New Privacy Preserving Back Propagation Learning for Secure Multiparty Computation

Hirofumi Miyajima[†1], Noritaka Shigei[†2], Hiromi Miyajima[†3],Yohtaro Miyanishi[†4] Shinji Kitagami[†5] and Norio Shiratori[†6]

***Abstract*—Many studies have been done with the security of cloud computing. Data encryption is one of typical approaches. However, complex computing requirement for encrypted data needs a great deal of time and effort for the system in this case. Therefore, another studies on secure sharing and computing methods are made to avoid secure risks being abused or leaked and to reduce computing cost. The secure multiparty computation (SMC) is one of these methods. So far, some studies have been done with SMC. Specifically, SMC with secure shared data in addition and multiplication forms is proposed and applied to arithmetic operation and simple statistical computation. However, complex calculation processing such as BP(Back Propagation) learning has never proposed yet. In this paper, we propose BP learning method for SMC on cloud computing system and prove the validity of it. Further, the performance of the proposed method is shown in numerical simulations.**

***Index Terms*—cloud computing, secure multiparty computation, BP learning, perceptron learning.**

I. Introduction

Cloud computing is service to provide computing resources through the Internet for the unspecified number of individual or a company [1], [2]. The client can realize reduction of the operational cost and the increase and decrease of resources flexibly without having one's resources. In addition, the spreading of computing of cloud computing allows use such as big data analysis to analyze enormous information accumulated by the client, and to create market value of data [1]. On the other hand, the client of cloud computing cannot escape from anxiety about the possibility of information being abused or leaked. How can we construct a cloud computing system to avoid the above risk? For this purpose, data encryption is one of typical approaches [3], [4]. Data encryption is an effective method to protect data from external risk, but the encrypted data must be decrypted to get plain texts when processed in a cloud. Therefore, safe system for distributed processing with secure data attracts attention, and a lot of studies have been done [5]–[9]. SMC is one of these methods. Most of works in SMC are developed on applying the model of SMC on different data distributions such as vertically, horizontally and arbitrary partitioned data [10]–[13]. They are the methods that each party performs its processing for the subset of data. Another method to realize secure computation is SMC for shared data, which is currently applied to certain applications for limited function calculations [16]–[19]. Further, Miyanishi proposed a simple method to share data and applied to statistical computation [17]. However, complex calculation processing such as data mining has never proposed yet.

In this paper, we propose BP learning method for SMC and prove the validity of it. Further, the performance of the proposed method is shown in numerical simulations. The aim is to realize learning using secure sharing and computing for data being on cloud system. In Section 2, we explain cloud computing system, related works on SMC and how to share the data used in this paper. Further, perceptron and BP learning are introduced. Though perceptron learning is a special case of BP learning, it is introduce to help understanding the proposed algorithm for BP learning. In Section 3, we propose perceptron and BP learning for SMC and show the validity of the algorithm. In section 4, numerical simulations of function approximation and pattern classification are performed to show the performance of the proposed method.

II. Preliminary

A. System configuration of cloud system and related works with privacy preserving BP

Fig.1 shows a system for SMC of cloud computing used in this paper. The system is composed of a client and cloud with m servers(parties). The client sends data to each server and each server memorizes them. If the client requires data processing, each server performs one's computation and sends each result to the client. The client computes the final result using them. If the result is not obtained by one processing, data processing between the client and servers are iterated until the final result is obtained. The problem is how data are shared and the computation for each server is carried out.

Let us consider about conventional works with them. In order to solve the problem, three partitioned representation of data such as horizontally, vertically and any partitioned methods for SMC are known [10]–[13]. Let us explain about them using an example of Table 1. In Table 1, a and b are original data (marks) and ID is student identifier. The purpose of computation is to get the average of them.

First, let us explain about horizontally partitioned method using Table 1. All the dataset are divided into two servers, Server 1 and 2 as follows:

Affiliation: Graduate School of Science and Engineering, Kagoshima University, 1-21-40 Korimoto, Kagoshima 890-0065, Japan
corresponding auther to provide email: miya@eee.kagoshima-u-ac.jp
†1 email: k3768085@kadai.jp
†2 email: shigei@eee.kagoshima-u-ac.jp
†3 email: miya@eee.kagoshima-u-ac.jp
Affiliation: Information Systems Engineering and Management, Tokyo, Japan
†4 email: miyanisi@jade.dti.ne.jp
Affiliation: Waseda University Graduate School of Global Information and Telecommunication Studies(GITS), Tokyo, Japan
†5 email: kitagami.shinji@meltec.co.jp
Affiliation: Waseda University GITS, Tokyo, Japan
†6 email: norio@shiratori.riec.tohoku.ac.jp

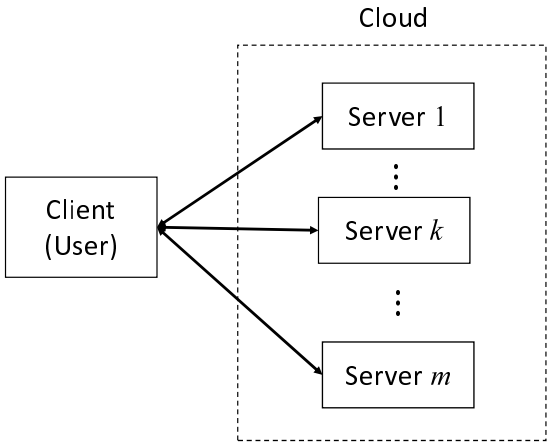

Fig. 1. A configuration of cloud computing system.

