JMLR: Workshop and Conference Proceedings 63:126–141, 2016 ACML 2016

Secure Approximation Guarantee for Cryptographically Private Empirical Risk Minimization

Toshiyuki Takada TAKADA.T.MLLAB.NIT@GMAIL.COM
Hiroyuki Hanada HANADA.HIROYUKI@NITECH.AC.JP
Nagoya Institute of Technology
Nagoya, Aichi, Japan

Yoshiji Yamada YAMADA@GENE.MIE-U.AC.JP
Mie University
Tsu, Mie, Japan

Jun Sakuma JUN@CS.TSUKUBA.AC.JP
University of Tsukuba
Tsukuba, Ibaraki, Japan

Ichiro Takeuchi TAKEUCHI.ICHIRO@NITECH.AC.JP
Nagoya Institute of Technology
Nagoya, Aichi, Japan

Editors: Robert J. Durrant and Kee-Eung Kim

Abstract

Privacy concern has been increasingly important in many machine learning (ML) problems. We study empirical risk minimization (ERM) problems under secure multi-party computation (MPC) frameworks. Main technical tools for MPC have been developed based on cryptography. One of limitations in current cryptographically private ML is that it is computationally intractable to evaluate non-linear functions such as logarithmic functions or exponential functions. Therefore, for a class of ERM problems such as logistic regression in which non-linear function evaluations are required, one can only obtain approximate solutions. In this paper, we introduce a novel cryptographically private tool called *secure approximation guarantee (SAG)* method. The key property of SAG method is that, given an arbitrary approximate solution, it can provide a non-probabilistic assumption-free bound on the approximation quality under cryptographically secure computation framework. We demonstrate the benefit of the SAG method by applying it to several problems including a practical privacy-preserving data analysis task on genomic and clinical information.

1. Introduction

Privacy preservation has been increasingly important in many machine learning (ML) tasks. In this paper, we consider empirical risk minimizations (ERMs) when the data is distributed among multiple parties, and these parties are unwilling to share their data to other parties. For example, if two parties have different sets of features for the same group of people, they might want to combine these two datasets for more accurate predictive model building. On the other hand, due to privacy concerns or legal regulations, these two parties might want to keep their own data private. The problem of learning from multiple confidential databases have been studied under the name of *secure multi-party computation (secure MPC)*.

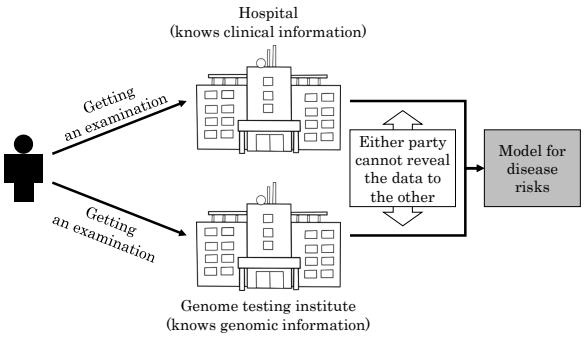
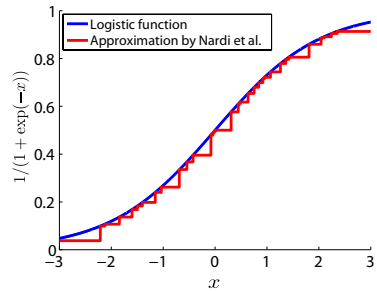


Figure 1: Our multi-party computation study for disease risk prediction based on genomic and clinical information

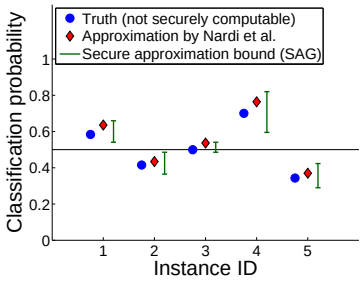
This paper is motivated by our recent secure MPC project on genomic and clinical data (Figure 1). Our task is to develop a model for predicting the risk of a disease based on genomic and clinical information of potential patients. The difficulty of this problem is that genomic information were collected in a research institute, while clinical information were collected in a hospital, and both institutes do not want to share their data to others. However, since the risk of the disease is dependent both on genomic and clinical features, it is quite valuable to use both types of information for the risk modeling.

Various tools for secure MPC have been taken from cryptography, and privacy-preserving ML approaches based on cryptographic techniques have been called *cryptographically private ML*. A key building block of cryptographically private ML is *homomorphic encryption* by which sum or product of two encrypted values can be evaluated without decryption. Many cryptographically private ML algorithms have been developed, e.g., for linear regression (Hall et al., 2011; Nikolaenko et al., 2013) and SVM (Laur et al., 2006; Yu et al., 2006) by using homomorphic encryption property. One of limitations in current cryptographically private ML is that it is computationally intractable to evaluate non-linear functions such as logarithmic functions or exponential functions in homomorphic encryption framework. Since non-linear function evaluations are required in many fundamental statistical analyses such as logistic regression, it is crucially important to develop a method that can alleviate this computational bottleneck. One way to circumvent this issue is to *approximate* non-linear functions. For example, in Nardi et al.'s work (Nardi et al., 2012) for secure logistic regression, the authors proposed to approximate a logistic function by sum of step functions, which can be computed under secure computation framework.

Due to the very nature of MPC, even after the final solution is obtained, the users are not allowed to access to private data. When the resulting solution is an approximation, it is important for the users to be able to check its approximation quality. Unfortunately, most existing cryptographically private ML method does not have such an approximation guarantee mechanism. Although a probabilistic approximation guarantee was provided in the aforementioned secure logistic regression study (Nardi et al., 2012), the approximation bound derived in that work depends on the unknown true solution, meaning that the users cannot make sure how much they can trust the approximate solution.



(A) A non-linear function $1/(1+\exp(-x))$ and its approximation with (Nardi et al., 2012)



(B) Class probabilities by true and approximate solutions and the bounds obtained by the SAG method.

Figure 2: An illustration of the proposed SAG method in a simple logistic regression example. The left plot (A) shows the logistic function (blue) and its approximation (red) proposed in (Nardi et al., 2012). The right plot (B) shows the true (blue) and approximate (red) class probabilities of five training instances (the instance IDs $1, \ldots, 5$ are shown in the horizontal axis), where the former is obtained with true logistic function, while the latter is obtained with the approximate logistic function. The green intervals in plot (B) are the approximation guarantee intervals provided by the SAG method. The key property of the SAG method is that these intervals are guaranteed to contain the true class probabilities.

The goal of this paper is to develop a practical method for secure computations of ERM problems. To this end, we introduce a novel secure computation technique called *secure approximation guarantee (SAG)* method. Given an arbitrary approximate solution of an ERM problem, the SAG method provides non-probabilistic assumption-free bounds on how far the approximate solution is away from the true solution. A key difference of our approach with existing ones is that our approximation bound is not for theoretical justification of an approximation algorithm itself, but for practical decision making based on a given approximate solution. Our approximation bound can be obtained without any information about the true solution, and it can be computed with a reasonable computational cost under secure computation framework, i.e., without the risk of disclosing private information.

