Secure Protocols for Privacy-preserving Outlier Detection Using Fully Homomorphic Encryption

Tushar Kanti Saha[1], Sakib Ahamed Shahon[2]

Abstract: Outlier detection is the technique of finding outliers or anomalies in a dataset. Outlier detection is an important data mining task to detect outliers because of its several application fields such as intrusion detection, credit card fraud, fraudulent financial transactions, environmental science, and many other machine learning applications. In this study, we consider a problem that an organization with less computational capability wants to detect the outliers from its categorical dataset using efficient data mining algorithms through outsourced computation. To address the above problems, we introduce two post-quantum secure protocols for detecting outliers from a dataset using attribute value frequency (AVF) and weighted attribute value frequency (WAVF) algorithms through outsourced computation. To ensure the security of the protocols, we use the modified technique of the CKK scheme (IEEE Trans. Inf. Forensics Security, 2016) with the BFV fully homomorphic encryption scheme since BFV's implementation works well for small plaintext moduli. In the end, we implement our protocols and evaluate their practical performance.

Keywords: Secure, Privacy-preserving, Protocol, Outlier, Detection

1. Introduction

In a dataset, an outlier is an object (point, observation) in a dataset that is significantly different from the other objects in that dataset. It is also known as an anomaly. Figure 1 shows an example of the outlier of a dataset in which outliers are pointed out by a circle. Outlier detection is the process of finding outliers or anomalies in a dataset. Outlier detection is an important data mining task because it finds several applications such as intrusion detection (Zhang & Zulkernine, 2006; García-Teodoro et al., 2009), credit-card fraud (El´ıas, 2011), fraudulent financial transactions (El´ıas, 2011), and environmental science (Garces & Sb´arbaro, 2011). Besides, the removal of anomalies in a dataset is an indispensable task before applying machine learning or data mining algorithms to that dataset to avoid erroneous results. In data mining, data are categorized into two types: quantitative (numerical)

[1] Dept. of Computer Science and Engineering, Jatiya Kabi Kazi Nazrul Islam University, Trishal, Mymensingh, Bangladesh. Email: tushar@jkkniu.edu.bd
[2] Dept. of Computer Science and Engineering, Jatiya Kabi Kazi Nazrul Islam University, Trishal, Mymensingh, Bangladesh. Email: sakib3201@gmail.com

and qualitative (categorical) data. For outlier detection from the categorical datasets, we consider marginal frequency-based approaches. This research considers outlier detection in categorical data because of its numerous practical applications. Some of those applications are network intrusion (Sarı, 2015), analyzing moving objects (Ge et al., 2010), medical health data analysis (Ieva & Paganoni, 2014), social networks (Aggarwal et al., 2011; Gao et al., 2010), credit fraud (Wang & Xu, 2018), questionnaire datasets (Zijlstra, et al., 2011), law enforcement (Lin & Brown, 2002), and so on. There exist several methods of finding outliers in categorical data. Some well-known methods among them are indicator variables, frequency-based, conditional frequency-based, density-based, clustering-based, and distance-based methods (Taha & Hadi, 2019). Among these methods, frequency-based outlier detection methods use the frequency of data or itemset instead of distances. Several works such as attribute value frequency (AVF) (Koufakou et al., 2007) and weighted attribute value frequency (WAVF) (Rokhman et al., 2016) have been proposed using the marginal frequency method which uses data item frequency rather than itemset frequency from a categorical dataset. For outlier detection from a categorical dataset, this study considers the private outsourced computation of marginal frequency-based outlier detection techniques rather than frequent itemset-based techniques (He et al., 2005; Otey et al., 2006) since the execution time of the frequent itemset-based techniques is exponential to the number of categorical attributes of that dataset.

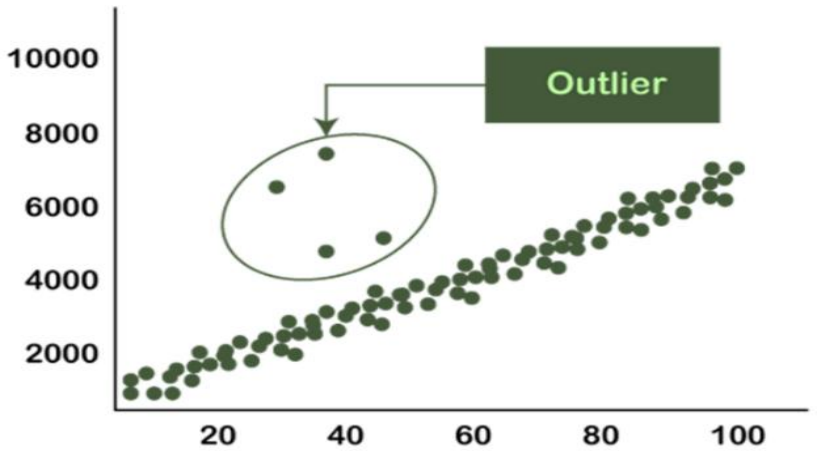

Figure 1: Example of Outlier in a dataset

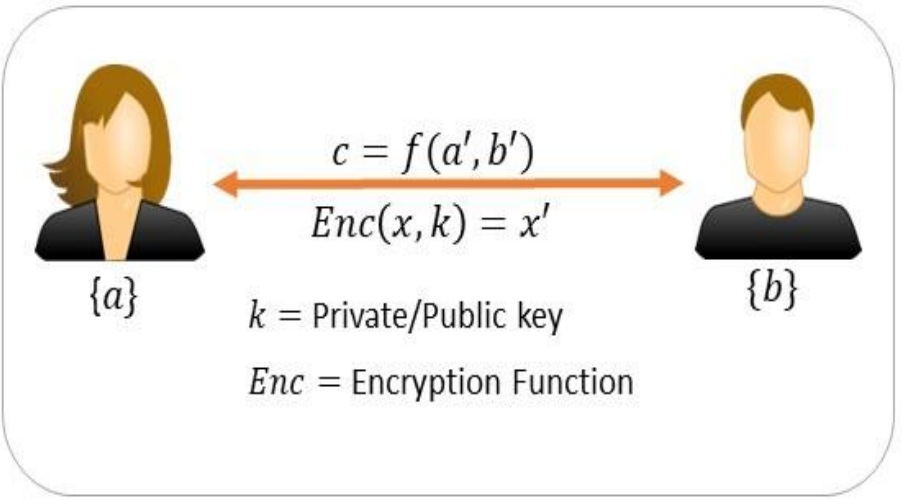

Figure 2: Example of two-party private computation

1.1 Privacy-preserving Computation

The process of computing a mathematical function securely with private inputs from two or more parties is called privacy-preserving computation or private computation or secure computation. In this subsection, we shall discuss several scenarios related to secure computation. First, consider the millionaire's problem in Figure 2 between Alice and Bob where they want to know who is rich without disclosing their wealth to each other (Yao, 1982). If they possess the wealth of a and b respectively. Here they want to use a comparison function $f(a', b')$ to securely find out the large value between the two, where a' and b' are the encrypted values of a and b respectively. Now we show the procedure of outsourcing such computation to the cloud in the following subsections.

1.2 Outsourcing Privacy-preserving Computation

In Figure 3, we show an example of outsourcing a two-party computation to the cloud. In the two-party computation, Alice, who has less computation capability, needs to know the summation of two numbers. In this case, she can outsource this computation to another party such as Bob in the cloud. However, Alice does not want to disclose his data to the cloud. In this case, she encrypts the data and outsources the data to the cloud so that the cloud can compute over encrypted data using the homomorphism property (Rivest & Dertouzos, 1978) and sends back the encrypted result to Alice. Then Alice can decrypt the encrypted result using his key to get his required result.

