Hindawi
Security and Communication Networks
Volume 2022, Article ID 3068199, 9 pages
https://doi.org/10.1155/2022/3068199

Research Article

Privacy-Preserving Minority Oversampling Protocols with Fully Homomorphic Encryption

Maohua Sun, Ruidi Yang, and Mengying Liu

School of Management and Engineering, Capital University of Economics and Business, Beijing 100070, China

Correspondence should be addressed to Maohua Sun; sunmaohua@cueb.edu.cn

Received 18 November 2021; Accepted 28 January 2022; Published 10 March 2022

Academic Editor: Yuling Chen

In recent years, blockchain and machine-learning techniques have received increasing attention both in theoretical and practical aspects. However, the applications of these techniques have many challenges, one of which is the privacy-preserving issue. In this paper, we focus on, specifically, the privacy-preserving issue of imbalanced datasets, a commonly found problem in real-world applications. Built based on the fully homomorphic encryption technique, this paper presents two new secure protocols, Privacy-Preserving Synthetic Minority Oversampling Protocol (PPSMOS) and Borderline Privacy-Preserving Synthetic Minority Oversampling Protocol (Borderline-PPSMOS). Our analysis reveals that PPSMOS is generally more efficient in performance than Borderline-PPSMOS. However, Borderline-PPSMOS achieves a better TP rate and F-Value than PPSMOS.

1. Introduction

In the past few years, new information technology techniques, such as blockchain [1–4] and machine-learning [5–15], have been developing rapidly and used successfully in various real-life applications. However, they still face a critical challenge in the privacy-preserving issue. For example, the openness of a blockchain system poses a serious threat to the privacy and security of any user transactions. Thus, research for privacy-preserving techniques is becoming even more crucial.

Datasets in the wild come with a variety of problems. One of the most common problems is the imbalanced issue of the datasets. Imbalanced datasets issue arises in many real-world sectors, such as disease detection [16], bankruptcy prediction [17], fraud detection [18], etc. As the distribution of samples is incorrect in imbalanced datasets, it may cause the classification algorithms to produce inaccurate results and further issues. An imbalanced dataset usually consists of a number of classes, which falls into one of these two types: majority classes, which has a bigger number of examples, and minority classes, in which there are fewer examples. In this paper, we consider the situation where there are only two classes in a dataset, i.e., one majority class and one minority class.

The existing solutions proposed to solve the imbalanced dataset problem are categorized according to which level the technique is solving the problem from, e.g., data level, feature level, and machine-learning algorithm level. In this paper, we focus on fixing the problems at the data level. There are two known data level techniques, namely undersampling and oversampling methods. The undersampling method works by removing parts of the samples from the majority class to balance the ratio of majority and minority samples, whereas oversampling method balances the majority and minority samples by generating new minority samples. In 1972, Wilson [19] proposed an undersampling method, in which a majority sample should be deleted if all of its neighbors are minority samples. In 2020, Wang et al. [20] proposed a novel entropy and confidence-based undersampling boosting framework to solve imbalanced dataset issues, which could be applied to noniterating algorithms such as decision trees.

Random oversampling of minority classes is the simplest oversampling method. Through sampling with replacement, samples are continuously drawn from the minority class. This method, however, can easily lead to data overfitting. In 2002, Chawla et al. [21] proposed the Synthetic Minority Oversampling Technique (SMOTE) algorithm, which is one of the best-known oversampling methods to date. The algorithm works by generating artificial data using bootstrapping and the K-nearest neighbor algorithm. Further improvising SMOTE algorithm, Han et al. [22] proposed Borderline-SMOTE in 2005. The algorithm focuses on working on samples that are on the boundary of both majority and minority classes. The demonstration showed that Borderline-SMOTE achieved a better TP rate and F-Value than its predecessor. In 2008, Douzas et al. [23] presented a simple and effective oversampling method based on K-means clustering and SMOTE, which is able to eliminate noise generation and effectively overcome imbalances between and within classes. Furthermore, Li et al. [24] presented three sampling approaches for imbalanced learning in 2020. Unlike the previous solutions, their approaches considered a new class-imbalance metric, which contains the differences of information contents between classes, instead of the traditional imbalance ratio.

Although so many solutions have been proposed to solve the imbalanced data sets problem, the privacy-preserving issue has not been well resolved. To the best of our knowledge, Hong et al. [25] proposed a secure collaborative machine-learning solution in which they used secure multiparty computation to adjust the class weight for the imbalanced dataset. That is, the privacy-preserving issue of the imbalanced dataset was tackled at the machine-learning algorithm level. The privacy-preserving solution in the machine-learning level is specific. That is, when we change the machine-learning algorithm, a new privacy-preserving solution to the imbalanced data set problem should be proposed. By contrast, as the privacy-preserving solutions in the data level solve the problem in the preprocessing stage, the output of these solutions can be widely used as they are independent of the machine-learning algorithms. So, in this paper, we focus on the privacy-preserving issue of imbalanced data sets at the data level.

Despite the numerous solutions proposed to solve the imbalanced data sets problems, there is almost none of them attempted to resolve the privacy-preserving issue. To the best of our knowledge, Hong et al. invented a secure collaborative machine-learning solution, in which they used a secure multiparty computation to adjust the class weight of the imbalanced dataset. They tackled the privacy-preserving issue of the imbalanced dataset on the machine-learning algorithm level. This solution, however, is sensitive to the algorithm used for machine-learning, i.e., when the machine-learning algorithm is changed, a new privacy-preserving solution must be proposed for the imbalanced dataset problem. In contrast, as the privacy-preserving solutions at the data level work by solving the problem in the preprocessing stage, their output can be used widely, regardless of the machine-learning algorithm adopted in the system. Hence, in this paper, we focus on tackling the privacy-preserving issue of imbalanced datasets on the data level.

