$, where satisfy $q'[i] + q''[i] = q[i]$. The encrypted value of q is the pair $\hat{q} = \{M_1^{-1} q', M_2^{-1} q''\}$.*
- ***Scalar product comparison**: Let $\hat{p}_1, \hat{p}_2$ and $\hat{q}$ be the encrypted value of index vector p_1, p_2 and query vector q. To compare the scalar product of p_1, q and p_2, q, it only need to compare $\hat{p}_1 \cdot \hat{q} \Leftrightarrow \hat{p}_2 \cdot \hat{q}$.*

Theorem 6.1. *The ASPEN is secure against the known-plaintext attack, if the random number γ for each query and δ for each object cannot be known by the adversary.*

Proof. In the known-plaintext attack model, the adversary can obtain a set of queries and their ciphertexts. For each query q, the adversary would have the encrypted pair $\hat{q} = \{M_1^{-1} q', M_2^{-1} q''\}$ used in scalar product comparison process. In the ASPEN described above, the scalar product between $\hat{q}$ and $\hat{p} = \{M_1^T p', M_2^T p''\}$ can be calculated as

$$\begin{aligned} \hat{p} \cdot \hat{q} &= M_1^T p' \cdot M_1^{-1} q' + M_2^T p'' \cdot M_2^{-1} q'' \\ &= (M_1^T p')^T M_1^{-1} q' + (M_2^T p'')^T M_2^{-1} q'' \\ &= (p')^T q' + (p'')^T q'' = \gamma \cdot (p \cdot q + \delta') + \delta''. \end{aligned} \tag{10}$$

As described in [21], [22], if the adversary can obtain the plaintext of q, Eq. 10 contains $n + 3$ unknowns (i.e., $\gamma, \delta', \delta''$, and the n dimensions of p). If the random number γ is same for each query, the adversary can solve the $n + 3$ unknowns in p and δ', δ'' by collect $n + 4$ plaintext-ciphertext pairs of query points to construct $n + 4$ equations like Eq. 10.

However, in the above ASPEN, the random numbers $\gamma, \delta', \delta''$ are generated different for each time. Therefore, there are $4n + 12$ unknowns (i.e., $n + 4$ random number

$\{\gamma_1, \gamma_2, ..., \gamma_{n+4}\}$, $n+4$ random number $\{\delta'_1, \delta'_2, ..., \delta'_{n+4}\}$, $n{+}4$ random number $\{\delta''_1, \delta''_2, ..., \delta''_{n+4}\}$, and n dimensions of p) in the equation set. Since there are only $n+4$ equation, the adversary does not have sufficient information to solve p, even if $n+4$ queries and corresponding scalar products are known by the adversary. Similarly, if the adversary obtains a set of objects in the dataset with their ciphertexts. Since the random number δ', δ'' for each object is unknown for the adversary, there are $3n+9$ unknowns in $n+4$ equations, the adversary also does not have sufficient information to solve q. Recently, [21] proposed an algorithm for a KPA adversary to reconstruct the plaintext of queries. However, their algorithm is focused on binary data domain, which cannot work on our work on the real data domain. Therefore, the ASPEN is secure against the known-plaintext attack. □

Theorem 6.2. *The ASPEN is secure against the known-plaintext attack, if the bit string S cannot be known by the adversary.*

Proof. Assume that the adversary knows the data object p with its corresponding ciphertext $\hat{p}$. For any data object, if the adversary does not know the bit string S used for splitting, p has to be modeled as two unknown $(n+U)$-dimensional vectors. The equations for solving the secret matrices M_1, M_2 can be constructed with p and $\hat{p}$. Notice that there are $2(n+U)|p|$ unknowns in p′ and p″, where $|p|$ is the number of data objects in the dataset. There are also $2(n+U)^2$ unknowns in the secret matrices, but only $2(n+U)|p|$ equations constructed. Similarly, using the queries, there are $2n|q|$ equations constructed which contain $2(n+U)|q| + 2(n+U)^2 + 1$ unknowns, where $|q|$ is the number of obtained queries. Therefore, the information to solve the unknowns is insufficient for the adversary, and the ASPEN is secure against the known-plaintext attack. □

6.2 Privacy of DKSE

Next, we analyze proposed DKSE concerning the privacy requirements as described in Section 3.3.

6.2.1 Index (trapdoor) confidentiality

Theorem 6.3. *Each index stored in the CS or search query is resilient to the level-2 attack defined in the attack model.*

Proof. In proposed DKSE, SU's personal data P_j or HU's search query is encrypted by $SK = \{S, M_1, M_2\}$ before sending to the CS. According to **Theorem** 6.1 and **Theorem** 6.2, if the secret key SK, random number γ and $\delta_1, \delta_2, ..., \delta_U$ are kept confidential, it is difficult for the adversary to deduce the meaning of each dimension in the index or query, each index stored in the CS or trapdoor is resilient to the *level-2* attack defined in the attack model. □

6.2.2 Keyword privacy

Theorem 6.4. *Assume that our DKSE is attacked by a level-2 attacker whose knowledge $H = \langle\{I, P\}, R_t, Q'_R, W, Scores\rangle$, where $\{I, P\}$ is a set of plaintex-ciphertext of indexes. The attacker cannot infer the value of digital keywords and which textual keywords are queried.*

Proof. For digital keyword, the adversary can infer search range from Q'_R as

$$\begin{aligned} &Q'_R[2i-1] - Q'_R[2i] \\ &= \gamma \cdot r_i(2q_i - q_{r2i-1} + s_i) + \delta' \\ &\quad - \gamma \cdot r_i(2q_i + q_{r2i} + s_i) + \delta'' \\ &= \gamma \cdot r_i(q_{r2i} + q_{r2i-1}) + \delta' - \delta''. \end{aligned} \tag{11}$$

