MAS-Encryption and Its Applications in Privacy-Preserving Classifiers

Chong-zhi Gao, Jin Li, Shibing Xia, Kim-Kwang Raymond Choo, Wenjing Lou, Changyu Dong

Abstract—Homomorphic encryption (HE) schemes, such as fully homomorphic encryption (FHE), support a number of useful computations on ciphertext in a broad range of applications, such as e-voting, private information retrieval, cloud security, and privacy protection. While FHE schemes do not require any interaction during computation, the key limitations are large ciphertext expansion and inefficiency. Thus, to overcome these limitations, we develop a novel cryptographic tool, MAS-Encryption (MASE), to support real-value input and secure computation on the multiply-add structure. The multiply-add structures exist in many important protocols, such as classifiers and outsourced protocols, and we will explain how MASE can be used to protect the privacy of these protocols, using two case study examples. Specifically, the first case study example is the privacy-preserving Naive Bayes classifier that can achieve minimal Bayes risk, and the other example is the privacy-preserving support vector machine. We prove that the constructed classifiers are secure and evaluate their performance using real-world datasets. Experiments show that our proposed MASE scheme and MASE based classifiers are efficient, in the sense that we achieve an optimal tradeoff between computation efficiency and communication interactions. Thus, we avoid the inefficiency of FHE based paradigm.

Index Terms—MAS-Encryption, Homomorphism, Privacy-preserving, Classifiers.

1 Introduction

Classifiers, such as Naive Bayes (NB) and decision trees, are key in machine learning. Given a dataset with multiple features, a classifier predicts the value of the target attribute for a new sample. Classifiers have been widely used in applications ranging from the detection of diseases to economic modeling, network optimization, and so on. A supervised classifier generally extracts classification rules from a dataset that contains a series of training samples. The rules are then represented as a prediction or discrimination function along with its parameters. Using the classification function, the algorithm will predict the value of the target attribute for any new sample.

The importance of ensuring privacy during classification has become more pronounced in recent years, due to the nature and amount of data, the type of computations performed on such data, and the increasing sophistication of classification algorithms to maximize the mining of data. For example, consider a situation where a server (say, a healthcare organization that owns a large dataset comprising patients' medical and diagnostic information, and a proprietary prediction function) provides prediction services to a user (say, a potential patient who wishes to analyze his health data, including his family history, to determine if he is at risk pertaining to a specific disease such as some autosomal recessive disorder). Privacy needs to be considered from the perspectives of both service providers and consumers/users. In this context, the service provider does not want to reveal/leak its proprietary prediction function (i.e., intellectual property), and the user clearly has vested interest not to reveal his/her private health related data as the leakage of such data and the prediction findings can have significant implications (e.g., being denied insurance coverage or at risk of facing discrimination in future job interviews). In other words, the service provider and the user should utilize a protocol, which provides accurate privacy-preserving prediction services to the user without leaking information about the underpinning service and infrastructure. Existing algorithms designed to protect the privacy of classifiers can be broadly categorized into those that protect the training phase's privacy, and those that protect the privacy during the prediction phase. In this paper, we focus on the second category.

Depending on the complexity of the classifier, different cryptographic tools are used to protect the privacy of the classifier. For example, for simple classifiers with only multiplication or addition structures, partially homomorphic encryption (PHE) tools can be used to achieve privacy protection. PHE mainly includes additively homomorphic encryption (AHE) and multiplicatively homomorphic encryption (MHE). The PHE approach not only protects the privacy, but also allows users to perform some simple operations on the ciphertext. For more complicated classifiers, existing approaches usually use FHE to facilitate secure prediction on the encrypted input. However, FHE is impractical for real-world deployment at the time of this research, due to the significant ciphertext expansion and computation complexity. For example, the ciphertext space in FHE is $\mathbb{Z}_q[x] \times \mathbb{Z}_q[x]$, where q is a large integer; hence, 100 bits' plaintext corresponds to a ciphertext above 10K bits.

Chong-zhi Gao and Shibing Xia are with the School of Computer Science, Guangzhou University, China. Chong-zhi Gao is also with the Guangxi Key Laboratory of Cryptography and Information Security, Guangxi, China. Emails: czgao@gzhu.edu.cn, shibing_xia@e.gzhu.edu.cn
Jin Li is with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, China. Email: jinli71@gmail.com.
K.-K. R. Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249-0631, USA. Email: raymond.choo@fulbrightmail.org
Wenjing Lou is with the Department of Computer Science, Virginia Polytechnic Institute and State University, USA, Email: wjlou@vt.edu
Changyu Dong is with the School of Computing Science, Newcastle University, Newcastle Upon Tyne, UK. Email:changyu.dong@newcastle.ac.uk

This paper focuses on the privacy of the family of multiply-add (M-A) type classifiers. The multiply-add type classifiers refer to the classifiers which can be expressed as the form of $\sum_i(\prod_j \alpha_{i,j})$. They can achieve the function that the classifier of simple structure (i.e., multiplicative-only or additive-only) cannot reach. Many important classifiers can be transformed into this type of classifier family, e.g., the NB classifiers and a part of support vector machine (SVM) classifiers. However, the M-A type structures also increase the difficulty of privacy protection, compared to the simple structure classifiers. For example, one cannot trivially use PHE to protect the privacy of M-A type classifiers, since PHE can only handle multiplicative or additive structure.

In this paper, we seek to solve the challenge in protecting privacy of M-A type classifiers. To avoid using FHE, we propose an efficient tool, coined MAS-Encryption, which allows us to protect the M-A structure's privacy with only additional two-round overhead compared to the FHE approach. In constructing the cryptographic tool MAS-Encryption, a novel blinding-then-normalization technique is used, which enable us to process real-value cleartexts, and handle multiple multiplications or additions in a batch mode.

1.1 Contributions

In summary, the contributions of this paper are as follows:

1). We propose a novel cryptographic tool MAS-Encryption, which is designed to support both real-value input and secure computations on a multiply-add structure. Compared to FHE and PHE approaches, it achieves an optimal tradeoff between computation efficiency and communication interactions.

2). We present two examples to illustrate the applications of MAS-Encryption in efficiently constructing private classifiers. In the first example, we construct a privacy-preserving Naive Bayes classifier with minimal Bayes risk (MBR-PPNBC). The MBR-PPNBC has a multiply-add structure, and prior approaches do not minimize the Bayes risk in a private NB classifier. The second application is to construct a privacy-preserving support vector machine classifier (PPSVMC). In prior approaches, the number of protocol interactions will be high if the multiplication operations in the support vector machine (SVM) classifier increase. We avoid such a limitation in the proposed protocol.

In the next two sections, we will present the related literature and background materials. In Section 4, we present the proposed cryptographic tool (i.e. MAS-Encryption), prior to presenting the proposed MBR-PPNBC and PPSVMC protocols in Sections 5 and 6 respectively. Findings from our evaluation are reported in Section 7. The concluding remarks are then presented in the last section.

2 Related Work

Protecting privacy of classifiers, or in a broader sense also the machine learning algorithms, is an ongoing research topic that has practical implications, particularly in our increasingly privacy-aware society [22], [39]. In supervised classifiers, the privacy issue lies in two phases, namely: the training phase and the prediction phase. Providing privacy in the training phase has been considered in secure training algorithms such as Naive Bayes [24], [48], [56], [60], decision tree [11], [58], and deep learning algorithms [53], [43]. Their applications include gene classification [36], electronic health system [30], preserving the privacy of outsourced data [23], [64], and so on. To preserve the privacy in the prediction phase, Erkin et al. [17] and Sadeghi et al. [49] designed protocols to enable a client to privately search for a specific face image in facial image databases. Barni et al. [5], [6] also designed protocols to securely evaluate linear branching programs, which can be used to construct privacy-preserving electrocardiogram classifications. However, preserving the privacy in the prediction phase appears to be a topic that is relatively understudied, in comparison to preserving the privacy in the training phase.

For simple classifiers with only multiplication or addition structures, simple structured cryptographic tools (e.g., PHE) can be used to achieve privacy protection. As discussed earlier, FHE are often used to protect the privacy in more complicated classifiers. M-A type classifiers are an important classifier family, in which NB and SVM classifiers are two typical representatives. Both NB and SVM classifiers are widely used in applications as diverse as spam filters [50], disease diagnosis [40], [41], [7] and face recognition [37]. In the following, we will review the related work for protecting the privacy of the NB classifier and the SVM classifier respectively.

To protect the privacy of NB classifiers, Vaidya et al. [57] introduced the use of differential privacy, and Bost et al. [10] explained how one can utilize a PHE scheme to compute the prediction value. Then, Aslett et al. [3] and Khedr et al. [26] demonstrated how one can construct secure NB classifiers based on FHE. David et al. [14] also proposed a scheme based on the so-called commodity-based model. In their scheme, they assumed that pre-distributed correlated randomness is available to both parties. In 2016, Liu et al. [35] proposed a privacy-preserving clinical decision support system based on NB classifications. These existing literature depend on either FHE (with a heavy computation overhead [3], [26]) or a strong security assumption [14]. In addition, these discussed works focus on how to construct privacy-preserving NB classifiers with the lowest misclassification rate. In other words, they assume the classifiers only simply output $\mathsf{argmax}_s \Pr(x, C_s)$ for prediction [19]. In a number of real-world applications, such as healthcare and network security, it is crucial to reduce the Bayes risk (BR), i.e., the expected misclassification loss. For example, a typical case is to distinguish between the false positive errors and the false negative errors and treat them differently [55], [54].

In this paper, we demonstrate how to use our proposed MAS-Encryption tool to minimize BR for NB classifiers, while preserving the privacy during prediction. This is, at the time of this research, the first construction of efficient privacy-preserving NB classifiers designed to minimize BR.

The work of protecting the privacy of SVM's training phase can be found in [63], [62], [59]. For the prediction phase, Rahulamathavan et al. [47], [46] proposed methods to protect Gaussian kernel-based and polynomial kernel-based SVM classifiers. Later in 2017, Li et al. [29] observed that Rahulamathavan et al.'s approaches [47], [46] have

security flaws, and hence proposed a revised version. The protocols of Rahulamathavan et al. and Li et al. [47], [46], [29] are based on additively homomorphic encryption, and are vulnerable to repetitive attacks. In other words, after multi-round execution of protocols, the client can obtain the number of the supported vectors and all their kernel values. In addition, in [47], [46], [29], to protect a value K, the cloud server blinds it with a natural number r, and sends the encryption of $K \cdot r$ to the client who has the decryption key. However, the client can repetitively request to obtain the value of $K \cdot r$ for different r values. Since all blind factors r are natural numbers, the client can get the value of K by finding the greatest common divisor between different $K \cdot r$.

In addition to the above discussed works, Lin and Chen's protocol [31] and Zhu et al.'s protocol [65] used lightweight multiparty random masking method to protect the privacy. However, both protocols cannot achieve semantic security (for the security model, please refer to [20]). In 2018, Liu et al. [34] presented a privacy-preserving outsourced SVM designed for secure drug discovery. However, for each multiplication operation in the protocol, their protocol has to perform two rounds of interaction between the cloud platform and the parties with private encryption keys [33]. Thus, the number of protocol interactions will be high if the multiplication operations in the SVM classifier increase.

We summarize several existing potential approaches for securely computing multiply-add structures as follows.

1) The general approach for two-party computation or multi-party computation can be found in [61], [9], [21], [44], where a classification function is expressed as a garbled circuit and some cryptographic tools can be used to achieve privacy-preserving for the protocol. However, such an approach usually represents the classification function as an arithmetic circuit or a binary circuit to achieve the privacy and security goals, and hence, incurs significant computation overhead. Moreover, it is known that garbled circuits are more suitable for Boolean operations or integer arithmetic operations [4], but are of high inefficiency for complicated arithmetic operations of real numbers. For example, It expresses each single multiplication or addition of two real numbers into about 13000 Non-XOR gates, and uses oblivious transfer and encryption techniques to securely compute these gates [45].
2) PHE approach. Trivially applying a PHE (e.g., Paillier Encryption, which is an AHE) cannot solve the M-A structure problem, since PHE is designed only for either multiplicative or additive homomorphism. However, PHE can achieve both kinds of homomorphism in an interactive way. Elmehdwi et al. [51], [16] used this interactive technique to realize secure k-nearest neighbor query over encrypted data. Their method is an AHE-based approach, which requires two interactions for one homomorphic multiplication of ciphertexts.
3) FHE is another commonly used method to construct privacy-preserving classifiers [3], [26], and similarly the limitation is the costs (i.e., significant ciphertext expansion and computation complexity). Cheon et al. [13] and Kim et al. [27] presented FHE-like encryption schemes that support approximate addition and multiplication of encrypted messages. However, in order to avoid message expansion, both schemes adopt a rescaling strategy each time a multiplication is performed. In other words, the ciphertext modulus decreases with homomorphic operations and their schemes no longer support homomorphic computation as the circuit depth becomes large.
4) Multi-server technique. Another line of work [25], [39], [38] use multi-server technique to protect user data privacy. This approach requires sharing users' data among non-colluding servers and assumes that the majority of servers are honest.
5) Functional Encryption (FE) is also adopted for privacy protection. This approach expresses classifiers as some specific functions and outputs the encryption of function value, which is then later used for classification purpose. However, existing FE schemes [52] can only efficiently compute simple functions such as inner product and quadratic schemes. For general multiply-add structures, there are no efficient FE solutions.

Therefore, we avoid using above approaches in this paper.

3 PRELIMINARIES

In this section, we will describe the notations and concepts relevant to the rest of this paper. Let $\mathbb{N}$ and $\mathbb{R}$ denote the set of natural numbers and the set of real numbers, respectively. If $k \in \mathbb{N}$, then $\{0,1\}^k$ denotes the set of bitstrings of bitlength k. If i, j are two integers and $i \leq j$, then we use $[i, j]$ to denote the set $\{i, i+1, \ldots, j\}$, and $[i, j]/r$ to denote the set of $\{i/r, (i+1)/r, \ldots, j/r\}$. If $r \in \mathbb{R}$, then we denote by $\lfloor r \rfloor$ the largest integral value that is not greater than r. "PPT" refers to "probabilistic polynomial-time", and all logarithms in this paper are of base 2.