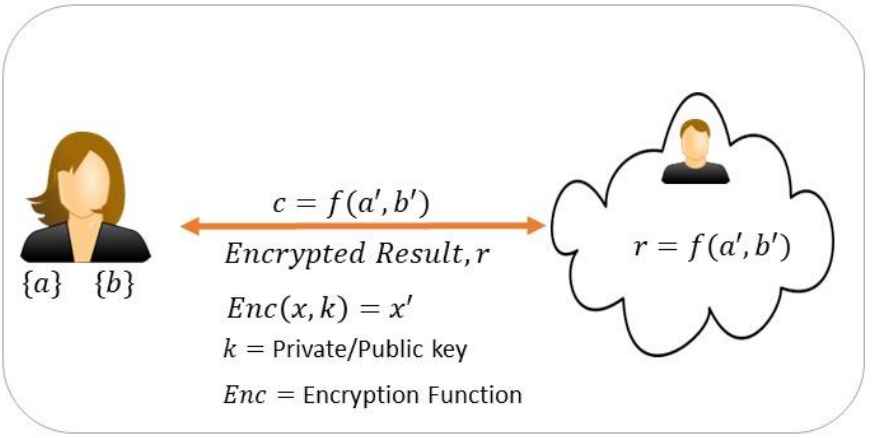

Figure 3: Example of outsourcing a two-party private computation to the cloud.

1.3 Privacy-preserving Outlier Detection Techniques

Privacy-preserving outlier detection (PPoD) is a subfield of data mining that focuses on detecting outliers in data while preserving the privacy of the data of the data owner. This is important in many applications such as healthcare, finance, and government, where the data may contain sensitive information about individuals. There are several approaches to performing the PPoD. One common approach is to add noise to the data before it is analyzed. This noise can make it more difficult to identify outliers, but the noise also makes it more difficult for anyone to learn anything about the individual data points. Another approach to the PPoD is to use secure multi-party computation (SMC). SMC allows multiple parties to jointly analyze data without revealing any of their individual data points. This can be used to detect outliers in data that is distributed across multiple organizations.

Furthermore, homomorphic encryption is a type of encryption that allows computations to be performed on encrypted data without decrypting them first. This can be used to detect outliers in data without decrypting the data, which can help to preserve the privacy of those data.

The PPoD is a relatively new field of research, but it has already made significant progress. There are now several effective PPoD algorithms available, and the field is continuing to grow. As the amount of data that is being collected and stored continues to grow, the PPoD will become increasingly important.

1.4 Background

The traditional way for outlier detection from a dataset is to use the data mining algorithms locally. It is a huge computational task for any

organization that needs outlier detection. To avoid such a burden of computation, users or organizations can outsource such computations to a distant third party like the cloud. However, it is hard to trust such a distant party. Therefore, secure outsourced computation is indispensable here. In 2019, Yang et al. showed several methods of secure outsourced computation such as secure multi-party computation (SMPC), pseudorandom functions (PRF), pseudorandom permutation (PRP), software guard extensions (SGX), perturbation-based method (PBM), and homomorphic encryption (HE) (Yang, 2019). Here SMPC requires many interactions among the parties participating in the private computation. For this reason, the communication complexity of this method is high. SGX is susceptible to certain software and physical attacks. PBM, PRP, and PRF have limited applications with low security levels. Moreover, none of these methods ensure post-quantum level security. If we use conventional encryption schemes without homomorphism property (Rivest & Dertouzos, 1978), it would be hard to compute a function on encrypted data. Besides, common encryption techniques such as deterministic encryption (Bellare, 2008), searchable encryption (Bellare, 2007), and order-preserving encryption (Boldyreva et al., 2009) are adopted for computing particular functions on the data. Therefore, an encryption scheme with homomorphism property can be a solution to the problem.

Homomorphic encryption was first coined by Rivest et al. in 1978 (Rivest & Dertouzos, 1978). Earlier homomorphic encryption schemes can perform a restricted number of operations on encrypted data. They are RSA cryptosystem (Rivest & Dertouzos, 1978), Goldwasser-Micali cryptosystem (Goldwasser & Micali, 2019), ElGamal cryptosystem (ElGamal, 1985), Benaloh cryptosystem (Benaloh, 1994), Paillier cryptosystem (Paillier, 1999), and Boneh-Goh-Nissim cryptosystem (Boneh et al., 2005). In 2009, Gentry devised a method called the fully homomorphic encryption scheme (FHE) which allows an unlimited number of additions and multiplications on encrypted data (Gentry, 2009). However, the efficiency of the scheme was not enough for practical use (Hu, 2013). Gentry developed the FHE scheme by modifying a somewhat homomorphic encryption scheme (SWHE) using bootstrapping. To increase the efficiency of the Gentry's FHE scheme, several FHE cryptosystems have been literally developed such as BGV (Brakerski et al., 2012), BFV (Brakerski, 2012; Fan & Vercauteren, 2012), TFHE (Chillotti, 2016), and CKKS (Cheon et al., 2017). In recent years, Saha et al. showed several privacy-preserving applications using somewhat homomorphic encryption (SwHE) scheme along with new data packing

methods for efficient computations which can be used in secure computation with FHE (Saha & Koshiba, 2017; Saha et al., 2018; Saha & Koshiba, 2018; Saha & Koshiba, 2018b; Saha & Koshiba, 2016; Saha & Koshiba, 2017; Saha & Koshiba, 2021). One of the above two homomorphic schemes can be used in private computations depending on the application areas and nature of those private computations.

To our knowledge, privacy-preserving outlier detection (PPoD) was first proposed by Vaidya and Clifton (2004). Here they presented a privacy-preserving technique to find distance-based outliers in distributed data. However, they require a communication complexity of $O(n^2)$. After that, Shaneck et al. (2006) addressed a privacy-preserving nearest neighbor search that applied local outlier factor-based outlier detection in horizontally partitioned data. But their approach leaked intermediate neighborhood information. Zhengyou et al. (2009) proposed two privacy-preserving distance-based outlier detection algorithms over vertically partitioned data. To maintain the privacy of the computation, they used the Paillier cryptosystem which can perform only addition on the encrypted data. In 2017, Bohler et al. (2017) proposed an algorithm using differential privacy, which allows private data perturbation of sensor streams for outlier detection. Here they obtained an accuracy of 80%, which should be improved. In 2021, Lu et al. (2021) proposed a privacy-preserving isolation forest (PIF) to detect outliers for multiple distributed data providers. Here they did not mention communication and encryption or decryption overhead. Recently, Itokazu et al. (2021) proposed the pp-iForest algorithm which extended the PIF algorithm for outlier detection for multiple organizations by employing an additively homomorphic encryption scheme with a federated learning scheme. This algorithm works well if the number of organizations is small. To our knowledge, we did not find any privacy-preserving protocol to ensure post-quantum security for detecting outliers from categorical data using marginal frequency-based methods. Therefore, new protocols for AFV and WAVF methods using fully homomorphic encryption are indispensable to estimate their efficiencies in the secured domain.

1.6 Problem Statements

Suppose an organization with less computation capability has several categorical datasets such as breast cancer, lymphography, post-operative data, and so on, where they want to acquire some knowledge from the dataset by applying some machine learning or data mining or statistical measures. Here they need to remove noise as a part of preprocessing before applying data

mining or machine learning algorithms to these data. To filter these data, the organization wants to detect the outliers in these data by using efficient outlier detection algorithms.

1.7 Our Contribution

To address the above problem of outlier detection computation in the categorical data using efficient algorithms, we propose:

- a fully blind PPoD protocol using AVF and WAVF algorithms without sending any query from the data owner.
- another fully blind PPoD protocol using the same algorithms with queries from the data owner.

To ensure post-quantum level security of the protocol, we use the modified technique of (Cheon et al., 2016) with the BFV encryption scheme (Brakerski, 2012; Fan & Vercauteren, 2012) since BFV implementation is faster for small plaintext moduli (Kim et al., 2021).

2. Preliminaries

To ensure secure computation in our protocols, we shall use a fully homomorphic encryption scheme (Brakerski, 2012; Fan & Vercauteren, 2012) which is reviewed as follows.

2.1 Fully Homomorphic Encryption Scheme

In this section, we discuss the parameters, algorithms of key generation, encryption, homomorphic evaluation, and decryption used for the BFV encryption scheme mentioned in (Brakerski, 2012; Fan & Vercauteren, 2012).