Since r_i is known by the adversary, if $\delta' - \delta'' = 0$, the adversary can construct two equations like Eq. 11 and infer γ by compute the gcd value among them. Then, the adversary can obtain $q_{r2i} + q_{r2i-1}$. However, such infer cannot work in our scheme, because the δ' and δ'' are independently drawn from Laplacian distributions. In search process, there are $2n+m+V+2$ unknowns (i.e., the $n+m$ dimensions of q, n random number s_i, V noise number δ_ξ, random δ', and γ) in Eq. 12.

$$\begin{aligned} &Scores_j \\ &= \gamma(\sum_{i=1}^{n} r_i(d_i + q_i + s_i) + BM(F_j, Q) + \sum \delta_\xi^{(V)}) + \delta'. \end{aligned} \tag{12}$$

To solve the search query, $2n+m+V+3$ equations can be constructed with I and P. However, each equation introduces additional $V+1$ unknowns, which makes the constructed equations contain $(2n+m+V+2)(V+2)$ unknowns. The adversary does not have sufficient information to solve q. As for textual keyword, Cao *et al.* proposed scale analysis attack [10], which can make the CS identify the keyword by referring to the keyword specific document frequency information about the dataset. To resist this attack, we set U noise keywords $\delta_\xi (1 \leq \xi \leq U)$ in each index and randomly select V noise keywords in each trapdoor T_Q. To make the probability of two $\sum \delta_\xi^{(V)}$ having the same value is less than $1/2^\omega$, every index should include at least 2ω noise entries, and every query vector will randomly select half noise entries. While $V = U/2$, the probability of two $\sum \delta_\xi^{(V)}$ have the same value is less than $1/2^U$. Therefore, even for the same $\vec{Q}$ and document, the final similarity score will be different because the $\sum \delta_\xi^{(V)}$ is different. The textual keyword privacy can be protected in DKSE. □

6.2.3 Trapdoor unlinkability

Definition 6.2 (Trapdoor unlinkability). *Let T'_s, T''_s be two trapdoors, T'_s, T''_s conform trapdoor unlinkability if the CS or a adversary cannot distinguish whether T'_s and T''_s are generated by the same Q.*

Theorem 6.5. *The trapdoor generated in DKSE is trapdoor unlinkability against the level-2 attack defined in the attack model.*

Proof. In DKSE, each query is random split according to S before encrypting as $\{M_1^{-1}Q_1, M_2^{-1}Q_2\}$. As shown in **Algorithm** 1, while $S[j] = 0$, $Q_1[i]$ and $Q_2[i]$ are randomly generated which satisfy $Q_1[i] + Q_2[i] = Q[i]$. The two trapdoors T'_s, T''_s will be exactly the same if and only if all the values in S are 1, which the probability is $1/2^{(3n+m+U)}$. In search process, the CS or a *level-2* adversary can obtain the final similarity scores, which may reveal the relationship between trapdoors. To randomize similarity scores, we introduce some random numbers (i.e., γ and s_i). Besides, the random choice of locations of noise keywords can also increase the trapdoor randomness in DKSE. The similarity scores of the same index I_j and T_s can be calculates by Eq. 12. As proved above, the probability of two $\sum \delta_{\xi}^{(V)}$ having the same value is less than $1/2^U$ and the $(\gamma', \gamma''), (s_i', s_i'')$ are totaly different for T_s' and T_s''. Thus, with different trapdoors generated by the same search query, different similarity scores will be produced even for the same index.

As for query range Q_R, there are 5 unknowns in Eq. 11. To solve the search range q_{r2i-1}, q_{r2i}, if the adversary constructs 6 equations like Eq. 11, 2 additional unknowns will introduced by each equation, which makes the constructed equations contain 15 unknowns. Therefore, it is hard for the CS or a *level-2* adversary to mine the relationship between two trapdoors by comparing them directly. The trapdoor generated in DKSE is trapdoor unlinkability against the *level-2* attack defined in the attack model. □

6.3 Privacy of Data Mining and Online Diagnosis

Theorem 6.6. *Privacy of Data mining and Online Diagnosis in our proposed PRIDO framework can be guaranteed against the level-2 attack defined in the attack model.*

Proof. In data mining process, HU sends a training reference vector Ref_t and a divide parameter set R_t to CS to train classifier, and then SU's historical personal data will be transformed by Ref and R_t. The value of SU's data d_x is hidden by $\gamma \sum_{i=x}^{n} r_i \cdot (d_i + c_i + s_i + \sum \delta_{\xi_i}^{(V)}) + \delta'$ after transforming. Even though the divide parameter set $R_t = \{r_1, ..., r_i, ..., r_n\}(1 \leq i \leq n)$ is known by CS or a *level-2* attacker, each personal data d_i is still protected by $s_i, \gamma, c_i, \delta'$, and $\sum \delta_{\xi_i}^{(V)}$. The trained classifier $\mathcal{C}$ is also protected by these random numbers.

In online pre-diagnosis process, HU uses the new secret key $SK^{'}$ to generate a new trapdoor $Ref_t^{'}$ of the reference vector Ref used in classifier training, and then sends $Ref_t^{'}$ to CS. Meanwhile, $SK^{'}$ is sent to authorized PU. PU encrypts his/her personal data and sends ciphertexts to CS for online pre-diagnosis. PU's personal data is protected by the encryption mechanism, and his diagnosis results are also encrypted. Therefore, PU privacy can be guaranteed. In addition, SU's historical personal data are secure because the secret key used in diagnosis is different from which is used to encrypt SU's data. The trained classifier is a private property of HU and of great value, it cannot be used by unauthorized users. In our framework, only transformed data can be predicted correctly, and only users who are authorized by HU can get $SK^{'}$ to encrypt their personal data and transform the encrypted data in CS. Therefore, if the adversary stoles the classifier, it still cannot use it without $SK^{'}$. The privacy of Dada mining and online diagnosis in our proposed PRIDO framework can be guaranteed against the *level-2* attack defined in the attack model. □

- Selective of QR: Although $error$ of range query will not be affected by query range. In border point, d_i is still cannot be searched. We suggest setting query range a little bigger than what you want to query if the border value is necessary to be searched.
- Selective of PD: The bigger PD is, the lower $error$ is, and it also depends on PTD. We suggest that PD is at least 10^3 bigger than PTD to ensure accuracy.