Additively homomorphic encryption. In an additively homomorphic encryption, given only the public-key and the encryptions of m_1 and m_2, one can compute the encryption of $m_1 + m_2$. We utilize the Paillier encryption scheme [42] supporting negative plaintexts as our additively homomorphic public-key encryption scheme. Let $\mathsf{Enc\text{-}P}_{pk}(\cdot,\cdot)$ be the encryption function of the Paillier cryptosystem with public key pk (see also Figure 1). Suppose the modulus of the cryptosystem is n, then $\mathsf{Enc\text{-}P}_{pk}(\cdot,\cdot)$ supports the following operations:

1). $\mathsf{Enc\text{-}P}_{pk}(m_1, r_1)\mathsf{Enc\text{-}P}_{pk}(m_2, r_2) \mod n^2 = \mathsf{Enc\text{-}P}_{pk}(m_1 + m_2, r_1 \cdot r_2)$.
2). $\mathsf{Enc\text{-}P}_{pk}(m_1, r_1)^{m_2} \mod n^2 = \mathsf{Enc\text{-}P}_{pk}(m_1 \cdot m_2, r_1^{m_2})$.

For simplicity, we denote an encryption of a message m as $\mathsf{Enc\text{-}P}_{pk}(m)$ or $[\![m]\!]$ with the randomness omitted.

3.1 Naive Bayes Classifiers

The NB classifier assumes each feature of the user data is conditionally independent given the target attribute, and trains a classifier by collecting a set of classification parameters. Suppose a sample's target attribute (Y-attribute) has k possible classes $C_1, \ldots, C_k$, and that $X = (X_1, \ldots, X_t)$ is its feature vector with dimension t. Let $x = (x_1, \ldots, x_t)$ be a concrete value of the feature vector. Typically, to minimize the misclassification rate, a NB classifier predicts the target attribute's class as $\mathsf{argmax}_{1 \leq s \leq k} \Pr(Y =$

Paillier encryption scheme

Public key: $pk = (n, g)$. Private key: $sk = (\lambda, \mu)$.
Encryption:
Plaintext $m \in [-\frac{n}{2}, \frac{n}{2}] \cap \mathbb{Z}$
Select random $r \in \mathbb{Z}_n^*$
Ciphertext $c = \mathsf{Enc\text{-}P}_{pk}(m, r) = g^{m \mod n} \cdot r^n \mod n^2$
Decryption:
Ciphertext $c \in \mathbb{Z}_{n^2}^*$
Compute $\hat{m} = \frac{(c^\lambda \mod n^2)-1}{n} \cdot \mu \mod n$, and

$$\mathsf{Dec\text{-}P}_{sk}(c) = \begin{cases} \hat{m} & \text{if } 0 \le \hat{m} < \frac{n}{2}, \\ \hat{m} - n & \text{otherwise.} \end{cases}$$

Fig. 1. Paillier encryption scheme supporting negative plaintexts

$C_s) \prod_{i=1}^{t} \Pr(X_i = x_i | Y = C_s)$. However, to minimize the Bayes risk (BR), the following MBR criterion should be used.

Classifiers with minimum Bayes risk (MBR). Reducing BR is formally described using a *loss function*, which is also known as a *cost function*. The loss function is defined using a *loss matrix*, which introduces the costs of various misclassifications. Suppose the loss matrix is L, and $L_{i,s}$ denotes the cost of misclassifying class C_i as class C_s, the BR of incorrectly assigning a sample to C_s is defined by

$$LP_s = \sum_{i=1}^{k} L_{i,s} \Pr(X = x, Y = C_i).$$

Normally, $L_{i,i}$ has a value of 0 for any i. Thus,

$$LP_s = \sum_{i=1, i \neq s}^{k} L_{i,s} \Pr(X = x, Y = C_i).$$

A MBR classifier's task is to make a prediction that minimizes BR (i.e., compute $k^* = \mathsf{argmin}_{s \in [1,k]} LP_s$).

For a MBR-NB classifier, by the NB independent assumption, it computes k^* as

$$k^* = \mathsf{argmin}_{s \in [1,k]} \left\{ \sum_{i=1, i \neq s}^{k} \left(L_{i,s} \cdot \Pr(Y = C_i) \cdot \prod_{j=1}^{t} \Pr(X_j = x_j | Y = C_i) \right) \right\}. \quad (1)$$

In fact, minimizing the misclassification rate is only a special case of minimizing BR if we assume $L_{i,j} = 1$ for all $i \neq j$. In this special case, computing $\mathsf{argmin}_{1 \le s \le k} LP_s$ is simplified as computing $\mathsf{argmax}_{1 \le s \le k} \Pr(Y = C_s) \prod_{i=1}^{t} \Pr(X_i = x_i | Y = C_s)$. The latter, clearly, is a relatively simple form. However, when $L_{i,j} (i \neq j)$ is not constant and the possible class number k is larger than 2, then computing Equation (1) needs using more complex calculations.

Security model. A privacy-preserving MBR-NB (MBR-PPNB) classifier is a two-party protocol, in which the server has its private classification parameter $\mathsf{par} = \big(\{\Pr(Y = C_i)\}_{i \in [1,k]}, \{\Pr(X_j = b | Y = C_i)\}_{b \in [1,k_j], i \in [1,k], j \in [1,t]}\big)$ as input, and the user has his/her personal data $x = (x_1, \ldots, x_t)$ as input. Here, t denotes the number of features, k_j denotes the number of possible values of the j-th feature, and k denotes the number of classes of the Y-attribute (i.e. the target attribute). At the end of the protocol execution, the user learns the value $k^* = \mathsf{argmin}_{s \in [1,k]} LP_s$, and the server learns nothing. The privacy of the classifier is considered under the framework of two-party secure computations, whose details can be found in [20]. From the view of two-party computations, the functionality of a MBR-PPNB classifier is defined as $\mathcal{F}_{\mathsf{MBR\text{-}PPNBC}} : (\mathsf{par}, x) \mapsto (\perp, k^*)$, where par is the server's private input, x is the user's private input, $\perp$ denotes the empty string (i.e., no output), and k^* is defined as above.

3.2 Support Vector Machine (SVM) Classifiers

The SVM classifier tries to separate the positive and the negative samples with a hyperplane. It predicts the label of a new sample by judging the sign of $\sum_{i=1}^{m} \alpha_i y^{(i)} K(z^{(i)}, x) + b$. Here $\{(z^{(i)}, y^{(i)})\}_{1 \le i \le m}$ are m training samples, $y^{(i)} \in \{-1, 1\}$ is the i-th sample's label, $\{\{\alpha_i\}_{1 \le i \le m}, b\}$ is a set of parameters obtained from the SVM training phase, x is a testing sample with unknown label, and $K(\cdot, \cdot)$ is a kernel function. If $\sum_{i=1}^{m} \alpha_i y^{(i)} K(z^{(i)}, x) + b \ge 0$, then the new sample x's label is predicted as +1, otherwise as -1. We show in Section 6 that our technique can be applied to the the cases of $p \ge 1$ for the polynomial kernels of form $K(z, x) = \langle z, x \rangle^p = (z_1 \cdot x_1 + z_2 \cdot x_2 + \cdots + z_t \cdot x_t)^p$.

Security model. A privacy-preserving SVM (PPSVM) classifier is a two-party protocol, in which the classifier server has its private classification parameter $\mathsf{par} = \big(\{\alpha_i, z^{(i)}, y^{(i)}\}_{1 \le i \le m, \alpha_i \neq 0}, b\big)$ as input, and the user has his/her personal data $x = (x_1, \ldots, x_t)$ as input. Here, t denotes the number of features of each sample and $\{z^{(i)}\}_{1 \le i \le m, \alpha_i \neq 0}$ are called support vectors. At the end of the protocol execution, the user learns the label y of the sample x which is determined by

$$y = \mathrm{sign}\big(\sum_{i=1}^{m} \alpha_i y^{(i)} K(z^{(i)}, x) + b\big), \quad (2)$$

and the server learns nothing. From the view of two-party computations, the functionality of a PPSVM classifier is defined as $\mathcal{F}_{\mathsf{PPSVM}} : (\mathsf{par}, x) \mapsto (\perp, y)$, where par is the server's private input, x is the user's private input, $\perp$ denotes the empty string (i.e., no output), and y is defined as above.

4 MAS-Encryption (MASE)

Background: Many classifiers must securely compute a structure of $f(\{\alpha_{i,j}\}) = \sum_i (\prod_j \alpha_{i,j})$ (i.e. a M-A structure). In the following we first give an outline of a HE-based framework for securely computing a M-A structure classifier. Previous FHE-based and PHE-based approaches, and our new solution all fall in this framework.

Suppose a classifier's goal is to compute $k^* = \mathsf{argmin}_{1 \le s \le k} f(\{\alpha_{i,j}^{(s)}\})$ where a party (say, Party A) has the set $\{\alpha_{i,j}^{(s)}\}$ as input and another party (say, Party B) can

compute function $f(\cdot)$. A HE-based framework executes the following steps to securely compute k^*.

(1) Party A encrypts all his input $\alpha_{i,j}^{(s)}$ as $\mathsf{Enc}(\alpha_{i,j}^{(s)})$ and sends them to Party B. Here $\mathsf{Enc}(\cdot)$ is a HE encryption whose private key is owned by A.

(2) Party A and Party B together compute $f(\{\mathsf{Enc}(\alpha_{i,j}^{(s)})\}) = \mathsf{Enc}(f(\{\alpha_{i,j}^{(s)}\}))$ for all $1 \le s \le k$ using some homomorphic property of the encryption scheme $\mathsf{Enc}(\cdot)$. In detail, they securely
   (a) compute $\prod_j(\mathsf{Enc}(\alpha_{i,j}^{(s)})) = \mathsf{Enc}(\prod_j \alpha_{i,j}^{(s)})$ (multiplicative homomorphism), and
   (b) compute $\sum_i(\mathsf{Enc}(\prod_j \alpha_{i,j}^{(s)})) = \mathsf{Enc}(\sum_i(\prod_j \alpha_{i,j}^{(s)}))$ (additive homomorphism).

   At the end of this step, B obtains $\mathsf{Enc}(f(\{\alpha_{i,j}^{(s)}\}))$ for all s.

(3) A and B together use a secure argmin algorithm over encrypted data $\mathsf{Enc}(f(\{\alpha_{i,j}^{(s)}\}))$ to obtain $k^* = \mathsf{argmin}_{1\le s\le k} f(\{\alpha_{i,j}^{(s)}\})$.

As previous discussed, although the FHE-based approach can execute step (2) without interaction, a significant computation overhead is needed. Another alternative approach is PHE-based approach. However, the number of interactions of current PHE-based approaches depends on the size of the set $\{\alpha_{i,j}^{(s)}\}$, which results in heavy interactions. In detail, Elmehdwi et al.'s solution [51], [16] requires two communication rounds for one multiplication, and requires $2\lceil \lg k \rceil$ rounds for k ciphertexts's parallel multiplication.

Therefore, in this section, we present a new cryptographic tool to securely compute a multiply-add structure. Specifically, the tool supports real value input (i.e. $\alpha_{i,j} \in \mathbb{R}$). Furthermore, its communication round does not increase with the number of multiplication or addition operations. When it is executed in parallel, it can output multiple encrypted values that can then be used for further comparison.

Now we explain more about the comparison issue. Our goal of securely computing $f(\{\alpha_{i,j}\})$ is for comparison, rather than obtaining the accurate value. For example, let us suppose we have k different sets $\{\alpha_{i,j}^{(s)}\}_{1\le s\le k}$. In this context, we are not seeking to obtain the accurate values of $f(\{\alpha_{i,j}^{(s)}\})$. Rather, our goal is to compare $f(\{\alpha_{i,j}^{(s)}\})$ for $1 \le s \le k$ (e.g. to compute $\mathsf{argmin}_s f(\{\alpha_{i,j}^{(s)}\})$ or $\mathsf{argmax}_s f(\{\alpha_{i,j}^{(s)}\})$).

To sum up, MASE is an interactive paradigm whose goal is to securely output the encrypted value of $f(\{\alpha_{i,j}\})$ where $f(\cdot)$ is a multiply-add function. It is composed of 6 basic algorithms, and in addition, the paradigm can be implemented based on any additive encryption schemes.

High-level intuition. A MAS-Encryption (MASE) is an encryption scheme which supports real-valued plaintext. As the traditional encryption, KeyGen is the key generation algorithm, Enc is the encryption algorithm, and Dec is the decryption algorithm. The difference from a traditional encryption is that MASE is a "composed" encryption paradigm built upon an additive encryption, and the ciphertext it produces is not for decryption (i.e., we do not need to ensure that the decryption of a ciphertext will get the original plaintext). MASE has two modes of encryption: type-1 encryption supports multiplications of ciphertext, i.e., the result of multiplying multiple type-1 ciphertexts will also get type-1 ciphertext. But for the addition, type-1 ciphertext will be converted to type-2 ciphertext. Parameters l_1 and l_2 are precision control parameters for type-1/type-2 encryption, respectively. Algorithm Dec returns a decrypted real-valued plaintext, which is a "virtual" algorithm only used in the security proof. Algorithm $\mathsf{Dec\text{-}int}$ behaves like a traditional decryption algorithm, which just outputs a integer value proportional to the real-valued plaintext. In MASE, the multiplicative and additive operations on ciphertext are explicitly defined as two operators: Mul and Add. Without depending on the FHE, our concrete implementation utilizes a two-rounds protocol to realize the Add algorithm, and needs no interactions in Mul. For further intuitive explanation of each component, please refer to the remarks after the formal definition.

MASE. MASE is given by the following algorithms (i.e. $\mathcal{MASE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec\text{-}int}, \mathsf{Dec}, \mathsf{Mul}, \mathsf{Add})$):

- KeyGen, the *key generation algorithm*, is a probabilistic algorithm that takes a security parameter $\kappa \in \mathbb{N}$ and returns a public/secret key pair (pk, sk).
- Enc, the *encryption algorithm*, is a probabilistic algorithm that takes as input an encryption mode $i \in \{1, 2\}$, a public key pk, a precision parameter $l_i \in \mathbb{N}$ and a message $Q \in \mathbb{R}^+$. The output is a type-i ciphertext c, denoted as $c = \mathsf{Enc}^{\text{type-}i}_{pk,l_i}(Q)$ (or $c =< Q >_{i,l_i}$ for simplicity).
- $\mathsf{Dec\text{-}int}$, the *integerized decryption algorithm*, is a deterministic algorithm that takes a secret key sk and a ciphertext c to produce a decrypted integer $m_Q \in \{0, 1\}^*$.
- Dec, the *decryption algorithm*, is a deterministic algorithm that takes a secret key sk, a ciphertext c, a decryption mode $i \in \{1, 2\}$, and a precision parameter $l_i \in \mathbb{N}$ to produce a decrypted message $Q \in \mathbb{R}^+$. We denote it as $Q = \mathsf{Dec}^{\text{type-}i}_{sk,l_i}(c)$.
- Mul, the *multiplicative operator*, is a probabilistic algorithm that takes a public key pk, and T type-1 ciphertexts $c_1, c_2, \ldots, c_T$, to produce a type-1 ciphertext c. We denote it as $c = \mathsf{Mul}_{pk}(c_1, c_2, \ldots, c_T)$ (or $c = c_1 \cdot c_2 \cdots c_T$ for simplicity).
- Add, the *additive operator*, is a probabilistic algorithm that takes a public key pk, and K type-1 ciphertexts $c_1, c_2, \ldots, c_K$, to produce a type-2 ciphertext c. We denote it as $c = \mathsf{Add}_{pk}(c_1, c_2, \ldots, c_K)$ (or $c = c_1 + c_2 + \cdots + c_K$ for simplicity).