2.1.1 Parameters

To illustrate the FHE scheme, we consider some parameters as follows. Let λ be the security parameter. $\phi(x)$ denotes a cyclotomic polynomial where $\phi(x) = x^n + 1$ where $n = 2^\mu$ such that $\mu \in \mathbb{Z}$. q is a coefficient modulus in the ciphertext space. Similarly, t is an integer defining plaintext modulus such that $t < q$. δ denotes the standard deviation where δ for a discrete Gaussian error distribution χ. $R = \mathbb{Z}[x]/\phi(x)$, the ring of integer polynomials modulo $\phi(x)$. $R_t = \mathbb{Z}_t[x]/\phi(x)$, the ring of integer polynomials modulo $\phi(x)$ and t.

2.1.2 Key Generation

Sample the secret key $sk = s$ such that $s \xleftarrow{\$} R_2$. Then, it samples a uniformly random element $a_1 \xleftarrow{\$} R_q$ and an error $e \leftarrow \chi$. Finally, it outputs a secret key s

and its public key $pk = (\rho_0, \rho_1) = [-(as + e)]_q, a$. The scheme requires an evaluation key ek. To generate this sample $a_i \xleftarrow{\$} R_q$ and $e \leftarrow \chi$ for $i = \{0,1,2, \dots, l\}$ and $ek = \left(\left[-(a_i s + e_i), w^i s^2\right]_q, a_i\right)$.

2.1.3 Encryption

If we have a message $m \in R_t$ and a public key $pk = (a_0, a_1)$, the encryption algorithm samples $e_1, e_2 \leftarrow \chi$, and $u \xleftarrow{\$} R_2$. Then the encryption algorithm outputs a ciphertext $(c_0, c_1) = ct$ which is defined as follows:

$$Enc(m, pk) = (c_0, c_1) = \rho_0 u + e_1 + \Delta mq, \rho_1 u + e_2 q. \quad (1)$$

Here, the plaintext $m \in R_t$ is also in R_q because $t < q$ and $\Delta = \lfloor q/t \rfloor$.

2.1.4 Homomorphic Evaluations

Generally, a homomorphic operation happens between the two ciphertexts $ct = (c_0, c_1)$ and $ct' = (c_0', c_1')$. The homomorphic addition ($\boxplus$) and homomorphic multiplication ($\boxdot$) between ct and ct' can be defined by

$$ct_{add} = ct \boxplus ct' = (c_0 + c_0', c_1 + c_1') \quad (2)$$

and

$$ct_{mul} = ct \boxdot ct' = (c_0, c_1) \boxdot (c_0', c_1'), \quad (3)$$

where $\boldsymbol{c_0} = \left[\left\lfloor \frac{t}{q} c_0 c_0' \right\rceil\right]_q$, $\boldsymbol{c_1} = \left[\left\lfloor \frac{t}{q} c_0 c_1' + c_1 c_0' \right\rceil\right]_q$, and $\boldsymbol{c_2} = \left[\left\lfloor \frac{t}{q} c_1 c_1' \right\rceil\right]_q$.

To relinearize ciphertext from three elements to two elements, express c_2 using the base w as $c_2 = \sum_{i=0}^{l} c_2^{(i)} w^i$. Using the evaluation key, we get relinearized ciphertext as follows:

$$\boldsymbol{c}_0 = c_0 + \sum_{i=0}^{l} ek[i][0] c_2^{(i)} \quad (4)$$

and

$$\boldsymbol{c}_1 = c_1 + \sum_{i=0}^{l} ek[i][1] c_2^{(i)}. \quad (5)$$

2.1.5 Decryption

The decryption algorithm decrypts a new or homomorphically operated ciphertext $ct = \left(c_0, \dots, c_\psi\right)$ in the following ways:

$$Dec(ct, sk) = [\widetilde{m}]_q \bmod t,$$

where $\sum_{i=0}^{\psi} c_i s^i$. That means the algorithm uses the secret key vector $\boldsymbol{s} = \left(1, s, s^2, \dots, s^\psi\right)$ such that $Dec(ct, sk) = [ct, \boldsymbol{s}]_q \bmod t$. For example, the

decryption of a fresh ciphertext $ct = (c_0, c_1)$ generated by Eq. (1) can be written as

$$\langle ct, s \rangle = \left[\frac{t}{q}[(a_0 u + e_1 + \Delta m) - (a_1 u + e_2) \cdot s]_q\right]_t = m$$

3. Methodology

In this study, we consider the outlier detection technique for categorical data. The categorical data are classified into two categories: nominal and ordinal data. Taha and Hadi (2019) reviewed several methods of outlier detection from categorical data. The methods are based on marginal frequency, itemset frequency, diversified frequency, and conditional frequency. Besides, others are density-based methods, clustering-based methods, distance-based methods, Shanon entropy, holo entropy, and so on. Here we use the attribute value frequency (AVF) and weighted attribute value frequency (WAVF) which are based on the marginal frequency of the data item.

3.1 Marginal Frequency-based Outlier Detection Techniques

To discuss AVF and WAVF methods, assume that a dataset contains n tuples with m attributes. Each record can be represented by x_i where $1 \leq i \leq n$ and $x_{i,j}$ indicates the value of the j-th attribute of the record x_i. Now let us discuss the methods as follows.

Table 1: Example of a categorical dataset.

Record number	ω_1	ω_2	ω_3
x_1	A	E	M
x_2	A	D	N
x_3	B	G	M
x_4	C	D	N
x_5	C	G	M
x_6	C	F	N

3.1.1 Attribute Value Frequency (AVF) Method

Koufakou et al. (2007) proposed a scalable and effective attribute value frequency (AVF) algorithm to detect outliers in categorical datasets. Their method performs better than existing methods to detect outliers from breast cancer, lymphography, and postoperative patients' datasets. They define the AVF score as

$$AVF(x_i) = \frac{1}{m}\sum_{j=1}^{m} f(x_{ij}), \qquad (7)$$

where $1 \leq i \leq n$ and $f(x_{i,j})$ is the number of times j-th attribute value of x_i appears in a dataset containing n records. Here record x_i is an outlier if it has

the lowest AVF score in the dataset. This algorithm considers δ objects as outliers in which δ is defined by the user known as threshold. Let us consider a categorical dataset with $m = 3$ attributes $\omega_1, \omega_2, \omega_3$ as shown in Table 1. This dataset contains six (n = 6) records $x_1, x_2, \ldots, x_6$. Now we calculate the AVF score according to Eq. (7), which is shown in Table 2. From this Table, it is justifiable that the records x_1 and x_3 are outliers because they have the lowest AVF scores among all records in the given dataset if $\delta = 2$.

Remark 1 *The AVF method does not consider the sparseness of the frequencies in the categorical variable. Furthermore, the performance of these methods can be improved by adding weight to the frequencies.*

Table 2: AVF(x_i) and WAVF(x_i) derived from Table 1 using $f(x_{i,j})$

Record Number	ω1	ω2	ω3	AVF(x_i)	WAVF(x_i)
x_1	2	1	3	2	1.67
x_2	2	2	3	2.33	2
x_3	1	2	3	2	1.33
x_4	3	2	3	2.67	2.67
x_5	3	2	3	2.67	2.67
x_6	3	1	3	2.33	2.33

3.1.2 Weighted Attribute Value Frequency (WAVF) Method

To consider the sparseness of the frequencies in the categorical variable, Rokhman et al. (2016) proposed a weighted attribute value frequency (WAVF) algorithm by extending the AVF method as

$$WAVF(x_i) = \frac{1}{m}\sum_{j=1}^{m} f(x_{ij}) \times R_j, \qquad (8)$$

where $1 \leq i \leq n$ and $R_j = max(\omega_j) - min(\omega_j)$ is the range of frequencies of the j-th categorical variable X_j. Using Eq. (8), we can easily obtain the WAVF score of the dataset shown in Table 1 and determine the outliers of this dataset. From Table 2, it is estimated that record x_1 and x_3 are outliers since we set the threshold $\delta = 2$.

3.2 Proposed Protocols

Now we show our protocols for privacy-preserving outlier detection from categorical dataset using AVF and WAVF algorithms through secure outsourced computation in the following subsections.