In order to develop the SAG method, we introduce two novel technical contributions in this paper. We first introduce a novel algorithmic framework for computing approximation guarantee that can be applied to a class of ERM problems whose loss function is non-linear and its secure evaluation is difficult. In this framework, we use a pair of surrogate loss functions that bounds the non-linear loss function from below and above. Our second contribution is to implement these surrogate loss functions by piecewise-linear functions, and show that they can be cryptographically securely computed. Furthermore, we empirically demonstrate that the bounds obtained by the SAG method are much tighter than the bounds in (Nardi et al., 2012) despite the former is non-probabilistic and assumption-free. Figure 2 is an illustration of the SAG method in a simple logistic regression example.

Notations We use the following notations in the rest of the paper. We denote the sets of real numbers and integers as $\mathbb{R}$ and $\mathbb{Z}$, respectively. For a natural number N, we define $[N] := \{1, 2, \ldots, N\}$ and $\mathbb{Z}_N := \{0, 1, \ldots, N-1\}$. The Euclidean norm is written as $\|\cdot\|$. Indicator function is written as I_χ i.e., $I_\chi = 1$ if χ is true, and $I_\chi = 0$ otherwise. For a protocol Π between two parties, we use the notation $\Pi(\mathcal{I}_A, \mathcal{I}_B) \to (\mathcal{O}_A, \mathcal{O}_B)$, where $\mathcal{I}_A$ and $\mathcal{I}_B$ are inputs from the parties A and B, respectively, and $\mathcal{O}_A$ and $\mathcal{O}_B$ are outputs given to A and B, respectively.

2. Preliminaries

2.1. Problem statement

Empirical risk minimization (ERM) Let $\{(\boldsymbol{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i\in[n]}$ be the training set, where the input domain $\mathcal{X} \subset \mathbb{R}^d$ is a compact region in $\mathbb{R}^d$, and the output domain $\mathcal{Y}$ is $\{-1, +1\}$ in classification problems and $\mathbb{R}$ in regression problems. In this paper, we consider the following class of empirical risk minimization problems:

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \ \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i\in[n]} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}), \tag{1}$$

where ℓ is a loss function subdifferentiable and convex with respect to $\boldsymbol{w}$, and $\lambda > 0$ is the regularization parameter. L_2 regularization in (1) ensures that the solution $\boldsymbol{w}$ is within a compact region $\mathcal{W} \subset \mathbb{R}^d$.

We consider the cases where ℓ is hard to compute in secure computation framework, i.e., ℓ includes non-linear functions such as $\log$ and $\exp$. Popular examples includes logistic regression $\ell(y, \boldsymbol{x}^\top\boldsymbol{w}) := \log(1 + \exp(-\boldsymbol{x}^\top\boldsymbol{w})) - y\boldsymbol{x}^\top\boldsymbol{w}$, Poisson regression $\ell(y, \boldsymbol{x}^\top\boldsymbol{w}) := \exp(\boldsymbol{x}^\top\boldsymbol{w}) - y\boldsymbol{x}^\top\boldsymbol{w}$, and exponential regression $\ell(y, \boldsymbol{x}^\top\boldsymbol{w}) := (y\exp(-\boldsymbol{x}^\top\boldsymbol{w})) - \boldsymbol{x}^\top\boldsymbol{w}$.

Secure two-party computation We consider secure two-party computation scenario where the training set $\{(\boldsymbol{x}_i, y_i)\}_{i\in[n]}$ is *vertically-partitioned* between two parties A and B (Vaidya and Clifton, 2003), i.e., A and B own different sets of features for common set of n instances. More precisely, let party A own the first d_A features and party B own the last d_B features, i.e., $d_A + d_B = d$. We consider a scenario where the labels $\{y_i\}_{i\in[n]}$ are also owned by either party, and we let party B own them here. We assume that both parties can identify the instance index $i \in [n]$, i.e., it is possible for both parties to make communications with respect to a specified instance. We denote the input data matrix owned by parties A and B as X_A and X_B, respectively. Furthermore, we denote the n-dimensional vector of the labels as $\boldsymbol{y} := [y_1, \ldots, y_n]^\top$.

Semi-honest model In this paper, we develop the SAG method so that it is secure (meaning that private data is not revealed to the other party) under the *semi-honest* model (Goldreich, 2001). In this security model, any parties are allowed to guess other party's data as long as they follow the specified protocol. In other words, we assume that any of the parties do not modify the specified protocol. The semi-honest model is a standard security model in cryptographically private ML.

2.2. Cryptographically Secure Computation

Paillier cryptosystem For secure computations, we use *Paillier cryptosystem* (Paillier, 1999) as an additive *homomorphic encryption* tool, i.e., we can obtain $E(a+b)$ from $E(a)$ and $E(b)$ without

decryption, where a and b are plaintexts and $E(\cdot)$ is the encryption function. Paillier cryptosystem has the *semantic security* (Goldreich, 2004) (the *IND-CPA security*), which roughly means that it is difficult to judge whether $a = b$ or $a \neq b$ by knowing $E(a)$ and $E(b)$.

Paillier cryptosystem is a public key cryptosystem with additive homomorphism over $\mathbb{Z}_N$ (i.e., $\mathrm{mod} N$). In this cryptosystem, the private key is two large prime numbers p and q, and the public key is $(N, g) \in \mathbb{Z} \times \mathbb{Z}_{N^2}$, where $N = pq$ and g is an integer co-prime with N^2. Given a plaintext $m \in \mathbb{Z}_N$, a ciphertext of $E(m)$ is obtained with a random integer $R \in \mathbb{Z}_N$ as $E(m) = g^m R^N \mod N^2$. Ciphertext $E(m)$ is decrypted with the private key whatever R is chosen. With the encryption, the additive homomorphism $E(a) \cdot E(b) = E(a+b)$ and $E(a)^b = E(ab)$ holds for any plaintexts $a, b \in \mathbb{Z}_N$. Hereafter, we denote by $E_{pk_A}(\cdot)$ and $E_{pk_B}(\cdot)$ the encryption functions with the public keys issued by party A and B, respectively.

Note that we need computations of real numbers rather than integers in data analysis tasks. First, negative numbers can be treated with the similar technique to the two's complement. In order to handle real numbers, we multiply a magnification constant M for each input real number for expressing it with an integer. Here, there is a tradeoff between the accuracy and range of acceptable real number, i.e., for large M, accuracy would be high, but only possible to handle a limited range of real numbers.

2.3. Related works

The most general framework for cryptographically private ML is the Yao's garbled circuit (Yao, 1986), where any desired secure computation is expressed as an electronic circuit with encrypted components. In principle, Yao's garbled circuit can evaluate any function securely, but its computational costs are usually extremely large. Unfortunately, it is impractical to securely compute the ERM problem with only the garbled circuit.

Nardi et al. (Nardi et al., 2012) studied cryptographically private approach for logistic regression. As briefly mentioned in §1, in order to circumvent the difficulty of secure non-linear function evaluations, the authors proposed to approximate logistic function by empirical cumulative density function (CDF) of logistic distributions (see Figure 2(A) as an example). Denoting the true solution and the approximate solution as $\boldsymbol{w}^*$ and $\hat{\boldsymbol{w}}$, respectively, the authors showed that the difference $\|\boldsymbol{w}^* - \hat{\boldsymbol{w}}\|$ is no greater than $\frac{nc_1 \max \|\boldsymbol{x}_i\|}{L^\gamma \lambda_{\min}}$ with probability greater than $1 - 2\exp(-cL^{1-2\gamma})$, where L is the sample size for the empirical CDF, $\lambda_{\min}$ is the smallest eigenvalue of Fisher information matrix, and $c > 0$, $c_1 > 0$, $\gamma \in (0, 1/2)$ are constants. This approximation error bound cannot be used for knowing the approximation quality of the given approximate solution $\hat{\boldsymbol{w}}$ because the bound depends on the unknown true solution $\boldsymbol{w}^*$ [1]. Furthermore, in experiment section, we demonstrate that the SAG method can provide much tighter non-probabilistic bounds than the above probabilistic bound in (Nardi et al., 2012).