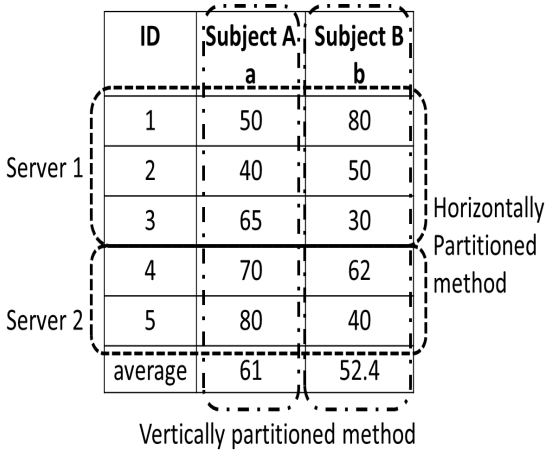

TABLE I
CONCEPT OF HORIZONTALLY AND VERTICALLY PARTITIONED METHOD COMPOSED OF ONE CLIENT AND TWO SERVERS.

Server 1: dataset for ID=1, 2, 3,
Server 2: dataset for ID=4,5.

In this case, two averages with subsets A and B for Server 1 are computed as $(50+40+65)/3$ and $(80+50+30)/3$, respectively. Likewise, two averages with subsets A and B for Server 2 are $(70+80)/2$ and $(62+40)/2$, respectively. As a result, two averages for subsets A and B are 61.0 and 52.4, respectively. Each server cannot know half of the dataset, so privacy preserving hold. In effect, raw data are not used, but encrypted data are used.

Next, let us consider about vertically partitioned method for SMC. All the dataset are divided into two servers, Server 1 and 2, as follows:

Server 1: dataset for subject A,
Server 2: dataset for subject B.

In this case, we can easily get two averages for subject A and B as usual. Each server can know only data for subject A or B, so privacy preserving hold.

At third, let us consider about any partitioned method for SMC. All the dataset are divided into two parties, mixed horizontally and vertically partitioned data. For example,

Server 1: dataset of ID=1, 2, 3 for subject A and ID=4, 5 for subject B,
Server 2: dataset of ID=4, 5 for subject A and ID=1, 2, 3 for subject B.

In this case, we can easily get two averages for subject A and B. Each server can know only partial dataset for subject A or B, so privacy preserving hold.

B. The representation of secure shared data

Let us explain data representation for the proposed method using Fig.2 [13], [17]. Let a and b be two positive integers and $m = 3$ for the number of servers. First, two integers a and b are shared into three real numbers. Let $a = a^{(1)} + a^{(2)} + a^{(3)}$ and $b = b^{(1)} + b^{(2)} + b^{(3)}$ as addition form and $a = A^{(1)}A^{(2)}A^{(3)}$ and $b = B^{(1)}B^{(2)}B^{(3)}$ as the multiplication form. Then the following results hold:

1) $a + b = (a^{(1)} + b^{(1)}) + (a^{(2)} + b^{(2)}) + (a^{(3)} + b^{(3)})$
2) $a - b = (a^{(1)} - b^{(1)}) + (a^{(2)} - b^{(2)}) + (a^{(3)} - b^{(3)})$
3) $ab = (A^{(1)}B^{(1)})(A^{(2)}B^{(2)})(A^{(3)}B^{(3)})$
4) $a/b = (A^{(1)}/B^{(1)})(A^{(2)}/B^{(2)})(A^{(3)}/B^{(3)})$

That is, four basic operations of arithmetic (addition, subtraction, multiplication, and division) hold as integration

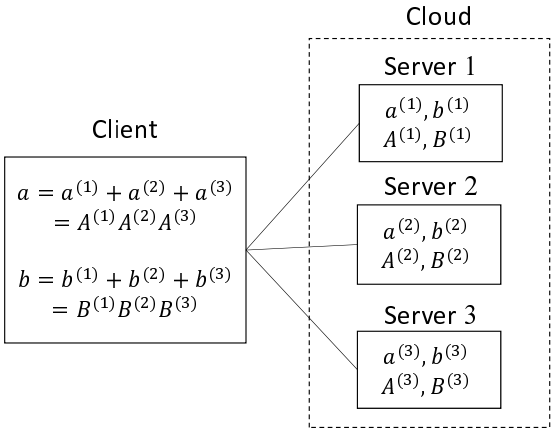

Fig. 2. The representation of secure shared data for $m = 3$.

of the result computed independently by each server [19]. In this case, each server can not know the original data a and b. Further, let us explain how to compute the sum and average for results processing using shared data in addition form.