7 Performance Analysis

We analyze the performance of our work in this section. We build our work in Python using NumPy[2] and gmpy2[3] extension modules. All experiments are run on a machine with one 3.3-GHz two-core processor and 8-GB RAM.

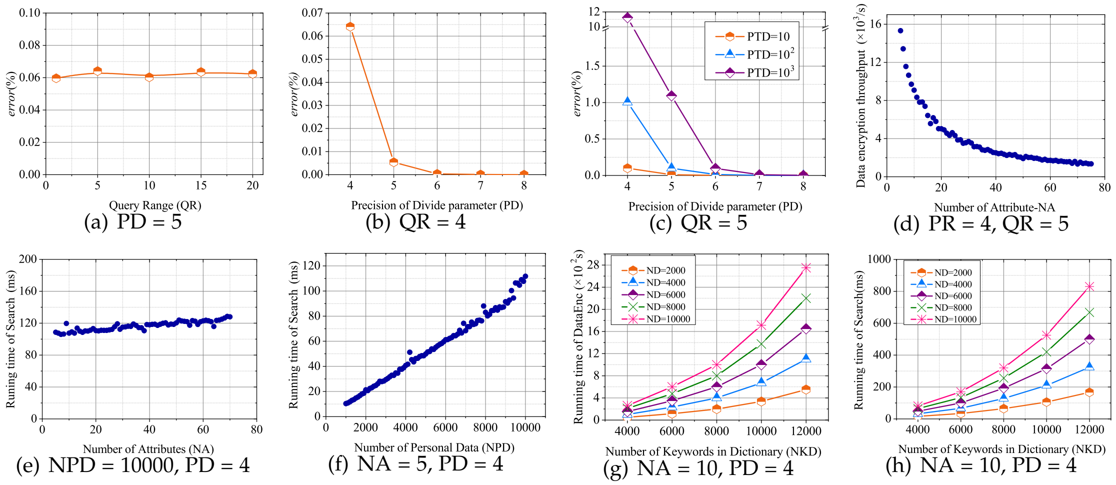
7.1 Performance of DKSE

7.1.1 Query precision

Precision of range query. Here, we analyze the precision of range query in DKSE. As descried in Section 4, for each attributes in personal data, the query range is $\gamma r_i(2q_i - q_{r2i-1} + s_i) + \delta_i', \gamma r_n(2q_i + q_{r2i} + s_i) + \delta_i'$, and the search score is $Scores = \gamma(\sum_i^n r_i(d_i+q_i+s_i)+BM(F,Q)+\sum \delta_{\xi}^{(V)}) + \delta'$. Hence, as shown in Fig. 4, search error occurs when $Scores \in [\gamma r_i(2q_i+q_{r2i-1}+s_i)+\delta_i', \gamma r_i(2q_i+q_{r2i-1}+s_i)+\delta_i'+\gamma(\sum_{j=i+1}^n r_j(d_j+q_j+s_j)+BM(F,Q)+\sum \delta_{\xi}^{(V)})+\delta']$ or $Scores \in [\gamma r_i(2q_i + q_{r2i} + s_i) + \delta_i', \gamma r_i(2q_i + q_{r2i} + s_i) + \delta_i' + \gamma(\sum_{j=i+1}^n r_j(d_j + q_j + s_j) + BM(F,Q) + \sum \delta_{\xi}^{(V)}) + \delta']$. AS $r_1 \gg r_2 \gg r_3 ... \gg r_n$, if query range $\{q_{r2i-1}, q_{r2i}\}$ is much wider than $|q_i - d_i|$, the small random numbers can be ignored in range query of q_i. However, if the query are on the boundary value, the small random numbers should be in consideration. There are two factors that may affecting query accuracy, namely Query Range (QR) and the Precision of Divide parameter (PD) (i.e., PD $= max(r_i)/max(r_{i+1})$). Define range error $error = Scores/r_i(2q_i + \text{QR} + \text{s}_\text{i})$ in each loop of **Search**, we consider a test vector $D = \{100, 100, 100, 100, 100\}$, and query it with different QR and PD. In Fig. 3(a), we plot $error$ in different QRs, where PD = 4. We can see that the $error$ is similar in different QRs. And we plot the range error in different PDs in Fig. 3(b), where QR is set to 5. As PD increases, the $error$ decreases, because the larger PD is, the larger r_i than r_{i+1}, the effect of small random number on query precision is reduced. In addition, the Precision of the Tuples in D between each other (PTD) (i.e., PTD $= D[i+1]/D[i]$) will also affect the query accuracy. We set QR = 5 to test the $error$ in different PTDs. The result is plotted in Fig. 3(c), we can see that $error$ in different PDs is very low when PTD$=$ 10. When PTD $= 10^2$, the error in PD = 4 is more than 1%, but is still less than 0.1% when PD$\geq$ 5. When PTD $= 10^3$, the $error$ in PD $= 4$ is more than 10%, and the $error$ in PD $= 5$ is more than 1%. From Eq. 6, we can see that $Scores$ of q_i is divided by multiplying a random number r_i. If $r_i/r_{i+1} \gg d_{i+1}/d_i$, the affect of rest values $\sum_{x=i+1}^n r_x \cdot (d_x + q_x + s_x + \delta_x^{\xi}) + \sum \delta_{\xi}^V$ can be ignored in search comparison. Else if r_i/r_{i+1} is close to d_{i+1}/d_i, the range error will be high. Based on the above analysis, we give some advice on use of DKSE as follows:

Precision of textual multi-keyword ranked search. As for textual multi-keyword ranked search, since Cao *et al.* [10] proposed Multi-keyword Ranked Search over Encrypted data (MRSE) based on tf-idf similarity score, many extended schemes [11], [13], [23] based on MRSE have been proposed to meet different application requirements. Our DKSE introduces the BM25 ranking model for similarity ranking, which has higher ranking precision than that of the MRSE.