Notations. For $c_1 \cdot c_2$: if $c_1, c_2 \in \mathbb{R}$, then we use "$\cdot$" to denote the multiplication in $\mathbb{R}$; if $c_1, c_2 \in \mathcal{C}_1$, where $\mathcal{C}_1$ is the type-1 ciphertext domain, then "$\cdot$" is the multiplicative operator Mul. For $c_1 \cdot c_2 \mod n^2$, we view c_1, c_2 as elements in $\mathbb{Z}_{n^2}$ and "$\cdot$" is the modular multiplication.

$c_1 + c_2$ is defined similarly: if $c_1, c_2 \in \mathbb{R}$, then we use "$+$" to denote the addition in $\mathbb{R}$; if $c_1, c_2 \in \mathcal{C}_1$, then we use "$+$" to denote the additive operator Add. When we use $c_1 + c_2 \mod n$, we view c_1, c_2 as elements in $\mathbb{Z}_n$ and "$+$" is the modular addition.

For "$\simeq$": when we write $c \simeq< Q >_{i,l}$ where c is a type-i ciphertext, we require that the decrypted value of c is sufficiently close to the value of Q. In other words, there exists a sufficiently small error bound ϵ, such that

$$\left| \frac{\mathsf{Dec}^{\text{type-}i}_{sk,l}(c)}{Q} - 1 \right| < \epsilon.$$

Requirements. $\mathcal{MASE}$ should satisfy the following properties.

(1). Multiplicative homomorphism: for appropriate $T \in \mathbb{N}$, $l_1 \in \mathbb{N}$ and $Q_1, Q_2, \ldots, Q_T \in \mathbb{R}^+$, it should hold that

$$< Q_1 >_{1,l_1} \cdot < Q_2 >_{1,l_1} \cdots < Q_T >_{1,l_1} \simeq < Q_1 \cdot Q_2 \cdots Q_T >_{1,l_1} .$$

(2). Additive homomorphism: for appropriate $K \in \mathbb{N}$ and $Q_1, Q_2, \ldots, Q_K \in \mathbb{R}^+$, it should hold that

$$\begin{aligned} &< Q_1 >_{1,l_1} + < Q_2 >_{1,l_1} + \cdots + < Q_K >_{1,l_1} \\ &\simeq < Q_1 + Q_2 + \cdots + Q_K >_{2,l_2} . \end{aligned}$$

(3). Correctness of Dec: for any $Q \in \mathbb{R}^+$, $i \in \{1,2\}$, $l_i \in \mathbb{N}$, and $c = \mathsf{Enc}^{\text{type-}i}_{pk,l_i}(Q)$, there exists a sufficiently small error bound ϵ, such that

$$\left| \frac{\mathsf{Dec}^{\text{type-}i}_{sk,l_i}(c)}{Q} - 1 \right| < \epsilon.$$

(4). Order-preservation of $\mathsf{Dec\text{-}int}$: for any $Q_1, Q_2 \in \mathbb{R}^+$, $i \in \{1,2\}$ and $l_i \in \mathbb{N}$, if $Q_1 > Q_2$, then it should hold that

$$\mathsf{Dec\text{-}int}(< Q_1 >_{i,l_i}) \geq \mathsf{Dec\text{-}int}(< Q_2 >_{i,l_i}).$$

(5). Computability and security of Mul and Add. The Mul operator is a PPT algorithm and there exists an efficient two-round protocol

$$\mathsf{P_{Add}}(c_1, c_2, \ldots, c_K; sk) \mapsto \mathsf{Add}_{pk}(c_1, c_2, \ldots, c_K)$$

that securely computes the Add function.

(6). Semantic security: both type-1 and the type-2 encryption should satisfy the semantic security (i.e. IND-CPA security [8]).

Remarks.

1). Encryption and homomorphism: The type-1 encryption has multiplicative homomorphism which requires no interaction. (The property of multiplicative homomorphism is stated in Property 1.) However, to achieve additive homomorphism, an interactive protocol is required and multiplying two type-1 ciphertexts will result in a type-2 encryption. (The property of additive homomorphism is stated in Property 2. And the computability of multiplicative and additive operators is stated in Property 5.)

2). Decryption:
   a). $\mathsf{Dec}^{\text{type-}i}$ is the corresponding decryption algorithm for $\mathsf{Enc}^{\text{type-}i}$ where $i \in \{1,2\}$.
   b). Because MASE is constructed based on ordinary encryption which has a integer plaintext domain, the decryption algorithm for the integer plaintext domain - we name it $\mathsf{Dec\text{-}int}$ - is needed for implementing the homomorphic addition.

3). Order-preservation: As stated in the background part, computing $f(\{\alpha_{i,j}\})$ is for comparison, rather than obtaining the accurate value. Thus, given two encryption $\mathsf{Enc}^{\text{type-}i}(Q_1)$ and $\mathsf{Enc}^{\text{type-}i}(Q_2)$, we only require the two integers decrypted by $\mathsf{Dec\text{-}int}$ to have the same numerical order as Q_1, Q_2, we name this property as Order-preservation. Note that this order-preservation requirement is different from the definition of order-preserving encryption [28], which may lead to semantic insecurity.

4). The last property (semantic security) is needed to guarantee that any adversary cannot obtain useful information from the ciphertext without sk.

4.1 Implementation of MASE

Based on any additively homomorphic encryption scheme which is IND-CPA secure, we can construct a secure MASE paradigm. Without loss of generality, we suppose the multiply-add structure we want to securely compute is $Q_{11}Q_{12}\cdots Q_{1T} + \cdots + Q_{K1}Q_{K2}\cdots Q_{KT}$ where $Q_{ij} \in \mathbb{R}^+$ and $2^{-L_{\max}} \leq Q_{ij} \leq 2^{L_{\max}}$. In the following we use the Paillier encryption scheme $\mathcal{PE} = (\mathsf{KeyGen}, \mathsf{Enc\text{-}P}, \mathsf{Dec\text{-}P})$, where $\mathsf{KeyGen}, \mathsf{Enc\text{-}P}, \mathsf{Dec\text{-}P}$ are the key generation algorithm, encryption algorithm and decryption algorithm of Paillier respectively. The keys of construction are how to preserve the precision of the real-valued plaintext and how to maintain the order-preservation property after the multiplicative and additive operations. The following is the construction.

- KeyGen is similar to that in the Paillier key generation algorithm. We let $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\kappa)$ and $pk = (n, g)$, $sk = (\lambda, \mu)$.
- The encryption algorithms.
  - $\mathsf{Enc}^{\text{type-1}}_{pk,l_1}(Q)$. We define it as the value of $\mathsf{Enc\text{-}P}_{pk}\left(\lfloor 2^{l_1} \lg Q \rfloor\right)$.
  - $\mathsf{Enc}^{\text{type-2}}_{pk,l_2}(Q)$. We define it as the value of $\mathsf{Enc\text{-}P}_{pk}\left(\lfloor 2^{l_2} Q \rfloor\right)$.
- $\mathsf{Dec\text{-}int}$. The integerized decryption algorithm is defined as $\mathsf{Dec\text{-}int}_{sk}(c) = \mathsf{Dec\text{-}P}_{sk}(c)$.
- The decryption algorithm:
  - $\mathsf{Dec}^{\text{type-1}}_{sk,l_1}(c)$. We define it as the value of $2^{\mathsf{Dec\text{-}P}_{sk}(c)/2^{l_1}}$.
  - $\mathsf{Dec}^{\text{type-2}}_{sk,l_2}(c)$. We define it as the value of $\mathsf{Dec\text{-}P}_{sk}(c)/2^{l_2}$.
- Mul. The multiplication algorithm is defined as $\mathsf{Mul}_{pk}(c_1, c_2, \ldots, c_T) = c_1 \cdot c_2 \cdots c_T \mod n^2$.
- Add. The addition function $\mathsf{Add}(< Q_1 >_{1,l_1}, < Q_2 >_{1,l_1}, \ldots, < Q_K >_{1,l_1})$ computes a ciphertext, which is "sufficiently close" to $< Q_1 + Q_2 + \cdots + Q_K >_{2,l_2}$. Here we use a two-party protocol $\mathsf{P_{Add}}$ to realize the Add operator. As shown in Figure 2, $\mathsf{P_{Add}}$ is executed between two parties, say A and B. The party B can be anyone who only has the public key and wants to make Add operation on ciphertext, and the party A is the secret key holder, who assists the party B to get the addition result he wants.

$\mathsf{P_{Add}}$'s design philosophy. $\mathsf{P_{Add}}$ is a core protocol of MASE. It is sufficient in our scenario to compute an encrypted value that is proportional to the original plaintext, and the following is our technical subtlety. The $\mathsf{P_{Add}}$ protocol uses a blinding-then-normalization approach to ensure additive homomorphism. Specifically, suppose B has ciphertexts $< Q_1 >_{1,l_1}, \ldots, < Q_K >_{1,l_1}$. Party B first sends to party A type-1 ciphertexts which blinds $\lg Q_i$ with $-\delta_i$ (Step 1). The party A then decrypts these ciphertexts and transforms them to type-2 (Step 2). A technical subtlety is that in this step, the function $\lg(\cdot)$ acting on Q_i is stripped off and now the resulted type-2 ciphertexts has additive homomorphism. Finally, Party B normalizes these type-2 ciphertexts to obtain

Algorithm $\mathsf{P}_{\mathsf{Add}}$

Public parameters: $T, L_{\max}, K, l_1, l_2, l_\triangle, \tau \in \mathbb{N}$, where τ is the bitlength of n, $l_2 = \lfloor \tau - 1 - T \cdot L_{\max} - \lg K \rfloor$, and $l_\triangle = 20$.
Private inputs: Party A has input sk, and party B has input $(c_1, c_2, \cdots, c_K)$ and pk, where $c_i =< Q_i >_{1,l_1}$ are type-1 encryption with precision parameter l_1, and $2^{-T \cdot L_{\max}} \leq Q_i \leq 2^{T \cdot L_{\max}}$.
Output: output $c \simeq < Q_1 + Q_2 + \cdots + Q_K >_{2,l_2}$ to the party B.

1. For all $i \in [1, K]$, party B randomly selects $\delta_i \in [2^{l_1} \cdot l_\triangle, 2^{l_1} \cdot (\tau - 1)]/2^{l_1}$ (i.e., δ_i is a random real number which has a $\lfloor \lg(\tau - 1) \rfloor$-bits integer part and an l_1-bits fraction part), computes $e_i = c_i \cdot [\![\lfloor -\delta_i \cdot 2^{l_1} \rfloor]\!] \mod n^2$, and sends e_i to party A.
   ▷ $e_i \simeq [\![\lfloor 2^{l_1} \cdot (\lg Q_i - \delta_i) \rfloor]\!]$
2. Party A computes $d_i = \mathsf{Dec\text{-}int}_{sk}(e_i)/2^{l_1}$, computes $\hat{e}_i = [\![\lfloor 2^{l_2 + d_i} \rfloor]\!]$, and sends $\hat{e}_i$ to party B.
   ▷ $d_i \simeq \lg Q_i - \delta_i$
   ▷ $\hat{e}_i \simeq [\![\lfloor 2^{l_2} \cdot Q_i \cdot 2^{-\delta_i} \rfloor]\!]$
3. Party B computes $\hat{c}_i = \hat{e}_i^{\lfloor 2^{\delta_i} \rfloor} \mod n^2$, multiply them to get $\hat{c} = \hat{c}_1 \cdot \hat{c}_2 \cdots \hat{c}_K \mod n^2$, and outputs $\hat{c}$.
   ▷ $\hat{c}_i \simeq [\![\lfloor 2^{l_2} \cdot Q_i \rfloor]\!]$
   ▷ $\hat{c} \simeq [\![\lfloor 2^{l_2} \cdot (Q_1 + Q_2 + \cdots + Q_K) \rfloor]\!]$

Fig. 2. Addition algorithm in MASE paradigm

ciphertexts $\hat{c}_i$ which have order-preserving plaintexts and multiplication of them can get an encrypted value which is proportional to $(Q_1 + .. + Q_K)$ (Step 3). We note that the algorithm is essentially running in a batch mode, because the number of its interactions does not depend on the number of addends (i.e., K). The following are two remarks focusing on some details.

1. δ_i is used to blind $\lg Q_i$ in the ciphertext e_i.
2. At the end of step 2, $\hat{e}_i$ has a form of $[\![\lfloor 2^{l_2} \cdot Q_i \cdot 2^{-\delta_i} \rfloor]\!]$(this is shown in the proof of Theorem 2). In order to get Q_i, a straightforward idea is to remove the factor $2^{-\delta_i}$ by computing $\hat{e}_i^{\zeta}$ where ζ is the inverse of $2^{-\delta_i}$, i.e., $\zeta = 2^{\delta_i} \mod n$. However we will find that doing this is not helpful because $2^{-\delta_i}$ is not an integer and we will not get desired result even if we let $\zeta = \lfloor 2^{(-\delta_i)} \rfloor^{-1} \mod n$. Fortunately, we do not need to accurately recover the value of Q_i, whilst we only need to add up all Q_i. Therefore in step 3, Party B adds δ_i to fill in the exponent, and get a common factor 2^{l_2} in the plaintext of $\hat{c}_i$. We note that $\hat{c}_i, \hat{c}$ are not held by Party A, thus Party A cannot benefit from these values in any attack.

4.2 Proof of Properties

We will now prove that the above implementation is a valid MASE satisfying the requirements described in Section 4.