Let us show a calcuration example of shared data using Table II as follows [17]:

$a = a^{(1)} + a^{(2)} : a^{(1)} = a(r_1/10)$ and $a^{(2)} = a(1 - r_1/10)$
$b = b^{(1)} + b^{(2)} : b^{(1)} = b(r_1/10)$ and $b^{(2)} = b(1 - r_2/10)$
$a = A^{(1)}A^{(2)} : A^{(1)} = \sqrt{a}(r_1/10)$ and $A^{(2)} = \sqrt{a}(10/r_1)$
$b = B^{(1)}B^{(2)} : B^{(1)} = \sqrt{b}(r_2/10)$ and $B^{(2)} = \sqrt{b}(10/r_2)$

Where r_1 and r_2 are real random numbers for $-9 \leq r_1 \leq 9$ and $0.2 \leq r_2 \leq 9$, $r_1 \neq 1$, $r_2 \neq 1$, respectively. For example, $a^{(1)}$ and $a^{(2)}$ for ID=1 are computed as $a^{(1)} = 50 \times (4/10) = 20$ and $a^{(2)} = 50 \times (1 - 4/10) = 30$ and data $A^{(1)}$ and $A^{(2)}$ for ID=1 are computed as $A^{(1)} = \sqrt{50} \times (9/10) = 6.31$ and $A^{(2)} = \sqrt{50} \times (10/9) = 7.86$, respectively. Note that Server 1 has all the data in column-wise of $a^{(1)}$, $b^{(1)}$, $A^{(1)}$ and $B^{(1)}$ for each ID and Server 2 has all the data in column-wise of $a^{(2)}$, $b^{(2)}$, $A^{(2)}$ and $B^{(2)}$ for each ID as shown in Fig.2.

Let us explain how to compute the sum and average for subject A using data a. Server 1 and Server 2 compute each sum of $a^{(1)}$ and $a^{(2)}$, respectively. In this case, each sum in column-wise for $a^{(1)}$ and $a^{(2)}$ is -23 and 328, respectively. As

TABLE II
DATA ON SERVER 1 AND SERVER 2.

ID	subject A	subject B	Additionbal form					Multiplication form				
				a		b			A		B	
	a	b	r_1	$a^{(1)}$	$a^{(2)}$	$b^{(1)}$	$b^{(2)}$	r_2	$A^{(1)}$	$A^{(2)}$	$B^{(1)}$	$B^{(2)}$
1	50	80	4	20	30	32	48	9	6.31	7.86	8.05	9.94
2	40	50	-6	-24	64	-30	80	2	1.27	31.62	1.41	35.36
3	65	30	2	13	52	6	24	0.8	0.65	100.78	0.44	68.47
4	70	62	-8	-56	126	-49.6	111.6	5	4.18	16.73	3.94	15.75
5	80	40	3	24	56	12	28	4	3.58	22.36	2.53	15.81
sum	305	262		-23	328	-29.6	291.6					
average	61	52.4		-4.6	65.6	-5.92	58.32					

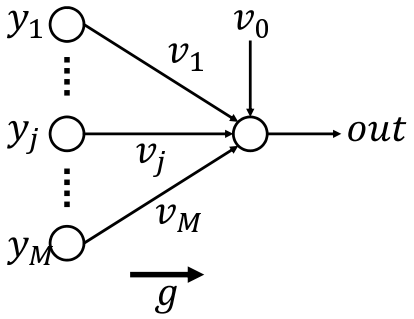

Fig. 3. A neural element.

a result, the total sum 305 is obtained from 328-23. Likewise, the average 61 is obtained from 65.6-4.6.

Remark that each data for server is randomized and the method does not need to use encrypted data.

In the following, we propose a learning method using secure shared data in multiplication form.

C. Perceptron learning for a neural element

Perceptron learning is a method to adjust the weight parameters to match the output of the neuron and the desired output. Desired output for input are called learning data. In percetron learning, the weight parameters are updated repeatedly using the difference between the output of the neuron and the desired output until the difference becomes sufficiently small.

Let us explain the conventional neuron [19]. Let $Z_i = \{1, - - -, i\}$ for a positive integer i. The output y for each neuron is given by

$$u = \sum_{j=0}^{M} v_j y_j \quad (1)$$

$$out = g(u) \quad (2)$$

where y_j is the j-th input, u is the internal potential of the neuron, $g(u)$ is the output function, v_j is the weight for the j-th input, v_0 is the threshold and $y_0 = 1$(See Fig. 3).

Let the evaluation function for learning data $Y = \{(\boldsymbol{y}^l, d(\boldsymbol{y})|l \in Z_L)\}$ be defined as follows:

$$E = \frac{1}{2L} \sum_{l=1}^{L} \left(g(\boldsymbol{y}^l) - d(\boldsymbol{y}^l)\right)^2 \quad (3)$$

, where $d(\boldsymbol{y}^l)$ is the desired output for the l-th data $\boldsymbol{y}^l$.