3. Secure Approximation Guarantee(SAG)

The basic idea behind the SAG method is to introduce two surrogate loss functions ϕ and ψ that bound the target non-linear loss function ℓ from below and above. In what follows, we show that, given an arbitrary approximate solution $\hat{\boldsymbol{w}}$, if we can securely evaluate $\phi(\hat{\boldsymbol{w}})$, $\psi(\hat{\boldsymbol{w}})$ and a subgra-

1. The minimum eigenvalue λ_1 depends on $\boldsymbol{w}^*$.

dient $\partial\phi/\partial\boldsymbol{w}\mid_{\boldsymbol{w}=\hat{\boldsymbol{w}}}$, we can securely compute bounds on the true solution $\boldsymbol{w}^*$ which itself cannot be computed under secure computation framework.

First, the following theorem states that we can obtain a ball in the solution space in which the true solution $\boldsymbol{w}^*$ certainly exists.

Theorem 1. *Let $\phi : \mathbb{R} \to \mathbb{R}$ and $\psi : \mathbb{R} \to \mathbb{R}$ be functions that satisfy $\phi(y, \boldsymbol{x}^\top \boldsymbol{w}) \leq \ell(y, \boldsymbol{x}^\top \boldsymbol{w}) \leq \psi(y, \boldsymbol{x}^\top \boldsymbol{w})\ \forall y \in \mathcal{Y}, \boldsymbol{x} \in \mathcal{X}, \boldsymbol{w} \in \mathcal{W}$, and assume that they are convex and subdifferentiable with respect to $\boldsymbol{w}$. Then, for any $\hat{\boldsymbol{w}} \in \mathcal{W}$,*

$$\|\boldsymbol{w}^* - \boldsymbol{m}(\hat{\boldsymbol{w}})\| \leq r(\hat{\boldsymbol{w}}),$$

i.e., the true solution $\boldsymbol{w}^*$ is located within a ball in $\mathcal{W}$ with the center $\boldsymbol{m}(\hat{\boldsymbol{w}}) := \frac{1}{2}\left(\hat{\boldsymbol{w}} - \frac{1}{\lambda}\nabla\Phi(\hat{\boldsymbol{w}})\right)$, and the radius $r(\hat{\boldsymbol{w}}) := \sqrt{\left\|\frac{1}{2}\left(\hat{\boldsymbol{w}} + \frac{1}{\lambda}\nabla\Phi(\hat{\boldsymbol{w}})\right)\right\|^2 + \frac{1}{\lambda}\left(\Psi(\hat{\boldsymbol{w}}) - \Phi(\hat{\boldsymbol{w}})\right)}$, where $\Phi(\hat{\boldsymbol{w}}) := \frac{1}{n}\sum_{i\in[n]} \phi(y_i, \boldsymbol{x}_i^\top \hat{\boldsymbol{w}})$, $\Psi(\hat{\boldsymbol{w}}) := \frac{1}{n}\sum_{i\in[n]} \psi(y_i, \boldsymbol{x}_i^\top \hat{\boldsymbol{w}})$ and $\nabla\Phi(\hat{\boldsymbol{w}})$ is a subgradient of Φ at $\boldsymbol{w} = \hat{\boldsymbol{w}}$.

The proof of Theorem 1 is presented in Appendix A.

Using Theorem 1, we can compute a pair of lower and upper bounds of any linear score in the form of $\boldsymbol{\eta}^\top \boldsymbol{w}^*$ for an arbitrary $\boldsymbol{\eta} \in \mathbb{R}^d$ as the following Corollary states.

Corollary 2. *For an arbitrary $\boldsymbol{\eta} \in \mathbb{R}^d$,*

$$LB(\boldsymbol{\eta}^\top \boldsymbol{w}^*) \leq \boldsymbol{\eta}^\top \boldsymbol{w}^* \leq UB(\boldsymbol{\eta}^\top \boldsymbol{w}^*), \tag{2}$$

where

$$LB(\boldsymbol{\eta}^\top \boldsymbol{w}^*) := \boldsymbol{\eta}^\top \boldsymbol{m}(\hat{\boldsymbol{w}}) - \|\boldsymbol{\eta}\| r(\hat{\boldsymbol{w}}) \tag{3a}$$

$$UB(\boldsymbol{\eta}^\top \boldsymbol{w}^*) := \boldsymbol{\eta}^\top \boldsymbol{m}(\hat{\boldsymbol{w}}) + \|\boldsymbol{\eta}\| r(\hat{\boldsymbol{w}}). \tag{3b}$$

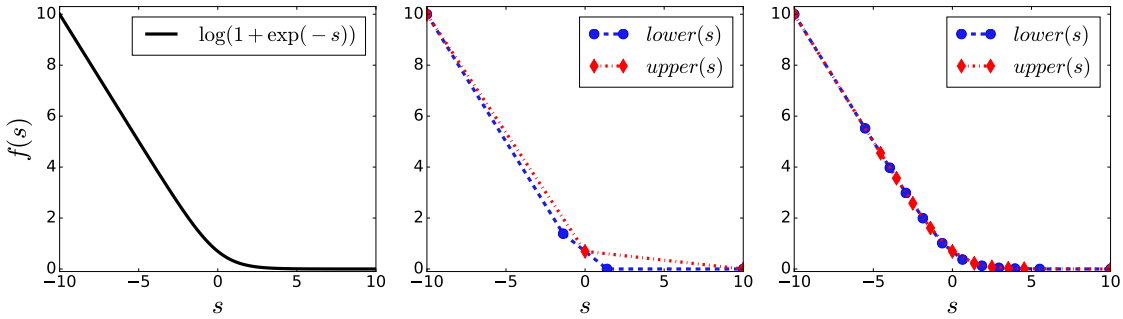
The proof of Corollary 2 is presented in Appendix A.

Many important quantities in data analyses are represented as a linear score. For example, in binary classification, the classification result $\tilde{y}$ of a test input $\tilde{\boldsymbol{x}}$ is determined by the sign of the linear score $\tilde{\boldsymbol{x}}^\top \boldsymbol{w}^*$. It suggests that we can certainly classify the test instance as $LB(\tilde{\boldsymbol{x}}^\top \boldsymbol{w}^*) > 0 \Rightarrow \tilde{y} = +1$ and $UB(\tilde{\boldsymbol{x}}^\top \boldsymbol{w}^*) < 0 \Rightarrow \tilde{y} = -1$. Similarly, if we are interested in each coefficient $w_h^*, h \in [d]$, of the trained model, by setting $\boldsymbol{\eta} = \boldsymbol{e}_h$ where $\boldsymbol{e}_h$ is a d-dimensional vector of all 1s except 0 in the h-th component, we can obtain a pair of lower and upper bounds on the coefficient as $LB(\boldsymbol{e}_h^\top \boldsymbol{w}^*) \leq w_h^* \leq UB(\boldsymbol{e}_h^\top \boldsymbol{w}^*)$.

