Vol.2020-DPS-185 No.13
2020/12/22

セキュアマルチパーティ計算によるエッジシステム上のBP学習法の提案

宮島 洋文[1,a)]　重井 徳貴[2,b)]　宮島 廣美[2,c)]　白鳥 則郎[3,d)]

概要：クラウドシステムのサーバの負担軽減の観点から, エッジシステムが提案されている. エッジステム上で安全に機械学習を行ういくつかのモデルが提案されている. その多くは, データを部分集合に分割して, 部分計算とその統合を繰り返すものであり, データ自身を使って更新を行う. 本稿では, 秘匿性を高めるため, あらかじめデータ自身を分解しておき, これら分解したデータを使った BP 学習法を提案する. また, 数値シミュレーションで, その有効性を示す.

Proposal of BP learning with IoT using Secure Multiparty Computation

1. はじめに

ICT 技術の 1 つとして, クラウドコンピューティングが広く用いられている. このクラウドコンピューティングは能力の高いサーバを複数のユーザ (クライアント) が利用するシステムであり, 運用コストを低く抑えることが可能である. 従来のクラウドコンピューティングにおいては, データの管理および計算処理はサーバ側で一括して行う. 一方で, IoT(Internet of Things) システムへの移行に伴って, サーバに接続するクライアントの数が多くなり, サーバの負担が増えて処理能力が低下することが知られている. この問題を解決する方法の一つとして, エッジコンピューティングが提案されている [1], [2], [3]. 従来のクラウドシステムにおいては, サーバは複数のクライアントから送られてきたデータを一括してすべて保管する. さらに, サーバ内部のデータを用いた計算を行い, 結果をクライアントに送る. このように, サーバ側の負担が大きなものとなる. 一方, エッジコンピューティングにおいては, エッジと呼ばれるサーバがクラウドとクライアント（端末や”もの”）間の近い距離に接続される. すなわち, エッジがショートカットとしての機能をはたす. エッジはクラウドシステムのサーバほど高い能力をもたないが, 他のサーバやエッジと組み合わせることによりシステム全体として高い能力の実現をはかる. 問題は, いかにして能力の低いエッジを効果的に組み合わせるかである. 特に, 近年では機械学習のように非常に多くの学習データを処理する問題を扱うことが多く, エッジを組み合わせた能力の高いシステムの実現が期待される. この問題はまた, クラウドシステムと同様, データの秘匿性をいかにして実現するかも問題となる. エッジシステムに関する研究には, 1)Aggregator を使うエッジステム, 2) データ圧縮を使う ProtoNN, 3) エッジシステムに対する Tree-based の機械学習, 4）Federated learning 等が知られている [3], [4]. 特に, 1) と 4）については, data 集合を複数の部分集合に分割する水平データ分割方式 (Horizontally Partitioned Data : HPD) や垂直データ分割方式 (Vertically Partitioned Data : VDP) のデータ構造による機械学習と捉えることができる [5]. いずれの場合も, 学習には個々のデータを使って, 計算処理を行う方法であり, データ漏洩の可能性は残る. これに対し, 筆者らの方法は, 予め個々のデータを乱数を使って複数に分解しておき, 学習は分解されたデータを使って実行する [5]. すなわち, この方法では, 学習データそのものを使うことなく, 分解されたデータによる学習となる. この方法ではまた, 目的とする学習パラメータ自身も分解されたデータとして処理できることから, その秘匿性を同時に保つことができる.

[1] 長崎大学
Nagasaki University, Nagasaki 852–8521, Japan
[2] 鹿児島大学
Kagoshima University
[3] 中央大学
Chuo University
[a)] miyajima@nagasaki-u.ac.jp
[b)] shigei@eee.kagoshima-u-ac.jp
[c)] k2356323@kadai.jp
[d)] norio@shiratori.riec.tohoku.ac.jp

本稿では, はじめに簡易秘匿計算が可能なデータ構造を導入する. このデータ構造に基づいた機械学習の並列計算をBP 学習を例として提案する. また, 数値実験により, これらの提案手法の有効性を示す

2. 予備概念

2.1 秘匿データによる機械学習の概念

今日のユビキタスネットワーク社会は, "いつでも, どこでも, だれとでも" 情報のやりとりを実現した社会である. 加えて, 昨今の AI 技術がこれを進歩させ, 大規模データから様々な知識が提供される社会となっている [6]. 一方で, ユーザはデータの提供方法にはほとんど関与できず, システム側に一方的にデータを提供するのが現状である. AI 技術がこれからも, 情報社会の基盤を支えるインフラとなるには, 利用者に対する安全や安心感を与える技術に進化していかなくてはならない. 言い換えると, AI 技術がユビキタスネットワーク社会を進展させる第 3 の柱となるには, データからいかにして安心かつ安全に知識を提供する仕組みを構築するかが大切である. この問に対する一つの解として, ユーザが提供するデータを秘匿したまま学習を実行するシステムの開発が行われている [7].

