ORIGINAL RESEARCH

Proposal of security preserving machine learning of IoT

Hirofumi Miyajima*[1], Norio Shiratori[2], Hiromi Miyajima[3]

[1] Okayama University of Science, Japan
[2] Chuo University, Tokyo, Japan
[3] Former Kagoshima University, Kagoshima, Japan

Received: June 10, 2018 **Accepted:** July 16, 2018 **Online Published:** August 9, 2018
DOI: 10.5430/air.v7n2p26 **URL:** https://doi.org/10.5430/air.v7n2p26

Abstract

The use of cloud computing system, which is the basic technology supporting ICT, is expanding. However, as the number of terminals connected to it increases, the limit of the capability is also becoming apparent. The limit of its capacity leads to the delay of significant processing time. As an architecture to improve this, the edge computing system has been proposed. This is known as a new paradigm corresponding the conventional cloud system. In the conventional cloud system, a terminal sends all data to the cloud and the cloud returns the result to the terminal or a thing directly connected to it. On the other hand, in the edge system, a plural of servers called edges are connected directly or to close distance between the cloud and the terminal (or thing). Then, let us consider the case of machine learning that requires big data. The purpose of learning is to find out the relationship (information) lurking in from the collected data. In order to realize this, a system with several parameters is assumed and estimated by repeatedly updating the parameters with learning data. Further, there is the problem of the security for learning data. In other words, users of cloud computing cannot escape the concern about the risk of information leakage. How can we build a cloud computing system to avoid such risks? Secure multiparty computation is known as one method of realizing safe computation. It is called SMC (Secure Multiparty Computation). Many studies on learning methods considering on SMC have also been proposed. Then, what kind of learning method is suitable for edge computing considering on SMC? In this paper, learning method suitable for edge computing considering on SMC is proposed. It is shown using an edge system composed of a client and m servers. Learning data are shared m pieces of subsets for m servers, learning is performed simultaneously in each server and system parameters are updated in the client using their results. The idea of learning method is shown using BP algorithm for neural network. The effectiveness is shown by numerical simulations.

Key Words: IoT, Machine learning, Security, Batch learning

1. Introduction

The use of cloud computing system, which is the basic technology supporting ICT, is expanding. However, as the number of terminals connected to it increases, the limit of the capability is also becoming apparent. The limit of its capability leads to the delay of significant processing time delay.[1–4] According to Cisco Comp., it is estimated that by 2019 data generated similar to people, machines and things will reach 500 Zettabytes. On the other hand, IP traffic of the Global Center is assumed to be only 10.4 Zettabytes by the same time.[4] In particular, in the field requiring online or fast processing such as automatic driving, this brings about fatal

*Correspondence: Hirofumi Miyajima; Email: miya@mis.ous.ac.jp; Address: Okayama University of Science, 1-1 Ridaicho, Kitaku, Okayama, Japan.

consequences. As an architecture to improve this, the edge computing system has been proposed.[2–4] It is desired as a new paradigm for IoT.[3,4] In the conventional cloud system, a terminal (or thing) sends all data to the cloud and the cloud returns the result to the terminal (or thing) directly connected to it. On the other hand, in the edge system, a plural of servers called edges are connected directly or to close distance between the cloud and the terminal (or thing).[3,4] That is, although edge system cannot use servers with high processing capability compared with the cloud system, it seems that high processing capability can be realized by efficiently combining a plural servers in the edge system. From the side of terminals, normal tasks can be handled at edges, and the cloud processing is done for tasks that require large computing power. With the push from cloud severs and pull from IoT, the edge of the network is changing from data consumer to data producer as well as consumer.[4,5] Then, what kind of paradigm for machine learning with the edge computing is needed? The purpose of learning is to find out the relationship (information) lurking in from the collected data. In order to realize this, a system with several parameters is assumed and estimated by repeatedly updating the parameters with learning data. Various solutions for learning have been proposed.[5,6] On the other hand, users of cloud or edge computing cannot escape the concern about the risk of information leakage. How can we build cloud or edge computing system to avoid such risks? One way to achieve this goal is data encryption. Data encryption is an effective way to protect data from risk, but data must be repeatedly encrypted and decrypted each time data processing is done. Therefore, a safe system using distributed processing has attracted attention, and a lot of studies with cloud have been done.[7,8] As a method for realizing distributed processing, secure multiparty computation which is currently applied to several application is known. It is called SMC (Secure Multiparty Computation). Many studies on learning methods of SMC have also been proposed.[5,8–10] So, what kind of learning method is suitable for edge computing considering on SMC?