We note that Theorem 1 and Corollary 2 are inspired by recent works on safe screening and related problems (El Ghaoui et al., 2012; Xiang et al., 2011; Ogawa et al., 2013; Liu et al., 2014; Wang et al., 2014; Xiang et al., 2014; Fercoq et al., 2015; Okumura et al., 2015), where an approximate solution is used for bounding the optimal solution without solving the optimization problem.

4. SAG implementation with piecewise-linear functions

In this section, we present how to compute the bounds on the true solution discussed in §3 under secure computation framework. Specifically, we propose using piecewise-linear functions for the two surrogate loss functions ϕ and ψ. In §4.1, we present a protocol of secure piecewise-linear function evaluation (SPL). In §4.2, we describe a protocol for securely computing the bounds. In full version of our paper (Takada et al.), we describe a specific implementation for logistic regression.



A. $\log(1+\exp(-s))$ B. Bounds with $K=2$ C. Bounds with $K=10$

Figure 3: An example of bounding convex function of one variable $\log(1+\exp(-s))$ with piecewise linear functions with K sections for $s \in [-10, 10]$

4.1. Secure piecewise-linear function computation

Let us denote a piecewise-linear function with K pieces defined in $s \in [T_0, T_K]$ as

$$g(s) = (\alpha_j s + \beta_j) I_{T_{j-1} \le s < T_j}, \tag{4}$$

where $\{(\alpha_j, \beta_j)\}_{j \in [K]}$ are the coefficients of the j-th linear segment and $T_0 < T_1 < \ldots < T_{K-1} < T_K$ are breakpoints. For continuity, we assume that $\alpha_j T_j + \beta_j = \alpha_{j+1} T_j + \beta_{j+1}$ for all $j \in \{0, 1, \ldots, K-1\}$.

An advantage of piecewise-linear functions is that, for any one-dimensional convex function, a lower bounding function can be easily obtained by using its tangents, while an upper bounding function can be also easily obtained by using its chords. In addition, we can easily control the trade-off between the accuracy and the computational complexity by changing the number of pieces K. Figure 3 shows examples of two piecewise-linear surrogate loss functions for a non-linear function $\log(1+\exp(-s))$ for several values of K.

The following theorem states that a piecewise-linear function $g(s)$ can be securely evaluated.

Theorem 3. *Suppose that party A has $E_{pk_B}(s_A)$ and party B has $E_{pk_A}(s_B)$ such that $s = s_A + s_B$. Then, the two parties can securely evaluate the encrypted value of the piecewise-linear function value $g(s)$ in the sense that there is a secure protocol that outputs $E_{pk_B}(g_A)$ and $E_{pk_A}(g_B)$ respectively to party A and party B such that $g_A + g_B = g(s)$.*

The proof of Theorem 3 is presented in full version (Takada et al.). In the proof, we develop such a protocol called SPL, whose input-output property is represented as

$$SPL(E_{pk_B}(s_A), E_{pk_A}(s_B)) \rightarrow (E_{pk_B}(g_A), E_{pk_A}(g_B)).$$

Let $o_j(s) := I_{s \in [T_{j-1}, T_j)}$, $j \in [K]$, denote the indicator of an event that a scalar s is in the j-th piece. The difficulty of secure piecewise-linear function evaluation is that we need to securely compute $E(o_j(s))$. We use a protocol presented in (Damgard et al., 2008) in order to compute $E(I_{a<b})$ from $E(a)$ and $E(b)$, and then compute $E(o_j(s))$ as

$$E(o_j(s)) = E(I_{s \ge T_{j-1}} - I_{s \ge T_j}) = E(I_{s \ge T_{j-1}}) E(I_{s \ge T_j})^{-1}.$$

Using the indicators $\{o_j(s)\}_{j\in[K]}$, the piecewise-linear function value $g(s)$ is written as

$$g(s) = \sum_{j\in[K]} o_j(s)(\alpha_j s + \beta_j), \tag{5}$$

which can be securely computed if $E(o_j(s))$ and $E(s)$ are available.

We finally note that, in Theorem 1, when $\phi(s)$ is represented as a piecewise-linear function, its subgradient $\partial\phi(s)/\partial s$ is represented as a piecewise-constant function and so is the subgradient $\nabla\Phi(\hat{\boldsymbol{w}})$. We can develop a secure piecewise-constant function evaluation protocol based on the same idea as above (detailed in the proof of Theorem 3 in full version (Takada et al.)).

4.2. Secure bound computation

We describe here how to compute the bounds on the true solution in the form of (2) when the surrogate loss functions ϕ and ψ are implemented with piecewise-linear functions. We consider a class of loss functions ℓ that can be decomposed as

$$\ell(y, \boldsymbol{x}^\top\boldsymbol{w}) = u(s(y, \boldsymbol{x}^\top\boldsymbol{w})) + v(y, \boldsymbol{x}^\top\boldsymbol{w}), \tag{6}$$

where u is a non-linear function whose secure evaluation is difficult, while $s(y, \boldsymbol{x}^\top\boldsymbol{w})$, $v(y, \boldsymbol{x}^\top\boldsymbol{w})$, and their subgradients are assumed to be securely evaluated. Note that most commonly-used loss functions can be written in this form. For example, in the case of logistic regression (§2.1), $u(s) = \log(1+\exp(-s))$, $s(y, \boldsymbol{x}^\top\boldsymbol{w}) = \boldsymbol{x}^\top\boldsymbol{w}$ and $v(y, \boldsymbol{x}^\top\boldsymbol{w}) = -y\boldsymbol{x}^\top\boldsymbol{w}$.

We consider a situation that two parties A and B own encrypted approximate solution $\hat{\boldsymbol{w}}$ separately for their own features, i.e., parties A and B own $E_{pk_B}(\hat{\boldsymbol{w}}_A)$ and $E_{pk_A}(\hat{\boldsymbol{w}}_B)$, respectively, where $\hat{\boldsymbol{w}}_A$ and $\hat{\boldsymbol{w}}_B$ the first d_A and the following d_B components of $\hat{\boldsymbol{w}}$.

4.2.1. SECURE COMPUTATIONS OF THE BALL

The following theorem states that the center $\boldsymbol{m}(\hat{\boldsymbol{w}})$ and the radius $r(\hat{\boldsymbol{w}})$ can be securely computed.

Theorem 4. *Suppose that party A has X_A and $E_{pk_B}(\hat{\boldsymbol{w}}_A)$, while party B has X_B, $\boldsymbol{y}$ and $E_{pk_A}(\hat{\boldsymbol{w}}_B)$. Then, the two parties can securely compute the center $\boldsymbol{m}(\hat{\boldsymbol{w}})$ and the radius $r(\hat{\boldsymbol{w}})$ in the sense that there is a secure protocol that outputs $E_{pk_B}(\boldsymbol{m}_A(\hat{\boldsymbol{w}}))$ and $E_{pk_B}(r_A(\hat{\boldsymbol{w}})^2)$ to party A, and $E_{pk_A}(\boldsymbol{m}_B(\hat{\boldsymbol{w}}))$ and $E_{pk_A}(r_B(\hat{\boldsymbol{w}})^2)$ to party B such that $\boldsymbol{m}_A(\hat{\boldsymbol{w}})+\boldsymbol{m}_B(\hat{\boldsymbol{w}}) = \boldsymbol{m}(\hat{\boldsymbol{w}})$ and $r_A(\hat{\boldsymbol{w}})^2 + r_B(\hat{\boldsymbol{w}})^2 = r(\hat{\boldsymbol{w}})^2$.*