(1). The multiplicative homomorphism is proven in Theorem 1.
(2). The additive homomorphism is proven in Theorem 2.
(3). The correctness of decryption algorithm Dec is proven in Theorem 3.
(4). Order-preservation of Dec-int. From the construction of the encryption algorithm, demonstrating the property of order-preservation is straightforward.
(5). The computability of Mul and $\mathsf{P}_{\mathsf{Add}}$ is also straightforward to demonstrate. The security of $\mathsf{P}_{\mathsf{Add}}$ is shown in Theorem 4.
(6). The semantic security can be easily derived from the IND-CPA security of the Paillier cryptosystem.

Theorem 1 shows the multiplicative homomorphism of our construction, which essentially states that

$$< Q_1 >_{1,l} \cdot < Q_2 >_{1,l} \cdots < Q_T >_{1,l} \simeq < Q_1 \cdot Q_2 \cdots Q_T >_{1,l}.$$

Theorem 1. The constructed MASE satisfies the multiplicative homomorphism. Formally, for any $l_1, T \in \mathbb{N}$ such that $\lg T \leq \min\{l_1 - l_\triangle, \tau - 1 - l_1 - \lg L_{\max}\}$, and $Q_1, \ldots, Q_T \in \mathbb{R}^+$ satisfying $|\lg Q_j| \leq L_{\max}$ for all $j \in [1, T]$,

$$\left| \frac{\mathsf{Dec}^{\text{type-1}}_{sk,l_1}(< Q_1 >_{1,l_1} \cdot < Q_2 >_{1,l_1} \cdots < Q_T >_{1,l_1})}{Q_1 \cdot Q_2 \cdots Q_T} - 1 \right| < 2^{-l_\triangle}. \quad (3)$$

Proof outline: We want to ensure that the multiplication of type-1 ciphertexts $< Q_1 >_{1,l_1}, \ldots, < Q_1 >_{T,l_1}$ results in a ciphertext which is "approximately" the type-1 encryption of the value $Q_1 \cdot Q_2 \cdots Q_T$. Therefore, we want to guarantee that:
1). The resulted plaintext filled in the Paillier plaintext domain does not exceed the value $n/2$ (recall that $n/2$ is the maximum positive plaintext in Paillier encryption);
2). The type-1 decryption of $c =< Q_1 >_{1,l_1}, \ldots, < Q_1 >_{T,l_1}$ is sufficiently close to $Q = Q_1 \cdot Q_2 \cdots Q_T$. The key to the proof is that we need to compare the decryption of c with Q, and prove that they are sufficiently close.

Proof 1. First of all, we need to ensure no overflow occurs during the calculation, i.e.,

$$|2^{l_1} \cdot (\lg Q_1 + \cdots + \lg Q_T)| < 2^{\tau - 1},$$

which is guaranteed by $|\lg Q_j| \leq L_{\max}$ and $\lg T \leq \tau - 1 - l_1 - \lg L_{\max}$.
Because $\mathsf{Dec}^{\text{type-1}}_{sk,l_1}(< Q_1 >_{1,l_1} \cdot < Q_2 >_{1,l_1} \cdots < Q_T >_{1,l_1}) = 2^\beta$, where $\beta = (\lfloor 2^{l_1} \lg Q_1 \rfloor + \cdots + \lfloor 2^{l_1} \lg Q_T \rfloor) \cdot 2^{-l_1}$, to prove Equation (3), we only need to show

$$\left| \frac{2^\beta}{Q_1 \cdot Q_2 \cdots Q_T} - 1 \right| < 2^{-l_\triangle}.$$

Since

$$\begin{aligned} & 2^{l_1} \lg Q_1 + \cdots + 2^{l_1} \lg Q_T - T \\ & \leq \lfloor 2^{l_1} \lg Q_1 \rfloor + \cdots + \lfloor 2^{l_1} \lg Q_T \rfloor \\ & \leq 2^{l_1} \lg Q_1 + \cdots + 2^{l_1} \lg Q_T, \end{aligned}$$

we know

$$Q_1 \cdot Q_2 \cdots Q_T \cdot 2^{-T \cdot 2^{-l_1}} \leq 2^\beta \leq Q_1 \cdot Q_2 \cdots Q_T.$$

So,

$$0 \le 1 - \frac{2^{\beta}}{Q_1 \cdot Q_2 \cdots Q_T} \le 1 - 2^{-T \cdot 2^{-l_1}}.$$

Because

$$1 - 2^{-x} \le x \text{ holds for any } x \ge 0, \tag{4}$$

we get

$$\left| \frac{2^{\beta}}{Q_1 \cdot Q_2 \cdots Q_T} - 1 \right| \le T \cdot 2^{-l_1} = 2^{\lg T - l_1} < 2^{-l_\triangle}.$$

This completes the proof of the theorem.
Thus, as long as $2^{l_\triangle}$ is sufficiently large (e.g. $l_\triangle > 20$), the property of multiplicative homomorphism will be satisfied.

Theorem 2 shows the additive homomorphism of our construction, which states that

$$\begin{aligned} &< Q_1 >_{1,l_1} + < Q_2 >_{1,l_1} + \cdots + < Q_K >_{1,l_1} \\ \simeq &< Q_1 + Q_2 + \cdots + Q_K >_{2,l_2}. \end{aligned}$$

The key to the proof is similar to that of Theorem 1.

Theorem 2. The constructed MASE satisfies the additive homomorphism. Formally, for any $l_1, l_2, K \in \mathbb{N}$ satisfying $l_2 + T \cdot L_{\max} + \lg K + 1 \le \tau$, and $Q_1, Q_2, \ldots, Q_K \in \mathbb{R}^+$ satisfying $|\lg Q_i| \le T \cdot L_{\max}$ for all $i \in [1, K]$,

$$\begin{aligned} &\left| \frac{\mathsf{Dec}^{\text{type-2}}_{sk,l_2}(< Q_1 >_{1,l_1} + \cdots + < Q_K >_{1,l_1})}{Q_1 + Q_2 + \cdots + Q_K} - 1 \right| \\ &< 2^{-l_\triangle}, \end{aligned} \tag{5}$$

where $l_\triangle$ is defined as $l_\triangle = \min\{l_1 - 1, \delta_i, l_2 + \lg Q_i - \delta_i\} - 3$, and δ_i are defined as the values in executing $\mathsf{P}_{\mathsf{Add}}$.

Proof 2. We first prove no overflow occurs during the calculations, i.e., the constraints $2^{\delta_i} \le 2^{\tau - 1}$ and $2^{l_2} \cdot (Q_1 + \cdots + Q_K) \le 2^{\tau - 1}$ are satisfied. From the setting of the values of δ_i, l_1, l_2, we can ensure that $\lg \delta_i \le \tau - 1$ and $l_2 + T \cdot L_{\max} + \lg K + 1 \le \tau$, which guarantees the constraints.
In order to prove Equation (5), we need to prove

$$\left| \frac{\beta}{Q_1 + Q_2 + \cdots + Q_K} - 1 \right| < 2^{-l_\triangle},$$

where $\beta = \mathsf{Dec}^{\text{type-2}}_{sk,l_2}(< Q_1 >_{1,l_1} + \cdots + < Q_K >_{1,l_1}) = (\lfloor 2^{l_2} \cdot 2^{d_1} \rfloor \cdot \lfloor 2^{\delta_1} \rfloor + \cdots + \lfloor 2^{l_2} \cdot 2^{d_K} \rfloor \cdot \lfloor 2^{\delta_K} \rfloor) \cdot 2^{-l_2 - r}$.
From $d_i = (\lfloor 2^{l_1} \lg Q_i \rfloor + \lfloor -\delta_i \cdot 2^{l_1} \rfloor) \cdot 2^{-l_1}$ and the equation

$$\begin{aligned} &2^{l_1} \lg Q_i - \delta_i \cdot 2^{l_1} - 2 \\ &\le 2^{l_1} \lg Q_i - \delta_i \cdot 2^{l_1}, \end{aligned}$$

we have the following:

$$\lg Q_i - \delta_i - 2/2^{l_1} \le d_i \le \lg Q_i - \delta_i.$$

Let $\beta_i = \lfloor 2^{l_2} \cdot 2^{d_i} \rfloor \cdot \lfloor 2^{\delta_i} \rfloor$, we arrive at

$$\begin{aligned} &(2^{l_2} \cdot 2^{\lg Q_i - \delta_i} \cdot 2^{-2^{1-l_1}} - 1) \cdot (2^{\delta_i} - 1) \\ &\le \beta_i \\ &\le 2^{l_2} \cdot Q_i \cdot 2^r. \end{aligned}$$

As we also know

$$\begin{aligned} &(2^{l_2} \cdot 2^{\lg Q_i - \delta_i} \cdot 2^{-2^{1-l_1}} - 1) \cdot (2^{\delta_i} - 1) \\ &= 2^{l_2} \cdot Q_i \cdot 2^r \cdot (2^{-2^{1-l_1}} - 2^{-2^{1-l_1}} \cdot 2^{-\delta_i} \\ &\quad - \frac{1}{2^{l_2} Q_i \cdot 2^{-\delta_i}} + \frac{1}{2^{l_2} Q_i \cdot 2^r}) \end{aligned}$$

Let

$$\alpha = 2^{-2^{1-l_1}} - 2^{-2^{1-l_1}} \cdot 2^{-\delta_i} - \frac{1}{2^{l_2} Q_i \cdot 2^{-\delta_i}} + \frac{1}{2^{l_2} Q_i \cdot 2^r},$$

and we can deduce that

$$1 - \alpha \le 2^{-l_\triangle}$$

where $l_\triangle$ is defined as

$$l_\triangle = \min\{l_1 - 1, \delta_i, l_2 + \lg Q_i - \delta_i\} - 3.$$

So,

$$\begin{aligned} &\left| \frac{\beta}{Q_1 + Q_2 + \cdots + Q_K} - 1 \right| \\ &= 1 - \frac{\beta_1 + \cdots + \beta_K}{2^{l_2} \cdot (Q_1 + \cdots + Q_K)} \\ &\le 1 - \alpha \\ &\le 2^{-l_\triangle}. \end{aligned}$$

Thus, as long as $2^{l_\triangle}$ is sufficiently large (e.g. the party B can ensure that $\min\{l_1 - 1, \delta_i, l_2 + \lg Q_i - \delta_i\} - 3 > 20$ and takes $l_\triangle = 20$), the property of additive homomorphism will be satisfied.

Theorem 3 shows that if $2^{\min(l_1 - 1, l_2 - T \cdot L_{\max})}$ is sufficiently large, the correctness of decryption algorithm Dec is satisfied.

Theorem 3. For all Q which are computed from the multiply-add structure, and $c = \mathsf{Enc}^{\text{type-}i}_{pk,l_i}(Q)$, it satisfies that

$$\left| \frac{\mathsf{Dec}^{\text{type-}i}_{sk,l_i}(c)}{Q} - 1 \right| < 2^{-\min(l_1 - 1, l_2 - T \cdot L_{\max})}.$$

Proof 3. Suppose $c = \mathsf{Enc}^{\text{type-}i}_{pk,l_i}(Q)$.
1). For the case of $i = 1$,

$$Q \cdot 2^{-2^{-l_1}} \le \mathsf{Dec}^{\text{type-1}}_{sk,l_1}(c) \le Q \cdot 2^{2^{-l_1}},$$

which means

$$2^{-2^{-l_1}} \le \frac{\mathsf{Dec}^{\text{type-1}}_{sk,l_1}(c)}{Q} \le 2^{2^{-l_1}}.$$

Then from inequality (4) we get

$$\left| \frac{\mathsf{Dec}^{\text{type-1}}_{sk,l_1}(c)}{Q} - 1 \right| \le \frac{1}{2^{l_1 - 1}}.$$

2).For the case of $i = 2$,

$$Q - 2^{-l_2} \le \mathsf{Dec}^{\text{type-2}}_{sk,l_2}(c) \le Q + 2^{-l_2},$$

which means

$$1 - \frac{1}{Q \cdot 2^{l_2}} \le \frac{\mathsf{Dec}^{\text{type-2}}_{sk,l_2}(c)}{Q} \le 1 + \frac{1}{Q \cdot 2^{l_2}}.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2020.3009221, IEEE Transactions on Knowledge and Data Engineering

Because Q is computed from $Q_{11}Q_{12}\cdots Q_{1T}+\cdots+Q_{K1}Q_{K2}\cdots Q_{KT}$ where $Q_{ij}\in\mathbb{R}^+$ and $2^{-L_{\max}}\le Q_{ij}\le 2^{L_{\max}}$, we get

$$\left|\frac{\mathsf{Dec}^{\text{type-2}}_{sk,l_2}(c)}{Q}-1\right|\le\frac{1}{Q\cdot 2^{l_2}}\le\frac{1}{(2^{-T\cdot L_{\max}})\cdot 2^{l_2}}=\frac{1}{2^{l_2-T\cdot L_{\max}}}.$$

This completes the proof.

Theorem 4 shows that the protocol $\mathsf{P}_{\mathsf{Add}}$ securely computes the function Add. In this paper, all two-party computation protocols work in the semi-honest model as described in [20], where all adversaries are curious-but-honest. In other words, these adversaries act honestly according as per protocol specifications, but they may attempt to learn or infer as much private information about the input/output of the other party by passively observing the entire transcript / communication.

Theorem 4. Our constructed protocol $\mathsf{P}_{\mathsf{Add}}$ securely computes the function Add in the semi-honest model if Paillier cryptosystem is semantically secure.

Proof 4. Now we follow the two-party computation's paradigm to show that there exist two PPT simulators that can simulate party A and B's views, separately.

Simulate party B's view. In the protocol's execution, party B's view is $\mathsf{View}_B=(pk,c_1,\ldots,c_K;\{\delta_i\}_{i\in[1,K]};\{\hat{e}_i\}_{i\in[1,K]})$. The simulator S_B, on input $pk,c_1,\ldots,c_K,\hat{c}$, performs the following:

(1). Randomly generate the blinding pairs $\{(r'_i,\delta'_i)\}_{i\in[1,K]}$ as in the real protocol $\mathsf{P}_{\mathsf{Add}}$.

(2). Randomly generate $\{\hat{c}'_i\}_{i\in[1,K]}$ such that $\hat{c}=\hat{c}'_1\cdot\hat{c}'_2\cdots\hat{c}'_K \mod n^2$.

(3). Compute $\hat{e}'_i=(\hat{c}'_i)^{\lfloor 2^{\delta'_i}\rfloor^{-1}}$, where $\lfloor 2^{\delta'_i}\rfloor^{-1}\in\mathbb{Z}^*_n$ is the inverse of $\lfloor 2^{\delta'_i}\rfloor \mod n$.