In this paper, learning method suitable for edge computing considering on SMC is proposed. That is, it is shown that batch learning method is one learning method suitable for edge system. The effectiveness of the idea is shown using BP algorithm of neural networks.

2. Preliminary

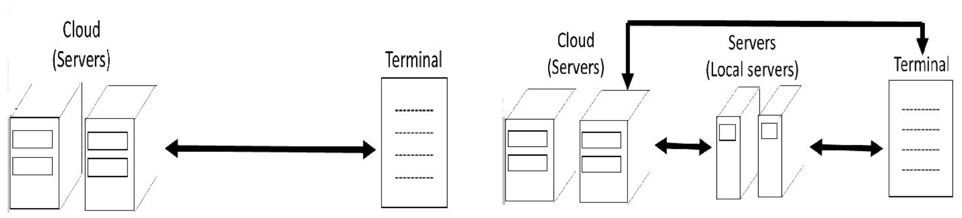
2.1 A configuration of edge computing system

The purpose of the edge system is how to combine multiple servers with low capabilities to build a system with high processing capability. In this paper, the following four items as processing capacity are considered.

(1) High speed
(2) Distributed hardware amount like memory
(3) Reduction of communication time
(4) Security preserving

In the following, a method to efficiently realize machine learning using big data on edge system is proposed.

Figure 1 shows a system of edge computing used in this paper. The system is composed of the terminals (or things) directly connected to the cloud and a plural servers (called edges) connected directly or to close distance between the cloud and the terminal (or things).



(a) Conventional system (b) Edge system

Figure 1. An example of secure shared data for $m = 2$

Each server is connected directly to each terminal (or thing). The client (user) sends data to each server and remembers them on the server. When a client needs data processing, each server performs a part of the calculation used by the client and sends it to the client. The client computes the result using them. If the result cannot be obtained in one process, data communication and calculation between the client and the cloud are repeated until the final result is obtained. The problem is how to share and distribute data between the client and the server in order to execute high-speed calculation while maintaining security.

In the conventional cloud system, each terminal that needs calculation processing sends all data to the cloud and the cloud returns the result to the terminal directly connected to the cloud (see Figure 1(a)). On the other hand, in the edge system, a plural servers called edges are connected directly or to close distance between the cloud and the terminal. Each data of terminal is sent to each edge (server) and data processing is performed in the edge system (see Figure 1(b)).

2.2 Gradient descent method in machine learning

The purpose of machine learning is to give a method to realize the input/output relation of given learning data by the parameters of one system. Normally, since appropriate parameters can not be found directly, parameters are estimated by sequentially updating the parameters based on SDM (Steepest Descent Method) or GDM (Gradient Descent Method). Applications of SDM include BP (Back Propagation) learning method of neural network, unsupervised learning like K-means, NG (Neural Gas) and SOM (Self Organization Mapping) and fuzzy modeling, etc. It is used for many problems.[5–7]

Gradient descent method is a way to minimize an objective function $J(\boldsymbol{\theta})$ parameterized by a model's parameters $\boldsymbol{\theta}\in R^d$ by updating the parameters in the opposite direction of the gradient of the objective function $\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ to the parameters, where R is the set of all real numbers. The learning rate η determines the size of the steps we take to reach a (local) minimum. The method is performed based on the following equations[7] :

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \eta \nabla J(\boldsymbol{\theta}) \tag{1}$$

That is, the parameter $\boldsymbol{\theta}$ is updated based on Eq.(1) using learning data in order to reach a minimum. There are three methods based on how to use learning data, online, mini batch and batch. In the following, let us explain the mini-batch method.