Currently, Secure Multiparty Computation (SMC) is one of the most widely used techniques to tackle the privacy-preserving issue. In SMC, multiple parties participate in the game with their individual secure inputs and nobody knows anything of each other's inputs. When the game ends, according to the game rules, some of the parties will obtain the output. The first SMC solution [26] to the millionaire problem was first presented by Yao. Since then, SMC has been developing rapidly. In 2017, Makri et al. [27] proposed SPDZ, a private image classification with SVM using the SMC framework. Mohassel et al. [28] presented a privacy-preserving machine-learning framework, SecureML, in which the privacy-preserving issue of the linear regression, logistic regression, and neural network training using the stochastic gradient descent method was considered.

In SMC, there are various underlying cryptographic tools, such as garbled circuit, homomorphic encryption scheme, oblivious transfer, and secret sharing scheme. In this paper, we focus on handling the imbalanced dataset problem with the privacy-preserving two-party computation using the homomorphic encryption scheme. Homomorphic encryption is one of the most active research areas in the field of cryptography. Homomorphic encryption was initially proposed by Rivest et al. [29] in 1978. In 1985, ElGamal et al. [30] proposed a widely used multiplicatively homomorphic encryption scheme, known as ElGamal scheme. In 2001, Damgard et al. [31] promoted an additively homomorphic encryption scheme, named Paillier scheme. In 2009, Gentry [32] proposed a fully homomorphic encryption scheme, a ground-breaking development to homomorphic encryption study. Currently, the two most widely used fully homomorphic encryption schemes are the BGV scheme [33] by Brakerski et al. and BFV scheme [34] by Fan et al. In 2021, Chen et al. [35] presented a dynamic multikey fully homomorphic encryption scheme based on LWE assumption in the public key setting.

1.1. Contributions. In this paper, we propose two novel privacy-preserving oversampling protocols, namely PPSMOS and Borderline-PPSMOS. Both PPSMOS and Borderline-PPSMOS are aimed to solve the problem of the imbalanced dataset while preserving the participants' input and output privacies. With the client and the service denoted as Bob and Alice, respectively, the work in this paper can be generally viewed as follows.

(1) PPSMOS: This algorithm works in a distributed architecture, where Bob inputs no examples at the beginning of the protocol. All the examples, both majority and minority, are provided by Alice. After the protocol, Bob gets the synthetic minority example while he learns nothing of Alice's examples. At the same time, Alice learns nothing of the output Bob receives. PPSMOS shows to be a good solution with a privacy-preserving manner for data balance problems encountered in the cold start phase of many real-life applications.

(2) Borderline-PPSMOS: In this algorithm, at the start of the protocol, Bob has some majority examples as his input. Meanwhile, Alice has a number of minority examples. After the protocol, Bob receives synthetic minority examples, while he learns nothing of Alice's

minority examples, and Alice learns nothing of Bob's input and output.

(3) PPSMOS and Borderline-PPSMOS performance analysis: Our analysis shows that PPSMOS generally works more efficiently than Borderline-PPSMOS, while Borderline-PPSMOS achieves a better TP rate and F-Value than PPSMOS. We also found that PPSMOS and Borderline-PPSMOS are both secure in the semihonest model.

1.2. Roadmap of This Paper. The rest of this paper is organized as follows. In Section 2, we introduce the preliminaries. We present the Privacy-Preserving Synthetic Minority Oversampling (PPSMOS) protocol in Section 3 and Borderline-PPSMOS in Section 4. We compare and analyse our protocols in Section 5. We, then, give our concluding remarks in Section 6.

2. Preliminaries

2.1. Homomorphic Encryption. The homomorphic encryption scheme allows us to operate the ciphertext directly. The result obtained after the application of this scheme is equivalent to the ciphertext obtained after performing an operation on a plaintext. Homomorphic encryption algorithms are divided into three categories: additive homomorphism, multiplicative homomorphism, and full homomorphism. For our protocols, we adopt the fully homomorphic encryption scheme. We describe the fully homomorphic encryption algorithm as follows.

We denote $(\mathrm{pk}, \mathrm{sk})$ as the system keys, where pk is the public key and *sk* is the secret key. Furthermore, $E(\alpha)$ is the encryption operation on the plaintext α and $D(\beta)$ is the decryption operation on the ciphertext β. The fully homomorphic encryption scheme follows the properties below.

$$\begin{aligned} E(\alpha) + E(\gamma) &= E(\alpha + \gamma), \\ E(\alpha) * E(\gamma) &= E(\alpha * \gamma). \end{aligned} \tag{1}$$

2.2. Semihonest Model. There are two widely used adversarial models in SMC, the semihonest model, and the malicious model. In this work, we design our protocols in the semihonest model.

In the semihonest model, there are two kinds of participants, the honest participants and the semihonest participants. The honest participants follow the protocol without doing any other activities. At the same time, the semihonest participants followed the protocol and collected the data they obtained during the process of the protocol. After the protocol, they may want to infer information from the data they collected. A protocol is secure in the semihonest model if the semihonest participants get no valuable information from the data they collected.

3. Privacy-Preserving Synthetic Minority Oversampling Protocol

In this section, we present our Privacy-Preserving Synthetic Minority Oversampling Protocol (PPSMOS) and analyze its security aspect.