3.2.1 Protocol Scenario

Suppose that a medical college maintains a database of its admitted patients. Now it owns a large dataset from which it wants to apply some machine learning computations or statistical measures to those data to gain some useful knowledge from the dataset. However, these data contain outliers that were generated due to software faults or wrong inputs of the users. As a part of preprocessing, the outliers need to be removed which is the precondition of machine learning or statistical measure to generate error-free results. In addition, the medical college does not have such an expert or the software by which it is possible to remove those outliers. Moreover, the medical college authority wants to access these data remotely. The possible solution to enable remote access to those data is to upload these data to a trusted cloud server. Nonetheless, such a trusted server is difficult to find in the real world. At the same time, it is required to apply some outlier detection techniques such as AVF and WAVF before applying any machine learning algorithms or statistical tools to gain some useful information from these outsourced data. Therefore, computation over encrypted data using the BFV encryption scheme mentioned in (Brakerski, 2012; Fan & Vercauteren, 2012) can be a solution to the problem.

3.2.2 Required Circuit for Computing the Equality

From Table 1 and above AVF and WAVF algorithms, it is obvious that we need an equality circuit for calculating the frequency of data from the encrypted dataset. Here we show the equality circuit mentioned in (Cheon et al., 2016) required for determining the frequency $f(x_{ij})$ as mentioned in Eq. (7) and Eq. (8), which is used in our PPoD protocol. Consider two integers a and b such that $a = (a_1, \ldots, a_l)$ and $b = (b_1, \ldots, b_l)$ where l denotes the number of bits. To determine the equality between a and b, the equality circuit can be written by the equation as follows:

$$\bar{d} = \prod_{i=1}^{l} \left(1 - \left(\bar{a} + \bar{b} - 2 \cdot \bar{a} \cdot \bar{b}\right)\right) = \begin{cases} 1, & a = b \\ 0, & otherwise \end{cases} \quad (9)$$

where $\bar{a}$, $\bar{b}$, and $\bar{d}$ are encrypted values of integers a, b, and the equality flag d respectively obtained through Eq. (1). After decryption, $d = 1$ indicates that $a = b$; otherwise, $a \neq b$.

3.2.3 Underlying Techniques

To reduce the cost of the circuit over the ring of integers $\mathbb{Z}_t$ mentioned in Sect. 2.1, we consider the following solution in which the query initiator will count the frequency according to the following algorithms. First, we focus on

the AVF algorithm. To count the frequency of X_{ij} as mentioned in Eq. (7), the value of X_{ij} is $\alpha_{i,j} = (a_1, \dots, a_l)$ which will be compared with every value $\beta_{i,j}$ at j-th column to determine equality where $1 \leq i \leq n$. In this respect, we show the required AVF algorithm supporting homomorphic computation as follows.

AVF Algorithm

Input: $\alpha_{i,j} = (a_1, \dots, a_l)$ and $\beta_{i,j} = (b_1, \dots, b_l)$

AVF circuit:

$$AVF(x_i) = \frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{n}\bar{d}_{i,j}, \qquad (10)$$

where l denotes the number of bits in $\alpha_{i,j}$ and $\beta_{i,j}$; the equality flag $\bar{d}$ is determined by Eq. (9) where two inputs are $\alpha_{i,j}$ and $\beta_{i,j}$.

Output: $\{\text{AVF}(x_i) | 1 \leq i \leq n\}$

From the above algorithm, we obtain the value of $AVF(x_i)$ from the two l-bit integers $\alpha_{i,j}$ and $\beta_{i,j}$ using Eq. (9). Here $AVF(x_i)$ can be determined through encrypted computation using the homomorphism property of addition and multiplication mentioned in Eq. (2) and Eq. (3). Now we consider the WAVF method mentioned in Sect. 3.1.2 whose algorithm is shown as follows.

WAVF Algorithm

Input: $\alpha_{i,j} = (a_1, \dots, a_l)$ and $\beta_{i,j} = (b_1, \dots, b_l)$

WAVF circuit:

$$WAVF(x_i) = \frac{1}{m}\sum_{j=1}^{m}\left(\sum_{i=1}^{n}\bar{d}_{i,j}\right) \times \bar{R}_j, \qquad (11)$$

where l denotes the number of bits in $\alpha_{i,j}$ and $\beta_{i,j}$; In addition, $\bar{d}$ is computed according to Eq. (9) and $\bar{R}_j$ is the range of frequencies encrypted by Eq. (1) of the j-th attribute in the dataset.

Output: $\{\text{WAVF}(x_i) | 1 \leq i \leq n\}$

In the WAVF algorithm, we see that $WAVF(x_i)$ can be obtained easily using Eq. (11) through encrypted computation using homomorphism property of addition and multiplication mentioned in Eq. (2) and Eq. (3).

3.3 Our Protocols

In this section, we show our privacy-preserving outlier detection (PPoD) protocols. Considering the same scenario mentioned in Section 3.2.1, let Alice and Bob are the representatives of the medical college hospital and cloud respectively for narrating the protocol. The hospital authority (Alice) has a dataset. Now Alice wants to detect outliers using AVF and WAVF techniques from his dataset through some secure outsourced computation. We know that AVF and WAVF scores require counting the frequency of every data that appeared in the dataset. The frequency can be counted from the dataset by sending queries from the data owner or without sending any query from the data owner. In this respect, we proposed two protocols for the computations as follows.

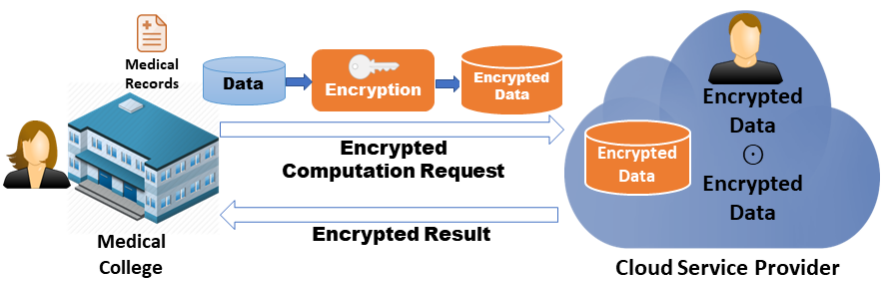

Figure 4: PPoD protocol for blind computation without any queries

3.3.1 PPoD Protocol without Any Query

In this protocol, we perform secure computation of the frequency required to find out AVF and WAVF scores without sending any queries from the data owner to the cloud. For this reason, we name this computation a fully blind computation without any query. Following the scenario mentioned in Section 3.2.1, our protocol (see Figure 4) works as follows.

1. First, Alice employs the key generation algorithm of the BFV's FHE to generate the public key, private key, and relinearization key and store them secretly.
2. Then Alice separates the required data from the hospital database. Then she encrypts the data through Eq. (1) using the public key and sends the corresponding record to Bob in the cloud.
3. Then Alice sends an encrypted computation request to Bob to find AVF and WAVF scores according to Eq. (10) and Eq. (11) respectively. Here Bob homomorphically counts the frequency of data using the

equality circuit mentioned in Eq. (9) and calculates the AVF and WAVF scores from the encrypted data.

4. Bob then sends back the encrypted result $\bar{r}$ to Alice to determine the actual AVF and WAVF scores of every row after decryption.
5. Bob then decrypts the result using Eq. (6) and determines the lowest AVF and WAVF scores as outliers in the dataset.

In this protocol, Bob needs to determine the frequency of every value of all attributes that appeared in the dataset blindly because he does not receive any prior knowledge about those values from the data owner. To mitigate this limitation, we propose another protocol that will be discussed in the next subsection.

3.3.2 PPoD Protocol with Queries

From the dataset shown in Table 1 and its frequency counting in Table 2, it is obvious that the data owner can determine attribute values whose frequencies need to be counted blindly by Bob in the cloud. Therefore, the data owner sends those values as queries to Bob in the cloud to calculate their frequencies from the dataset blindly. Since this protocol accomplishes secure computation of counting frequency required to determine AVF and WAVF scores by sending queries from the data owner to the cloud, we name this computation a fully blind computation with queries. Following the scenario mentioned in Section 3.2.1, this protocol shown in Figure 5 works as follows.