We call such a protocol as secure ball computation (SBC) protocol. whose input-output property is characterized as

$$\begin{aligned} &SBC((X_A, E_{pk_B}(\hat{\boldsymbol{w}}_A)), (X_B, \boldsymbol{y}, E_{pk_A}(\hat{\boldsymbol{w}}_B)) \\ &\quad\rightarrow\ ((E_{pk_B}(\boldsymbol{m}_A(\hat{\boldsymbol{w}})), E_{pk_B}(r_A(\hat{\boldsymbol{w}})^2)), \\ &\qquad (E_{pk_A}(\boldsymbol{m}_B(\hat{\boldsymbol{w}})), E_{pk_A}(r_B(\hat{\boldsymbol{w}})^2))) \end{aligned}$$

To prove Theorem 4, we only describe secure computations of three components in the SBC protocol. We omit the security analysis of the other components because they can be easily derived from the security properties of Paillier cryptosystem (Paillier, 1999), comparison protocol (Damgard et al., 2008) and multiplication protocol (Nissim and Weinreb, 2006).

Encrypted values of $\Psi(\hat{\boldsymbol{w}}) - \Phi(\hat{\boldsymbol{w}})$ This quantity can be obtained by summing $\psi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_i)$ for $i \in [n]$. Denoting $\phi := \underline{u}(s) + v$ and $\psi := \overline{u}(s) + v$, where $\underline{u}$ and $\overline{u}$ are lower and upper bounds of u implemented with piecewise-linear functions, respectively, we can compute $\psi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_i) = \overline{u} - \underline{u}$ by using SPL protocol for each of $\overline{u}$ and $\underline{u}$.

Encrypted values of $\nabla\Phi(\hat{\boldsymbol{w}})$ This quantity can be obtained by summing $\nabla\phi$ at $\boldsymbol{w} = \hat{\boldsymbol{w}}$. Since $\nabla\phi = \frac{\partial}{\partial \boldsymbol{w}}\underline{u}(s) + \frac{\partial v}{\partial \boldsymbol{w}} = \underline{u}'(s)\frac{\partial s}{\partial \boldsymbol{w}} + \frac{\partial v}{\partial \boldsymbol{w}}$, its encrypted version can be written as $E(\nabla\phi) = E(\underline{u}'(s)\frac{\partial s}{\partial \boldsymbol{w}})E(\frac{\partial v}{\partial \boldsymbol{w}})$. Here, $\underline{u}'(s)$ can be securely evaluated because $\underline{u}'$ is piecewise-constant function, while $\frac{\partial s}{\partial \boldsymbol{w}}$ and $\frac{\partial v}{\partial \boldsymbol{w}}$ are securely computed from the assumption in (6). For computing $E(\underline{u}'(s)\frac{\partial s}{\partial \boldsymbol{w}})$ from $E(\underline{u}'(s))$ and $E(\frac{\partial s}{\partial \boldsymbol{w}})$, we can use the secure multiplication protocol in (Nissim and Weinreb, 2006).

Encrypted value of $r(\hat{\boldsymbol{w}})^2$ In order to compute this quantity, we need the encrypted value of $\|\frac{1}{2}(\hat{\boldsymbol{w}} + 1/\lambda\nabla\Phi)\|^2$, which can be also computed by using the secure multiplication protocol in (Nissim and Weinreb, 2006).

4.2.2. SECURE COMPUTATIONS OF THE BOUNDS

Finally we discuss here how to securely compute the upper and the lower bounds in (2) from the encrypted $\boldsymbol{m}(\hat{\boldsymbol{w}})$ and $r(\hat{\boldsymbol{w}})^2$. The protocol depends on who owns the test instance and who receives the resulted bounds. We describe here a protocol for a particular setup where the test instance $\tilde{\boldsymbol{x}}$ is owned by two parties A and B, i.e., $\tilde{\boldsymbol{x}} = [\tilde{\boldsymbol{x}}_A^\top \ \tilde{\boldsymbol{x}}_B^\top]^\top$ where $\tilde{\boldsymbol{x}}_A$ and $\tilde{\boldsymbol{x}}_B$ are the first d_A and the following d_B components of $\tilde{\boldsymbol{x}}$, and that the lower and the upper bounds are given to either party. Similar protocols can be easily developed for other setups.

Theorem 5. *Let party A owns $\tilde{\boldsymbol{x}}_A$, $E_{pk_B}(\boldsymbol{m}_A(\hat{\boldsymbol{w}}))$ and $E_{pk_B}(r_A(\hat{\boldsymbol{w}})^2)$, and party B owns $\tilde{\boldsymbol{x}}_B$, $E_{pk_A}(\boldsymbol{m}_B(\hat{\boldsymbol{w}}))$ and $E_{pk_A}(r_B(\hat{\boldsymbol{w}})^2)$, respectively. Then, either party A or B can receive the lower and the upper bounds of $\tilde{\boldsymbol{x}}^\top\boldsymbol{w}^*$ in the form of (2) without revealing $\tilde{\boldsymbol{x}}_A$ and $\tilde{\boldsymbol{x}}_B$ to the others.*

The proof of Theorem 5 is presented in full version (Takada et al.). We note that a party who receives bounds from the protocol would get some information about the center $\boldsymbol{m}_B(\hat{\boldsymbol{w}})$ and the radius $\boldsymbol{r}_B(\hat{\boldsymbol{w}})$, but no other information about the original dataset is revealed.

5. Experiments

We conducted experiments for illustrating the performances of the proposed SAG method. The experimental setup is as follows. We used Paillier cryptosystem with $N = 1024$-bit public key and comparison protocol (Damgard et al., 2008) for 60 bits of integers. The program is implemented with Java, and the communications between two parties are implemented with sockets between two processes working in the same computer. We used a single computer with 3.07GHz Xeon CPU and 48GB RAM. Except when we investigate computational costs, computations were done on unencrypted values. Note that the proposed SAG method provide bounds on the true solution $\boldsymbol{w}^*$ based on an arbitrary approximate solution $\hat{\boldsymbol{w}}$. In all the experiments presented here, we used approximate solutions obtained by Nardi et al. (2012) approach as the approximate solution $\hat{\boldsymbol{w}}$. In what follows, we call the bounds or intervals obtained by the SAG method as SAG bounds and SAG intervals, respectively.

Table 1: Data sets used for the logistic regression. All are from UCI Machine Learning Repository.

data set	training set	validation set	d
Musk	3298	3300	166
MGT	9510	9510	10
Spambase	2301	2301	57
OLD	1268	1268	72

5.1. Logistic regression

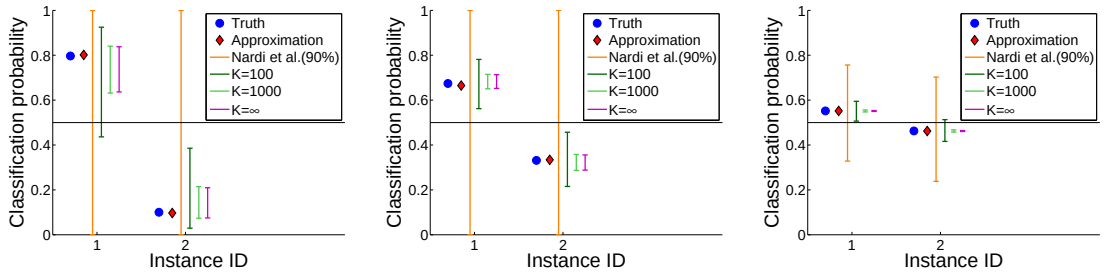
We first investigated several properties of the SAG method by applying it to four benchmark datasets summarized in Table 1. Due to the space limitation, only the results on Musk dataset are shown in the main text, and results on other datasets are presented in full version (Takada et al.).