Suppose that Alice has the total dataset $P = \{p_1, p_2, \ldots, p_h\}^T$ with $p_i = (p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(n)})$ with $1 \le i \le t$. To simplify, Alice puts all the minority samples in front of P. In other words, we denote the minority subclass by $P_{\min} = \{p_1, p_2, \ldots, p_m\}^T$ where m is the number of the minority samples. Both Alice and Bob wish to generate a minority sample p_{new} based on P. After the protocol, Bob gets the output p_{new} under the condition that Alice and Bob cannot know any information about p_{new} and P, respectively.

3.1. PPSMOS

3.1.1. Input. Alice inputs $P = \{p_1, p_2, \ldots, p_h\}^T$, where $p_i = (p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(n)})$ and $1 \le i \le h$, with the first m elements $P_{\min} = \{p_1, p_2, \ldots, p_m\}^T$ belonging to the minority class. Bob inputs nothing.

3.1.2. Output. Bob obtains a newly synthesized minority sample p_{new} while Alice gets nothing.

3.1.3. Preprocessing Stage

(1) Alice calls the key generation algorithm of the fully homomorphic encryption system to generate the system key $(\mathrm{pk}, \mathrm{sk})$.

(2) Alice computes the ciphertext $E(P)$ as follows.

 (i) $E(P) = \{E(p_1), E(p_2), \ldots, E(p_h)\}^T$ where $E(p_i) = (E(p_i^{(1)}), E(p_i^{(2)}), \ldots, E(p_i^{(n)}))$

(3) Alice constructs a matrix S that contains the indices of the k-nearest neighbors of every element in $P_{\min}$, i.e., every s_{ij} in S presents the index of the jth nearest neighbor of p_i in the minority class $P_{\min}$.

$$S = \begin{pmatrix} s_{11} & \cdots & s_{1k} \\ \vdots & \ddots & \vdots \\ s_{m1} & \cdots & s_{mk} \end{pmatrix}. \tag{2}$$

(4) Alice discloses $\mathrm{pk}, E(P)$ and S on the network.

(5) Bob gets $\mathrm{pk}, E(P)$ and S published by Alice.

3.1.4. Processing Stage

(1) Bob generates two random integers, α and β, where $1 \le \alpha \le m$ and $1 \le \beta \le k$.

(2) Bob generates two random numbers gap and noise where $0 < \text{gap} < 1$. Then, using the public key pk and the encryption algorithm $E(*)$, he computes the ciphertext $E(\text{gap})$ and $E(\text{noise})$.

(3) Using both ciphertexts obtained in (2), Bob does the following operation to produce X. Then he sends X to Alice.

$$\begin{aligned} X &= \left[E(\text{gap}) * \left(E\left(p_{s_{\alpha\beta}}\right) - E(p_\alpha)\right) + E(p_\alpha)\right] * E(\text{noise}) \\ &= \left[\left(E(\text{gap}) * \left(E\left(p_{s_{\alpha\beta}}^{(1)}\right) - E\left(p_\alpha^{(1)}\right)\right) + E\left(p_\alpha^{(1)}\right)\right) * E(\text{noise}) \right. \\ &\quad \left(E(\text{gap}) * \left(E\left(p_{s_{\alpha\beta}}^{(2)}\right) - E\left(p_\alpha^{(2)}\right)\right) + E\left(p_\alpha^{(2)}\right)\right) * E(\text{noise}) \\ &\quad \cdots \\ &\quad \left.\left(E(\text{gap}) * \left(E\left(p_{s_{\alpha\beta}}^{(n)}\right) - E\left(p_\alpha^{(n)}\right)\right) + E\left(p_\alpha^{(n)}\right)\right) * E(\text{noise})\right]^T. \end{aligned} \tag{3}$$

(4) Alice decrypts X using the secret key *sk* and obtains $D(X) = (\gamma^{(1)}, \gamma^{(2)}, \ldots, \gamma^{(n)})$, before sending it to Bob.

(5) Bob gets the final result p_{new} as follows.

$$p_{\text{new}} = \left(\frac{\gamma^{(1)}}{\text{noise}}, \frac{\gamma^{(2)}}{\text{noise}}, \ldots, \frac{\gamma^{(n)}}{\text{noise}}\right). \tag{4}$$

3.2. Security Analysis

Theorem 1. *Under the assumption that the underlying fully homomorphic encryption scheme is secure, PPSMOS securely generates the minority samples in the semihonest model.*

Proof. First, we analyse the situation where Alice is corrupted. In PPSMOS, Alice receives X from Bob. Using the secret key, Alice is able to recover the plaintext:

$$D(X) = \left(\gamma^{(1)}, \gamma^{(2)}, \ldots, \gamma^{(n)}\right) = \left[\text{gap} * \left(p_{s_{\alpha\beta}} - p_\alpha\right) + p_\alpha\right] * \text{noise}. \tag{5}$$

As α, β, and gap are random numbers, Alice does not have the ability to infer the matchup between $D(X)$ and its samples. Furthermore, since $D(X)$ are confused by the random number noise, Alice has no way of knowing Bob's newly generated point p_{new}. Hence, even if Alice is corrupted, Bob's output is isolated from Alice and, thus, secure.

Next, we analyze the case that Bob is corrupted. In the preprocessing stage, Bob gets the ciphertext $E(P)$ and a matrix S, which are both disclosed by Alice. As the underlying homomorphic encryption scheme is secure in the semihonest model, Bob will not be able to infer any information regarding Alice's private input from $E(P)$. As S presents the index of the j^{th} nearest neighbor of p_i in the minority class $P_{\min}$, Bob is unable to get any information of the specific point of P through S. Therefore, even if Bob is corrupted, Alice's private information is still secure and undisclosed from Bob.