それでは, 現状のシステムについて考えてみよう. ユビキタスネットワーク社会を支えるクラウドシステムのサーバの負担軽減の観点から, エッジシステムが提案されている. 先に述べたように, 問題は, いかにしてこれら能力の低いエッジを効果的に組み合わせるかである.

それでは, 生の学習データを使わずに秘匿性を保ったままの機械学習法とは, どのようなものであろうか? 暗号をそのまま使った学習方法が開発されれば, この問題は解決できるが, 現状ではそのような一般的な方法は開発されていない. そこで筆者らは, 予め個々のデータを乱数を使って複数に分解しておき, 学習は分解されたデータを使って実行する方法を提案している [5]. この方法は SMC に基づく簡易秘匿計算を使った学習方法であり, データそのものを使うことなく, 分解されたデータのみを使って学習を行うので, 秘匿性は高くなる. その意味で, 従来の学習データを使う学習から, 秘匿データ（学習データそのものでない仮のデータ）を使った学習方法開発への道を開いたと言える. 提案法の長所としては, 学習に学習データそのものを用いないことによる安心感であり, 短所としては, 分解して学習することによる計算量の増加が挙げられる.

本稿では, エッジシステムへの応用について説明するが, クラウドシステムへの応用も可能である.

2.2 簡易秘密計算法 (SMC)

本稿で用いる簡易秘密計算法 (Secure Multiparty Computation : SMC) について説明する [5]. 図 1 のような L 個の端末と $Q+1$ 個のサーバからなる分散処理方式のモデルを使って, 実数データ x の関数 $f(x)$ を計算する場合について説明する (文献 [2] では, サーバ 0 は aggregator と呼ばれている). いま, L 個の端末の一つから実数データ x が分解されて各サーバに送られる場合を考える.

はじめに, 1 つの端末から与えられた実数データ x を Q 個のランダムな実数に分解する. q 番目のサーバには, データ x の一部である x^q が送られる. ここに, $x=\sum_{q=1}^{Q} x^q$ とする.

q 番目のサーバで $f_q(x^q)$ を計算し, サーバ 0 に結果を送る. ここに, $f_q(\cdot)$ は q 番目のサーバが行う計算処理とする. サーバ 0 では, 結果 $f_q(x^q)$ を統合して最終結果 $\odot_{q=1}^{Q} f_q(x^q)$ を得る. 1 回の処理で結果が得られない場合は, サーバ 0 の結果が各サーバに送られて, 同様の過程が繰り返される.

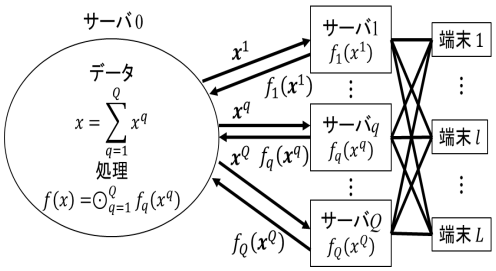

図 1 SMC の提案モデル

2.3 簡易秘密計算法と従来法によるデータ分割

はじめに, SMC の従来法として知られる HPD, VPD および提案法のデータ構造を例を使って説明する. ここでは, 表 1 に示す 2 つの項目からなる 4 個のデータの平均値を求める (表 1). HPD において, サーバ 1 にはデータ 1 と 2, サーバ 2 にはデータ 3 と 4 が保管されているとする. このとき, サーバ 1 でデータ 1 と 2 の平均値 3 と 5, サーバ 2 でデータ 3 と 4 の平均値 3 と 5 が計算される. サーバ 0 側においてこれらの計算結果の平均値 $(3+5+3+5)/4$ を求める. また, VPD では, サーバ 1 にはデータ 1 から 4 の項目 1, サーバ 2 にはデータ 1 から 4 の項目 2 が保管されている. このとき, サーバ 1 で項目 1 の平均値 5, サーバ 2 で項目 2 の平均値 3 が計算される. サーバ 0 においてこれらの計算結果の平均値 $(5+3)/2$ を求める. HPD では, サーバ 1 はデータ 1 と 2, サーバ 2 はデータ 3 と 4 のみを知ることができる. VPD では, サーバ 1 はデータの項目 1, サーバ 2 は項目 2 のみを知ることができる.