[2] https://numpy.org/
[3] https://pypi.org/project/gmpy2/



(a) PD = 5 (b) QR = 4 (c) QR = 5 (d) PR = 4, QR = 5

(e) NPD = 10000, PD = 4 (f) NA = 5, PD = 4 (g) NA = 10, PD = 4 (h) NA = 10, PD = 4

Fig. 3. Actual performance analysis of DKSE (test 10 times for average). (a) (b) (c) are range error in each loop of **Search** in different situations; (d) is the data encryption throughput of **DataEnc** algorithm varying with Number of Attributes (NA), where Precision of Divide parameter (PD) = 4, Query Range (QR) = 5; (e) (f) are computational cost of **Search** algorithm for range query in different situations; (g) is the computational cost of index encryption varying with Number of Keywords in Dictionary (NKD) in different Number of Documents (ND); (h) is the computational cost of textual multi-keyword ranked search varying with NKD in different ND.

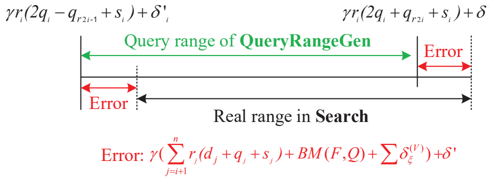

$\gamma r_i(2q_i - q_{r2i-1} + s_i) + \delta'_i$
$\gamma r_i(2q_i + q_{r2i} + s_i) + \delta'_i$
Query range of QueryRangeGen
Error
Error
Real range in Search
Error: $\gamma(\sum_{j=i+1}^{n} r_j(d_j + q_j + s_j) + BM(F,Q) + \sum \delta_{\varsigma}^{(V)}) + \delta'$

Fig. 4. Range error in **Search**.

TABLE 2
Precision Comparison of Textual Multi-keyword Ranked Search

Method	P@1	P@5	P@10	MAP
MRSE (b=1)	0.1673	0.1389	0.1102	0.1395
Our DKSE (b=1)	0.2967	0.2132	0.1692	0.2661
MRSE (b=0.5)	0.1225	0.1014	0.0816	0.1095
Our DKSE (b=0.5)	0.2153	0.1536	0.1217	0.1914

We use LETOR 4.0 dataset [24] to test the ranking precision of our DKSE and MRSE. The LETOR 4.0 dataset is a package of benchmark data sets for research on web search, which contains standard features, relevance judgments, and data partitioning. There are many kinds of features included in LETOR 4.0, such as tf-idf, BM25, and LMIR. We use Mean Average Precision (MAP) and Precision at position k (P@k), which are widely used to analyze retrieval performance, to evaluate the precision of our work. P@k measures the relevance of the top k results of the ranking list with respect to a given query, which can be calculated as P@k = (Number of relevant docs in top k results) $/k$. For a single query, Average Precision (AP) is defined as the average of the P@k values for all relevant documents as

$$AP = \frac{\sum_{k=1}^{N}(\text{P@}k * rel(k))}{N_{docs}}, \quad (13)$$

where N is the number of retrieved documents, and $rel(k)$ is a binary function on the relevance of the k-th document. If the k-th doc is relevant, set $rel(k) = 1$; otherwise, $rel(k) = 0$. We get MAP by averaging the AP values of all the queries.

In this evaluation, we use our DKSE to encrypt the BM25 features of keywords and we also use MRSE to encrypt the tf-idf features of keywords. The number of noise keywords is set as $U = 100$, and the scale parameter of noise is set to $b = 1, b = 0.5$, respectively. As shown in Table. 2, the textual multi-keyword ranked search precision of our DKSE is significantly higher than that of MRSE. Besides, we can see that big d leads to a higher precision of search results.

7.1.2 Computational cost of DKSE

Theoretical analysis. Let $|d|$ be the bit-length of each element in personal data and SK, $|r|$ be the bit-length of each divide parameter. The secret key $SK = \{S, M_1, M_2\}$ is generated as two $(3n+m+U) \times (3n+m+U)$ matrices and a $(3n+m+U)$-dimension boolean vector, the storage cost of SK is $O((3n+m+U)^2) \cdot |d|$ bits. In **DataEnc**, each personal data is extended into $(3n+m+U)$-dimension, split into two $(3n+m+U)$-dimension vectors and encrypted by M_1^T and M_2^T, respectively. The storage cost of each encrypted personal data is $O(3n+m+U) \cdot |d|$ bits and the computational cost of data encryption is $O((3n+m+U)^2)$. As for **TrapGen**, each search query is generated as $(3n+m+U)$-dimension, split into two $(3n+m+U)$-dimension vectors and encrypted by M_1^{-1} and M_2^{-1}, respectively. Since the divide parameters are much larger than elements in personal data, the storage cost is at most $O(3n+m+U) \cdot |r|$ bits. The computational cost of trapdoor generation is $O((3n+m+U)^2)$. In **Search**, the CS first performs $I_j \cdot T_s$ and then performs n modular operations. The computational cost of **Search** is $O(4n+m+U)$, and the memory space cost is at most $O(n+m) \cdot |r|$ bits.