Thus, we can deduct that that Theorem 1 holds. □

4. Borderline Privacy-Preserving Synthetic Minority Oversampling Protocol

In this section, we present our Borderline Privacy-Preserving Synthetic Minority Oversampling Protocol (Borderline-PPSMOS) and analyze its security aspect.

Suppose that Alice has a minority class $P = \{p_1, p_2, \ldots, p_m\}^T$, where $p_i = (p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(n)})$. Bob has a majority class $Q = \{q_1, q_2, \ldots, q_t\}^T$, where $q_i = (q_i^{(1)}, q_i^{(2)}, \ldots, q_i^{(n)})$. Both Alice and Bob wish to generate a minority sample p_{new} based on P and Q. After the protocol, Bob gets the output p_{new}. Meanwhile, Alice cannot know any information about p_{new} and Q, and Bob cannot know any information about P.

4.1. Borderline-PPSMOS

4.1.1. Input. Alice inputs $P = \{p_1, p_2, \ldots, p_m\}^T$ where $p_i = (p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(n)})$. Bob inputs $Q = \{q_1, q_2, \ldots, q_t\}^T$ where $q_i = (q_i^{(1)}, q_i^{(2)}, \ldots, q_i^{(n)})$.

4.1.2. Output. Alice gets nothing. Bob obtains a newly synthesized minority sample p_{new}.

4.1.3. Preprocessing Stage

(1) Alice generates the key $s(\text{pk}, \text{sk})$ of the fully homomorphic encryption system.

(2) Alice computes the ciphertext $E(P)$ as follows.

(i) $E(P) = \{E(p_1), E(p_2), \ldots, E(p_m)\}^T$ where $E(p_i) = (E(p_i^{(1)}), E(p_i^{(2)}), \ldots, E(p_i^{(n)}))$

(3) Alice constructs a matrix D, where d_{ij} in D represents the square power of the Euclidean distance between the point p_i and its j^{th}-nearest neighbors.

$$D = \begin{pmatrix} d_{11} & \cdots & d_{1k} \\ \vdots & \ddots & \vdots \\ d_{m1} & \cdots & d_{mk} \end{pmatrix}. \tag{6}$$

(4) Alice encrypts every element in D and obtains $E(D)$.

(5) Alice discloses pk, $E(P)$ and $E(D)$ on the network.

(6) Bob gets pk, $E(P)$ and $E(D)$ which were published by Alice.

4.1.4. Processing Stage

(1) Bob computes $E(Q)$ using pk and the encryption algorithm $E(*)$.

(i) $E(Q) = \{E(q_1), E\quad(q_2), \ldots, E(q_t)\}^T$, where $E(q_i) = (E(q_i^{(1)}), E\ (q_i^{(2)}), \ldots, E(q^{(n)}))$

(2) For every element p_i in P, Bob calculates the ciphertext of the square power of the Euclidean distance between the p_i and the elements in Q.

$$E(V)=\begin{pmatrix} E(v_{11}) & \cdots & E(v_{1t}) \\ \vdots & \ddots & \vdots \\ E(v_{m1}) & \cdots & E(v_{mt}) \end{pmatrix}. \tag{7}$$

(i) where $E(v_{ij}) = (E(p_i^{(1)}) - E(q_j^{(1)}))^2 + (E(p_i^{(2)}) - E(q_j^{(2)}))^2 + \cdots + (E(p_i^{(n)}) - E(q_j^{(n)}))^2$

(3) Bob generates m random numbers $\sigma_1, \sigma_2, \ldots, \sigma_m$. Then, he obtains the ciphertext $E(\sigma_1)$, $E(\sigma_2), \ldots, E(\sigma_m)$, using the public key pk and the encryption algorithm $E(*)$.

(4) Bob connects $E(V)$ to the encryption matrix $E(D)$ to form a new matrix $E(G)$.

$$E(G)=\begin{pmatrix} E(d_{11})+E(\sigma_1) & \cdots & E(d_{1k})+E(\sigma_1) & E(v_{11})+E(\sigma_1) & \cdots & E(v_{1t})+E(\sigma_1) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ E(d_{m1})+E(\sigma_m) & \cdots & E(d_{mk})+E(\sigma_m) & E(v_{m1})+E(\sigma_m) & \cdots & E(v_{mt})+E(\sigma_m) \end{pmatrix}. \tag{8}$$

(5) Bob performs row and column confusion on $E(G)$ to obtain the confused matrix $E(G')$. Then he sends $E(G')$ to Alice.

(6) Alice receives $E(G')$ and decrypts it with the private key sk to obtain the matrix G'.

$$G'=\begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1(n+t)} \\ \vdots & \vdots & \ddots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{m(n+t)} \end{pmatrix}. \tag{9}$$

(7) For every row in G', Alice computes the k-smallest value and denotes the position of these elements in the matrix R. Then, Alice sends R to Bob.

$$R=\begin{pmatrix} r_{11} & \cdots & r_{1k} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{mk} \end{pmatrix}. \tag{10}$$

(8) Bob performs the inverse obfuscation on matrix R to get matrix B.

$$B=\begin{pmatrix} b_{11} & \cdots & b_{1k} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mk} \end{pmatrix}. \tag{11}$$

(9) For the i^{th} row in B, where $1 \le i \le k$, Bob counts the number cnt_i of the elements smaller than $k+1$. Similar to that in Borderline-SMOTE, we call the point $p_i \in$ Danger, if $(k/2) \le \mathrm{cnt}_i < k$.