In order to minimize the function E, the weights for the neuron are updated as follows:

$$\triangle v_j = \frac{\partial E}{\partial v_j} = \left(g(\boldsymbol{y}^l) - d(\boldsymbol{y}^l)\right) \frac{\partial g(u)}{\partial u} y_j^l \quad (4)$$

$$v_j(t+1) = v_j(t) + K \triangle v_j(t) \quad (5)$$

, where K is a learning constant and $j \in Z_M$

In order to consider the simple case, let us assume that $\frac{\partial g(u)}{\partial u} = 1$. As a result, learning formula for perceptron learning is shown as follows:

$$v_j(t+1) = v_j(t) + K \left(g(\boldsymbol{y}^l) - d(\boldsymbol{y}^l)\right) y_j \quad (6)$$

, where t is the number of learning time.

Then, learning algorithm for learning data $\{(\boldsymbol{y}^l, d(\boldsymbol{y}^l))|l \in Z_L\}$ is shown as follows [19]:

Step1: Given the maximum number of learning t_{max}. Let $t = 1$. The initial assignment of $v_j (j \in Z_M)$ is set randomly.
Step 2: Let $l = 1$.
Step 3: Select a data $(\boldsymbol{y}^l, d(\boldsymbol{y}^l))$.
Step 4: Compute the output $g(\boldsymbol{y}^l)$ for input $\boldsymbol{y}^l$.
Step 5: Compute the error $\left(g(\boldsymbol{y}^l) - d(\boldsymbol{y}^l)\right)$.
Step 6: Compute the updated value $\triangle v_j(t)$ using Eq.(4).
Step 7: Update the weight as $v_j(t+1) = v_j(t) + K \triangle v_j(t)$.
Step 8: If $l \neq L$, then go to Step 3 with $l \leftarrow l+1$ and $t \leftarrow t+1$ else go to Step 9.
Step 9: If $t \neq T_{max}$, then go to Step 2 with $t \leftarrow t+1$ else the algorithm terminates.

D. BP method for neural networks

Let us consider the case of n input and one output without loss of generality. Let $\boldsymbol{z}^l \in J^N$ for $l \in Z_L$ and $d : J^N \to J$, where $J = [0, 1]$. Giving learning data $\{(\boldsymbol{z}^l, d(\boldsymbol{z}^l))|l \in Z_L\}$, let us determine the three-layered neural network identifying learning data by BP method [19], [20]. Let $h = g \circ e : J^N \to J$ be the function defined by a neural network. Let the number of elements in second layer be M. Let $\boldsymbol{w}_j$ for $j \in Z_M$ and $\boldsymbol{v}$ be weights for the second and output layers, respectively. Then g and e are defined as follows (See Fig.4):

$$y_j = e_j(\boldsymbol{z}) = \tau\left(\sum_{i=0}^{N} w_{ij} z_i\right),$$

$$z_0 = 1,$$

$$\tau(u) = \frac{1}{1 + \exp(-u)}$$

where

$$\boldsymbol{z} = (z_1, \cdots, z_N) \in J^N$$

$$\boldsymbol{y} = (y_1, \cdots, y_M) \in J^M$$

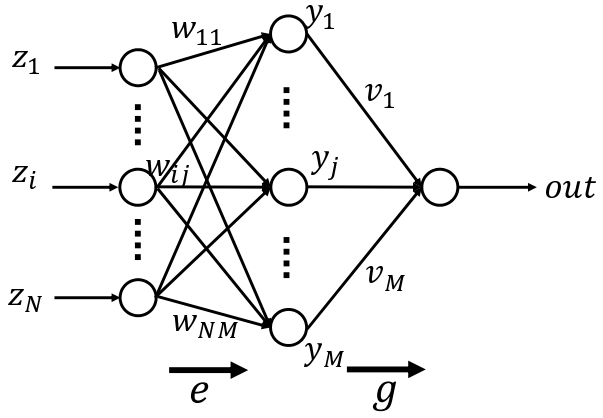

Fig. 4. Three-layered neural network

and w_{0j} means the threshold value.

Further,

$$\begin{aligned} g(\boldsymbol{y}) &= \tau\left(\sum_{j=0}^{M} v_j y_j\right), \\ y_0 &= 1, \end{aligned}$$

where v_0 means the threshold value.

Then, the evaluation function is defined as follow:

$$E = \frac{1}{2L}\sum_{l=1}^{L}\left(h(\boldsymbol{z}^l) - d(\boldsymbol{z}^l)\right)^2 \tag{7}$$

The weights $\boldsymbol{w}$ and $\boldsymbol{v}$ are updated based on BP method as follow [19]:

$$\triangle v_j = -\alpha_1 \delta_{1j}(\boldsymbol{z}^l) e_j(\boldsymbol{z}^l) \tag{8}$$

$$\triangle w_{ij} = -\alpha_2 \delta_{1j}(\boldsymbol{z}^l) z_i^l \tag{9}$$

$$(i = 0, \cdots, N, j = 1, \cdots, M)$$

where α_1 and α_2 are learning coefficients,

$$\delta_{1j}(\boldsymbol{z}) = (h(\boldsymbol{z}) - d(\boldsymbol{z}))h(\boldsymbol{z})(1 - h(\boldsymbol{z})) \tag{10}$$

and

$$\delta_{2j}(\boldsymbol{z}) = \delta_{1j} v_j e_j(\boldsymbol{z})(1 - e_j(\boldsymbol{z})). \tag{11}$$

Then, BP learning method is shown in Fig.5 [19].