First, in Figure 4, we compared the tightness of the bounds on the predicted classification probabilities for two randomly chosen validation instances $\boldsymbol{x}_i$ defined as $p(\boldsymbol{x}_i) := 1/(1+\exp(-\boldsymbol{x}_i^\top \boldsymbol{w}^*))$, $i = 1, 2$. In the figure, four types of intervals are plotted. The orange ones are probabilistic bounds in (Nardi et al., 2012) with the probability 90%. The blue, green and purple ones were obtained by the SAG method with $K = 100, 1000$ and ∞, respectively, where K is the number of pieces in the piecewise-linear approximations. Here, $K = \infty$ means that the true loss function ℓ was used as the two surrogate loss functions ϕ and ψ. The results clearly indicate that bounds obtained by the SAG method are clearly tighter than those by Nardi et al.'s approach despite that the latter is probabilistic and cannot be securely computed in practice. When comparing the results with different K, we can confirm that large K yields tighter bounds. The results with $K = 1000$ are almost as tight as those obtained with the true loss function ($K = \infty$).

Figure 5 also shows similar plots. Here, we investigated how the tightness of the SAG bounds changes with the quality of the approximate solution $\hat{\boldsymbol{w}}$. In order to consider approximate solutions with different levels of quality, we computed three approximate solutions with $L = 10, 100$ and 1000 in Nardi et al.'s approach, where L is the sample size used for approximating the logistic function (see §2). The results clearly indicate that tighter bounds are obtained when the quality of the approximate solution is higher (i.e., larger L).

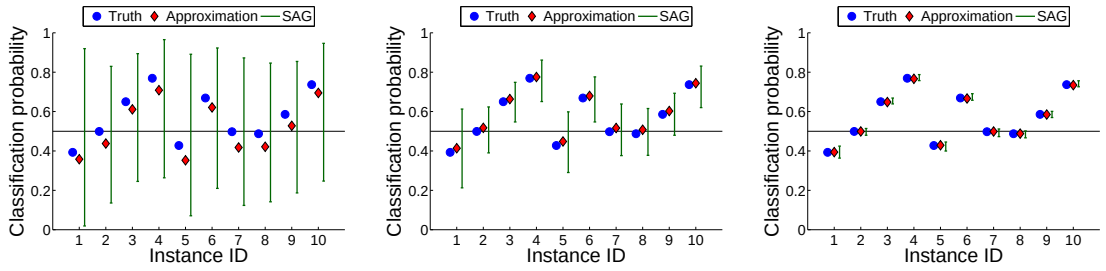
Figure 6 illustrates how the SAG bounds can be useful in binary classification problems. In binary classification problems, if a lower bound of the classification probability is greater than 0.5, the instance would be classified to positive class. Similarly, if an upper bound of the classification probability is smaller than 0.5, the instance would be classified to negative class. The green histograms in the figure indicate how many percent of the validation instances can be certainly classified as positive or negative class based on the SAG bounds. The blue lines indicate the average length of the SAG intervals, i.e., the difference between the upper and the lower bounds. The results clearly indicate that, as the number of pieces K increases in the SAG method, the tighter bounds are obtained, and more validation instances can be certainly classified. On the other hand, probabilistic bounds in Nardi et al.'s approach cannot provide certain classification results because their bounds are too loose.

Finally, we explain the computational cost for computing the SAG bounds. The computational cost mainly depends on the number of the comparison protocols. In fact, when we use K-piece piecewise linear functions as ϕ and ψ, we need to conduct the comparison for nK time to compute each of $\Phi(\hat{\boldsymbol{w}})$ and $\Psi(\hat{\boldsymbol{w}})$ (§3). In our setting, one comparison needs about 0.8 second. In the real



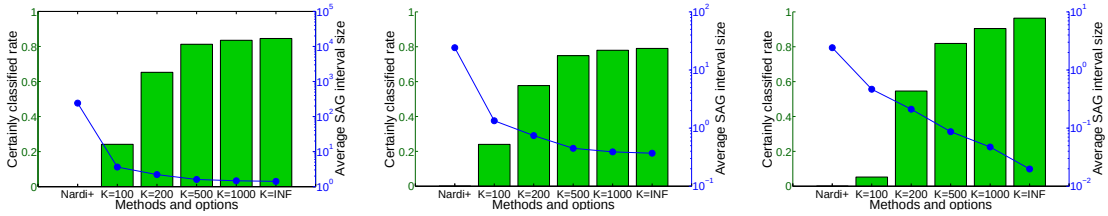
$\lambda = 0.1$ $\lambda = 1$ $\lambda = 10$

Figure 4: The result of proposed bounds for some test instances (data set Musk)



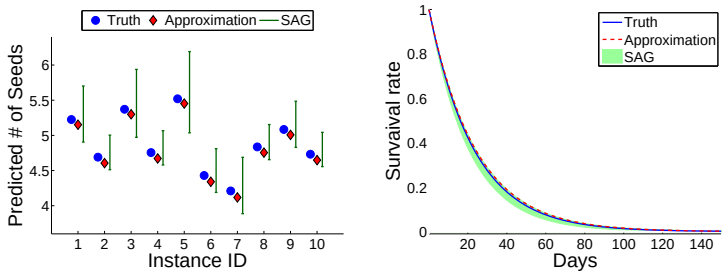
$L = 10$ $L = 100$ $L = 1000$

Figure 5: Change of bounds for the change of the accuracy of the approximated solution $\hat{\boldsymbol{w}}$ (data set Musk)



$\lambda = 0.1$ $\lambda = 1$ $\lambda = 10$

Figure 6: Rate of successfully classified test instances and the average of size of bounds by different bound calculations (Nardi's, $K \in \{100, 200, 500, 1000, \infty\}$) (data set Musk)



(A) Poisson regression (B) Exponential regression

Figure 7: Proposed bounds for Poisson and exponential regressions

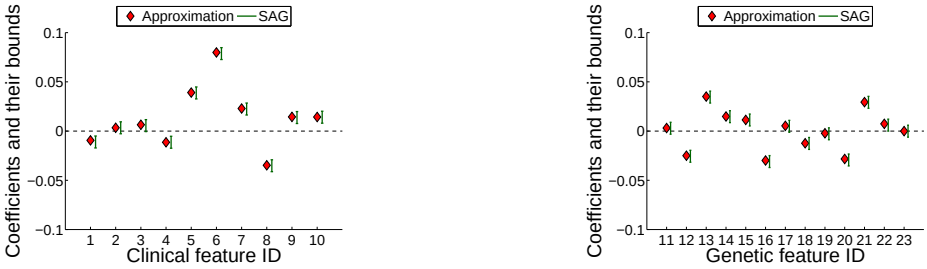


Figure 8: Bounds of coefficients for disease risk evaluation

application, we set K for a trade-off between the tightness of the bounds and the computational cost.