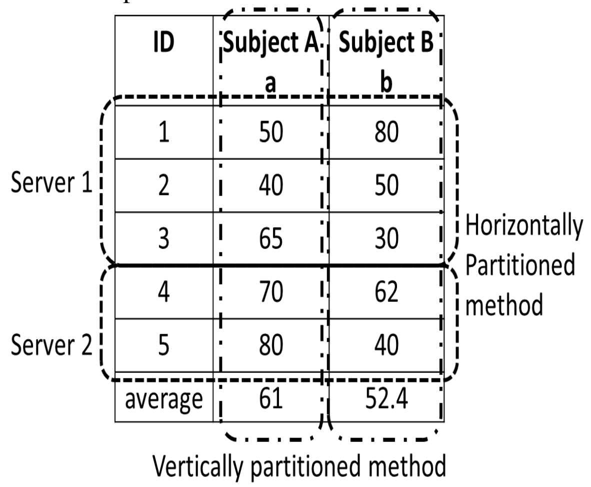
Let D be the set of learning data. The set D is composed of L subsets such as $D = \bigcup_{l=1}^{L} B_l$ and $B_i \cap B_j = \varnothing$ for $i \neq j \in Z_L$ where $|B_l| = b_l$ for $l \in Z_L$, $|D| = \sum_{l=1}^{L} b_l$, and $Z_i = \{1, 2, \cdots, l\}$. Let $t = 1$.
Step 1: The sets B_l is given.
Step 2: Update $\boldsymbol{\theta}$ based on Eq.(1) for B_l.
Step 3: If $\nabla J(\boldsymbol{\theta})$ is not sufficiently small, then go to Step 1 with $t \leftarrow t+1$ else algorithm terminates.

If $L = 1$ and $L = |D|$, then the methods are called online and batch ones, respectively.

2.3 System configuration of secure shared data for SMC

Let us consider about conventional works with secure shared data for SMC. In order to solve the problem, three partitioned representation of data such as horizontally, vertically and any partitioned methods for SMC are known. Let us explain about them using an example of Table 1.[5] 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.

Table 1. Concept of horizontally and vertically partitioned methods composed of one client and two servers


Server 1
Server 2
Horizontally Partitioned method
Vertically partitioned method

ID	Subject A a	Subject B b
1	50	80
2	40	50
3	65	30
4	70	62
5	80	40
average	61	52.4

First, let us explain about horizontally partitioned method using Table 1.[7, 10] All the dataset are shared into two servers, Server 1 and 2 as follows:

Server 1: dataset for ID=1, 2, 3,

Server 2: dataset for ID=4,5.

In Server 1, each average for A or B is computed as $(50+40+65)/3$ and $(80+50+30)/3$, respectively. In Server 2, each average for A or B is computed as $(70+80)/2$ 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 security preserving holds. In effect, it is possible not to use raw learning data. But, in order to understand the idea easily, raw learning data are used in the following.

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 security preserving hold.

In the following, the horizontally partitioned method is used and the proposed method is applied to BP method of neural network as an example of SDM.

2.4 Perceptron learning

Let us explain the conventional neuron and its learning method to understand BP method easily.[5] The output y is computed as

$$y = g(u) \tag{2}$$

$$u = \sum_{j=0}^{N} v_j x_j \tag{3}$$

where u is the internal potential of the neural element and $g(u)$ is the output function. v_j is the weight for the j-th input, v_0 is the threshold and $x_0 = 1$(see Figure 2).

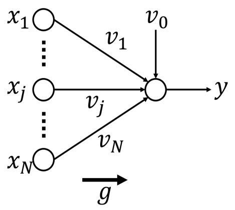


Figure 2. A neural element

Let $d(\boldsymbol{x}^l)$ be the desired output for the l-th input $\boldsymbol{x}^l$. Let the error function for learning data $D = \{(\boldsymbol{x}^l, d(\boldsymbol{x}^l)|l \in Z_L)\}$ be defined by

$$E = \frac{1}{2L}\sum_{l=1}^{L}\left(g(\boldsymbol{x}^l) - d(\boldsymbol{x}^l)\right)^2 \tag{4}$$

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

$$\triangle v_j = \frac{\partial E}{\partial v_j} = \left(g(\boldsymbol{x}^l) - d(\boldsymbol{x}^l)\right)\frac{\partial g(u)}{\partial u}x_j^l \tag{5}$$

$$v_j(t+1) = v_j(t) + \alpha \triangle v_j(t) \tag{6}$$

where $j \in Z_N$, and α is a learning coefficient.