次に, 簡易秘密計算法により, 平均値を求める場合について説明する. 項目 1 の各データを分割する. データ 1 は $5+(-4)$ として 2 つのデータ 5 と-4 に分解する場合を考える. 分解された数値 5 はサーバ 1, 数値-4 はサーバ 2 に保管する. 同様に, データ 2, 3, 4 はそれぞれ $7=2+5$, $4=3+1$, $8=6+2$ と分解され, それぞれサーバ 1 とサーバ 2 に保管される. このとき, サーバ 1 とサーバ 2 では, それぞれ格納されている項目 1 の分割データの平均値

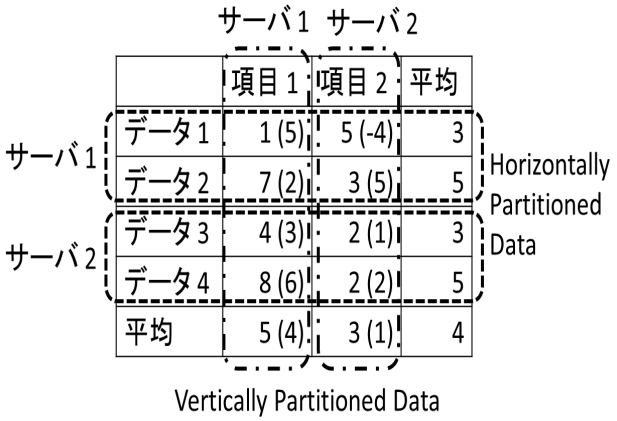

表 1 VPD, HPD, SMC の例：( ) 内の数値は, 項目 1 の各データを分割して各サーバに記憶することを示す.

$(5+2+3+6)/4=4$, $(-4+5+1+2)/4=1$ が導出される (表 1 の (·) の数値参照). サーバ 0 においてこれらの計算結果の和 $4+1=5$ を求めることで, 項目 1 の平均値を求めることができる. 同様にして, 項目 2 についてもデータを分解し, サーバ 1 と 2 により結果を得ることができる. この方法では, 各データは分割されるので各サーバはどのデータも知ることができない.

2.4 階層型ニューラルネットワークと BP 法

ここでは, 図 2 に示す 3 層の階層型ニューラルネットワークの BP 学習について説明する [8]. 任意の整数 i に対して, 集合 $Z_i=\{1,2,\cdots,i\}$ と $Z_i^*=\{0,1,\cdots,i\}$ を定義する. ニューラルネットワークにより構成される写像 $\boldsymbol{h}: J_{in}^n \to J_{out}^R$, J_{out}^R を, $\boldsymbol{x}\in J_{in}^n$ に対して $\boldsymbol{h}(\boldsymbol{x})=(h_1(\boldsymbol{x}),\cdots,h_R(\boldsymbol{x}))$ と定義する. ただし, $J_{in}=[0,1]$ または $[-1,1]$, $J_{out}=\{0,1\}$ とする. この場合, L 個の学習データの集合 $X=\{(\boldsymbol{x}^l,\boldsymbol{d}(\boldsymbol{x}^l))|\boldsymbol{x}^l\in J_{in}^n, \boldsymbol{d}(\boldsymbol{x}^l)\in J_{out}^R, l\in Z_L\}$ を使って, ネットワークのパラメータである重みを決定する. ここに, $\boldsymbol{d}(\boldsymbol{x}^l)=\big(d_1(\boldsymbol{x}^l),\ldots,d_R(\boldsymbol{x}^l)\big)$ は $\boldsymbol{x}^l$ に対する (教師データの) 出力をあらわす.

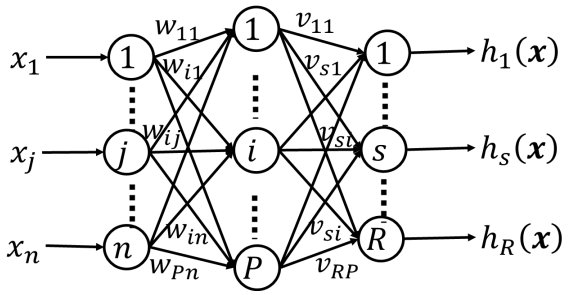

図 2 3 層の階層型ニューラルネットワークの概略図

ネットワークの重みを $W=\{w_{ij}|i\in Z_P, j\in Z_n^*\}$, $V=\{v_{si}|s\in Z_R, i\in Z_P^*\}$ とする. このとき, ネットワークの出力は次式により得られる [8].