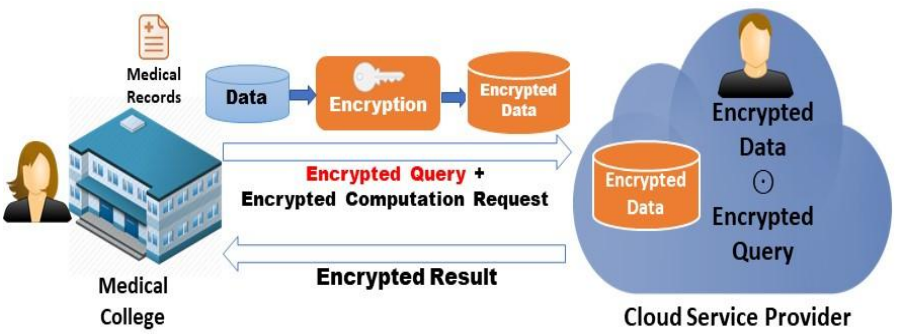

Figure 5: PPoD protocol for blind computation with queries

1. Like our first protocol, Alice employs the same key generation algorithm of the BFV's FHE to generate the public key, private key, and relinearization key and keep them secret.
2. Then Alice separates the necessary data from the hospital database for outliers' computation. According to Eq. (1) she encrypts the data using the public key and sends the encrypted data to Bob in the cloud.

3. Next, Alice determines the attribute values whose frequencies are needed to determine and send those values as queries to Bob in the cloud.
4. After this, Alice sends an encrypted computation request to Bob to determine AVF and WAVF scores according to Eq. (10) and Eq. (11) respectively. Here Bob homomorphically counts the frequency of data using the equality circuit mentioned in Eq. (9) and calculates the AVF and WAVF scores from the encrypted data.
5. Bob then sends back the encrypted result $\bar{r}$ to Alice to determine the actual AVF and WAVF scores of every row after decryption.
6. Bob then decrypts the obtained result through Eq. (6) and determines the lowest AVF and WAVF scorers as outliers in the dataset.

Remark 2: *Though our protocols reveal the computation circuits of equality, Bob in the cloud does not learn anything about the data and results due to encrypted computation except the circuits. Therefore, we claim that our protocols are secure in the semi-honest model whose security is ensured by the FHE mentioned in Sect.2.1.*

4. Results and Discussion

In this section, we show the experimental results of the two protocols mentioned in Sect.3.3.1 and Sect.3.3.2. We also show a comparative performance analysis between the two protocols.

4.1 Data Source

We collected the data of the experiment from the Lymphography dataset of the UCI machine learning repository (Zwitter & Soklic, 1988). We took five attributes of the data for our experiment in which the attributes are nominal. Moreover, we consider record sizes of 10, 25, and 50 for our experiments on the two protocols.

4.2 Parameter Settings

We have written the required code in C++ programming language with the Microsoft Simple Encrypted Arithmetic Library (SEAL) library 3.7.0 (Microsoft, n.d.). Then we ran the code on a single 64-bit machine (1.8 GHz Intel core–i7 CPU and 8 GB RAM) in a Windows environment. We took the library for the experiment with BFV's FHE scheme. We set the lattice dimension n and plaintext modulus t as $(n, t) = (16384, 1024)$. We know that

the FHE standardization consortium set the values of the parameters used in the FHE encryption schemes to achieve a security level of 128-bit (Albrecht et al., 2019). According to (Albrecht et al., 2019), our protocol achieves a security level of 128-bit from our parameters' settings because of using BFV's FHE, which ensures post-quantum security of the protocol.

Table 3: Performances of the PPoD protocol without any query

Number of records	Frequency counting time (sec.)	AVF score calculation time (sec.)	Total time (Frequency counting+ AVF time) (sec.)	WAVF score calculation time (sec.)	Total time (Frequency counting + WAVF time) (sec.)
10	778.226	0.157	778.383	6.912	785.295
25	5743.2	0.506	5743.706	15.553	5759.259
50	19004.3	4.405	19008.705	31.445	19040.15

4.3 Performance Analysis of Our Protocols

In this section, we discuss the practical performances of our two PPoD protocols.

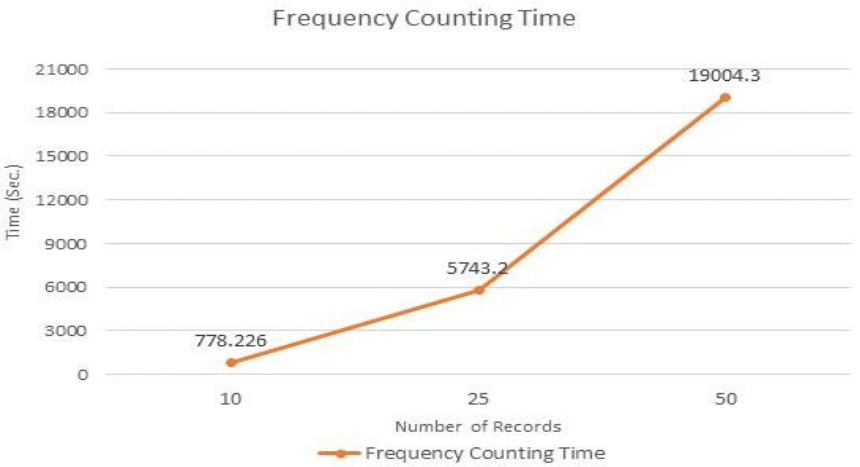

Figure 6: Frequency computation time of the PPoD protocol without any query from the data owner having record sizes of 10, 25, and 50

4.3.1 PPoD Protocol without Any Query

We show the performances of our PPoD protocol without any query from the data owner for the dataset having 10, 25, and 50 records in Table 3. Here we show the time required for frequency counting, AVF score calculation, and WAVF score calculation. Also, we show the total time of frequency counting

and AVF score computation. Moreover, we show the total time required for frequency counting and WAVF score computation. We plot the frequency calculation time against the number of records and show it in Figure 6. From this figure, we see that the time required for calculating the frequency for 10, 25, and 50 records are 778.226 sec., 5743.2 sec., and 19004.3 sec. respectively. Here it is obvious that the frequency counts work well when the number of records is as low as 10. However, the computation time increases exponentially with the increase in the number of records.

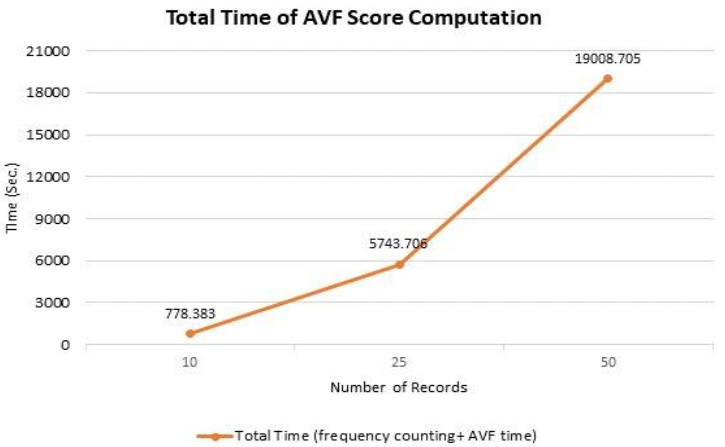

Figure 7: Total time required for AVF score computation of the PPoD protocol without any query from the data owner having record sizes of 10, 25, and 50

Moreover, the total time required for AVF score calculation is shown in Figure 7 according to Eq. (10). Here the total time is the sum of frequency counting time and AVF score calculation time using the frequencies. As shown in Table 3, the AVF score calculation time is 0.157 sec., 0.506 sec., and 4.405 sec. for record sizes 10, 25, and 50 respectively, whereas the total time of AVF score calculation is 778.383 sec., 5743.706 sec., and 19008.705 sec. respectively for the same record sizes. Here it is obvious that most of the time in AVF score computation is consumed in the frequency calculations.