(10) Bob randomly selects a point p_i from the class Danger and randomly selects an element b_{ix}, which is greater than k from the i^{th} row of B.

(11) Bob generates two random numbers, gap and noise, where $0 < \text{gap} < 1$. Next, he generates the ciphertext $E(\text{gap})$ and $E(\text{noise})$ by using the public key pk and the encryption algorithm $E(*)$.

(12) Bob performs an operation using the ciphertext $E(\text{gap})$ and $E(\text{noise})$ to obtain X as follows. Then he sends X to Alice.

$$\begin{aligned} X &= \left[E(\text{gap}) * \left(E\left(q_{b_{ix}}\right) - E(p_i)\right) + E(p_i)\right] * E(\text{noise}) \\ &= \left[\left(E(\text{gap}) * \left(E\left(q_{b_{ix}}^{(1)}\right) - E\left(p_i^{(1)}\right)\right) + E\left(p_i^{(1)}\right)\right) * E(\text{noise})\right. \\ &\quad \left(E(\text{gap}) * \left(E\left(q_{b_{ix}}^{(2)}\right) - E\left(p_i^{2}\right)\right) + E\left(p_i^{2}\right)\right) * E(\text{noise}), \ldots, \\ &\quad \left.\left(E(\text{gap}) * \left[E\left(q_{b_{ix}}^{n}\right) - E\left(p_i^{n}\right)\right] + E\left(p_i^{n}\right)\right) * E(\text{noise})\right]^T. \end{aligned} \tag{12}$$

(13) Alice decrypts X using the secret key sk and obtains $D(X) = (\gamma^{(1)}, \gamma^{(2)}, \ldots, \gamma^{(n)})$. She then proceeds to send $D(X)$ to Bob.

(14) Bob gets the final result p_{new} as follows.

$$p_{\text{new}} = \left(\frac{\gamma^{(1)}}{\text{noise}}, \frac{\gamma^{(2)}}{\text{noise}}, \ldots, \frac{\gamma^{(n)}}{\text{noise}}\right). \tag{13}$$

4.2. Security Analysis

Theorem 2. *Under the assumption that the underlying fully homomorphic encryption scheme is secure, Borderline-PPSMOS securely generates the minority sample in the semihonest model.*

Proof. First, we analyse the situation where Alice is corrupted. In Borderline-PPSMOS, Alice receives $E(G')$ from Bob. Alice is able to recover the plaintext using the private key. However, since Bob obtained $E(G')$ by applying row and column confusion operation on $E(G)$, Alice will not be able to infer the true rank order of $E(G)$. Furthermore, as $E(G')$ is confused by using $E(\sigma_1), E(\sigma_2), \ldots, E(\sigma_m)$, where $\sigma_1, \sigma_2, \ldots, \sigma_m$ are random numbers, Alice will not be able to know the information of set Q owned by Bob.

Also, when Alice receives X from Bob, she can recover the plaintext:

$$\begin{aligned} D(X) &= \left(\gamma^{(1)}, \gamma^{(2)}, \ldots, \gamma^{(n)}\right) \\ &= \left[\text{gap} * \left(q_{b_{ix}} - p_i\right) + p_i\right] * \text{noise}. \end{aligned} \tag{14}$$

TABLE 1: Performance analysis of PPSMOS and Borderline-PPSMOS.

Protocol	Preprocessing stage		Processing stage	
	Computational complexity	Communication complexity	Computational complexity	Communication complexity
PPSMOS	hn	hn	$5n+2$	n
Borderline-PPSMOS	$mn+mk$	$tn+mk$	$(m+mt+2t+3)n+4mk+2mt+m+2$	$m(k+t)+n$

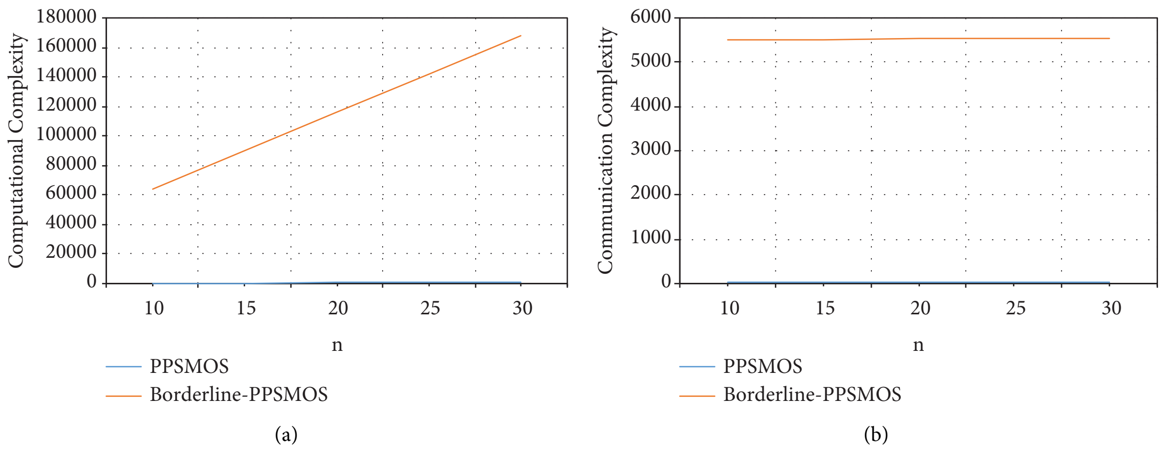

FIGURE 1: Computational Complexity (left) and Communication Complexity (right) of PPSMOS and Borderline-PPSMOS by varying n when $m = 100, t = 50, k = 5$.