Experimental results. Here, we evaluate the computational cost of DKSE. There are three factors affecting the running time of DKSE, namely the Number of Personal Data (NPD), the Number of Attributes (NA) contained in data, and the Precision of Divide parameter (PD). To fully evaluate the performance of DKSE, we performed experiments using synthetic datasets with different NPD and NA. In Fig. 3(d), we plot the data encryption throughput of **DataEnc** which varies with NA in personal data. From Fig. 3(d), we can see that the data encryption throughput decreases with increasing NA. Even if NA $= 60$, more than $2,000$ personal data can be encrypted per second, which can meet real-time requirements. We also plot the running time of **Search** varying with NA and NPD in Fig. 3(e) and Fig. 3(f), respectively. We can observe that the running time of **Search** increases linearly with increasing NA and NPD, and $10,000$ personal data can be searched in 120 ms. Moreover, we evaluate the running time of textual multi-keyword ranked search in DKSE. Assume that each personal documents contain 10 attributes and all of them are in query range, there are two factors affecting the running time of multi-keyword ranked search, namely Number of Documents (ND) and Number of Keywords in Dictionary (NKD). In Fig. 3(g), we plot the running time of **DataEnc** varying with NKD in different ND. We can see that the running of index encryption is increasing with increasing ND and NKD. From Fig. 3(h), we can notice that the running time of **Search** also increases with increasing ND and NKD, 10,000 indexes with 12,000 keywords can be searched in 900 ms.

7.2 Performance of PRIDO

In this experiment, we evaluate the performance of the proposed PRIDO framework.

Datasets. We consider three datasets, where two of them (i.e., Pima Indians Diabetes Dataset (PIDD) and Breast Cancer Wisconsin Dataset (BCWD)) are used by the UCI machine learning repository[4], and a synthetic dataset is used to test all factors which affect the performance of our framework. The PIDD is created by the National Institute of Diabetes and Digestive and Kidney Diseases, which is used to predict whether a female of Pima Indian heritage suffers from diabetes. This dataset contains 768 instances, and each instance contains 8 attributes. The BCWD is created by Dr. WIlliam H. Wolberg from the University of Wisconsin Hospitals, which is used to predict whether a female suffers from Breast Cancer. This dataset contains 683 intact instances, and each instance contains 10 attributes. We delete the useless attributes (i.e., id numbers) in BCWD. The synthetic dataset contains 10,000 instances, and each instance 25 attributes randomly chosen from 10 to 100.

7.2.1 Accuracy of online pre-diagnosis

Here, we use PIDD and BCWD to train naïve Bayesian classifier and decision tree classifier using our framework for performance evaluation. For better comparison, we also implement original naïve Bayesian and decision tree algorithms used in plaintext. As discussed in Section 7.1.1, we set PD = 4 to ensure the range query precision.

Table. 3 shows that the diagnosis accuracy in our PRIDO is evidently affected by the scale parameter b of the noise variable δ. We can see that smaller b leads to higher accuracy of diagnosis but lower privacy guarantee, while large b results in higher diagnosis privacy guarantee but lower accuracy. In other words, our scheme provides a balance parameter for users to satisfy their different requirements on accuracy and privacy.

TABLE 3
Average Accuracy of Online Pre-diagnosis (Test 10 times)

Dataset	BCWD		PIDD	
Classifier	NB	DT	NB	DT
plaintext	95.707%	94.706%	73.113%	72.281%
b=0.9	95.23%	94.79%	74.81%	72.95%
b=0.7	94.54%	93.32%	73.05%	70.97%
b=0.5	92.81%	92.43%	72.85%	70.64%
b=0.3	90.94%	90.26%	71.12%	69.84%
b=0.1	88.45%	89.05%	69.62%	68.93%

7.2.2 Computational cost of PRIDO

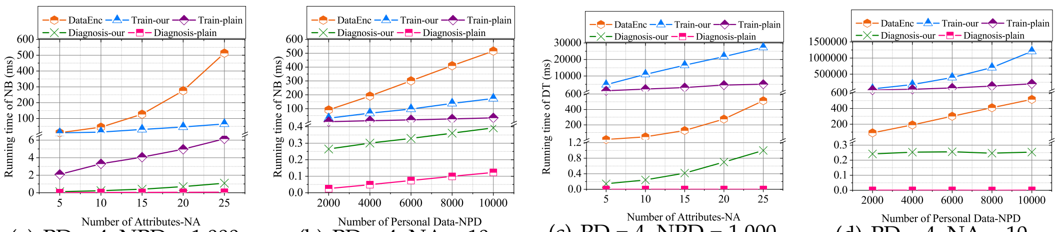
Theoretical analysis. Here, we analysis the complexity of our PRIDO framework. In **ReferGen**, the PU generates a $(3n+m+U)$-dimension training reference trapdoor Ref_t according to **TrapGen**, where the computational cost is $O((3n+m+U)^2)$ and storage cost is $O(3n+m+U)\cdot|r|$ bits. As for **DataTrans**, the CS costs $O(|I|(4n+m+U))$ operations, where $|I|$ is the number of personal data. The memory space cost when performing **DataTrans** is at most $O(n+m)\cdot|r|$ and the storage cost of transformed dataset D_t is $O(|I|n)\cdot|r|$bits. In **Train**, the CS cost the same operations as the plaintext to train classifiers. The cost of **DiagInit** is the same as that of **ReferGen** since its just generates a new training reference trapdoor Ref'_t. The **Diagnosis** performs one-time **DataEnc** and one-time **DataTrans**, which cost $O((3n+m+U)^2)$ operations and at most $O(n+m)\cdot|r|$ bits memory space.