In addition to this, the total time required for WAVF score calculation is shown in Figure 8 according to Eq. (11). Here the total time is the sum of frequency counting time and WAVF score calculation time using those frequencies. As shown in Table 3, the WAVF score calculation time is 6.912 sec., 15.553 sec., and 31.445 sec. for record sizes 10, 25, and 50 respectively. From the above performances, it is evident that the total time is very close to the frequency computation time. Therefore, we can conclude that most of the time is spent on frequency computation required for AVF and WAVF algorithms used in the PPoD protocol without any query.

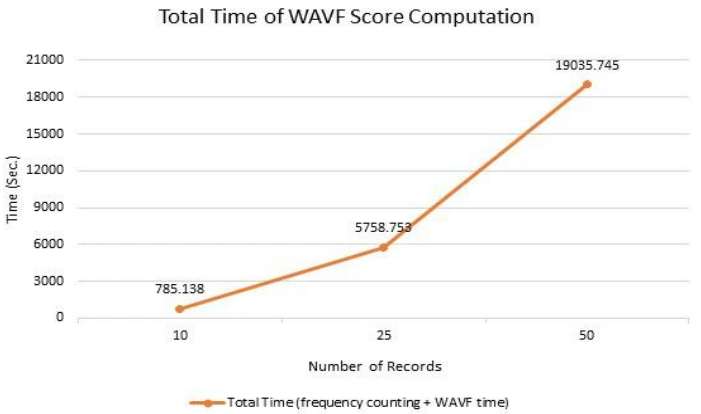

Figure 8: Total time required for WAVF score computation of the PPoD protocol without any query from the data owner having record sizes of 10, 25, and 50

Table 4: Performances of the PPoD protocol with queries from the data owner having record sizes of 10, 25, and 50

Number of records	Frequency counting time (sec.)	AVF score calculation time (sec.)	Total time (Frequency counting + AVF time) (sec.)	WAVF score calculation time (sec.)	Total time (Frequency counting + WAVF time) (sec.)
10	120.095	0.15	120.245	6.374	126.469
25	276.183	0.354	276.537	15.497	291.68
50	579.702	0.841	580.543	30.993	610.695

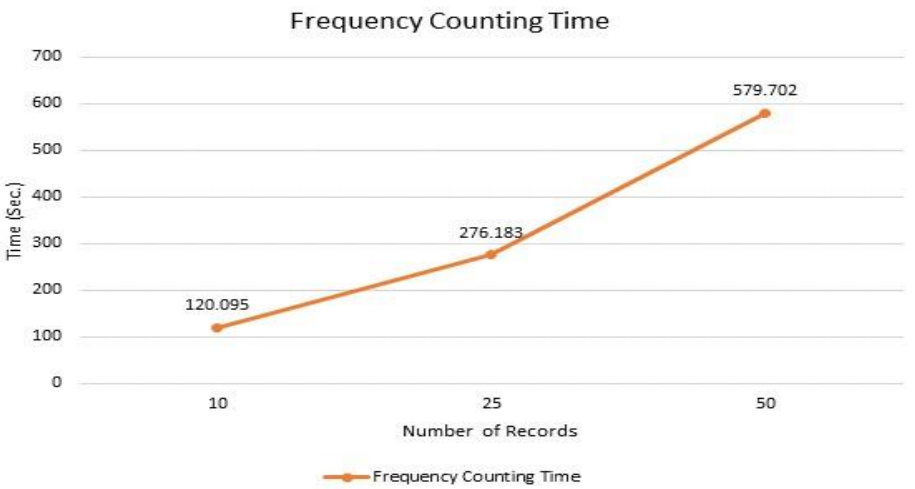

Figure 9: Frequency computation time of the PPoD protocol with queries from the data owner having record sizes of 10, 25, and 50

4.3.2 PPoD Protocol with Queries

To reduce frequency counting costs, we proposed another PPoD protocol using queries sent from the data owner for determining their frequencies in the dataset as mentioned in Sect. 3.3.2. The practical performance of this protocol for determining AVF and WAVF scores for record sizes of 10, 25, and 50 is shown in Table 4. Here we present the time required for frequency counting using queries from the data owner, AVF score calculation, and WAVF score calculation. Also, we show the total time of frequency counting and AVF score computation. Moreover, we show the total time required for frequency counting and WAVF score computation. We plot the frequency calculation time against the number of records and show it in Figure 9. As shown in Figure 9, we see that the time required for calculating the frequency for 10, 25, and 50 records are 120.095 sec., 276.183 sec., and 579.702 sec. respectively. Here it is obvious that the frequency counts work well when the number of records is as low as 10. However, the computation time is not increasing exponentially with the increase in the number of records that we also observed in the PPoD protocol without any query mentioned in Sect. 4.3.1.

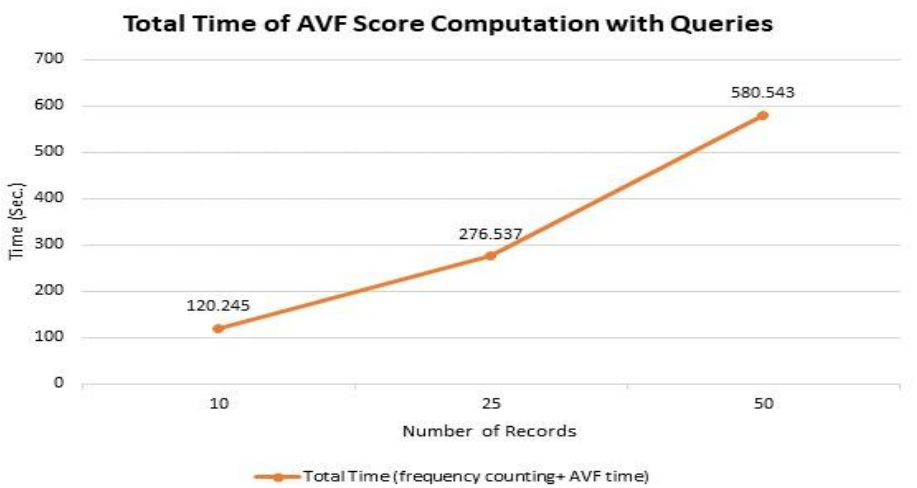

Figure 10: Total time required for AVF score computation of the PPoD protocol with queries from the data owner having record sizes of 10, 25, and 50

According to Eq. (10), the total time required for AVF score calculation with queries sent from the data owner is shown in Figure 10 in which the total time of AVF score calculation is 120.245 sec., 276.537 sec., and 580.543 sec. for record sizes of 10, 25, and 50 respectively. Here the total time is measured by summing up frequency counting time with AVF score calculation time using those frequencies. As shown in Table 4, only AVF

score calculation time is 0.150 sec., 0.354 sec., and 0.841 sec. for record sizes of 10, 25, and 50 respectively.

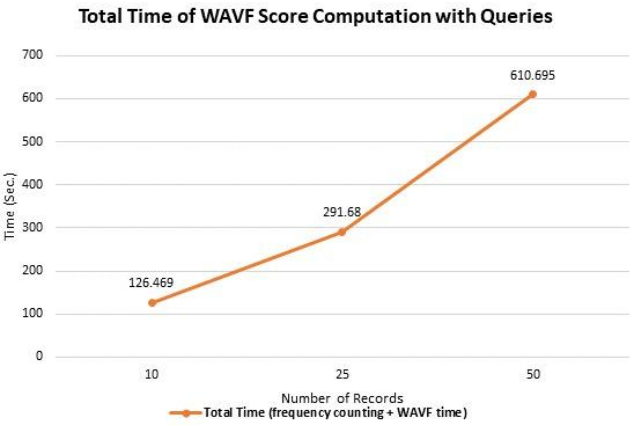

Figure 11: Total time required for WAVF score computation of the PPoD protocol with frequency and range queries from the data owner having record sizes of 10, 25, and 50

According to Eq. (11), it is obvious that the range and frequencies of different values of an attribute are required to determine the WAVF score. To determine the range and frequencies of different values from the dataset in our protocol, the data owner sends queries to the cloud to reduce the cost of the computation. Therefore, the total time (126.469 sec., 291.68 sec., and 610.695 sec. for record sizes of 10, 25, and 50 respectively) required for WAVF score calculation using queries is reduced as shown in Figure 11. Here the total time is the summation of frequency counting time and WAVF score calculation time using the frequencies. As shown in Table 4, the WAVF score calculation time of the PPoD protocol with queries is 6.374 sec., 15.497 sec., and 30.993 sec. for record sizes of 10, 25, and 50 respectively.