III. Back propagation learning for secure multiparty computation

A. Perceptron learning for secure shared data

Let us consider a system composed of client and m servers(See Fig.1). In learning on cloud system, learning data and weight parameters are shared to each server in multiplication form. Each server updates shared weight parameters and sends the computation result to the client. The client can get new weight parameters by multiplying the results of m servers. This is verified from Eqs.(15) and (16). The process is iterated until the error (difference) between the output of the neuron and the desired output becomes sufficiently small. The problem is how the weight parameter $\boldsymbol{w}$ on the user is updated using the set of learning data shared on each server. The shared representation of learning data $\{(\boldsymbol{z}^l, d(\boldsymbol{z}^l))|l \in Z_L\}$and parameters are given as follows:

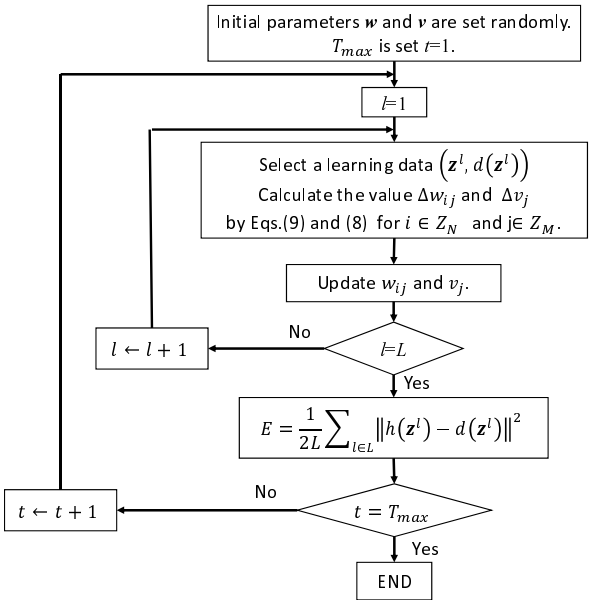

Fig. 5. BP learning for three-layered neural network.

$$\boldsymbol{z}^l = (z_1^l, \cdots, z_i^l, \cdots, z_N^l) \tag{12}$$

for $l \in Z_L$ and

$$z_i^l = \Pi_{k=1}^m (z_i^l)^k \tag{13}$$

for $i \in Z_N$,

$$d(\boldsymbol{z}^l) = \sum_{k=1}^{m} (d(\boldsymbol{z}^l))^k \tag{14}$$

Note that Eqs.(13) and (14) are in multiplication form for shared data.

In this case, Eqs.(4) and (5) are renewed as follows:

$$\triangle v_j^k = \frac{\partial E}{\partial v_j^k} = (g(\boldsymbol{z}^l) - d(\boldsymbol{z}^l)) z_j^l v_j / v_j^k \tag{15}$$

$$(v_j^k)(t+1) = (v_j^k)(t) + K \triangle v_j^k(t) \tag{16}$$

for $\boldsymbol{v} = (v_1, \cdots, v_j, \cdots, v_M)$ and $v_j = \Pi_{k=1}^m v_j^k$.

Eq.(16) means that each server can update the weight by dividing by v_j^k for the conventional method.

The learning process of the perceptron learning is shown in TableIII.

B. BP learning for secure shared data

In this section, we propose BP learning method for SMC as the same method as the section III.A. The problem is how weights $\{w_{ij}\}$ and $\{v_j\}$ are updated using learning data shared on each server. The shared representation of learning data and parameters are given as follows:

$$\boldsymbol{z}^l = (z_1^l, \cdots, z_i^l, \cdots, z_N^l) \tag{17}$$

for $l \in Z_L$,

$$z_i^l = \Pi_{k=1}^m (z_i^l)^k \tag{18}$$

TABLE III
LEARNING PROCESS OF PERCEPTRON LEARNING FOR SMC.

	Client	k-th Server
Initial condition	The weight $\boldsymbol{v} = (v_1, \cdots, v_M)$ is selected randomly, and send $v_j^k (v_j = \Pi_{k=1}^m v_j^k)$ to each server for $k \in Z_m$ and $j \in Z_M$. Set $t = 1$.	$\{(\boldsymbol{z}^e)^k, (d(\boldsymbol{z}^e))^k \mid e \in Z_L\}$
Step 1	A number l is selected randomly	$\{v_j^k\}$ for $j \in Z_M$
Step 2		Compute $\mu_j^k = v_j^k (z_j^l)^k$ for $j \in Z_M$ and send μ_j^k and $(d(\boldsymbol{z}^l))^k$ to Client.
Step 3	Compute $\mu_j = \Pi_{k=1}^m \mu_j^k$ for $j \in Z_M$, and $z_j v_j = \Pi_{k=1}^m (z_j^l)^k v_j^k$	
Step 4	Compute $\phi = Kg(\sum_{i=1}^N \mu_i) z_j v_j$-$K \sum_{k=1}^m \left(d(\boldsymbol{z}^l)\right)^k (z_j^l)^k v_j^k$ and send ϕ to each server	
Step 5		Compute $\triangle v_j^k = \phi / v_j^k$ and $v_j^k \leftarrow v_j^k + \triangle v_j^k$ for $j \in Z_M$ and send them to Client
Step 6	If $t \neq T_{max}$ then go to Step 1 with $t \leftarrow t+1$ else the algorithm terminates.	