With batch learning, the following equation is used instead of Eq.(5).

$$\triangle v_j = \sum_{l=1}^{L}\frac{\partial E}{\partial v_j} = \sum_{l=1}^{L}\left(g(\boldsymbol{x}^l) - d(\boldsymbol{x}^l)\right)\frac{\partial g(u)}{\partial u}x_j^l \tag{7}$$

2.5 BP method

Let $\boldsymbol{x}^l \in J^N$ for $l \in Z_L$ and $d : J^N \to J$, where $J = [0,1]$ or $[-1,1]$. For data $\{(\boldsymbol{x}^l, d(\boldsymbol{x}^l))|l \in Z_L\}$, let us determine the three-layered neural network identifying learning data by BP method. Let $h = g \circ e : J^N \to J$. Let M be the number of elements for the second layer. Let $\boldsymbol{w}_j$ for $j \in Z_M$ and $\boldsymbol{v}$ be weights for the second and output layer, respectively. Then g and e are as follows (see Figure 3):

$$y_i = e_i(\boldsymbol{x}) = \tau\left(\sum_{j=0}^{N} w_{ij}x_j\right),$$
$$x_0 = 1,$$
$$\tau(u) = \frac{1}{1+\exp(-u)}$$

where

$$\boldsymbol{x} = (x_1, \cdots, x_j, \cdots, x_N) \in J^N$$
$$\boldsymbol{y} = (y_1, \cdots, y_i, \cdots, y_M) \in J^M$$

and w_{0j} means the threshold value.

Further,

$$g(\boldsymbol{y}) = \tau\left(\sum_{i=0}^{M} v_i y_i\right), \tag{8}$$
$$y_0 = 1,$$

where v_0 means the threshold value.

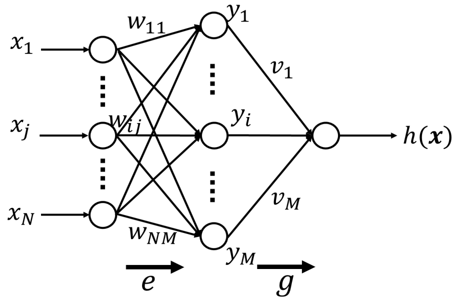


Figure 3. An example of three-layered neural network

Then, the evaluation function is defined as follow:

$$E = \frac{1}{2L}\sum_{l=1}^{L}\left(h(\boldsymbol{x}^l) - d(\boldsymbol{x}^l)\right)^2 \tag{9}$$

The weights $\boldsymbol{w}$ and $\boldsymbol{v}$ are updated based on BP method as follows[5,6] :

$$\triangle v_i = -\alpha_1 \delta_2(\boldsymbol{x}^l)e_i(\boldsymbol{x}^l) \tag{10}$$

$$\triangle w_{ij} = -\alpha_2 \delta_{1i}(\boldsymbol{x}^l)x_j^l \tag{11}$$

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

where α_1 and α_2 are learning coefficients,

$$\delta_2(\boldsymbol{x}) = (h(\boldsymbol{x}) - d(\boldsymbol{x}))h(\boldsymbol{x})(1 - h(\boldsymbol{x})) \tag{12}$$

and

$$\delta_{1i}(\boldsymbol{x}) = \delta_2(\boldsymbol{x})v_i e_i(\boldsymbol{x})(1 - e_i(\boldsymbol{x})). \tag{13}$$

Then, BP learning method is shown in Figure 4.

In batch learning, the following equations are used instead of Eqs.(10) and (11).

$$\triangle v_i = -\sum_{l=1}^{L} \alpha_1 \delta_2(\boldsymbol{x}^l) e_i(\boldsymbol{x}^l) \tag{14}$$

$$\triangle w_{ij} = -\sum_{l=1}^{L} \alpha_2 \delta_{1i}(\boldsymbol{x}^l) x_j^l \tag{15}$$