Table 5: Comparative performances between the PPoD protocol (without any query) and the PPoD protocol (with queries) for computing the frequencies

Number of records	**Time of frequency counting without any query (sec.)**	**Time of frequency counting with queries (sec.)**
10	778.226	120.095
25	5743.2	276.183
50	19004.3	579.702

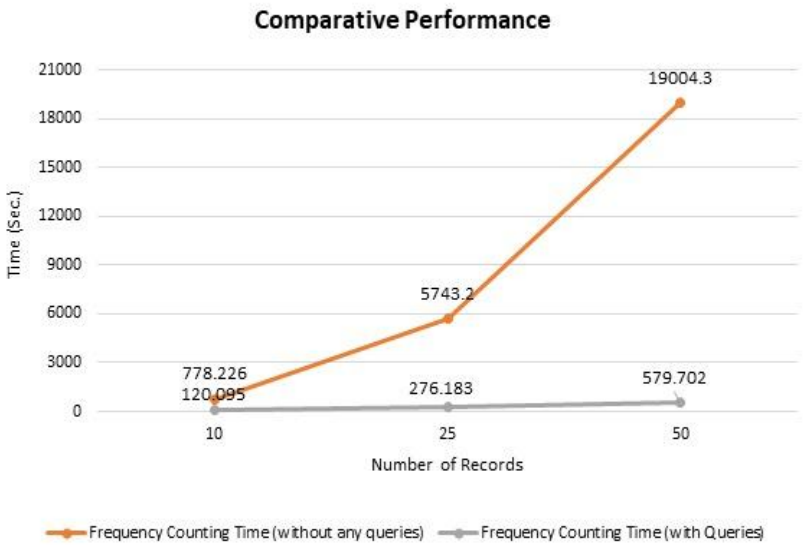

Figure 12: Comparative performances between the PPoD protocol (without any query) and the PPoD protocol (with queries) for computing frequencies

4.3.3 Comparative Performances

In this section, we show the comparative performances between the PPoD protocol (without any query) and the PPoD protocol (with queries) for computing the frequencies in Table 5 which is depicted in Figure 12. From the figure, it is evident that frequency counting time increases exponentially with the increase of record sizes for the PPoD protocol (without any query). On the contrary, the frequency counting time increases linearly with the increase of record sizes. Therefore, we conclude that frequency computation works faster in the PPoD protocol (with queries) than in the PPoD protocol (without any query).

5. Conclusion

In this study, we have presented post-quantum secure protocols for privacy-preserving outlier detection. We have shown two protocols — a fully blind PPoD protocol without any query and another PPoD protocol with queries for detecting outliers from a dataset using AVF and WAVF algorithms. Moreover, we have implemented our protocols using C++ with the Microsoft SEAL library and tested them with a real-world dataset. We also measured the performance of our protocols. From the performances of the protocols, it can be decided that the PPoD protocol with queries works faster than the PPoD protocol without queries. Though the PPoD protocol without queries

works slower, it removes most of the computation burden from the data owner. Since the computation time of this protocol grows exponentially, we can improve the protocol performances by using block-wise computation techniques in a distributive computation environment. We tested our protocols on a partial dataset taken from the real dataset in (Zwitter & Soklic, 1988) due to a lack of high-performance computers. Moreover, we can also extend our protocols to enable encrypted computation of MR-AVF protocol (Sequeira et al., 2014), a map-reduce version of AVF.

References

Zhang, J., & Zulkernine, M. (2006). Anomaly-based network intrusion detection with unsupervised outlier detection. 2006 IEEE International Conference on Communications. https://doi.org/10.1109/icc.2006.255127

García-Teodoro, P., Díaz-Verdejo, J. E., Maciá-Fernández, G., & Vazquez, E. A. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security, 28(1–2), 18–28. https://doi.org/10.1016/j.cose.2008.08.003

El´ıas, A., Ochoa, A., Alejandro, P., & Ponce, J. (2011). Outlier analysis for plastic card fraud detection a hybridized and multi-objective approach. In Hybrid Artificial Intelligent Systems. HAIS 2011, Lecture Notes in Computer Science (vol 6679. pp. 1–9). Springer. https://doi.org/10.1007/ 978-3-642-21222-2_1

Garces, H. O., & Sb´ arbaro, D. (2011). Outliers detection in environmental´ monitoring databases. Engineering Applications of Artificial Intelligence, 24(2), 341–349. https://doi.org/10.1016/j.engappai.2010.10.018

Sarı, A. (2015). A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications. Journal of Information Security, 06(02), 142–154. https://doi.org/10.4236/jis.2015.62015

Ge, Y., Xiong, H., Zhou, Z., Ozdemir, H., Yu, J., & Lee, K. C. (2010). Top-Eye: Top-k evolving trajectory outlier detection. In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, 1733–1736. https://doi.org/10.1145/1871437.1871716

Ieva, F., & Paganoni, A. M. (2014). Detecting and visualizing outliers in provider profiling via funnel plots and mixed effect models. Health Care Management Science, 18(2), 166–172. https://doi.org/10.1007/s10729-013-9264_9

Aggarwal, C. C., Zhao, Y., & Yu, P. S. (2011). Outlier detection in graph streams. In 2011 IEEE 27th International Conference on Data Engineering, 399–409. https://doi.org/10.1109/icde.2011.5767885

Gao, J., Liang, F., Wang, F., Wang, C., Sun, Y., & Han, J. (2010). On community outliers and their efficient detection in information networks. In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 813–822. https://doi.org/10.1145/1835804.1835907

Wang, Y., & Xu, W. (2018). Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud. Decision Support Systems, 105, 87–95. https://doi.org/10.1016/j.dss.2017.11.001

Zijlstra, W. P., Van Der Ark, L. A., & Sijtsma, K. (2011). Outliers in questionnaire data: Can they be detected and should they be removed?. Journal of Educational and Behavioral Statistics, 36(2), 186–212. https://doi.org/10.3102/1076998610366263

Lin, S., & Brown, D. E. (2002). Outlier detection and data association for data mining criminal incidents. WIT Transactions on Information and Communication Technologies, 28.

Taha, A., & Hadi, A. S. (2019). Anomaly detection methods for categorical data: A review. ACM Computing Surveys, 52(2), 1–35. https://doi.org/10.1145/3312739

Koufakou, A., Ortiz, E., Georgiopoulos, M., Anagnostopoulos, G. C., & Reynolds, K. M. (2007). A scalable and efficient outlier detection strategy for categorical data. In Proceedings of the IEEE International Conference on Tools With Artificial Intelligence (ICTAI'07), 210–217. https://doi.org/10.1109/ictai.2007.125

Sequeira, H., Carreira, P., Goldschmidt, T., & Vorst, P. (2014, December). Energy cloud: Real-time cloud-native energy management system to monitor and analyze energy consumption in multiple industrial sites. In 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (pp. 529-534). IEEE.

Rokhman, N. Subanar, & Winarko, E. (2016). Improving the performance of outlier detection methods for categorical data by using weighting function. J. Theor. Appl.d Info.n Technol. 83, 327–336.

He, Z., Xu, X., Huang, J. Z., & Deng, S. (2005). FP-outlier: Frequent pattern based outlier detection. Computer Science and Information Systems, 2(1), 103–118. https://doi.org/10.2298/csis0501103h

Otey, M. E., Ghoting, A., & Parthasarathy, S. (2006). Fast distributed outlier detection in Mixed-Attribute data sets. Data Mining and Knowledge Discovery, 12(2–3), 203–228. https://doi.org/10.1007/s10618-005-0014-6

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In Advances in Cryptology — EUROCRYPT '99. EUROCRYPT 1999. Lecture Notes in Computer Science, vol 1592. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-48910-X 16

Rivest, R. L., & Dertouzos, M. L. (1978). On data banks and privacy homomorphisms. Found. Secure Comput., 4(11), 169–180.

Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. In Proc. ACM Symp. Theory Comput. (STOC), 169–178. https://doi.org/10.1145/1536414.1536440

Saha, T. K., & Koshiba, T. (2017). An efficient Privacy-Preserving comparison protocol. In Lecture notes on data engineering and communications technologies (pp. 553–565). Springer. https://doi.org/10.1007/978-3-319-65521-5_48

Saha, T. K., Mayank, Deevashwer, & Koshiba, T. (2018). Private comparison protocol and its application to range queries. In Internet and Distributed Computing Systems. IDCS 2017, Lecture Notes in Computer Science (vol 10794. pp. 128–141). Springer. https://doi.org/10.1007/978-3-319-97795-9_12

Saha, T. K., & Koshiba, T. (2018). Privacy-Preserving Equality test towards big data. In Foundations and Practice of Security. FPS 2017, Lecture Notes in Computer Science (vol 10723. pp. 95–110). Springer. https://doi.org/10.1007/978-3-319-75650-9_7

Saha, T. K., & Koshiba, T. (2018b). Outsourcing private equality tests to the cloud. Journal of Information Security and Applications, 43, 83–98. https://doi.org/10.1016/j.jisa.2018.09.002

Saha, T. K., & Koshiba, T. (2016). Private equality test using ring-LWE somewhat homomorphic encryption. In 3rd Asia-Pacific World Congress on Computer Science and Engineering (APWConCSE), 1–9. https://doi.org/10.1109/apwc-oncse.2016.013

Vaidya, J., & Clifton, C. W. (2005). Privacy-preserving outlier detection. In Fourth IEEE International Conference on Data Mining (ICDM'04), 233–240. https://doi.org/10.1109/icdm.2004.10081

Shaneck, M., Kim, Y., & Kumar, V. (2006). Privacy preserving nearest neighbor search. In Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06), 541–545. https://doi.org/10.1109/icdmw.2006.133

Zhengyou, Z., Huang, L., Yang, W., & Ye, Y. (2009). Privacy Pre-serving Outlier Detection over Vertically Partitioned Data. In 2009 International Conference on E-Business and Information System Security, 1–5. https://doi.org/10.1109/ebiss.2009.5138025

Bohler, J., Bernau, D., & Kerschbaum, F. (2017). Privacy-preserving outlier¨ detection for data streams. In Lecture Notes in Computer Science (pp. 225–238). https://doi.org/10.1007/978-3-319-61176-1 12

Lu, G., Duan, C., Zhou, G., Ding, X., & Liu, Y. (2021). Privacy-preserving outlier detection with high efficiency over distributed datasets. In IEEE INFOCOM 2021-IEEE Conference on Computer Communications, 1–10. https://doi.org/10.1109/infocom42981.2021.9488710

Itokazu, K., Wang, L., & Ozawa, S. (2021). Outlier detection by privacy-preserving ensemble decision tree using homomorphic encryption. In 2021 International Joint Conference on Neural Networks (IJCNN), 1–7. https://doi.org/10.1109/ijcnn52387.2021.9534464

Cheon, J. H., Kim, M., & Kim, M. (2016). Optimized Search-and-Compute circuits and their application to query evaluation on encrypted data. IEEE Transactions on Information Forensics and Security, 11(1), 188–199. https://doi.org/10.1109/tifs.2015.2483486

Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Halevi, S., Hoffstein, J., Laine, K., Lauter, K., Lokam, S., Micciancio, D., Moody, D., Morrison, T.,

Sahai, A., & Vaikuntanathan, V. (2019). Homomorphic encryption standard. IACR Cryptology ePrint Archive, 2019, 939. https://eprint.iacr.org/2019/939.pdf

Brakerski, Z. (2012). Fully homomorphic encryption without modulus switching from classical GapSVP. In Advances in Cryptology – CRYPTO 2012, Lecture Notes in Computer Science (vol 7417. pp. 868–886). Springer. https://doi.org/10.1007/978-3-642-32009-5_50

Fan, J., & Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive, 2012, 144. https://eprint.iacr.org/2012/144.pdf

Yang, Y., Huang, X., Liu, X., Cheng, H., Weng, J., Luo, X., & Chang, V. (2019). A comprehensive survey on secure outsourced computation and its applications. IEEE Access, 7, 159426–159465. https://doi.org/10.1109/access.2019.2949782

ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 31(4), 469–472. https://doi.org/10.1109/tit.1985.1057074

Bellare, M., Boldyreva, A., & O'Neill, A. (2007). Deterministic and efficiently searchable encryption. In Advances in Cryptology - CRYPTO 2007, Lecture Notes in Computer Science (vol 4622. pp. 535–552). https://doi.org/10.1007/978-3-540-74143-5_30

Bellare, M., Fischlin, M., O'Neill, A., & Ristenpart, T. (2008). Deterministic encryption: definitional equivalences and constructions without random oracles. In Proc. Annu. Int. Cryptol. Conf., Lecture Notes in Computer Science (pp. 360–378). https://doi.org/10.1007/978-3-540-85174-5_20

Boldyreva, A., Chenette, N., Lee, Y., & O'Neill, A. (2009). Order-preserving symmetric encryption. In Proc. Annu. Int. Conf. Theory Appl. Crypto-graph. Techn., Lecture Notes in Computer Science (pp. 224–241). https://doi.org/10.1007/978-3-642-01001-9_13

Benaloh, J. (1994). Dense probabilistic encryption. In Proceedings of the Workshop on Selected Areas in Cryptography, pp. 120–128.

Boneh, D., Goh, E., & Nissim, K. (2005). Evaluating 2-DNF formulas on ciphertexts. In Proc. 2nd TCC, Lecture Notes in Computer Science (vol. 3378, pp. 325–341). https://doi.org/10.1007/978-3-540-30576-7_18

Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2012). (Leveled) fully homomorphic encryption without bootstrapping. In Proc. Innov. Theor. Comput. Sci. (ITCS), (pp. 309–325). https://doi.org/10.1145/2090236.2090262

Goldwasser, S., & Micali, S. (2019). Probabilistic encryption & how to play mental poker keeping secret all partial information. In Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali (pp. 173-201).

Hu, Y. (2013). Improving the efficiency of homomorphic encryption schemes (Doctoral dissertation, Worcester Polytechnic Institute).

Chillotti, I., Gama, N., Georgieva, M., & Izabachene, M. (2016). Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22 (pp. 3-33). https://doi.org/10.1007/978-3-662-53887-6_1

Cheon, J. H., Kim, M., & Kim, M. (2016). Optimized search-and-compute circuits and their application to query evaluation on encrypted data. In *IEEE Trans. Inf. Forensics Security* vol. 11(1), pp. 188–199. https://doi.org/10.1109/TIFS.2015.2483486.

Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science, pp. 409–437. https://doi.org/10.1007/978-3-319-70694-8_15

Kim, A., Polyakov, Y., & Zucca, V. (2021). Revisiting homomorphic encryption schemes for finite fields. In ASIACRYPT, Lecture Notes in Computer Science (pp. 608–639). https://doi.org/10.1007/978-3-030-92078-4_21

Saha, T. K., & Koshiba, T. (2017). Private Conjunctive Query over Encrypted Data. In Progress in Cryptology—AFRICACRYPT 2017, Lecture Notes in Computer Science (pp. 149–164). https://doi.org/10.1007/978-3-319-57339-7_9

Saha, T. K., & Koshiba, T. (2021). Efficient private conjunctive query protocol over encrypted data. Cryptography, 5(1), 2. https://doi.org/10.3390/cryptography5010002

Microsoft. (n.d.). GitHub - Microsoft/SEAL: Microsoft SEAL is an easy-to-use and powerful homomorphic encryption library. GitHub. https://github.com/microsoft/SEAL

Yao, A. C. (1982, November). Protocols for secure computations. In 23rd annual symposium on foundations of computer science (SCFS 1982) (pp. 160-164). IEEE.

Zwitter, M. & Soklic, M. (1988). Lymphography. UCI Machine Learning Repository (n.d.). http://archive.ics.uci.edu/dataset/63/lymphography