for $i \in Z_N$,

$$d(\boldsymbol{z}^l) = \sum_{k=1}^m (d(\boldsymbol{z}^l))^k \quad (19)$$

$$\boldsymbol{v} = (v_1, \cdots, v_j, \cdots, v_M) \quad (20)$$

$$v_j = \Pi_{k=1}^m v_j^k \quad (21)$$

$$\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_j, \cdots, \boldsymbol{w}_M) \quad (22)$$

$$\boldsymbol{w}_j = (w_{1j}, \cdots, w_{ij}, \cdots, w_{Nj}) \quad (23)$$

$$w_{ij} = \Pi_{k=1}^m (w_{ij})^k \quad (24)$$

Further, Eqs. (8) and (9) are renewed as follows:

$$\triangle v_j^k = -\alpha_1 \delta_{1j}(\boldsymbol{z}^l) v_j e_j(\boldsymbol{z}^l) / v_j^k \quad (25)$$

$$\triangle w_{ij}^k = -\alpha_2 \delta_{2j}(\boldsymbol{z}^l) w_{ij} x_i^l / w_{ij}^k \quad (26)$$

Eqs.(25) and (26) mean that each server can update the weight by dividing by v_j^k and w_{ij}^k in the proposed method.

The learning process of the three-layered neural network is shown in TalbeIV.

IV. NUMERICAL SIMULATIONS FOR THE PROPOSED ALGORITHM

In this section, numerical simulations of function approximation and pattern classification for conventional and proposed methods are performed. The conventional method means the conventional BP method without sharing data and the proposed method is one with $m = 3$ and $m = 10$. In this case, the multiplication form is used as $a = A^{(1)} \cdot \cdots \cdot A^{(m)}$, where $A^{(k)}$ for $1 \leq k \leq m-1$ is a random number in $[0, 1]$ or $[-1, 1]$, and $A^{(m)} = a / (A^{(1)} \cdot \cdots \cdot A^{(m-1)})$.

A. Function Approximation

This simulation uses three systems specified by the following functions with 4-dimensional input space $[0, 1]^4$ (for Eq.(27)) and $[-1, 1]^4$ (for Eqs.(28) and (29)). The simulation condition is shown in Table V. The numbers of learning and test data randomly selected are 512 and 6400, respectively.

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{37.21} \times \frac{(4\sin(\pi x_3) + 2\cos(\pi x_4) + 6}{12} \quad (27)$$

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{74.42} + \frac{4\sin(\pi x_3) + 2\cos(\pi x_4) + 6}{446.52} \quad (28)$$

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{74.42} + \frac{(3e^{3x_3} + 2e^{-4x_4})^{-0.5} - 0.077}{4.68} \quad (29)$$

Table VI shows the results of comparison of accuracy between the conventional and the proposed methods. In each box of Table VI, three numbers from the top to the bottom show MSE of training ($\times 10^{-4}$), MSE of test ($\times 10^{-4}$) and the number of parameters, respectively. The result of simulation is the average value from twenty trials.

The result shows that the accuracy between the conventional and the proposed methods is almost the same. Further, it is shown that the results for $m = 3$ and $m = 10$ is also about the same accuracy.

Next, we will compare the learning speed of the conventional method with one of proposed method using numerical simulation of Fig.(27). The simulation condition is the same as Table V. Fig. 6 shows the graph of MSE for learning time. The result shows that learning speed of the proposed methods is fast compared to the conventional method. In another functions, the same results are also shown.

B. Pattern Classification

Let us show the result for pattern classification using benchmark problems of Iris, Wine, Sonar and BCW in UCI database [21]. See Table VII.

In this simulation, 5-fold cross validation as an evaluation method is used: In 5-fold cross validation, all data are randomly partitioned into 5 equal size subsets. Of the 5 subsets, a single subset is kept as data for testing the model, and the remaining 4 subsets are used as training data. The cross validation process is repeated 5 times (the folds) with each of 5 subsets used exactly once as the validation (test)

TABLE IV
LEARNING PROCESS OF BP FOR SMC.