(4). Output $\mathsf{S\text{-}View}_B=(pk,c_1,\ldots,c_K;\{r'_i,\delta'_i\}_{i\in[1,K]};\{\hat{e}'_i\}_{i\in[1,K]})$.

Since Paillier cryptosystem is semantically secure, and the constraint

$$\hat{c}=(\hat{e}'_1)^{\lfloor 2^{\delta'_1}\rfloor}\cdot(\hat{e}'_2)^{\lfloor 2^{\delta'_2}\rfloor}\cdots(\hat{e}'_K)^{\lfloor 2^{\delta'_K}\rfloor} \mod n^2$$

is also satisfied, it follows that both distributions View_B and $\mathsf{S\text{-}View}_B$ are computationally indistinguishable.

Simulate party A's view. In the protocol's execution, party A's view is $\mathsf{View}_A=(sk,\{e_i\}_{i\in[1,K]})$. S_A, on input sk, performs the following:

(1). Randomly generate K Paillier ciphertexts $\{e'_i\}_{i\in[1,K]}$.

(2). Output $\mathsf{S\text{-}View}_A=(sk,\{e'_i\}_{i\in[1,K]})$

Since the plaintext in e_i is blinded by the random factor $-\delta_i$, both distributions View_A and $\mathsf{S\text{-}View}_A$ are indistinguishable. Formally, the integer $\lfloor 2^{l_1}\cdot\lg Q_i\rfloor$, which has at most $l_1+\lg(T\cdot L_{\max})$ bits, is blinded by an integer $\lfloor-\delta_i\cdot 2^{l_1}\rfloor$, which has $l_1+\lg(\tau-1)$ bits. So the distributions are distinguishable with an advantage of $\frac{T\cdot L_{\max}}{\tau-1}$ at most.

5 A Privacy-preserving Naive Bayes Classifier with Minimum Bayes Risk (MBR-PPNBC)

Based on our MASE, we give a simple and elegant construction for the MBR-PPNBC. In the proposed construction, a user with personal data x and a classifier with classification parameter par can collectively output a private prediction for the user. We depict the architecture of our MBR-PPNBC and the key technique in Figure 3.

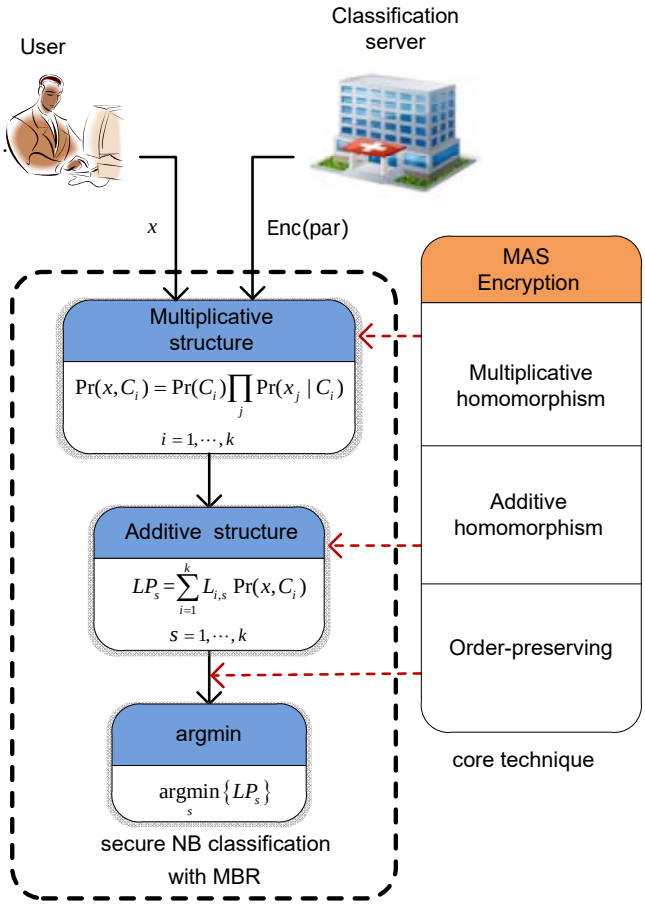


Fig. 3. Architecture of the MBR-PPNB classifier.

5.1 Construction

Intuition. A MBR-PPNB classifier wishes to securely compute $\mathsf{argmin}_{s\in[1,k]}LP_s$, where LP_s has a multiply-add structure (i.e. $LP_s=\sum_{i=1,i\neq s}^{k}\left(L_{i,s}\Pr(Y=C_i)\prod_{j=1}^{t}\Pr(X_j=x_j|Y=C_i)\right)$). The server first uses the encryption algorithm of MASE to encrypt its secret parameters and transfers them to the user. Then, the user utilizes the multiplicative and additive operators to generate the encryption of LP_s's for $i\in[1,k]$. Finally, the prediction k^* can be obtained by a secure comparison protocol being applied on the ciphertext of LP_s's. The correctness of k^* can be guaranteed by the order-preservation property of MASE, and the security can be guaranteed by the security of MASE.

MBR-PPNBC construction. The protocol is split into an online phase and an offline phase. In the latter phase, the server undertakes preparatory activities prior to the user launching the classification service request. When the input x is known (i.e. a user initiates the request), the online phase will commence. In this phase, the server and user utilize

the result of the pre-computation and the execution of the protocol is relatively brief. The use of such an online/offline technique is generally to improve the efficiency of network protocols.

Public input: l_1, l_2 are public precision parameters.

Server's input: The server has its NB classifier's secret parameters $\{\Pr(Y = C_i)\}_{i\in[1,k]}, \{\Pr(X_j = b|Y = C_i)\}_{b\in[1,k_j],i\in[1,k],j\in[1,t]}$, and the MASE's secret key sk as input.

User's input: The user has his/her personal data $x = (x_1, \ldots, x_t)$, and the MASE's public key pk as input.

Output: The prediction k^* with minimum expected loss is given as the output to the user.

Collectively, both server and user execute the following protocol π.

1) Off-line stage. The server computes $< \Pr(X_j = b|C_i) >_{1,l_1}$ and $< \Pr(Y = C_i) >_{1,l_1}$ for all $i \in [1,k], j \in [1,t], b \in [1,k_j]$ using the type-1 encryption algorithm, and transmits them to the user.

2) On-line stage.

2.1 For all $i, s \in [1,k]$, the user computes

$$\tilde{P}_i =< \Pr(Y = C_i) >_{1,l_1} \cdot \prod_{j=1}^{t} < \Pr(x_j|C_i) >_{1,l_1} \quad (6)$$

and

$$\widetilde{LP_{i,s}} =< L_{i,s} >_{1,l_1} \cdot \tilde{P}_i \quad (7)$$

using the multiplicative operator.

2.2 The server and the user jointly execute

$$\widetilde{LP_s} \leftarrow \mathsf{P_{Add}}(\{\widetilde{LP_{i,s}}\}_{i\in[1,k],i\neq s}) \quad (8)$$

in parallel for all $s \in [1,k]$. Here the server/user play the roles of Party A/Party B respectively.

2.3 The server and the user jointly execute

$$k^* \leftarrow \mathsf{P_{argmin}}(\{\widetilde{LP_s}\}_{s\in[1,k]}), \quad (9)$$

where $\mathsf{P_{argmin}}$ is a two-party protocol to jointly compute argmin over encrypted data. Here we adopt the secure comparison protocol proposed by Bost et al., whose details and security proof can be found in [10]. The protocol proposed in [10] is secure in the semi-honest model and has $8(k-1)$ rounds of communication.

The above proposed paradigm also works if we replace MASE with FHE, as long as the FHE supports real-valued plaintext, e.g., Kim et al.'s FHE-like encryption [13]. The only difference when applying FHE is that we should instead use multiplicative and additive homomorphism of FHE to accomplish the step 2.1 and 2.2 in the MBR-PPNB protocol. We'll see in Section 7 that the MASE based paradigm outperforms the FHE-based one.

5.2 Security and Efficiency

Correctness. We show that the MBR-PPNB classifier outputs the correct prediction satisfying Equation (1).

From Equation (6), (7) and the multiplicative homomorphism, we have

$$\widetilde{LP_{i,s}} \simeq< L_{i,s} \cdot \Pr(Y = C_i) \cdot \prod_{j=1}^{t} \Pr(x_j|C_i) >_{1,l_1} \quad (10)$$

From Equation (6), (10) and the additive homomorphism, we have

$$\widetilde{LP_s} = \sum_{i\neq s} \widetilde{LP_{i,s}} \simeq< \sum_{i\neq s} \left(L_{i,s} \cdot \Pr(Y = C_i) \cdot \prod_{j=1}^{t} \Pr(x_j|C_i) \right) >_{2,l_2} \quad (11)$$

By the order-preserving property,

$$k^* = \mathsf{P_{argmin}}(\{\widetilde{LP_s}\}_{s\in[1,k]}) = \mathsf{argmin}_{s\in[1,k]}\{LP_s\},$$

where

$$LP_s = \sum_{i\neq s} \left(L_{i,s} \cdot \Pr(Y = C_i) \cdot \prod_{j=1}^{t} \Pr(x_j|C_i) \right).$$

Thus, the output of the protocol is what we expected.

Security. Theorem 5 describes the privacy–preserving property of the protocol in the semi-honest model.

Theorem 5. The MBR-PPNBC protocol $\pi^{\mathsf{P_{Add}},\mathsf{P_{argmin}}}$ securely computes $\mathcal{F}_{\mathsf{MBR\text{-}PPNBC}}$ in the semi-honest model.

Proof 5. To prove this theorem, we only need to show that $\pi^{\mathsf{Add},\mathsf{argmin}}$ securely computes $\mathcal{F}_{\mathsf{MBR\text{-}PPNBC}}$ in the semi-honest model. Since $\mathsf{P_{Add}}$ securely computes Add, and $\mathsf{P_{argmin}}$ securely computes argmin, then by the composition theorem [12], [32], we can conclude that $\pi^{\mathsf{P_{Add}},\mathsf{P_{argmin}}}$ securely computes $\mathcal{F}_{\mathsf{MBR\text{-}PPNBC}}$ in the semi-honest model. Now we show that there exist two PPT simulators that can simulate the user and the server's view in executing $\pi^{\mathsf{Add},\mathsf{argmin}}$.

Simulate the user's view. In the protocol's execution, the user's view is $\mathsf{View}_U = (pk; x; \{< \Pr(X_j = b|C_i) >_{1,l_1}\}_{i\in[1,k],j\in[1,t],b\in[1,k_j]}, \{< \Pr(Y = C_i) >_{1,l_1}\}_{i\in[1,k]}, \{\widetilde{LP_s}\}_{s\in[1,k]}; k^*)$. The simulator S_U, on input pk, x, k^* performs the following:

(1). Randomly generate $k*(\sum_{j=1}^{t} k_j+1)$ type-1 ciphertexts: $\{e_{i,j}^{(b)}\}_{i\in[1,k],j\in[1,t],b\in[1,k_j]}, \{e_i\}_{i\in[1,k]}$,

(2). Randomly generate k type-2 ciphertexts: $\{\hat{e}_i\}_{i\in[1,k]}$,

(3). Output $\mathsf{S\text{-}View}_U = (pk; x; \{e_{i,j}^{(b)}\}_{i\in[1,k],j\in[1,t],b\in[1,k_j]}, \{e_i\}_{i\in[1,k]}, \{\hat{e}_i\}_{i\in[1,k]}; k^*)$. Since the MASE cryptosystem is semantically secure, both distributions View_U and $\mathsf{S\text{-}View}_U$ are computationally indistinguishable.

Simulate the server's view. In the $(\mathsf{Add}, \mathsf{argmin})$-hybrid model where we view Add and argmin as ideal functionalities, the server receives nothing in executing the protocol $\pi^{\mathsf{Add},\mathsf{argmin}}$. Thus, the simulator S_S for the server is trivially constructed in the sense that it only outputs the server's initial input and random input (coins) used.

Efficiency. The computation/communication cost of above MBR-PPNB protocol is summarized as follows. The analysis is applicable to both MASE-based and FHE-based paradigms. Let t_e, t_m, t_a denote the time of encryption, multiplication, and addition per ciphertext of the basic encryption scheme, and let l_e denote the length of a ciphertext. Computational cost: $O(t_e \cdot k(k_1 + .. + k_t))$ for Server, and $O(t_a \cdot k^2 + t_m \cdot (k^2 + kt))$ for User. Communicational cost: $O(l_e \cdot k(k + k_1 .. + k_t))$ for Server, and $O(l_e \cdot k)$ for User.

In addition, MASE based paradigm has 1 round of offline communication and $8k - 6$ rounds of online communications, namely: 2 rounds in the $\mathsf{P_{Add}}$ protocol and $8(k-1)$

rounds in the argmin protocol. The communication round of the FHE-based MBR-PPNB has two less rounds than the MASE based paradigm (the FHE needs no communications in the Add operation).

Elmehdwi et al.'s AHE-based approach uses an interactive method to obtain multiplicative homomorphism. Their method needs $2\lceil \lg k \rceil$ communication rounds to perform a batch multiplication of k ciphertexts [51]. Thus the communication rounds of their approach is $2\lceil \lg(t+1) \rceil + 8(k-1)$, and the communication/computational complexity are $O(l_e \cdot k(4k+4t+k_1..+k_t))$ and $O(t_e \cdot k(k_1+..+k_t)+t_a \cdot k^2+t_m \cdot (k^2+kt))$ respectively.

6 Privacy-preserving SVM (PPSVM) classifiers

We show another application of the MASE: a privacy-preserving SVM classifier. The SVM classifier we consider in this section is a binary classifier and the kernels we use is a set of polynomial kernels of form $K(z,x) = \langle z,x \rangle^p = (z_1 \cdot x_1 + z_2 \cdot x_2 + \cdots + z_t \cdot x_t)^p$, $p \geq 1, p \in \mathbb{N}$.

6.1 Construction

Intuition. A PPSVM classifier wishes to securely compute $y = \text{sign}\big(\sum_{i=1}^m \alpha_i y^{(i)} K(z^{(i)},x) + b\big)$. We first convert the problem into computing $\mathsf{argmax}_{i\in\{+1,-1\}}\{\gamma_{+1},\gamma_{-1}\}$ such that γ_i has a multiply-add form, and then we can apply our MASE paradigm. Here we adopt the server-centric model in order to reduce transmissions. In protocol construction, the user first uses the type-1 encryption algorithm of MASE to encrypt its feature values and transfers them to the classification server. At the server side, the server utilizes the multiplicative and additive operators to generate the encryption of γ_i's for $i \in \{+1,-1\}$. Finally, the prediction y can be obtained by a secure comparison protocol being applied on the ciphertext of γ_{+1} and γ_{-1}. The correctness of y can be guaranteed by the order-preservation property of MASE, and the security can be guaranteed by the security of MASE.