Experimental results. In order to test all factors affecting our framework, we use the synthetic dataset. There are three factors which affect the running time of our framework, namely the Number of Personal Data (NPD) used to train a classifier, and the Number of Attributes (NA) contained in data. In Fig. 5(a) and Fig. 5(c), we plot the running time of our NB, DT and original NB, DT varying with PD, respectively. We can see that the running time of all algorithms increases with NA because more tuples need to be calculated. In Fig. 5(b) and Fig. 5(d), we plot the running time of our NB, DT and original NB, DT varying with NPD, respectively. As more data need to be processed, the running time of all algorithms increases with NPD. From all aforementioned figures, we can see that computational cost in our framework is comparable with the original algorithm used in plaintext in classifier training and online diagnosis. In the classifier training process, our framework spends more time on **DataTrans** than the original algorithm, and larger data values after transform also lead to larger computational overhead. Similarly, the running time of our framework is larger than that of the original algorithm in the diagnosis process. However, only one personal data needed to be transformed in **Diagnosis**, which cause it takes less than 1 ms for a user to make an online pre-diagnosis.

7.3 Comparative Analysis

Here, we make a comparison between our work and the most recent works of outsourced Personal Health Records (PHR) search and privacy-preserving online pre-diagnosis systems, which is shown in Table 4. From Table 4, we can conclude that the recent works [3], [4], [5], [7], [9], [10] support only one or a few functions which is important in

[4] http://archive.ics.uci.edu/ml/.



(a) PD = 4, NPD = 1,000 (b) PD = 4, NA = 10 (c) PD = 4, NPD = 1,000 (d) PD = 4, NA = 10

Fig. 5. Average computational cost of PRIDO (test 10 times). (a) is the running time of our NB and original NB varying with PD, where PD = 4, NPD = 1,000; (b) is the running time of our NB and original NB varying with NPD, where PD = 4, NA = 10; (c) is the running time of our DT and original DT varying with NA, where PD = 4, NPD = 1,000; (d) is the running time of our DT and original DT varying with NPD, where PD = 4, NA = 10.

MHMN scheme as we discussed in Section 1. None of the recent works support both digital vectors range query, textual multi-keyword ranked search, and online pre-diagnosis efficiently. Yao *et al.* [3] present a scheme that just supports digital vectors range query and the performance is limited. Li *et al.* [4] proposed a scheme which supports both digital vectors range query and textual keyword search, but its performance is low and cannot support ranked search. Cao *et al.* [10] first proposed an efficient **M**ulti-keyword **R**anked **S**earch over **E**ncrypted cloud data (MRSE) scheme, but their work just supports textual multi-keyword ranked search and the ranking model is tf-idf, which has lower search precision than that of BM25 model. All of the above works just need 1 round of communication to complete the function. Several homomorphic encryption based schemes [5], [9] are proposed to support privacy-preserving online pre-diagnosis. However, their works only support specific classification algorithms and required 2 or 3 rounds of communication, with huge communication and computational overhead. Moreover, [3], [4], [9] are designed based on the predicate encryption [25], which are **S**electively secure under the **C**hosen-**P**laintext **A**ttack (SCPA) model. [7] and [5] are based on the Paillier cryptosystem [26], which are secure under **C**hosen-**P**laintext **A**ttack(CPA) model. MRSE [10] is secure under the **K**onwn-**B**ackground **A**ttack (KBA) model for computational cost saving. In order to meet the performance requirements of the actual application scenario, our solution is secure under the KPA model, which is weaker than CPA and SCPA but similar to MRSE and most of the practical searchable encryption schemes [11], [12], [13]. In general, our solution is a trade-off between functionality, security, and performance. Compared with the above recent works, our work can support digital vector range query, textual multi-keyword ranked search, and online pre-diagnosis in reasonable security.

8 Related Work

8.1 Searchable Encryption

In 2000, Song *et al.* [27] first proposed the notion of Searchable Encryption (SE) in the symmetric key setting. Following this work, Curtmola *et al.* [28] gave the improvements and advanced security definitions of SE. Next, Wang *et al.* [29] solved the keyword ranked search problem. However, all of the schemes mentioned above only support single keyword search. Cao *et al.* [10] first defined the problem of Multi-keyword Ranked Search over Encrypted cloud data (MRSE) and proposed corresponding schemes based on the Vector Space Model (VSM). To achieve higher search result accuracy, Sun *et al.* [30] generated the search index based on term frequency and the VSM with cosine similarity measure. Next, Fu *et al.* [31] adopted parallel computing to increase the effectiveness of multi-keyword search. Moreover, many extended schemes [11], [12], [13] based on MRSE have been proposed to meet different application requirements. Xia *et al.* [11] proposed a dynamic multi-keyword ranked search scheme which can realize dynamic update operations (i.e., deletion, insertion, etc.). Fu *et al.* [12] generated a user interest model as the importance model, which can return search results personalized search results combining with the keyword vector space model. To achieve smart semantic search, Fu *et al.* [13] proposed a modified linear form of Conceptual Graphs and use Conceptual Graphs as knowledge representation. In addition, some researches [4], [32], [33], [34] studied the SE schemes in multi-user settings. First, Li *et al.* enabled efficient multi-dimensional keyword searches with range query based on the hierarchical predicate encryption. Then, Miao *et al.* [32] devised a hierarchical attribute-based keyword search scheme supporting multi-keyword search and user revocation. To achieve expressive keyword search and improve computational efficiency, Hui *et al.* [33] present a public-key searchable encryption scheme in the prime-order groups, which allows keyword search policies to be expressed in conjunctive, disjunctive or any monotonic Boolean formulas and achieves significant performance improvement over existing schemes. In [34], Wang *et al.* proposed an efficient hidden policy ABE scheme with keyword search, which enables efficient keyword search with constant computational overhead and constant storage overhead. Unfortunately, as far as we know, there is currently no searchable encryption scheme that can support machine learning.