	Client	k-th Server
Initial condition	The weight $\boldsymbol{v} = (v_1, \cdots, v_M)$ and $\boldsymbol{w}_j = (w_{1j}, \cdots, w_{ij}, \cdots, w_{Nj})$ for $j \in Z_M$ are selected randomly, $v_j = \Pi_{k=1}^m v_j^k$ and $w_{ij} = \Pi_{k=1}^m w_{ij}^k$. Send v_j^k and w_{ij}^k to each server.	$\{(\boldsymbol{z}^e)^k, (d(\boldsymbol{z}^e))^k \mid e \in Z_L\}$
Step 1	A number l is selected randomly and send it to Server.	$\{w_{ij}^k, v_j^k\}$ for $\in Z_N$ and $j \in Z_M$
Step 2		Compute $\mu_{ij}^k = w_{ij}^k (z_i^l)^k$ for $i \in Z_N$ and $j \in Z_M$. Send them to Client
Step 3	Compute $w_{ij} z_i = \Pi_{k=1}^m \mu_{ij}^k$, $h_j = \sum_{i=0}^L w_{ij} z_i$ and $y_j = e(h_j)$	
Step 4	Send y_j^k for $k \in Z_m$ to each setver, where $y_j = \Pi_{k=1}^m y_j^k$	
Step 5		Compute $\phi_j^k = v_j^k y_j^k$ for $j \in Z_L$ and send it to Client
Step 6	Compute $v_j y_j = \Pi_{k=1}^m \phi_j^k$, $z_0^l = \sum_{j=0}^M v_j y_j$ and $z^l = g(z_0^l)$	
Step 7	Compute $\phi_1 = g(\boldsymbol{z}^l)^2 (1 - g(\boldsymbol{z}^l)) v_j y_j$ $-g(\boldsymbol{z}^l)(1 - g(\boldsymbol{z}^l)) \sum_{k=1}^m \left(d(\boldsymbol{z}^l)\right)^k v_j^k y_j^k$ and $\phi_2 = g(\boldsymbol{z}^l)^2 (1 - g(\boldsymbol{z}^l)) v_j y_j (1 - y_j) w_{ij} z_i^l$ $-g(\boldsymbol{z}^l)(1 - g(\boldsymbol{z}^l)) v_j y_j (1 - y_j) \sum_{k=1}^m \left(d(\boldsymbol{z}^l)\right)^k w_{ij} z_i^l$ and send them to each server	
Step 8		Compute $v_j^k \leftarrow v_j^k + K_1 \phi_1 / v_j^k$ and $w_{ij}^k \leftarrow w_{ij}^k + K_2 \phi_2 / w_{ij}^k$ and send them to Client
Step 9	If $t \neq T_{max}$ then go to Step 1 with $t \leftarrow t + 1$ else the algorithm terminates.	

TABLE V
THE INITIAL CONDITIONS FOR SIMULATIONS OF FUNCTION APPROXIMATION

	conventional method	proposed method
K_w	0.01	0.01
K_v	0.01	0.01
T_{max}	50000	50000
Initial w_{ij}	random on [0,1]	
Initial v_i	random on [0,1]	

TABLE VI
RESULT OF SIMULATION OF FUNCTION APPROXIMATION

		Eq.(27)	Eq.(28)	Eq.(29)
conventional method	Learn	0.80	2.06	1.22
	Test	1.00	2.66	1.41
	#Para	121	121	121
proposed method ($m = 3$)	Learn	0.21	0.52	0.27
	Test	0.38	1.02	0.41
	#Para	333	333	333
proposed method ($m = 10$)	Learn	0.35	0.44	0.99
	Test	0.54	0.63	1.63
	#Para	1210	1210	1210

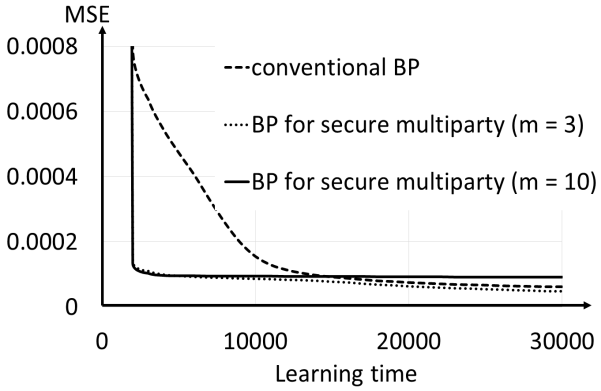

Fig. 6. The comparison of learning speed between conventional and proposed BP methods for a function approximation problem

data. The 5 results from the folds can then be averaged to produce a single estimation.

Table VIII shows the results of comparison between the conventional and the proposed methods. In each box of Table VIII, three numbers from the top to the bottom show the rate of misclassified data for training and test(%) and the number of parameters, respectively. Each value is average from five trials.

TABLE VII
THE DATASET FOR PATTERN CLASSIFICATION

	Iris	Wine	Sonar	BCW
The number of data	150	178	208	683
The number of input	4	13	60	9
The number of class	3	3	2	2

The result shows that accuracy between the conventional and the proposed methods is almost same. Further, it is shown that the results for $m = 3$ and $m = 10$ are also about the same accuracy.