Converting to the multiply-add structure. To apply MASE, we need to convert the SVM classifier into a multiply-add structure first. When the parameter p of the kernel equals to 1, the classifier is already an additive structure and the construction is trivial. In the following we focus on the case of $p > 1$.

Firstly, the task of computing $\text{sign}\big(\sum_{i=1}^m \alpha_i y^{(i)} K(z^{(i)},x) + b\big)$ is equivalent of computing $\mathsf{argmax}_{j\in\{0,1\}}\{\beta_j\}$ where $\beta_1 = \sum_{i=1}^m \alpha_i y^{(i)} \langle z^{(i)},x \rangle^p + b$ and $\beta_0 = 0$. This is derived from the definition of the SVM algorithm, which predicts the new sample x's label as:

$$\text{Predicted label of } x = \begin{cases} +1 & \text{if } \beta_1 \geq \beta_0, \\ -1 & \text{if } \beta_1 < \beta_0. \end{cases}$$

Denote $S = \{i : 1 \leq i \leq m, \alpha_i \neq 0\}$ be the index set of support vectors, and let $c^+_{j_1..j_p} = \sum_{i\in S, y^{(i)}=1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p}$, $c^-_{j_1..j_p} = \sum_{i\in S, y^{(i)}=-1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p}$. To ensure that each item in the formula has a positive value, we rearrange the formula as follows. [1]

$$\begin{aligned} & \text{argmax}\{\sum_{i=1}^m \alpha_i y^{(i)} \langle z^{(i)},x \rangle^p + b, 0\} \\ = \; & \text{argmax}\{\sum_{i\in S, y^{(i)}=1} \alpha_i \langle z^{(i)},x \rangle^p + b, \sum_{i\in S, y^{(i)}=-1} \alpha_i \langle z^{(i)},x \rangle^p\} \\ = \; & \begin{cases} \text{argmax}\{\beta^+ + |b|, \beta^-\} & \text{if } b > 0, \\ \text{argmax}\{\beta^+, \beta^- + |b|\} & \text{if } b < 0 \end{cases} \end{aligned} \tag{12}$$

where

$$\beta^+ = \sum_{j_1,..,j_p=1}^t c^+_{j_1..j_p} x_{j_1} \cdots x_{j_p} \tag{13}$$

and

$$\beta^- = \sum_{j_1,..,j_p=1}^t c^-_{j_1..j_p} x_{j_1} \cdots x_{j_p} \tag{14}$$

are both multiply-add structures, and we are done.

PPSVM construction. The protocol is in a server-centric model and is also divided into on-line/off-line phases.
Public input: l_1, l_2 are public precision parameters.
Server's input: The server has its SVM classifier's secret parameters $\mathsf{par} = \big(\{\alpha_i, z^{(i)}, y^{(i)}\}_{1\leq i\leq m, \alpha_i\neq 0}, b\big)$, and the MASE's public key pk as input.
User's input: The user has his/her personal data $x = (x_1,\ldots,x_t)$, and the MASE's private key sk as input.
Output: The prediction y denoting the sample x's label is given as the output to the user.

The server and user execute the following protocol π.
1) Off-line stage. The server computes $\tilde{c}^+_{j_1..j_p} =< \sum_{i\in S, y^{(i)}=1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p} >_{1,l_1}$, $\tilde{c}^-_{j_1..j_p} =< \sum_{i\in S, y^{(i)}=-1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p} >_{1,l_1}$, and $\tilde{b} =< |b| >_{1,l_1}$ for all $j_1,\ldots,j_p \in [1,t]$ using the type-1 encryption algorithm.
2) On-line stage.
2.1 The user computes $< x_j >_{1,l_1}$ for all $j \in [1,t]$ and transmits them to the server.
2.2 For all $j_1,\ldots,j_p \in [1,t]$, the server computes

$$\tilde{\beta}^+_{j_1..j_p} = \tilde{c}^+_{j_1..j_p} \cdot < x_{j_1} >_{1,l_1} \cdots\cdots < x_{j_p} >_{1,l_1} \tag{15}$$

and

$$\tilde{\beta}^-_{j_1..j_p} = \tilde{c}^-_{j_1..j_p} \cdot < x_{j_1} >_{1,l_1} \cdots\cdots < x_{j_p} >_{1,l_1} \tag{16}$$

using the multiplicative operator.
2.3 The server and user jointly execute

$$\begin{cases} \tilde{\gamma}_{+1} \leftarrow \mathsf{P}_{\mathsf{Add}}(\{\tilde{\beta}^+_{j_1..j_p}\}_{j_1,..,j_p\in[1,t]} \cup \tilde{b}) \\ \tilde{\gamma}_{-1} \leftarrow \mathsf{P}_{\mathsf{Add}}(\{\tilde{\beta}^-_{j_1..j_p}\}_{j_1,..,j_p\in[1,t]} \cup \tilde{\perp}) \end{cases} \tag{17}$$

if $b > 0$, or

$$\begin{cases} \tilde{\gamma}_{+1} \leftarrow \mathsf{P}_{\mathsf{Add}}(\{\tilde{\beta}^+_{j_1..j_p}\}_{j_1,..,j_p\in[1,t]} \cup \tilde{\perp}) \\ \tilde{\gamma}_{-1} \leftarrow \mathsf{P}_{\mathsf{Add}}(\{\tilde{\beta}^-_{j_1..j_p}\}_{j_1,..,j_p\in[1,t]} \cup \tilde{0}) \end{cases} \tag{18}$$

1. Without loss of generality, we assume before the training phase, we have already converted the feature values of each sample to positive values by a standard procedure. For example, to achieve this we can add a positive constant to each feature value.

if $b < 0$. Here the server/user play the roles of Party B/Party A respectively, and $\tilde{\perp}$ denotes that the party B (i.e., the server) selects a random dummy message's encryption, which does not really be added to the final result of addition.

2.4 The server and user jointly execute

$$y \leftarrow \mathsf{P}_{\text{re-argmax}}(\widetilde{\gamma}_{+1}, \widetilde{\gamma}_{-1}), \tag{19}$$

where $\mathsf{P}_{\text{re-argmax}}$ is a reversed argmax protocol, which securely outputs the indicator $y \in \{+1, -1\}$ of the largest element to the private-key holder (in our case, the user obtains the final result). Here $\mathsf{P}_{\text{re-argmax}}$ can be easily derived from the reversed encrypted comparison protocol proposed by Bost et al. (Appendix A.2 of [10]).

6.2 Security and Efficiency

Correctness. We show that the PPSVM classifier outputs the correct prediction satisfying Equation (2).

From Equation (15), (16) and the multiplicative homomorphism, we have

$$\beta^{+}_{j_1,..,j_p} \simeq< \Big(\sum_{i \in S, y^{(i)}=1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p} \Big) \cdot (x_{j_1} \cdots x_{j_p}) >_{1,l_1} \tag{20}$$

and

$$\beta^{-}_{j_1,..,j_p} \simeq< \Big(\sum_{i \in S, y^{(i)}=-1} \alpha_i z^{(i)}_{j_1} \cdot z^{(i)}_{j_2} \cdots z^{(i)}_{j_p} \Big) \cdot (x_{j_1} \cdots x_{j_p}) >_{1,l_1} . \tag{21}$$

From Equation (13), (14), (17), (18) and the additive homomorphism, we have

$$\begin{cases} \widetilde{\gamma}_{+1} \simeq< \beta^{+} + |b| >_{2,l_2} \\ \widetilde{\gamma}_{-1} \simeq< \beta^{-} >_{2,l_2} \end{cases} \tag{22}$$

if $b > 0$, and

$$\begin{cases} \widetilde{\gamma}_{+1} \simeq< \beta^{+} >_{2,l_2} \\ \widetilde{\gamma}_{-1} \simeq< \beta^{-} + |b| >_{2,l_2} \end{cases} \tag{23}$$

if if $b < 0$.

By Equation (12) and the order-preserving property,

$$\begin{aligned} y \quad &= \mathsf{P}_{\text{re-argmax}}(\widetilde{\gamma}_{+1}, \widetilde{\gamma}_{-1}) = \mathsf{argmax}\{\gamma_{+1}, \gamma_{-1}\} \\ &= \text{sign}\Big(\sum_{i=1}^{m} \alpha_i y^{(i)} K(z^{(i)}, x) + b \Big) \end{aligned}$$

where $\gamma_{+1} = \mathsf{Dec}(\widetilde{\gamma}_{+1})$ and $\gamma_{-1} = \mathsf{Dec}(\widetilde{\gamma}_{-1})$. Thus, the output of the protocol is what we expected.

Security. Theorem 6 describes the privacy–preserving property of the protocol in the semi-honest model.

Theorem 6. The PPSVM protocol $\pi^{\mathsf{P}_{\text{Add}}, \mathsf{P}_{\text{re-argmax}}}$ securely computes $\mathcal{F}_{\text{PPSVM}}$ in the semi-honest model.

Proof 6. As in proving Theorem 5, we only need to show that $\pi^{\text{Add,re-argmax}}$ securely computes $\mathcal{F}_{\text{PPSVM}}$ in the semi-honest model.

Now we show that there exist two PPT simulators that can simulate the user and the server's view in executing $\pi^{\text{Add,re-argmax}}$.

Simulate the server's view. In the protocol's execution, the server's view is $\mathsf{View}_S = (pk; \{\alpha_i, z^{(i)}, y^{(i)}\}_{1 \le i \le m}, b; \{< x_j >_{1,l_1}\}_{j \in [1,t]}, \widetilde{\gamma}_{+1}, \widetilde{\gamma}_{-1})$. The simulator S_S, on input $(pk, \{\alpha_i, z^{(i)}, y^{(i)}\}_{1 \le i \le m}, b)$ performs the following:

(1). Randomly generate t type-1 ciphertexts: $\{e_j\}_{j \in [1,t]}$,
(2). Randomly generate 2 type-2 ciphertexts: ζ_+, ζ_-,
(3). Output $\mathsf{S\text{-}View}_S = (pk; \{\alpha_i, z^{(i)}, y^{(i)}\}_{1 \le i \le m}, b; \{e_j\}_{j \in [1,t]}, \zeta_+, \zeta_-)$.

Since the MASE cryptosystem is semantically secure, both distributions View_S and $\mathsf{S\text{-}View}_S$ are computationally indistinguishable.

Simulate the user's view. In the $(\mathsf{Add}, \mathsf{re\text{-}argmax})$-hybrid model where we view Add and re-argmax as ideal functionalities, the user's view is $\mathsf{View}_U = (pk, sk; x; y)$ where (pk, sk, x) is the input and y is the output. Thus, the simulator S_U for the user is trivially constructed in the sense that it just outputs $\mathsf{S\text{-}View}_S = (pk, sk; x; y)$.

Efficiency. The computation/communication cost of above MBR-PPNB protocol is summarized as follows. Since the classifier is a multiply-add structure, the following analysis is applicable to both MASE based and FHE-based paradigms. The notations t_e, t_m, t_a, l_e are defined in Section 5.2. Define $\xi = \frac{(p+t-1)!}{p!(t-1)!}$, which represents the number of different $\tilde{c}^{+}_{j_1..j_p}$ or $\tilde{c}^{-}_{j_1..j_p}$. Computational cost: $O(\xi \cdot (t_e + p \cdot t_m + t_a))$ for Server, and $O(t \cdot t_e + 2\xi \cdot t_a)$ for User. Communicational cost: $O(\xi \cdot l_e)$ for Server, and $O((t+2\xi) \cdot l_e)$ for User.

In addition, MASE-based paradigm has a constant round of communication. It has a total of 9 rounds, namely: 1 round in transmitting the user's data, 2 rounds in the P_{Add} protocol and 6 rounds in the re-argmax protocol. The communication round of the FHE-based PPSVM has two less rounds than the MASE based paradigm (the FHE needs no communications in the Add operation).

For Elmehdwi et al.'s AHE-based approach, the communication rounds is $12 + 2\lceil \lg p \rceil$, and the communication/computational complexity are $O(l_e \cdot (8p\xi + t))$ and $O(2\xi \cdot (t_e + p \cdot t_m + t_a) + t_e \cdot t)$ respectively.

7 Evaluation

In the following, we first give a comparison of the performance of the FHE and our MASE. Then we evaluate our MASE based MBR-PPNB and PPSVM protocol.

The evaluation is performed on a personal computer (PC) with a 4-core 2.5GHz Intel Core i5 CPU and 4GB RAM. The algorithms are implemented in C++ and compiled with g++ version 5.4.0 on a 64-bit version of Ubuntu (ubuntu-16.04-desktop-amd64). The program uses the GNU Multi-Precision (GMP) library [1] (version 6.1.1) and the OpenSSL Library [2] (version 1.0.2g).

Our protocol works well with the floating-point representation system of real numbers. In our implementation, IEEE 754 standard is used to store floating point values, which is a double-precision floating-point format. This representation is simple and convenient. Compared to the fix-point representation system, where a number must be represented as shared integers for multiplication [39], our approach is very efficient and achieves good accuracy. In addition, the modulus n of the Paillier cryptosystem is 2048 bits long. Thus in general, our implementation has security equivalent to integer factorization cryptography's 2048 bit-key security.

Schemes	t_e	t_m	t_a	l_e
MASE	11.2 ms	0.006 ms	30.1 ms	0.5KB
FHE [13]	856 ms	2573 ms	41 ms	32KB

TABLE 1
Comparison between our proposed MASE and FHE [13].

Schemes	Additive round	Multiplicative round	Ciphertext expansion	Encryption time
AHE [51] (parallel)	0	$2\lceil \lg k \rceil$	2	10.7 ms
FHE [13]	0	0	128	856 ms
MASE	2	0	2	11.2 ms

TABLE 2
Comparison of our proposed MASE with FHE and PHEs.

7.1 Comparison between MASE and FHE

As shown in Sections 5.2 and 6.2, the MASE based classifier and the FHE-based classifier have the same asymptotic complexity, except that they have different values of t_e, t_m, t_a, l_e and the FHE-based paradigm has two less communication rounds. In Table 1, we present the comparative summary for the performance of FHE (Cheon et al. [13] , ASIACRYPT'2017, which also supports real-valued plaintext) and the Paillier-based MASE using the same security level and setting (we take 2048 bit modulus for MASE and 2^{17} ring dimension for FHE). Findings indicate that MASE outperforms FHE-based solutions at the price of adding two communication rounds in $\mathsf{P}_{\mathsf{Add}}$.