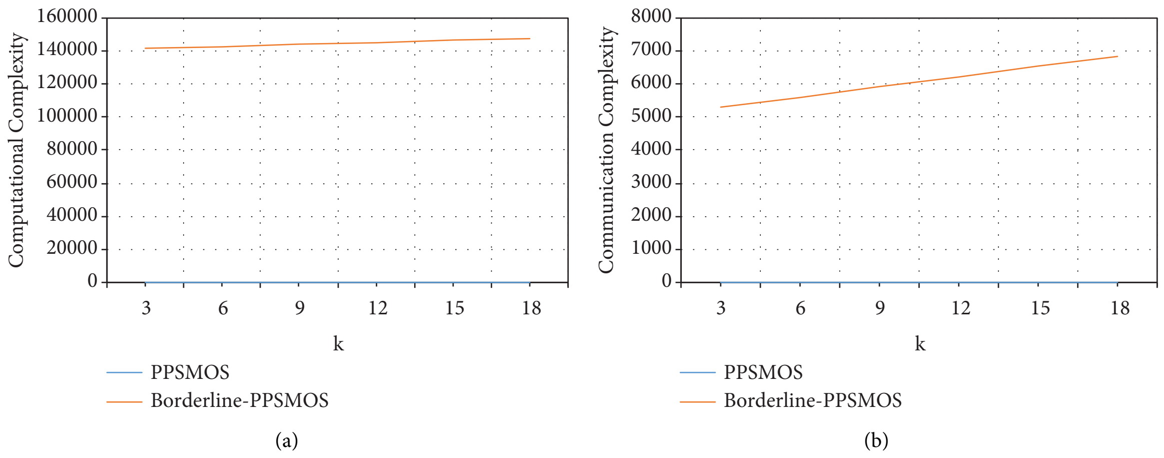

FIGURE 2: Computational Complexity (left) and Communication Complexity (right) of PPSMOS and Borderline-PPSMOS by varying k when $m = 100, t = 50, n = 25$.

However, as i, x and gap are random numbers, Alice does not have the ability to infer the matchup between $D(X)$ and its samples. In addition, as $D(X)$ is confused by the random number noise, Alice has no way of knowing Bob's newly generated point p_{new}. Hence, even if Alice is corrupted, Bob's input and output are totally isolated from Alice and still secure.

Next, we analyze the case where Bob is corrupted. In the preprocessing stage, Bob gets the ciphertext $E(P)$, $E(D)$ and public key pk. As the underlying homomorphic encryption scheme is secure in the semihonest model, Bob is unable to infer any information regarding Alice' private input from $E(P)$ and $E(D)$. In the processing stage, Alice computes the k-smallest value and denotes the position of these elements in matrix R. During this step, Alice only sends the location index to Bob, which does not reveal any information of Alice's input. Next, Bob gets $D(X) = (\gamma^{(1)}, \gamma^{(2)}, \ldots, \gamma^{(n)})$ from Alice. Similarly, as the homomorphic encryption scheme is secure in the semihonest model, Bob cannot infer p_i from gap $* (q_{b_{ix}} - p_i) + p_i$. Thus, even if Bob is corrupted, Alice's private information is still secure and undisclosed from Bob.

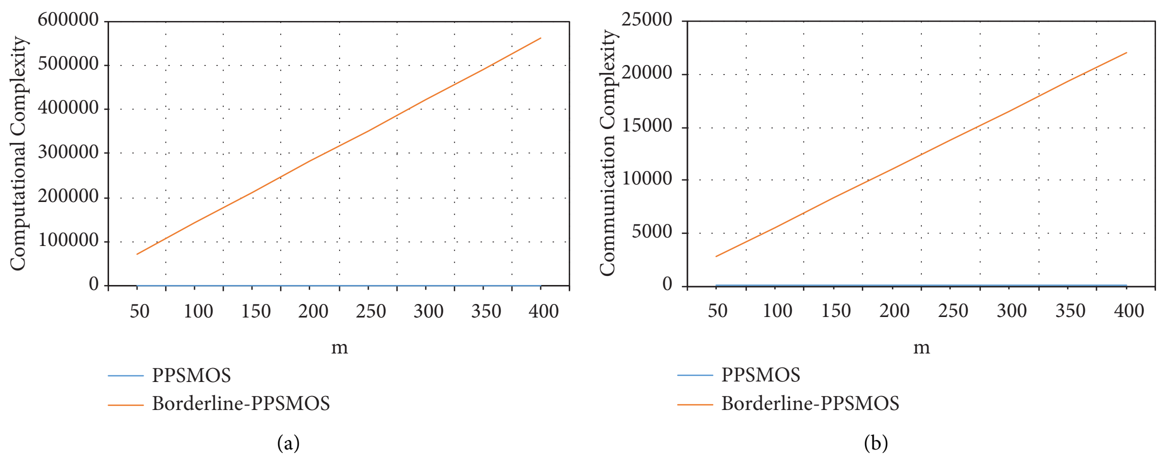

Figure 3: Computational Complexity (left) and Communication Complexity (right) of PPSMOS and Borderline-PPSMOS by varying m when $n = 25, t = 50, k = 5$.