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

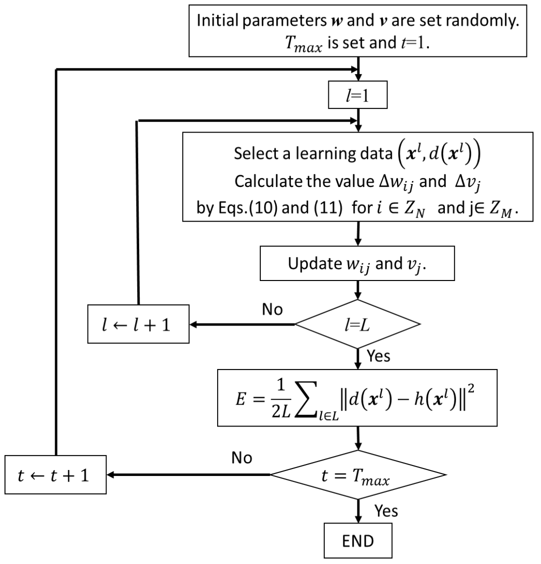

Initial parameters $\boldsymbol{w}$ and $\boldsymbol{v}$ are set randomly. T_{max} is set and t=1.
l=1
Select a learning data $(\boldsymbol{x}^l, d(\boldsymbol{x}^l))$ Calculate the value Δw_{ij} and Δv_j by Eqs.(10) and (11) for $i \in Z_N$ and $j \in Z_M$.
Update w_{ij} and v_j.
l=L
No
$l \leftarrow l+1$
Yes
$E = \frac{1}{2L}\sum_{l\in L}\|d(\boldsymbol{x}^l) - h(\boldsymbol{x}^l)\|^2$
$t = T_{max}$
No
$t \leftarrow t+1$
Yes
END

Figure 4. BP learning

3. Batch BP learning for secure multiparty computation

3.1 SMC for perceptron learning

Let us consider a system composed of a client and m servers. It is assumed that the k-th server has subset B_k for learning data, where $D = \bigcup_{k=1}^{m} B_k$ and $B_i \bigcap B_j = \phi$ for $i \neq j \in Z_m$. Each server has parameter v_j for $j \in Z_N$. The k-th server calculates the error for the set B_k and the client updates weight parameters using the result of each server.

In this case, Eq.(7) is renewed as follows:

$$\begin{aligned} \triangle v_j &= \sum_{k=1}^{m} \sum_{\boldsymbol{x}\in B_k} (g(\boldsymbol{x}) - d(\boldsymbol{x})) \frac{\partial g(u)}{\partial u} x_j \\ &= \sum_{k=1}^{m} \sum_{\boldsymbol{x}\in B_k} \frac{\partial E}{\partial v_j} \end{aligned} \tag{16}$$

Eq.(16) means that the calculation $\sum_{k=1}^{m}\sum_{\boldsymbol{x}\in B_k}$ is used instead of the calculation $\sum_{l=1}^{L}$. That is, the set D is shared horizontally to m pieces of datasets, each subset is processed at a time in each server and the result is sent to the client.

The batch learning for the perceptron learning is shown in Table 2.

Table 2. SMC for Perceptron Learning

	Client	k-th Server
Initial condition	The weight $\boldsymbol{v} = (v_1, \cdots, v_N)$ is selected randomly and sent to each server. α, T_{max} and θ are given. Set $t = 1$.	The set B_k is given
Step 1		Calculate $g(\boldsymbol{x}^l)$ for $\boldsymbol{x}^l \in B_k$ by using Eq.(3). Calculate $\triangle_j^k = \sum_{\boldsymbol{x}\in B_k}(g(\boldsymbol{x}) - d(\boldsymbol{x}))$ $g(\boldsymbol{x}^l)(1 - g(\boldsymbol{x}))x_j$ and send them to the client.
Step 2	Calculate $v_j \leftarrow v_j + \alpha\sum_{k=1}^{m}\triangle_j^k$ and send them to each server.	
Step 3		Compute $E_k(t) = \sum_{\boldsymbol{x}\in B_k}(g(\boldsymbol{x}) - d(\boldsymbol{x}))^2$ and send them to the client.
Step 4	Compute $E(t) = \sum_{k=1}^{m} E_k(t)$. If $E < \theta$ or $t \geq T_{max}$ then the algorithm terminates else go to Step 1 with $t \leftarrow t + 1$	

3.2 Batch BP learning for three-layered neural network for secure multiparty computation

In this section, a batch learning method for SMC for three-layered neural network is proposed as the same method as the section 3.1. The problem is how weights w_{ij} and v_j are updated using learning data shared on each server.