7.2 Comparison between MASE and PHEs

Our proposed MASE finds good tradeoff between computation efficiency and communication interactions. In particular, it can significantly reduce communication rounds of batch addition/multiplication. For AHE, $2\lceil \lg k \rceil$ communication rounds are needed for a batch multiplication of k ciphertexts [51]. For MHE, to our knowledge there are no current methods to obtain additive homomorphism.

Our proposed MASE scheme optimizes this to only 2 communication rounds for batch addition, while no interactions are needed for multiplication. Table 2 in Section 7 clearly shows this advantage.

7.3 Evaluation of MBR-PPNB Protocol

We use two real world datasets from UCI Machine Learning Repository [15] to evaluate the performance of our protocol. The choice of these two medical datasets is due to the need for privacy protection in medical and healthcare industry. For the rest of this section, t, k_j and k respectively denote the number of features, the number of possible values of the j-th feature, and the number of classes of the Y-attribute (i.e. target attribute).

1) **Dataset 1: cardiotocography dataset.** In this dataset, $t = 21, k = 3$, and we reduce the real value feature type to the categorical type such that k_1 to k_{21} take the values of 9, 9, 9, 8, 8, 8, 8, 8, 9, 2, 5, 8, 8, 8, 9, 11, 9, 8, 8, 9, 3. The loss matrix is assigned to

$$L = \begin{pmatrix} 0 & 1 & 2 \\ 50 & 0 & 1 \\ 100 & 50 & 0 \end{pmatrix}.$$

2) **Dataset 2: lymphography dataset.** In this dataset, $t = 18, k = 4$, and k_1 to k_{18} take the values of 4, 2, 2, 2, 2, 2, 2, 2, 4, 4, 3, 4, 4, 8, 3, 2, 2, 8. The loss matrix is assigned to

$$L = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 30 & 0 & 1 & 2 \\ 70 & 30 & 0 & 1 \\ 100 & 50 & 30 & 0 \end{pmatrix}.$$

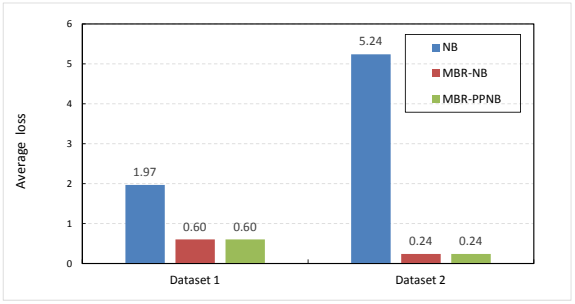


Fig. 4. Average loss comparison of the NB, MBR-NB and MBR-PPNB classifiers

Risk reduction. To show that the MBR-PPNB classifier reduces the Bayes risk, i.e., the expected misclassification loss, we compare the average loss of the three methods (i.e. NB, MBR-NB and MBR-PPNB). The comparative summary is presented in Figure 4. Here, the average loss is defined as

$$loss_{\mathrm{a}} = \frac{\sum_{i\in[1,m]} L_{j(i),s(i)}}{m},$$

where $j(i)$ and $s(i)$ represent the true category and the predicted category of the i-th testing sample respectively, and the dataset has m testing samples.

Classification accuracy. Findings from our evaluation show that the MBR-PPNB classifier has almost the same accuracy as the original MBR-NB classifier. First of all, the final outputs (i.e. predicted k^*) of the MBR-PPNB classifier and the MBR-NB classifier are the same for all samples in both datasets. In addition, in the following we will show that the intermediate results of the two methods are also consistent.

Let $\hat{\alpha}$ be the computed value from the MBR-PPNB classifier, and α be the corresponding value computed from the MBR-NB classifier. We define the relative error between these two methods as $e = \left|\frac{\hat{\alpha}}{\alpha} - 1\right|$. We also define the global maximum error as

$$e_{\mathrm{m}} = \max_{i\in[1,m], s\in[1,k]} \left| \frac{\widehat{LP}_s^{(i)}}{LP_s^{(i)}} - 1 \right|,$$

where $\widehat{LP}_s^{(i)}$ denotes the decrypted LP_s value of the i-th sample from the MBR-PPNB classifier and the dataset has m testing samples. Similarly, we define the global average error as

$$e_{\mathrm{a}} = \frac{\sum_{i\in[1,m], s\in[1,k]} \left| \frac{\widehat{LP}_s^{(i)}}{LP_s^{(i)}} - 1 \right|}{m \cdot k}.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2020.3009221, IEEE Transactions on Knowledge and Data Engineering

Dataset	T	$L_{\max}$	K	l_2
1	23	10	2	1816
2	20	10	3	1845

TABLE 3
Parameters for MBR-PPNB Protocol on two datasets.

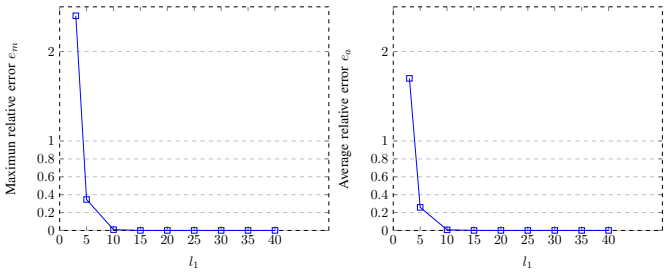


Fig. 5. Relative error between MBR-PPNB and MBR-NB on Dataset 1.

In order to ensure security and avoid overflow, we set l_2 to a fixed value $l_2 = \lfloor \tau - 1 - T \cdot L_{\max} - \lg K \rfloor$ that satisfies the constraints of Theorem 1 and Theorem 2. The parameters for the two datasets are summarized in Table 3. To determine the appropriate parameter l_1 that satisfies the accuracy requirement, we experiment with different values. The global maximum error e_{m} and the global average error e_{a} with different l_1 performed on Dataset 1 and Dataset 2 are respectively shown in Figures 5 and 6. Our experiments show that the parameter l_1 affects the accuracy of the multiplication. For both datasets, selecting $l_1 = 50$ has a global maximum error less than 10^{-6}; thus it satisfies the accuracy requirement.

Computational cost. We now evaluate the computational cost for the MBR-PPNB classifier. For a dataset with k classes of Y-attribute, the protocol has $8k - 5$ rounds of communications. For Dataset 1 (with $k = 3$), the MBR-PPNB classifier has 19 rounds of communications; and for Dataset 2 (with $k = 4$), the classifier has 27 rounds. The computation and communication overhead are summarized in Figure 7.

Comparison among different approaches for MBR-PPNB We compare the performance among different approaches for constructing MBR-PPNB. For AHE-based approach, Elmehdwi et al. [51], [16] introduced a method to obtain multiplicative homomorphism through an interactive approach on AHE, which can used to securely compute the multiply-add structure. As we stated before, FHE-based approach [27] is another alternative solution. To our knowledge, there is no MHE-based approach to solve this problem. The comparison is listed in Table 4, and it shows that our proposed MASE-based approach finds good tradeoff between computation efficiency and communication interactions.

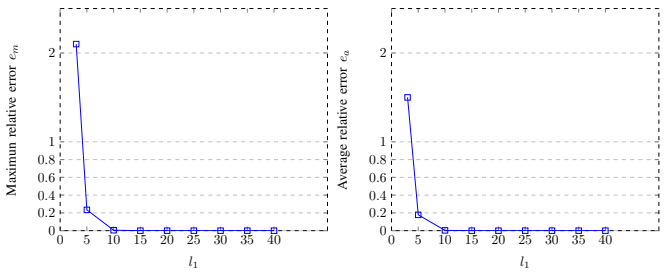
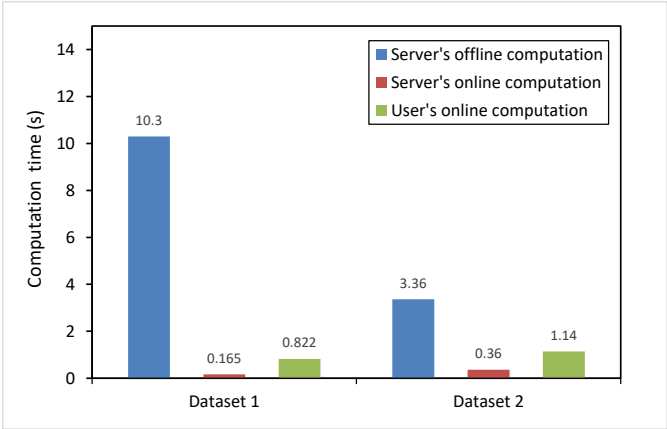


Fig. 6. Relative error between MBR-PPNB and MBR-NB on Dataset 2.



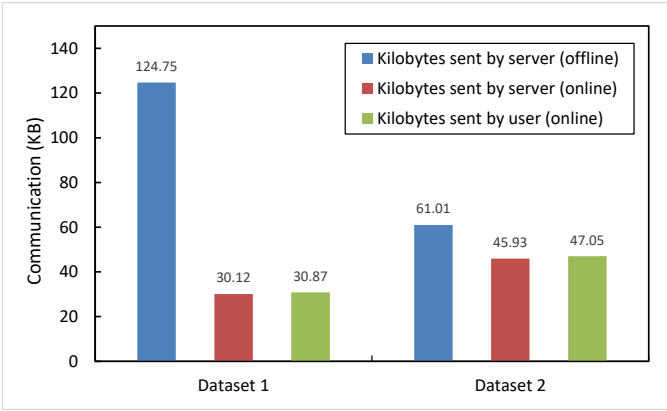


Fig. 7. Computational cost of the MBR-PPNB classifier for making a classification. Dataset 1 has $t = 21, k = 3, k_j = 3$ to 11 and Dataset 2 has $t = 18, k = 4, k_j = 2$ to 8.

7.4 Evaluation of PPSVM Protocol

We use a non-linearly separable dataset to evaluate the PPSVM protocol. The Iris dataset is a classic dataset in the pattern recognition literature [18]. We set $t = 2$, i.e., we only use two key features: sepal length and petal width. We also reduce it into a dataset with only two categories: the virginica iris and non-virginica iris.

To evaluate performance with different degree of the polynomial kernel, we take $p = 2, 4$ and 6. The values of parameters are shown in Table 5.

Classification accuracy. As with the NB classifier, our PPSVM classifier and the original SVM classifier have the same final output for all testing samples. The classification accuracy for $p = 1, 2, 4, 6$ is shown in Table 6. We can see that taking $p \geq 2$ can significantly improve the classification accuracy compared to $p = 1$ (the case of $p = 1$ does not require the use of MASE). In the following, we will show that the intermediate results of the two methods are also consistent.

MBR-PPNB Protocols	Dataset 1				Dataset 2			
	Communi-cation(KB)	Time(s)		Round	Communi-cation(KB)	Time(s)		Round
		Offline	Online			Offline	Online	
AHE-based	445.98	10.19	4.41	27	380.71	3.32	5.39	35
FHE-based	3118592.02	241.27	100.52	17	1707140.71	118.44	124.01	25
MASE-based	185.74	10.3	0.99	19	153.99	3.36	1.5	27

TABLE 4
Comparison among different approaches for MBR-PPNB

p	T	$L_{\max}$	K	l_2
2	3	10	5	2014
4	5	10	17	1992
6	7	10	65	1970

TABLE 5
Parameters for PPSVM Protocol with different p.

Phase	$p=1$	$p=2$	$p=4$	$p=6$
Training	79.17%	91.67%	94.17%	94.17%
Prediction	90%	96.67%	96.67%	96.67%

TABLE 6
SVM classification accuracy on Iris dataset.

The indicators $e_{\mathrm{m}}, e_{\mathrm{a}}$ (the global maximum/average error) are defined in Section 7.3. Their evaluations with different l_1 and p performed on the Iris dataset are respectively shown in Figures 8, 9 and 10. In general, selecting $l_1 = 40$ achieves a sufficient accuracy (global maximum error less than 10^{-4}).

Computational cost. The computation and communication overheads of PPSVM on the Iris dataset are summarized in Figure 11. One can observe that our protocol is very efficient. For example at $p = 6$, the protocol requires less than 0.3 seconds and 20 KB of communication to make a private prediction. In addition, the PPSVM protocol has a total of 9 rounds, regardless of the values of t and p.

Comparison among different approaches for PPSVM We compare the performance when AHE-based, AHE-based, and our MASE-based approaches are used to construct PPSVM. The comparative summary outlined in Table 7 shows that our MASE-based approach is efficient and its communication round does not increase with the value p.

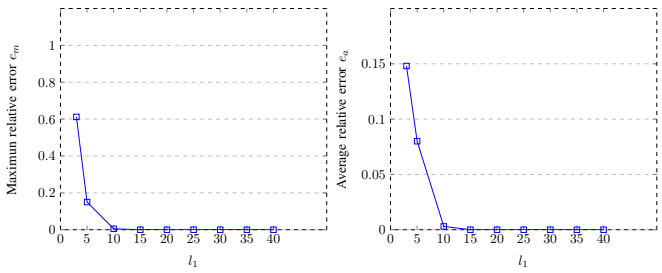


Fig. 8. Relative error between PPSVM and SVM on Iris ($p = 2$).

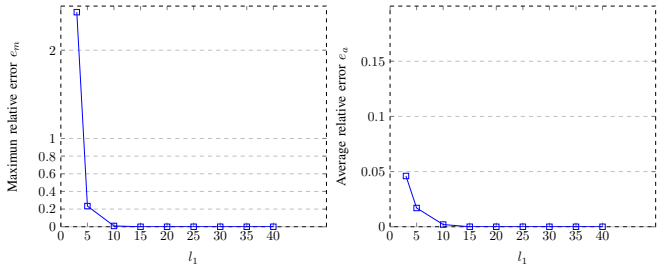


Fig. 9. Relative error between PPSVM and SVM on Iris ($p = 4$).

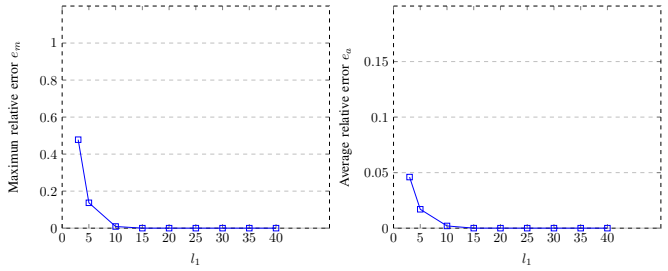


Fig. 10. Relative error between PPSVM and SVM on Iris ($p = 6$).