8.2 Privacy-preserving Online Diagnosis

To provide privacy-preserving online diagnosis, many works [5], [6], [7], [8], [35], [36], [37], [38] have been proposed. Ayday *et al.* [36] employed logistic regression model to calculate the disease probability while protecting the privacy of patients. While in all the above two settings, the prediction model is publicly known, and their proposed schemes can only protect the patient's information. In [35], Bost *et al.* constructed three major classification protocols based on homomorphic encryption for hyperplane decision, naïve Bayesian, and decision trees. Liu *et al.* also presented

TABLE 4
Comparison Summary

Scheme	Range query	Textual keyword	Ranking model	Data mining	Online Diagnosis (Health monitoring)	# of rounds	Performance	Security
[3]	✓	✗	N/A	✗	✗	1	Medium	SCPA
[4]	✓	Func 1	N/A	✗	✗	1	Low	SCPA
[10]	✗	Func 2	tf-idf	✗	✗	1	High	KBA
[9]	✗	✗	N/A	✗	Binary decision tree	2	Low	SCPA
[7]	✗	✗	N/A	✗	SVM	3	Low	CPA
[5]	✗	✗	N/A	✓	Naïve Bayesian	2	Low	CPA
Our work	✓	Func 2	BM25	✓	Diverse algorithms	1	High	KPA

Notes. Func 1: Multi-keyword search;
Func 2: Multi-keyword ranked search.

secure multiparty protocols based on Paillier cryptosystem [26] for privacy-preserving naïve Bayesian [5] and single-layer neural network [6], which can help clinicians to securely diagnose the risk of patients' diseases. Similarly, Rahulamathavan *et al.* [7] achieved Support Vector Machine (SVM) for online diagnosis by using Paillier cryptosystem. In addition, Liu *et al.* [37] proposed a privacy-preserving reinforcement learning framework for a patient-centric dynamic treatment regime. Unfortunately, the computational and communication overhead of all these homomorphic encryption based schemes is huge. To improve efficiency, Zhu *et al.* [8] proposed a novel framework based on lightweight multi-party random masking and polynomial aggregation techniques that greatly improve the prediction efficiency without disclosing any sensitive medical information. Moreover, Zhang *et al.* [38] present an efficient and privacy-preserving disease prediction system. They trained prediction models by using Single-Layer Perceptron (SLP) learning algorithm and utilize random matrices to protect the privacy and facilitate secure outsourced computation of SLP. However, none of the above privacy-preserving online diagnosis is able to achieve secure search on encrypted data.

9 Conclusion

In this paper, we proposed practical techniques for diverse keyword search, data mining, and online pre-diagnosis over encrypted cloud data. By taking our scheme, the CS can provide range query and textual multi-keyword ranked search services for HU in a privacy-preserving way. Furthermore, HU could use big medical dataset stored in CS to train classifiers, and then applied the classifier for disease diagnosis without compromising the privacy of PU. Thorough privacy analysis and performance analysis demonstrated that our scheme is practicable. As a part of future work, we will continue to improve the security of our work.

Acknowledgments

This work was supported by the The Key Program of NSFC Grant (U1405255), Shaanxi Science & Technology Coordination & Innovation Project (2016TZC-G-6-3), the National Natural Science Foundation of China (No. 61702404, No.U1804263, No.61702105), the Fundamental Research Funds for the Central Universities (SA-ZD161504, No.JB171504), the China Postdoctoral Science Foundation (No. 2017M613080).

References

[1] W. Walker, A. P. Aroul, and D. Bhatia, "Mobile health monitoring systems," in *Proc. Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'09)*. IEEE, 2009, pp. 5199–5202.

[2] N. H. Ab Rahman and K.-K. R. Choo, "A survey of information security incident handling in the cloud," *Computers & Security*, vol. 49, pp. 45–69, 2015.

[3] X. Yao, Y. Lin, Q. Liu, and J. Zhang, "Privacy-preserving search over encrypted personal health record in multi-source cloud," *IEEE Access*, vol. 6, pp. 3809–3823, 2018.

[4] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in *2011 International Conference on Distributed Computing Systems (ICDCS'11)*, 2011, pp. 383–392.

[5] X. Liu, R. Lu, J. Ma, L. Chen, and B. Qin, "Privacy-preserving patient-centric clinical decision support system on naive bayesian classification," *IEEE journal of biomedical and health informatics*, vol. 20, no. 2, pp. 655–668, 2016.

[6] X. Liu, R. H. Deng, Y. Yang, H. N. Tran, and S. Zhong, "Hybrid privacy-preserving clinical decision support system in fog–cloud computing," *Future Generation Computer Systems*, 2017.

[7] Y. Rahulamathavan, S. Veluru, R. C.-W. Phan, J. A. Chambers, and M. Rajarajan, "Privacy-preserving clinical decision support system using gaussian kernel-based classification," *IEEE journal of biomedical and health informatics*, vol. 18, no. 1, pp. 56–66, 2014.

[8] H. Zhu, X. Liu, R. Lu, and H. Li, "Efficient and privacy-preserving online medical prediagnosis framework using nonlinear svm," *IEEE journal of biomedical and health informatics*, vol. 21, no. 3, pp. 838–850, 2017.

[9] H. Lin, J. Shao, C. Zhang, and Y. Fang, "Cam: cloud-assisted privacy preserving mobile health monitoring," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, pp. 985–997, 2013.

[10] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel & Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2014.

[11] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2016.

[12] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE transactions on parallel and distributed systems*, vol. 27, no. 9, pp. 2546–2559, 2016.

[13] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang, "Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1874–1884, 2017.

[14] X. Wang, J. Ma, Y. Miao, R. Yang, and Y. Chang, "EPSMD: an efficient privacy-preserving sensor data monitoring and online diagnosis system," in *Proc. the 37th IEEE Conference on Computer Communications (INFOCOM'18)*, 2018, pp. 819–827.

[15] K. P. Murphy, "Naive bayes classifiers," *University of British Columbia*, 2006.

[16] K. S. Jones, S. Walker, and S. E. Robertson, *A probabilistic model of information retrieval: development and comparative experiments Part 2*. Pergamon Press, Inc., 2000.