The following equations are used instead of Eqs.(14) and

(15).

$$\triangle v_i = -\sum_{k=1}^{m}\sum_{\boldsymbol{x}\in B_k} \alpha_1\delta_2(\boldsymbol{x})e_i(\boldsymbol{x}) \quad (17)$$

$$\triangle w_{ij} = -\sum_{k=1}^{m}\sum_{\boldsymbol{x}\in B_k} \alpha_2\delta_{1i}(\boldsymbol{x})x_j \quad (18)$$

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

Eqs.(17) and (18) mean that the calculation $\sum_{k=1}^{m}\sum_{\boldsymbol{x}\in B_k}$ is used instead of the calculation $\sum_{l=1}^{L}$.

The learning process of the three-layered neural network is shown in Table 3.

Batch processing takes longer time than online one, but generalization ability is known to be high.[7] The mini-batch has properties between them. The learning method of SDM for SMC by sharing learning data into several subsets as described in Chapter 2. That is, method of minimizing disclosure of data to each server is taken. This realizes confidentiality of learning data. Therefore, increasing the number of servers leads to an improvement in confidentiality.

Table 3. SMC for BP method

	Client	k-th Server
Initial condition	The weight $\boldsymbol{v} = (v_1,\cdots,v_M)$, $\boldsymbol{w} = (w_{ij},\cdots,w_{iN})$ are selected randomly and sent to each server. α, T_{max} and θ are given. Set $t = 1$.	The set B_k is given. Let the parameters v_i and w_{ij} be denoted by v_i^k and w_{ij}^k.
Step 1		Calculate $g(\boldsymbol{x})$ for $\boldsymbol{x}\in B_k$ by using Eqs.(17) and (18). Calculate $\triangle v_i^k = \sum_{\boldsymbol{x}\in B_k}(g(\boldsymbol{x}) - d(\boldsymbol{x}))g(\boldsymbol{x})(1 - g(\boldsymbol{x}))$ $e_i(\boldsymbol{x})$ and $\triangle w_{ij}^k = \sum_{\boldsymbol{x}\in B_k}(g(\boldsymbol{x}) - d(\boldsymbol{x}))$ $g(\boldsymbol{x})(1 - g(\boldsymbol{x}))e_i(\boldsymbol{x})(1 - e_i(\boldsymbol{x}))x_j$ and send them to the client.
Step 2	Calculate $v_i \leftarrow v_i + \alpha\sum_{k=1}^{m}\triangle v_i^k$ and $w_{ij}\leftarrow w_{ij} + \alpha\sum_{k=1}^{m}\triangle w_{ij}^k$ and send them to each server.	
Step 3		Compute $E_k(t) = \sum_{\boldsymbol{x}\in B_k}(g(\boldsymbol{x}) - d(\boldsymbol{x}))^2$ and send them to the client.
Step 4	Compute $E(t) = \sum_{k=1}^{m} E_k(t)$. If $E < \theta$ or $t\geq T_{max}$ then the algorithm terminates else go to Step 1 with $t\leftarrow t + 1$	

On the other hand, in the case of the online method, parameters must be updated for each learning data. Therefore, as the number of servers increases, the communication cost increases and leading to an increase in learning time. In Figure 5(a), the error for each data of each server is computed, the result is sent to client and the client updates the parameters. The new parameters are set to the same server and the same processing for another data is performed (see the thick arrows of Figure 5(a)). The processing of learning data for the next server is performed until all the processing of all learning data for the server finished. In batch processing, each server calculates the error for a subset of learning data at a time, adds these results on the client side, and updates the parameters. That is, as shown in Figure 5(b), error calculation is performed at a time, updating is performed on the client side, the result of new parameters is sent to all servers, and the next processing of learning is performed. Therefore, the increase of the number of servers realizes the reduction of confidentiality and improvement of calculation speed.

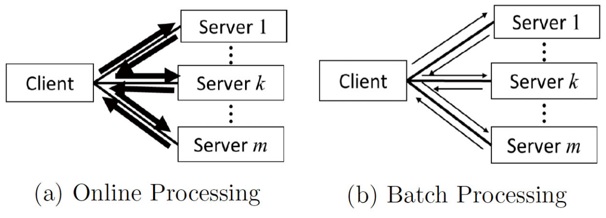


Figure 5. Online and Batch Procesing

3.3 Why is batch processing effective for IoT?

There are three methods, batch, mini-batch and online, of learning for data processing using SDM. Batch processing takes longer time than the online one, but generalization ability is known to be high. The mini-batch has properties between them.[7] In the learning method of SDM for SMC, as described in Chapter 2, by sharing learning data into several subsets, a method of security preserving is performed. In particular, batch processing as a learning method of IoT for SMC is superior to online and mini-batch as follows:

In batch processing, data sent from the terminal to the server is processed, it is without being sent to the client, and only the result is sent to the client. On the client side, the parameters of the system are updated by adding the results sent from each server. The new parameters are sent to each server and the next learning step is performed. Compared with conventional cloud systems, in edge systems, servers and client are located nearby, so calculation processing and data transmission/reception can be realized in short time. Furthermore, since the server cannot access learning data other than the self, security is preserved. In this case, by increasing the number of servers, learning by a safer and faster IoT system can be realized. Also, the high capability on each server is not always required. In the case of online processing, calculation processing for each data and updating of parameters in each server are required, so that the time needed for transmission and reception becomes longer. Also, by increasing the number of servers, the time to change the server is also added. However, compared to online processing of a conventional cloud system, fast calculation can be realized from the merit that the server is located nearby.

Since it is difficult to evaluate direct computing time, estimate the time required for parameter updating of one time for learning data. In online processing, the client sends learning parameters to the first server. The server updates parameters using (L/m) pieces of learning data, where L and m are the numbers of learning data and servers, respectively. Let computing time for updating be $O_1(L/m)$. After completion, the server returns the updated parameters to the client. The client sends the parameter to the next server and performs similar parameter update processing. In this case, the time complexity is $O(m{\cdot}O_1(L/m))$, if the communication cost between the client and the server is neglected. On the other hand, in batch processing, when the computing time of parameter updating for (L/m) pieces of learning data is assumed to be $O_2(L/m)$, the computing time for all learning data is $O(1{\cdot}O_2(L/m))$. The time complexity is $O(O_3(m)+1{\cdot}O_2(L/m))$, assuming that the computing time required for updating at the client is $O_3(m)$ by using the update amount of each server. In this case, since the first term is smaller than the second term, $O(O_3(m)+1{\cdot}O_2(L/m)){\approx}O(1{\cdot}O_2(L/m))$ is estimated. If $O_1(L/m)$ and $O_2(L/m)$ are assumed to be about the same, speedup of m times in batch processing is expected.

4. Numerical simulations for the proposed algorithm

In this section, numerical simulations of function approximation and pattern classification are performed. The BP and Batch methods mean the conventional BP and Batch methods without sharing data, respectively. The proposed method means the proposed method with $m = 10$ for SMC.

4.1 Function approximation

This simulation uses four systems specified by the following functions with $[0,1]^4$ (for Eqs.(19) and 20)) and $[-1,1]^4$ (for Eqs.(21) and (22)). The simulation conditions are $K_w = 0.01$, $K_v = 0.01$ and $T_{max} = 1,000,000$ for each method. Further, the initial values w_{ij} and v_i are set to randomly on $[0,1]$, respectively. The numbers of learning and test data randomly selected are $1,000$ and $1,000$, respectively.

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{37.21}\times\frac{(4\sin(\pi x_3)+2\cos(\pi x_4)+6}{12} \tag{19}$$

$$y = \frac{\sin(2\pi x_1)\times\cos(x_2)\times\sin(\pi x_3)\times x_4+1.0}{2.0} \tag{20}$$

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{74.42}+\frac{4\sin(\pi x_3)+2\cos(\pi x_4)+6}{446.52} \tag{21}$$

$$y = \frac{(2x_1+4x_2^2+0.1)^2}{74.42}+\frac{(3e^{3x_3}+2e^{-4x_4})^{-0.5}-0.077}{4.68} \tag{22}$$

Table 4 shows the results of comparison of accuracy for each method. In each box of Table 4, Training and Test mean MSE of training ($\times 10^{-4}$), MSE of test ($\times 10^{-4}$), respectively.

The result of simulation is the average value from twenty trials. The result shows that the accuracy for each method is almost the same. That is, the accuracy of calculation processing for SMC is held and the security is preserved.

4.2 Pattern classification

Let us show the result for pattern classification using benchmark problems of Iris, Wine, Sonar and BCW in UCI database[11] as shown in Table 5.