8 Conclusion

Privacy will be increasingly important, as our society becomes more networked and data about individuals are increasingly digitalized.

In this paper, we presented a new cryptographic tool, MAS-Encryption (MASE), which is designed to allow one to securely compute a multiply-add structure for comparative purposes. Such a tool has several potential applications, for example to protect the privacy of classifiers with multiply-add structures. To demonstrate the utility of MASE, we use two examples, namely: to construct a privacy-preserving NB classifier with minimal Bayes risk (i.e. MBR-PPNB classifier) and to construct a privacy-preserving SVM classifier (i.e., PPSVM classifier). In order to be practical for real-world applications, our constructions do not rely on the FHE. We also proved the security of the two private classifiers, as well as demonstrating that both classifiers are very efficient.

Acknowledgements: This work is supported by the Natural Science Foundation of China (U1936116, 61772148, 61802078), and the Guangxi Key Laboratory of Cryptography and Information Security (GCIS201807).

References

[1] The GNU Multiple Precision Arithmetic Library. https://gmplib.org/.

PPSVM Protocols	p=2				p=4				p=6			
	Communi-cation(KB)	Time(s)		Round	Communi-cation(KB)	Time(s)		Round	Communi-cation(KB)	Time(s)		Round
		Offline	Online			Offline	Online			Offline	Online	
AHE-based	48.74	0.05	0.68	14	80.74	0.13	1.70	16	128.74	0.20	3.29	18
FHE-based	5115.53	0.33	1.92	7	10155.53	0.90	10.80	7	15195.53	1.90	36.82	7
MASE-based	32.99	0.05	0.31	9	34.98	0.13	0.46	9	36.98	0.20	0.60	9

TABLE 7
Comparison among different approaches for PPSVM

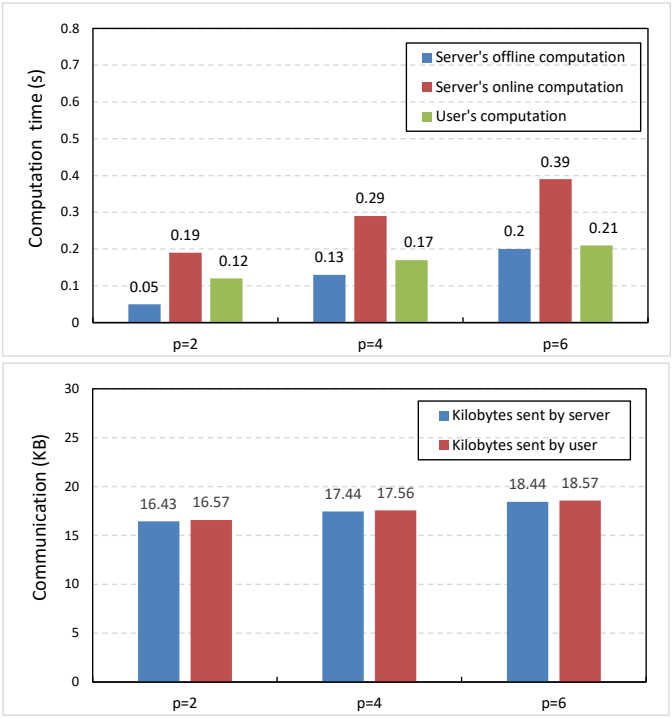


Fig. 11. Computational cost of the PPSVM classifier for making a classification on Iris dataset.

[2] OpenSSL: Cryptography and SSL/TLS Toolkit. https://www.openssl.org/.

[3] L. J. M. Aslett, P. M. Esperança, and C. C. Holmes. Encrypted statistical machine learning: new privacy preserving methods. *CoRR*, abs/1508.06845, 2015.

[4] M. Ball, T. Malkin, and M. Rosulek. Garbling gadgets for boolean and arithmetic circuits. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 565–577, New York, NY, USA, 2016. ACM.

[5] M. Barni, P. Failla, V. Kolesnikov, R. Lazzeretti, A.-R. Sadeghi, and T. Schneider. Secure evaluation of private linear branching programs with medical applications. In *Proceedings of the 14th European Conference on Research in Computer Security*, ESORICS'09, pages 424–439, Berlin, Heidelberg, 2009. Springer-Verlag.

[6] M. Barni, P. Failla, R. Lazzeretti, A. Paus, A. Sadeghi, T. Schneider, and V. Kolesnikov. Efficient privacy-preserving classification of ECG signals. In *First IEEE International Workshop on Information Forensics and Security, WIFS 2009, London, UK, December 6-9, 2009*, pages 91–95. IEEE, 2009.

[7] S. Bauer, L.-P. Nolte, and M. Reyes. Fully automatic segmentation of brain tumor images using support vector machine classification in combination with hierarchical conditional random field regularization. In G. Fichtinger, A. Martel, and T. Peters, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011*, pages 354–361, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[8] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. *Lecture Notes in Computer Science*, 1462:26–46, 1998.

[9] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10, New York, NY, USA, 1988. ACM.

[10] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser. Machine learning classification over encrypted data. In *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*. The Internet Society, 2015.

[11] J. Brickell and V. Shmatikov. *Privacy-Preserving Classifier Learning*, pages 128–147. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[12] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, Jan 2000.

[13] J. H. Cheon, A. Kim, M. Kim, and Y. Song. Homomorphic encryption for arithmetic of approximate numbers. In T. Takagi and T. Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 409–437, Cham, 2017. Springer International Publishing.

[14] B. David, R. Dowsley, R. Katti, and A. C. Nascimento. Efficient unconditionally secure comparison and privacy preserving machine learning classification protocols. In *Proceedings of the 9th International Conference on Provable Security - Volume 9451*, ProvSec 2015, pages 354–367, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

[15] D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017.

[16] Y. Elmehdwi, B. K. Samanthula, and W. Jiang. Secure k-nearest neighbor query over encrypted data in outsourced environments. In *2014 IEEE 30th International Conference on Data Engineering*, pages 664–675, March 2014.

[17] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, PETS '09, pages 235–253, Berlin, Heidelberg, 2009. Springer-Verlag.

[18] R. A. FISHER. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188.

[19] C.-z. Gao, Q. Cheng, P. He, W. Susilo, and J. Li. Privacy-preserving naive bayes classifiers secure against the substitution-then-comparison attack. *Information Sciences*, 444:72 – 88, 2018.

[20] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York, NY, USA, 2004.

[21] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual Symposium on Theory of Computing (STOC)*, pages 218–229, New York, NY USA, May 1987. ACM Press.

[22] F.-J. Gonzlez-Serrano, ngel Navia-Vzquez, and A. Amor-Martn. Training support vector machines with privacy-protected data. *Pattern Recognition*, 72:93 – 107, 2017.

[23] G. Hu, D. Xiao, T. Xiang, S. Bai, and Y. Zhang. A compressive sensing based privacy preserving outsourcing of image storage and identity authentication service in cloud. *Information Sciences*, 387:132–145, 2017.

[24] M. Kantarcıoglu, J. Vaidya, and C. Clifton. Privacy preserving naive bayes classifier for horizontally partitioned data. In *IEEE ICDM workshop on privacy preserving data mining*, pages 3–9, 2003.

[25] M. Kesarwani, A. Kaul, P. Naldurg, S. Patranabis, G. Singh, S. Mehta, and D. Mukhopadhyay. Efficient secure k-nearest neighbours over encrypted data. In *Proceedings of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018*, pages 564–575. OpenProceedings.org, 2018.

[26] A. Khedr, P. G. Gulak, and V. Vaikuntanathan. SHIELD: scalable homomorphic implementation of encrypted data-classifiers. *IEEE Trans. Computers*, 65(9):2848–2858, 2016.

[27] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang. Secure logistic regression based on homomorphic encryption: Design and evaluation. *JMIR Med Inform*, 6(2):e19, Apr 2018.

[28] V. Kolesnikov and A. Shikfa. On the limits of privacy provided by order-preserving encryption. *Bell Labs Technical Journal*, 17(3):135–146, Dec 2012.

[29] X. Li, Y. Zhu, J. Wang, Z. Liu, Y. Liu, and M. Zhang. On the soundness and security of privacy-preserving svm for outsourcing data classification. *IEEE Transactions on Dependable and Secure Computing*, 15(5):906–912, Sept 2018.

[30] Y. Li, C. Bai, and C. K. Reddy. A distributed ensemble approach for mining healthcare data under privacy constraints. *Information Sciences*, 330:245 – 259, 2016.

[31] K.-P. Lin and M.-S. Chen. Privacy-preserving outsourcing support vector machines with random transformation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 363–372, New York, NY, USA, 2010. ACM.

[32] Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. Cryptology ePrint Archive, Report 2008/197, 2008.

[33] X. Liu, K. R. Choo, R. H. Deng, R. Lu, and J. Weng. Efficient and privacy-preserving outsourced calculation of rational numbers. *IEEE Transactions on Dependable and Secure Computing*, 15(1):27–39, Jan 2018.

[34] X. Liu, R. Deng, K. R. Choo, and Y. Yang. Privacy-preserving outsourced support vector machine design for secure drug discovery. *IEEE Transactions on Cloud Computing*, pages 1–1, 2018.

[35] X. Liu, R. Lu, J. Ma, L. Chen, and B. Qin. Privacy-preserving patient-centric clinical decision support system on naïve bayesian classification. *IEEE J. Biomedical and Health Informatics*, 20(2):655–668, 2016.

[36] M. R. Mendoza and A. L. C. Bazzan. Social choice in distributed classification tasks: Dealing with vertically partitioned data. *Information Sciences*, 332:56 – 71, 2016.

[37] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, 2000.

[38] P. Mohassel and P. Rindal. Aby3: A mixed protocol framework for machine learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 35–52, New York, NY, USA, 2018. ACM.

[39] P. Mohassel and Y. Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 19–38, May 2017.

[40] J. Nahar, Y.-P. P. Chen, and S. Ali. Kernel-based naive bayes classifier for breast cancer prediction. *Journal of Biological Systems*, 15(01):17–25, 2007.

[41] G. Orrù, W. Pettersson-Yeo, A. F. Marquand, G. Sartori, and A. Mechelli. Using support vector machine to identify imaging biomarkers of neurological and psychiatric disease: A critical review. *Neuroscience & Biobehavioral Reviews*, 36(4):1140 – 1152, 2012.

[42] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[43] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, May 2018.

[44] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams. Secure two-party computation is practical. *IACR Cryptology ePrint Archive*, 2009:314, 2009.

[45] P. Pullonen and S. Siim. Combining secret sharing and garbled circuits for efficient private ieee 754 floating-point computations. In M. Brenner, N. Christin, B. Johnson, and K. Rohloff, editors, *Financial Cryptography and Data Security*, pages 172–183, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[46] Y. Rahulamathavan, R. C. . Phan, S. Veluru, K. Cumanan, and M. Rajarajan. Privacy-preserving multi-class support vector machine for outsourcing the data classification in cloud. *IEEE Transactions on Dependable and Secure Computing*, 11(5):467–479, Sept 2014.

[47] Y. Rahulamathavan, S. Veluru, R. C. . Phan, J. A. Chambers, and M. Rajarajan. Privacy-preserving clinical decision support system using gaussian kernel-based classification. *IEEE Journal of Biomedical and Health Informatics*, 18(1):56–66, Jan 2014.

[48] O. Regnier-Coudert and J. McCall. Privacy-preserving approach to bayesian network structure learning from distributed data. In *Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '11, pages 815–816, New York, NY, USA, 2011. ACM.

[49] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient privacy-preserving face recognition. In *Proceedings of the 12th International Conference on Information Security and Cryptology*, ICISC'09, pages 229–244, Berlin, Heidelberg, 2010. Springer-Verlag.

[50] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin, 1998. AAAI Technical Report WS-98-05.

[51] B. K. Samanthula, Y. Elmehdwi, and W. Jiang. k-nearest neighbor classification over semantically secure encrypted relational data. *IEEE Transactions on Knowledge and Data Engineering*, 27(5):1261–1273, May 2015.

[52] E. D. Sans, R. Gay, and D. Pointcheval. Reading in the dark: Classifying encrypted digits with functional encryption. Cryptology ePrint Archive, Report 2018/206, 2018. https://eprint.iacr.org/2018/206.

[53] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1310–1321, New York, NY, USA, 2015. ACM.

[54] L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik. Identifying missed monitoring alerts based on unstructured incident tickets. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, pages 143–146, Oct 2013.

[55] L. Tang, T. Li, L. Shwartz, F. Pinel, and G. Y. Grabarnik. An integrated framework for optimizing automatic monitoring systems in large it infrastructures. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1249–1257, New York, NY, USA, 2013. ACM.

[56] J. Vaidya, M. Kantarcioglu, and C. Clifton. Privacy-preserving naive bayes classification. *The VLDB Journal*, 17(4):879–898, July 2008.

[57] J. Vaidya, B. Shafiq, A. Basu, and Y. Hong. Differentially private naive bayes classification. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 1, pages 571–576, Nov 2013.

[58] J. Vaidya, B. Shafiq, W. Fan, D. Mehmood, and D. Lorenzi. A random decision tree framework for privacy-preserving data mining. *IEEE Transactions on Dependable and Secure Computing*, 11(5):399–411, Sept 2014.

[59] J. Vaidya, H. Yu, and X. Jiang. Privacy-preserving svm classification. *Knowledge and Information Systems*, 14(2):161–178, Feb 2008.

[60] R. Wright and Z. Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 713–718, New York, NY, USA, 2004. ACM.

[61] A. C. Yao. How to generate and exchange secrets. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 162–167. IEEE Computer Society Press, 1986.

[62] H. Yu, X. Jiang, and J. Vaidya. Privacy-preserving svm using nonlinear kernels on horizontally partitioned data. In *Proceedings of the 2006 ACM Symposium on Applied Computing*, SAC '06, pages 603–610, New York, NY, USA, 2006. ACM.

[63] H. Yu, J. Vaidya, and X. Jiang. Privacy-preserving svm classification on vertically partitioned data. In W.-K. Ng, M. Kitsuregawa, J. Li, and K. Chang, editors, *Advances in Knowledge Discovery and Data Mining*, pages 647–656, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[64] Y. Zhang, X. Chen, J. Li, D. S. Wong, H. Li, and I. You. Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing. *Information Sciences*, 379:42–61, 2017.

[65] H. Zhu, X. Liu, R. Lu, and H. Li. Efficient and privacy-preserving online medical prediagnosis framework using nonlinear svm. *IEEE Journal of Biomedical and Health Informatics*, 21(3):838–850, May 2017.