5.2. Poisson and exponential regressions

We applied the SAG method to t Poisson regression and exponential regression. Poisson regression was applied to a problem for predicting the number of produced seeds [2]. Exponential regression was applied to a problem for predicting survival time of lung cancer patients [3]. The results are shown in Figure 7. The left plot (A) shows the result of Poisson regression, where the SAG intervals on the predicted number of seeds are plotted for several randomly chosen instances. The right plot (B) shows the SAG bounds on the predicted survival probability curve, in which we can confirm that the true survival probability curve is included in the SAG bound.

5.3. Privacy-preserving logistic regression to genomic and clinical data analysis

Finally, we apply the SAG method to a logistic regression on a genomic and clinical data analysis, which is the main motivation of this work (§1). We apply the SAG method for computing the bounds of coefficients of the logistic regression model as described in §3.

In this experiment, 13 genomic (SNP) and 10 clinical features of 134 potential patients are provided from a research institute and a hospital, respectively [4]. The SAG bounds on each of these 23 coefficients are plotted in Figure 8. Although we do not know the true coefficient values, we can at least identify features that positively/negatively correlated with the disease risk (note that, if the lower/upper bound is greater/smaller than 0, the feature is guaranteed to have positive/negative coefficient in the logistic regression model).

6. Conclusions

We studied empirical risk minimization (ERM) problems under secure multi-party computation (MPC) frameworks. We developed a novel technique called secure approximation guarantee (SAG) method that can be used when only an approximate solution is available due to the difficulty of secure non-linear function evaluations. The key property of the SAG method is that it can securely provide the bounds on the true solution, which is practically valuable as we illustrated in benchmark data experiments and in our motivating problem on genomic and clinical data.

2. http://hosho.ees.hokudai.ac.jp/~kubo/stat/2015/Fig/poisson/data3a.csv
3. http://help.xlstat.com/customer/portal/kb_article_attachments/60040/original.xls
4. Due to confidentiality reasons, we cannot describe the details of the dataset. Here, we only analyzed a randomly sampled small portion of the datasets just for illustration purpose.

Acknowledgement

It was partially supported by JST CREST 13217726, CREST 15656320, JSPS KAKENHI Grant Number 26280083, MEXT KAKENHI Grant Number 16H06538, JST support program for starting up innovation-hub on materials research by information integration initiative, and RIKEN AIP center.

References

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

Ivan Damgard, Martin Geisler, and Mikkel Kroigard. Homomorphic encryption and secure comparison. *International Journal of Applied Cryptography*, 1(1):22–31, 2008.

L. El Ghaoui, V. Viallon, and T. Rabbani. Safe feature elimination for the lasso and sparse supervised learning problems. *Pacific Journal of Optimization*, 8(4):667–698, 2012.

O. Fercoq, A. Gramfort, and J. Salmon. Mind the duality gap: safer rules for the lasso. In *The 32nd International Conference on Machine Learning (ICML 2015)*, 2015.

O. Goldreich. *Foundations of cryptography: volume 1, basic tools*. Cambridge university press, 2001.

O. Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2004.

R. Hall, S. E. Fienberg, and Y. Nardi. Secure multiple linear regression based on homomorphic encryption. *Journal of Official Statistics*, 27(4):669, 2011.

S. Laur, H. Lipmaa, and T. Mielikäinen. Cryptographically private support vector machines. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD2006)*, pages 618–624. ACM, 2006.

J. Liu, Z. Zhao, J. Wang, and J. Ye. Safe Screening with Variational Inequalities and Its Application to Lasso. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Y. Nardi, S. E. Fienberg, and R. J. Hall. Achieving both valid and secure logistic regression analysis on aggregated data from different private sources. *Journal of Privacy and Confidentiality*, 4(1): 9, 2012.

V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *2013 IEEE Symposium on Security and Privacy (SP)*, pages 334–348. IEEE, 2013.

K. Nissim and E. Weinreb. Communication efficient secure linear algebra. In *Theory of Cryptography*, pages 522–541. Springer, 2006.

K. Ogawa, Y. Suzuki, and I. Takeuchi. Safe screening of non-support vectors in pathwise svm computation. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1382–1390, 2013.

S. Okumura, Y. Suzuki, and I. Takeuchi. Quick sensitivity analysis for incremental data modification and its application to leave-one-out cv in linear classification problems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 885–894. ACM, 2015.

P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology – EUROCRYPT'99*, pages 223–238. Springer, 1999.

T. Takada, H. Hanada, Y. Yamada, J. Sakuma, and I. Takeuchi. Secure approximation guarantee for cryptgraphically private empirical risk minimization. https://arxiv.org/abs/1602.04579.

J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2003)*, pages 206–215. ACM, 2003.

J. Wang, J. Zhou, J. Liu, P. Wonka, and J. Ye. A safe screening rule for sparse logistic regression. In *Advances in Neural Information Processing Systems*, pages 1053–1061, 2014.

Z. J Xiang, H. Xu, and P. J. Ramadge. Learning sparse representations of high dimensional data on large scale dictionaries. In *Advances in Neural Information Processing Systems*, pages 900–908, 2011.

Z. J. Xiang, Y. Wang, and P. J. Ramadge. Screening tests for lasso problems. *arXiv preprint arXiv:1405.4897*, 2014.

C. A. Yao. How to generate and exchange secrets. In *The 27th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1986)*, pages 162–167. IEEE, 1986.

H. Yu, J. Vaidya, and X. Jiang. Privacy-preserving svm classification on vertically partitioned data. In *Advances in Knowledge Discovery and Data Mining*, pages 647–656. Springer, 2006.

Appendix A

Proofs of Theorem 1 and Corollary 2 (bounds of w^* from $\hat{w}$)

First we present the following proposition which will be used for proving Theorem 1.

Proposition 6. *Consider the following general problem:*

$$\min_{z} \; g(z) \quad \text{s.t. } z \in \mathcal{Z}, \tag{7}$$

where $g : \mathcal{Z} \to \mathbb{R}$ is a subdifferentiable convex function and $\mathcal{Z}$ is a convex set. Then a solution z^* is the optimal solution of (7) if and only if

$$\nabla g(z^*)^\top (z^* - z) \leq 0 \quad \forall\, z \in \mathcal{Z},$$

where $\nabla g(z^*)$ is the subgradient vector of g at $z = z^*$.

See, for example, Proposition B.24 in (Bertsekas, 1999) for the proof of Proposition 6.