[17] J. R. Quinlan, “Induction on decision tree,” *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[18] C. Ecient and S. Ruggieri, “Efficient c4.5,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 438–444, 2000.

[19] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, “Secure knn computation on encrypted databases,” in *Proc. International Conference on Management of data (SIGMOD’09)*. ACM, 2009, pp. 139–152.

[20] H. Delfs and H. Knebl, *Introduction to Cryptography - Principles and Applications, Third Edition*, ser. Information Security and Cryptography. Springer, 2015.

[21] W. Lin, K. Wang, Z. Zhang, and H. Chen, “Revisiting security risks of asymmetric scalar product preserving encryption and its variants,” in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 1116–1125.

[22] B. Yao, F. Li, and X. Xiao, “Secure nearest neighbor revisited,” in *Proc. IEEE International Conference on Data Engineering (ICDE’13)*, 2013, pp. 733–744.

[23] C. Chen, X. Zhu, P. Shen, J. Hu, S. Guo, Z. Tari, and A. Y. Zomaya, “An efficient privacy-preserving ranked keyword search method,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 4, pp. 951–963, 2016.

[24] T. Qin and T. Liu, “Introducing LETOR 4.0 datasets,” *CoRR*, vol. abs/1306.2597, 2013. [Online]. Available: http://arxiv.org/abs/1306.2597

[25] J. Katz, A. Sahai, and B. Waters, “Predicate encryption supporting disjunctions, polynomial equations, and inner products,” in *Advances in Cryptology – EUROCRYPT ’08*, N. Smart, Ed., 2008, pp. 146–162.

[26] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *Proc. Advances in Cryptology — EUROCRYPT ’99*, J. Stern, Ed., 1999, pp. 223–238.

[27] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in *Proc. IEEE Symposium on Security and Privacy (S&P’00)*, 2000, pp. 44–55.

[28] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: improved definitions and efficient constructions,” vol. 19, no. 5. IOS Press, 2011, pp. 895–934.

[29] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, “Secure ranked keyword search over encrypted cloud data,” in *Proc. IEEE 30th International Conference on Distributed Computing Systems (ICDCS’10)*. IEEE, 2010, pp. 253–262.

[30] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, “Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking,” in *Proc. The 8th ACM Symposium on Information, Computer and Communications Security (ASIACCS’13)*. ACM, 2013, pp. 71–82.

[31] Z. Fu, X. Sun, Q. Liu, L. Zhou, and J. Shu, “Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing,” *IEICE Transactions on Communications*, vol. 98, no. 1, pp. 190–200, 2015.

[32] Y. Miao, J. Ma, X. Liu, X. Li, Q. Jiang, and J. Zhang, “Attribute-based keyword search over hierarchical data in cloud computing,” *IEEE Transactions on Services Computing*, 2017.

[33] H. Cui, Z. Wan, R. H. Deng, G. Wang, and Y. Li, “Efficient and expressive keyword search over encrypted data in cloud,” *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 3, pp. 409–422, 2018.

[34] H. Wang, J. Ning, X. Huang, G. Wei, G. S. Poh, and X. Liu, “Secure fine-grained encrypted keyword search for e-healthcare cloud,” *IEEE Transactions on Dependable and Secure Computing*, 2019.

[35] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, “Machine learning classification over encrypted data,” in *Proc. 22nd Annual Network and Distributed System Security Symposium (NDSS’15)*, 2015.

[36] E. Ayday, J. L. Raisaro, P. J. McLaren, J. Fellay, and J.-P. Hubaux, “Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data,” in *Proc. the 2013 USENIX Workshop on Health Information Technologies*. USENIX, 2013.

[37] X. Liu, R. Deng, K. R. Choo, and Y. Yang, “Privacy-preserving reinforcement learning design for patient-centric dynamic treatment regimes,” *IEEE Transactions on Emerging Topics in Computing*, 2019.

[38] C. Zhang, L. Zhu, C. Xu, and R. Lu, “PPDP: an efficient and privacy-preserving disease prediction scheme in cloud-based e-healthcare system,” *Future Generation Comp. Syst.*, vol. 79, pp. 16–25, 2018.

Xiangyu Wang currently is a Ph.D candidate in Xidian University. He received the B.E. degree with the School of Cyber Engineering from Xidian University, Shannxi, China, in 2017. His research interests include data security and secure computation outsourcing.

Jianfeng Ma received the B.S. degree in mathematics from Shaanxi Normal University, Xi’an, China, in 1985, and the M.S. degree and the Ph.D. degree in computer software and telecommunication engineering from Xidian University, Xi’an, China, in 1988 and 1995, respectively. He is currently a professor with the School of Cyber Engineering, Xidian University, Xi’an, China. He is also the Director of the Shaanxi Key Laboratory of Network and System Security. His current research interests include information and network security and mobile computing systems.

Yinbin Miao received the B.E. degree with the Department of Telecommunication Engineering from Jilin University, Changchun, China, in 2011, and Ph.D. degree with the Department of Telecommunication Engineering from xidian university, Xi’an, China, in 2016. He is currently a Lecturer with the Department of Cyber Engineering in Xidian university, Xi’an, China. His research interests include information security and applied cryptography.

Ximeng Liu (M’16) received the B.Sc. degree in electronic engineering from Xidian University, Xi’an, China, in 2010 and Ph.D. degrees in Cryptography from Xidian University, China, in 2015. Now, he is a full professor at College of Mathematics and Computer Science, Fuzhou University, China. Also,he is a research fellow at School of Information System, Singapore Management University, Singapore. He has published over 100 research articles include IEEE TIFS, TDSC, TC, TII, TSC and TCC. His research interests include cloud security, applied cryptography and big data security.

Ruikang Yang received the BE degree from the School of Cyber Engineering, Xidian University, Shannxi, China, in 2017. He is currently working towards the PhD degree in the Xidian University. His research interest includes data security and cloud computing security.