Table 4. Result for function approximation

		Eq.19	Eq.20	Eq.21	Eq.22
BP	Training	2.64	40.49	4.40	5.58
	Test	2.64	43.75	4.90	6.16
Batch	Training	1.00	26.45	1.06	2.30
	Test	1.17	28.08	1.18	2.68
Proposed (m = 10)	Learn	1.00	21.83	1.10	2.06
	Test	1.11	24.00	1.23	2.49

Table 5. 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

In this simulation, 10-fold cross validation as an evaluation method is used: In 10-fold cross validation, all data are randomly partitioned into 10 equal size subsets. Of the 10 subsets, a single subset is kept as data for testing the model, and the remaining 9 subsets are used as training data. The cross validation process is repeated 10 times (the folds) with each of 10 subsets used exactly once as the validation (test) data. The ten results from the folds can then be averaged to produce a single estimation.

Table 6 shows the results of comparison between the conventional and the proposed methods. In each box of Table 6, Training and Test mean the rate (%) of misclassified data for training and test, respectively. Each value is average from five trials.

Table 6. Result of simulation of pattern classification

		Iris	Wine	Sonar	BCW
BP	Training	3.60	1.96	1.29	2.39
	Test	3.83	5.33	15.81	2.79
Batch	Training	3.56	1.81	1.30	2.35
	Test	3.87	5.50	16.60	2.78
Proposed (m = 10)	Training	3.54	1.81	1.25	2.37
	Test	3.93	5.17	16.45	2.88

The result shows that accuracy between the conventional and the proposed methods is almost same. Therefore, the proposed method has good accuracy and security preserving.

5. Conclusion

In this paper, a method of security preserving machine learning for IoT was proposed and its effectiveness was shown by numerical simulations. In Section 2, cloud and edge computing systems, a secure data sharing mechanism used in this paper was explained. Further, the conventional BP method was introduced. In Section 3, learning method suitable for edge computing considering on SMC was proposed. In section 4, numerical simulations were performed to show the performance of the proposed method. Generally speaking, it becomes as follows: It was shown using an edge system composed of a client and m servers. Learning data were shared m pieces of subsets for m servers, learning was performed simultaneously in each server and system parameters were updated in the client using their results. The processing was iterated until sufficient results were obtained. The effectiveness of the idea was shown using BP algorithm of neural network. In the future, we will consider other application for SDM and another method for data sharing.

References

[1] Al-Fuqaha A, Guizani M, Mohammadi M, et al. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. IEEE Communication Surveys & Tutorials. 2015; 17(4): 2347-2376. `https://doi.org/10.1109/COMST.2015.2444095`

[2] Kitagami S, Suganuma T, Ogino T, et al. Proposal of a Multi-agent Based Flexible IoT Edge Computing Architecture Harmonizing Its Control with Cloud Computing. The Fifth International Symposium on Computing and Networking (CANDAR2017). 2017 November.

[3] Abdelshkour M. IoT, from Cloud to Fog Computing. Available from: `http://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing`. Mar.2015 (accessed 18 Jun.2017).

[4] Lopez PG, Montresor A, Epema D, et al. Edge-centric Computing: Vision and Challenges. ACM SIGCOMM Computer Communication Review. 2015 Oct.; 45(5): 37-42. `https://doi.org/10.1145/2831347.2831354`

[5] Miyajima H, Shigei N, Miyajima H, et al. New Privacy Preserving Back Propagation Learning for Secure Multiparty Computation. IAENG International Journal of Computer Science. 2016; 43(3): 270-276.

[6] Gupta MM, Jin L, Homma N. Static and Dynamic Neural Networks. IEEE Press. 2003.

[7] Ruder S. An Overview of Gradient Descent Optimization Algorithms. Available from: `http://ruder.io/optimizing-gradient-descent/`. 2016 (accessed 14 Mar. 2018).

[8] Aggarwal CC, Yu PB. Privacy-Preserving Data Mining: Models and Algorithms. ISBN 978-0-387-70991-8, Springer-Verlag, 2009.

[9] Yuan J, Yu S. Privacy Preserving Back-Propagation Neural Network Learning Made Practical with Cloud Computing. IEEE Trans. on Parallel and Distributed Systems. 2013; 25(1): 212-221.

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

[11] UCI Repository of Machine Learning Databases and Domain Theories. Available from: `ftp://ftp.ics.uci.edu/pub/machinelearning-Databases`