TABLE VIII
RESULT OF SIMULATION OF PATTERN CLASSIFICATION

		Iris	Wine	Sonar	BCW
conventional method	Learn	3.7	8.8	9.2	0.8
	Test	4.0	13.7	18.5	4.3
	#Para	121	301	1241	221
proposed method ($m = 3$)	Learn	0.6	0.0	1.3	0.8
	Test	9.6	5.0	19.5	3.8
	#Para	363	903	3723	663
proposed method ($m = 10$)	Learn	3.5	0.1	2.8	1.3
	Test	4.0	8.3	21.0	3.8
	#Para	1210	3010	12410	2210

V. CONCLUSION

The secure multi-party computation(SMC) is one of secure sharing and computing methods. The SMC with secure shared data in addition and multiplication forms is known, but complex computation processing such as BP learning has never proposed yet. In this paper, we proposed a learning method for SMC on cloud computing system and proved the validity of it. Further, the performance of the proposed method was shown in numerical simulations.

The idea of our study is to perform privacy preserving data mining as "shared data + parallel algorithm". That is, it is to find the representation of shared data and to construct parallel algorithm. Specifically, we introduced BP learning algorithm(as one of data mining) using the representation of data in multiplication form.

In the future work, we consider the improved method to reduce the computation of client and develop AUI(Application User Interface) for the client.

REFERENCES

[1] C. C. Aggarwal, and P. S. Yu, "Privacy-Preserving Data Mining: Models and Algorithms", ISBN 978-0-387-70991-8, Springer-Verlag, 2009.

[2] S. Subashini, and V. Kavitha, "A survey on security issues in service delivery models of cloud computing", J. Network and Computer Applications, Vol.34,pp.1-11, 2011.

[3] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices", STOC2009, pp.169-178, 2009.

[4] HElib, "An Implementation of homomorphic encryption", https://github.com/shaih/HElib

[5] A. Shamir, "How to share a secret", Comm. ACM, Vol. 22, No. 11, pp. 612-613, 1979.

[6] A. Beimel, "Secret-sharing schemes: a survey", in Proc. of the Third international conference on Coding and cryptology (IWCC 11), 2011.

[7] R. Canetti, et al., "Adaptively secure multi-party computation", STOC' 96, pp. 639-648, 1996.

[8] R. Cramer, et al., "General secure multi-party computation from any linear secret-sharing scheme", EUROCRYPT' 00, 2000.

[9] A. Ben-David, et al., "Fair play MP: a system for secure multi-party computation", ACM CCS' 08, 2008.

[10] J. Yuan, S. Yu, "Privacy Preserving Back-Propagation Neural Network Learning Made Practical with Cloud Computing", IEEE Trans. on Parallel and Distributed Systems, Vol.25, Issue 1, pp.212-221, 2013.

[11] N. Schlitter, "A Protocol for Privacy Preserving Neural Network Learning on Horizontal Partitioned Data", Privacy Statistics in Database(PSD), 2008.

[12] T. Chen, S. Zhong, "Privacy-Preserving Back Propagation Neural Network Learning", IEEE Trans. on NN, Vol.20, Issue 10, pp.1554-1564, 2009.

[13] K. Chida, et al., "A Lightweight Three-Party Secure Function Evaluation with Error Detection and Its Experimental Result", IPSJ Journal Vol. 52 No. 9, pp. 2674-2685, Sep. 2011(in Japanese).

[14] N. Rajesh, K. Sujatha, A. A. Lawrence, "Survey on Privacy Preserving Data Mining Techniques using Recent Algorithms", International Journal of Computer Applications Vol.133, No.7, pp.30-33, 2016.

[15] S. S. Rathna, T. Karthikeyan, "Survey on Recent Algorithms for Privacy Preserving Data mining", International Journal of Computer Scienceand Information Technologies, Vol. 6 (2), pp. 1835-1840, 2015.

[16] O. Catrina, et al., "Secure multiparty linear programming using fixed point arithmetic", ESORICS 2010, 2010.

[17] Y. Miyanishi, A. Kanaoka, F. Sato, X. Han, S. Kitagami, Y.Urano, N. Shiratori, "New Methods to Ensure Security to Increase User's Sense of Safety in Cloud Services", Proc. of The 14th IEEE Int. Conference on Scalable Computing and Communications (ScalCom-2014), pp.859-865, Bali, Dec.2014.

[18] H. Miyajima, N. Shigei, H. Miyajima, Y. Miyanishi, S. Kitagami, N. Shiratori, "A Proposal of Back Propagation Learning for Secure Multi-party computation methods", Lecture Notes in Engineering and Computer Science : Proc. of IMECS 2016, IMECS 2016, 16-18 March, 2016, Hong Kong, Vol.I, pp.381-386.

[19] M. M. Gupta, L. Jin and N. Homma, "Static and Dynamic Neural Networks", IEEE Press, 2003.

[20] H. Miyajima, N. Shigei, H. Miyajima, "Performance Comparison of Hybrid Electromagnetism-like Mechanism Algorithms with Descent Method", JAISCR,vol.5, no.4, 2015.

[21] UCI Repository of Machine Learning Databases and Domain Theories, ftp://ftp.ics.uci.edu/pub/machinelearning-Databases.