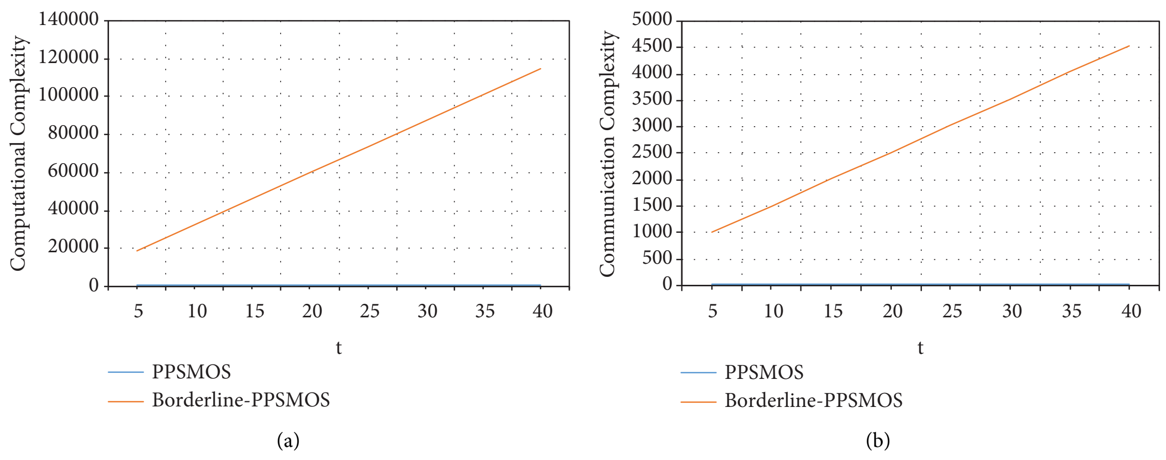

Figure 4: Computational Complexity (left) and Communication Complexity (right) of PPSMOS and Borderline-PPSMOS for the cost by varying t when $m = 100, n = 25, k = 5$.

Therefore, we can deduce that Borderline-PPSMOS is secure in the semihonest model, under the fully homomorphic encryption scheme, i.e., Theorem 2 holds. □

5. Performance Analysis

In this section, we present the performance analysis of both PPSMOS and Borderline-PPSMOS. On the efficiency analysis, we look at computational complexity and communication complexity. Given that h is the size of Alice's input in PPSMOS, n is the feature size, t is the size of the majority class, m is the size of the minority class, and k is a parameter, we analyze the protocols' performances as follows.

First, we analyze the performance of PPSMOS. During the preprocessing stage, Alice performs the encryption operation for hn times while Bob gets hn ciphertexts. Furthermore, in the processing stage, Bob performs encryption operation 2 times and $2n$ times for each homomorphic additive operation and homomorphic multiplicative operation. Alice, then, performs decryption operations for n times. Bob sends n ciphertexts to Alice.

Secondly, we analyze the efficiency of Borderline-PPSMOS. In the preprocessing stage, Alice performs the encryption operations for $mn + mk$ times, while Bob gets $tn + mk$ ciphertexts. Next, in the processing stage, Bob performs encryption operation for $tn + m + 2$ times, homomorphic additive operation for $mtn + (t + k)m + 2n$ times, and homomorphic multiplicative operation for $mtn + 2n$ times. Alice performs decryption operations for $m(k + t) + n$ times. Finally, Bob transferred $m(k + t) + n$ ciphertexts to Alice.

We summarise the computational complexity and communication complexity of both protocols below in Table 1.

We visualize the operational efficiency of both PPSMOS and Borderline-PPSMOS during the processing stage by instantiating the parameters, as shown below in Figures 1–4.

From these figures, we can conclude the following: (1) the computational complexity and communication complexity of

PPSMOS are lower than those of Borderline-PPSMOS; (2) the computational complexity and communication complexity of PPSMOS depend only on the feature size *n*; (3) the computational complexity and communication complexity of Borderline-PPSMOS depend on almost all of the parameters, i.e., *n*, *t*, *m* and *k*; (4) Borderline Synthetic Minority Oversampling Protocol achieves better TP rate and F-Value than Synthetic Minority Oversampling Protocol [22]. Furthermore, as our privacy-preserving schemes do not affect the TP rate and F-Value of the underlying Minority Oversampling protocol, we can further deduct that Borderline-PPSMOS achieves a better TP rate and F-Value than PPSMOS.

6. Conclusion

In this paper, we propose two novel privacy-preserving oversampling protocols, PPSMOS and Borderline-PPSMOS, that are aimed to address the imbalanced dataset issue while preserving the privacy of the participants' input and output.

PPSMOS works in a manner where the client inputs no majority examples, as opposed to Borderline-PPSMOS, where the client has some majority examples. Both PPSMOS and Borderline-PPSMOS are secure in the semihonest model. This means that both methods are suitable for the preprocessing stage of machine-learning and applicable to any cases where synthesizing minority examples in a privacy-preserving manner is needed. Our results show that PPSMOS is more efficient than Borderline-PPSMOS in general, while Borderline-PPSMOS achieves better TP rate and F-Value than PPSMOS.

While doing our work in the semihonest model and through our analysis, we found that our protocols are unable to resist malicious attacks, and their efficiency needs improvements. As future work, we will continue improving our research on these two aspects, as well as focusing on designing better privacy-preserving protocols that are to be used in the preprocessing stage of machine-learning.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by Project of High-level Teachers in Beijing Municipal Universities in the Period of 13th Five-year Plan (CIT&TCD201904097) and The Fundamental Research Funds for Beijing Local Universities From Capital University of Economics and Business.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[2] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proceedings of the International conference on financial cryptography and data security*, pp. 436–454, Christ Church, Barbados, May 2014.

[3] C. Hou, M. Zhou, Y. Ji et al., "SquirRL: Automating attack analysis on blockchain incentive mechanisms with deep reinforcement learning," arXiv preprint arXiv:1912.01798, 2019.