Proof of Theorem 1. Using a slack variable $\xi \in \mathbb{R}$, let us first rewrite the minimization problem (1) as

$$\begin{aligned} \min_{\boldsymbol{w} \in \mathbb{R}^d, \xi \in \mathbb{R}} \; & J(\boldsymbol{w}, \xi) := \xi + \frac{\lambda}{2} \|\boldsymbol{w}\|^2 \\ \text{s.t. } & \xi \geq \frac{1}{n} \sum_{i \in [n]} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}). \end{aligned} \tag{8}$$

Note that the optimal solution of the problem (8) is $\boldsymbol{w} = \boldsymbol{w}^*$ and $\xi = \xi^* := \frac{1}{n} \sum_{i \in [n]} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}^*)$. Using the definitions of ψ and Ψ, we have $\frac{1}{n} \sum_{i \in [n]} \ell(y_i, \boldsymbol{x}_i^\top \hat{\boldsymbol{w}}) \leq \frac{1}{n} \sum_{i \in [n]} \psi(y_i, \boldsymbol{x}_i^\top \hat{\boldsymbol{w}}) = \Psi(\hat{\boldsymbol{w}})$. It means that $(\hat{\boldsymbol{w}}, \Psi(\hat{\boldsymbol{w}}))$ is a feasible solution of the problem (8). Applying this fact into Proposition 6, we have

$$\nabla J(\boldsymbol{w}^*, \xi^*)^\top \left(\begin{bmatrix} \boldsymbol{w}^* \\ \xi^* \end{bmatrix} - \begin{bmatrix} \hat{\boldsymbol{w}} \\ \Psi(\hat{\boldsymbol{w}}) \end{bmatrix} \right) \leq 0, \tag{9}$$

where $\nabla J(\boldsymbol{w}^*, \xi^*) \in \mathbb{R}^{d+1}$ is the gradient of the objective function in (8) evaluated at $(\boldsymbol{w}^*, \xi^*)$. Since $J(\boldsymbol{w}, \xi)$ is a quadratic function of $\boldsymbol{w}$ and ξ, we can write $\nabla J(\boldsymbol{w}^*, \xi^*)$ explicitly, and (9) is written as

$$\begin{aligned} & \lambda \|\boldsymbol{w}^2\| + \xi^* - \lambda \boldsymbol{w}^{*\top} \hat{\boldsymbol{w}} - \Psi(\hat{\boldsymbol{w}}) \leq 0 \\ \Leftrightarrow \quad & \lambda \|\boldsymbol{w}^{*2}\| + \frac{1}{n} \sum_{i \in [n]} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}^*) \\ & - \lambda \boldsymbol{w}^{*\top} \hat{\boldsymbol{w}} - \Psi(\hat{\boldsymbol{w}}) \leq 0 \end{aligned} \tag{10}$$

From the definition of ϕ and Φ, we can plug $\frac{1}{n} \sum_{i \in [n]} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}^*) \geq \frac{1}{n} \sum_{i \in [n]} \phi(y_i, \boldsymbol{x}_i^\top \boldsymbol{w}^*) = \Phi(\boldsymbol{w}^*)$ into (10). Then,

$$\lambda \|\boldsymbol{w}^{*2}\| + \Phi(\boldsymbol{w}^*) - \lambda \boldsymbol{w}^{*\top} \hat{\boldsymbol{w}} - \Psi(\hat{\boldsymbol{w}}) \leq 0 \tag{11}$$

Furthermore, noting that ϕ and Φ are convex with respect $\boldsymbol{w}$, by the definition of convex functions we get

$$\Phi(\boldsymbol{w}^*) \geq \Phi(\hat{\boldsymbol{w}}) + \nabla\Phi(\hat{\boldsymbol{w}})^\top(\boldsymbol{w}^* - \hat{\boldsymbol{w}}). \tag{12}$$

By plugging (12) into (11),

$$\begin{aligned}\lambda\|\boldsymbol{w}^{*2}\| + \Phi(\hat{\boldsymbol{w}}) + \nabla\Phi(\hat{\boldsymbol{w}})^\top(\boldsymbol{w}^* - \hat{\boldsymbol{w}}) \\ -\lambda\boldsymbol{w}^{*\top}\hat{\boldsymbol{w}} - \Psi(\hat{\boldsymbol{w}}) \leq 0\end{aligned} \tag{13}$$

Noting that (13) is a quadratic function of $\boldsymbol{w}^*$, we obtain

$$\begin{aligned}&\left\|\boldsymbol{w}^* - \frac{1}{2}\left(\hat{\boldsymbol{w}} - \frac{1}{\lambda}\nabla\Phi(\hat{\boldsymbol{w}})\right)\right\|^2 \\ &\leq \left\|\frac{1}{2}\left(\hat{\boldsymbol{w}} + \frac{1}{\lambda}\nabla\Phi(\hat{\boldsymbol{w}})\right)\right\|^2 + \frac{1}{\lambda}\left(\Psi(\hat{\boldsymbol{w}}) - \Phi(\hat{\boldsymbol{w}})\right).\end{aligned}$$

It means that the optimal solution $\boldsymbol{w}^*$ is within a ball with the center $\boldsymbol{m}(\hat{\boldsymbol{w}})$ and the radius $r(\hat{\boldsymbol{w}})$, which completes the proof. ■

Next, we prove Corollary 2.

Proof of Corollary 2. We show that the lower bound of the linear model output value $\boldsymbol{w}^{*\top}\boldsymbol{x}$ is $\boldsymbol{x}^\top\boldsymbol{m}(\hat{\boldsymbol{w}}) - \|\boldsymbol{x}\|r(\hat{\boldsymbol{w}})$ under the constraint that

$$\|\boldsymbol{w}^* - \boldsymbol{m}(\hat{\boldsymbol{w}})\| \leq r(\hat{\boldsymbol{w}}).$$

To formulate this, let us consider the following constrained optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \boldsymbol{w}^\top\boldsymbol{x} \quad \text{s.t.} \quad \|\boldsymbol{w} - \boldsymbol{m}(\hat{\boldsymbol{w}})\|^2 \leq r(\hat{\boldsymbol{w}})^2. \tag{14}$$

Using a Lagrange multiplier $\mu > 0$, the problem (14) is rewritten as

$$\begin{aligned}&\min_{\boldsymbol{w}\in\mathbb{R}^d} \boldsymbol{w}^\top\boldsymbol{x} \quad \text{s.t.} \quad \|\boldsymbol{w} - \boldsymbol{m}(\hat{\boldsymbol{w}})\|^2 \leq r(\hat{\boldsymbol{w}})^2, \\ =&\min_{\boldsymbol{w}\in\mathbb{R}^d}\max_{\mu>0}\left(\boldsymbol{w}^\top\boldsymbol{x} + \mu(\|\boldsymbol{w} - \boldsymbol{m}(\hat{\boldsymbol{w}})\|^2 - r(\hat{\boldsymbol{w}})^2)\right) \\ =&\max_{\mu>0}\left(-\mu r(\hat{\boldsymbol{w}})^2 + \min_{\boldsymbol{w}}\left(\mu\|\boldsymbol{w} - \boldsymbol{m}(\hat{\boldsymbol{w}})\|^2 + \boldsymbol{w}^\top\boldsymbol{x}\right)\right) \\ =&\max_{\mu>0} H(\mu) := \left(-\mu r(\hat{\boldsymbol{w}})^2 - \frac{\|\boldsymbol{x}\|^2}{4\mu} + \boldsymbol{x}^\top\boldsymbol{m}(\hat{\boldsymbol{w}})\right),\end{aligned}$$

where μ is strictly positive because the constraint $\|\boldsymbol{w} - \boldsymbol{m}(\hat{\boldsymbol{w}})\|^2 \leq r(\hat{\boldsymbol{w}})^2$ is strictly active at the optimal solution. By letting $\partial H(\mu)/\partial\mu = 0$, the optimal μ is written as

$$\mu^* := \frac{\|\boldsymbol{x}\|}{2r(\hat{\boldsymbol{w}})} = \arg\max_{\mu>0} H(\mu).$$

Substituting μ^* into $H(\mu)$,

$$\boldsymbol{x}^\top\boldsymbol{m}(\hat{\boldsymbol{w}}) - \|\boldsymbol{x}\|r(\hat{\boldsymbol{w}}) = \max_{\mu>0} H(\mu).$$

The upper bound part can be shown similarly. ■