[4] T. Li, Y. Chen, Y. Wang et al., "Rational protocols and Attacks in blockchain system," *Security and Communication Networks*, vol. 2020, no. 44, 11 pages, Article ID 8839047, 2020.

[5] K. Suzuki, "Pixel-based machine learning in medical imaging," *International Journal of Biomedical Imaging*, vol. 2012, Article ID 792079, 1 page, 2012.

[6] M. L. Giger and K. Suzuki, "Computer-aided diagnosis (CAD)," *Biomedical Information Technology*, vol. 31, pp. 359–374, 2007.

[7] Y. Chen, J. Sun, Y. Yang, T. Li, X. Niu, and H. Zhou, "PSSPR: A source location privacy protection scheme based on sector phantom routing in WSNs," *International Journal of Intelligent Systems*, vol. 37, no. 2, pp. 1204–1221, 2022.

[8] K. Doi, "Current status and future potential of computer-aided diagnosis in medical imaging," *British Journal of Radiology*, vol. 78, no. 1, pp. S3–S19, 2005.

[9] S. M. Usman, M. Usman, and S. Fong, "Epileptic seizures prediction using machine learning methods," *Computational and Mathematical Methods in Medicine*, vol. 2017, Article ID 9074759, 2017.

[10] S. R. Girard, V. Legault, G. Bois, and J. F. Boland, "Avionics graphics hardware performance prediction with machine learning," *Scientific Programming*, vol. 2019, Article ID 9195845, 15 pages, 2019.

[11] T. Ali, H. Khazaei, M. H. Y. Moghaddam, and Y. Hassan, "Machine learning in transportation," *Journal of Advanced Transportation*, vol. 2019, Article ID 4359785, 3 pages, 2019.

[12] Y. Chen, K. Liu, Y. Xie, and M. Hu, "Financial trading strategy system based on machine learning," *Mathematical Problems in Engineering*, vol. 2020, Article ID 3589198, 13 pages, 2020.

[13] J. Mu, F. Wu, and A. Zhang, "Housing value forecasting based on machine learning methods," *Abstract and Applied Analysis*, vol. 2014, Article ID 648047, 7 pages, 2014.

[14] T. Li, Z. Wang, Y. Chen, C. Li, Y. Jia, and Y. Yang, "Is semi-selfish mining available without being detected?" *International Journal of Intelligent Systems*, pp. 1–22, 2021.

[15] T. Li, Z. Wang, G. Yang, Y. Cui, Y. Chen, and X. Yu, "Semi-selfish mining based on hidden Markov decision process," *International Journal of Intelligent Systems*, vol. 36, no. 7, pp. 3596–3612, 2021.

[16] J. Zhang, L. Chen, and F. Abid, "Prediction of Breast cancer from imbalance respect using cluster-based undersampling method," *Journal of Healthcare Engineering*, vol. 2019, Article ID 7294582, 10 pages, 2019.

[17] H. Wang and X. Liu, "Undersampling bankruptcy prediction: taiwan bankruptcy data," *PLoS ONE*, vol. 16, no. 7, pp. 1–17, 2021.

[18] H.-L. Dai, "Class imbalance learning via a fuzzy total margin based support vector machine," *Applied Soft Computing*, vol. 31, pp. 172–184, 2015.

[19] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-2, no. 3, pp. 408–421, 1972.

[20] Z. Wang, C. Cao, and Y. Zhu, "Entropy and confidence-based undersampling boosting random forests for imbalanced problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 12, pp. 5178–5191, 2020.

[21] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[22] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *Proceedings of the International conference on intelligent computing*, Springer, Berlin, Heidelberg, pp. 878–887, 2005.

[23] G. Douzas, F. Bacao, and F. Last, "Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE," *Information Sciences*, vol. 465, pp. 1–20, 2018.

[24] L. Li, H. He, and J. Li, "Entropy-based sampling Approaches for multi-class imbalanced problems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 11, pp. 2159–2170, 2020.

[25] C. Hong, Z. Huang, W. J. Lu et al., "Privacy-preserving collaborative machine learning on genomic data using TensorFlow," in *Proceedings of the ACM Turing Celebration Conference-China*, pp. 39–44, Hefei, China, May 2020.

[26] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pp. 160–164, Chicago, IL, USA, November 1982.

[27] E. Makri, D. Rotaru, N. P. Smart, and F. Vercauteren, "EPIC: efficient private image classification (or: Learning from the masters)," *Cryptographers' Track at the RSA Conference, Lecture Notes in Computer Science*, vol. 11405, pp. 473–492, 2019.

[28] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proceedings of the 2017 IEEE symposium on security and privacy*, pp. 19–38, San Jose, CA, USA, May 2017.

[29] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data Banks and privacy homomorphisms," *Foundations of Secure Compuation*, vol. 4, no. 11, pp. 120–126, 1978.

[30] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," *International Cryptology Conference*, vol. 31, no. 4, pp. 10–18, 1985.

[31] I. Damgård and M. Jurik, "A generalisation, a simplification and some Applications of paillier's probabilistic public-key system," *Public Key Cryptography*, Springer, in *Proceedings of the International workshop on public key cryptography*, pp. 119–136, Berlin, Heidelberg, 2001.

[32] C. Gentry and D. Boneh, "A fully homomorphic encryption scheme," *Stanford university*, vol. 20, no. 9, 2009.

[33] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *ACM Transactions on Computation Theory*, vol. 6, no. 3, pp. 1–36, 2014.

[34] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, p. 144, 2012.

[35] Y. Chen, S. Dong, T. Li, Y. Wang, and H. Zhou, "Dynamic multikey FHE in asymmetric key setting from LWE," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 5239–5249, 2021.