$$y_i(\boldsymbol{x})=\frac{1}{1+\exp\Big(-\Big(\sum_{j=0}^{n} w_{ij}x_j\Big)\Big)}) \tag{1}$$

$$h_s(\boldsymbol{x})=\frac{1}{1+\exp\Big(-\Big(\sum_{i=0}^{P} v_{si}y_i(\boldsymbol{x})\Big)\Big)} \tag{2}$$

ここに, $x_0=1$, $y_0=1$ であり, w_{i0}, v_{s0} はしきい値をあらわす.

学習の評価関数を式 (3) として与える.

$$E(X,W,V)=\frac{1}{2L}\sum_{l=1}^{L}\sum_{s=1}^{R}\big(d_s(\boldsymbol{x}^l)-h_s(\boldsymbol{x}^l)\big)^2 \tag{3}$$

重み W,V は, 与えられた学習用データ X に対する式 (3) の最小化問題として BP 法により解くことができる. BP 法のアルゴリズムを以下のように与える [8]. X は学習用データ, T は重みの最大更新回数, θ はしきい値, α は学習係数とする.

[**B**P 法のアルゴリズム]BP(X, W, V)

[**S**tep 1]

$W=\{w_{ij}|i\in Z_P, j\in Z_n^*\}$, $V=\{v_{si}|s\in Z_R, i\in Z_P^*\}$ を初期化, $t\leftarrow 0$.

[**S**tep 2]

データ $(\boldsymbol{x}^l,\boldsymbol{d}(\boldsymbol{x}^l))\in X$ をランダムに選択. 式 (1), (2) により $y_i(\boldsymbol{x}^l)$, $h_s(\boldsymbol{x}^l)$ を求める.

[**S**tep 3]

次式に基づき $w_{ij}\in W$, $v_{si}\in V$ を更新する.

$$w_{ij} \leftarrow w_{ij}+\alpha\sum_{s=1}^{S}(d_s(\boldsymbol{x}^l)-h_s(\boldsymbol{x}^l))(1-h_s(\boldsymbol{x}^l))v_{si} \times y_i(\boldsymbol{x}^l)(1-y_i(\boldsymbol{x}^l))x_j^l \tag{4}$$

$$v_{si} \leftarrow v_{si}+\alpha(d_s(\boldsymbol{x}^l)-h_s(\boldsymbol{x}^l))h_s(\boldsymbol{x}^l)(1-h_s(\boldsymbol{x}^l))y_i(\boldsymbol{x}^l) \tag{5}$$

[**S**tep 4]

式 (3) に基づき評価値 $E(X,W,V)$ を求める.

[**S**tep 5]

もし, $\cdot E<\theta$ または $t<T$ ならば, アルゴリズムを終了する. そうでなければ, $t\leftarrow t+1$ として Step 2 へ.

3. 提案手法

提案アルゴリズムの目的は, 学習データを各サーバに知らせることなく学習を行うことである. 特に, 学習データとパラメータを 2.3 で導入した簡易秘匿計算によるデータの分割方法を用いて, 図 1 の提案モデル上で BP 学習を実現する.

3.1 秘匿計算用データの表現

ここでは, 学習用データ $(\boldsymbol{x}^l,\boldsymbol{d}(\boldsymbol{x}^l))\in X$ は以下のように Q 個にランダムに分割され, Q 個のサーバに分けて管理されるものとする [5].

$$x_j^l = \Pi_{q=1}^{Q} x_j^{l(q)} \tag{6}$$

$$d_s(\boldsymbol{x}^l) = \sum_{q=1}^{Q} d_s^{(q)}(\boldsymbol{x}^l) \tag{7}$$

また, 重み $w_{ij}\in W$, $v_{si}\in V$ もまた以下のように Q 個に分割され, Q 個のサーバに分けて管理されるものとする.

$$w_{ij} = \Pi_{q=1}^{Q} w_{ij}^{(q)} \tag{8}$$

$$v_{si} = \Pi_{q=1}^{Q} v_{si}^{(q)} \tag{9}$$

このとき, $W^{(q)} = \{w_{ij}^{(q)} | i\in Z_P, j\in Z_n^*\}$, $V^{(q)} = \{v_{si}^{(q)} | s\in Z_S, i\in Z_P^*\}$ とおく.

ここで, $\Pi_{q=1}^{Q} x_0^{(q)} = 1$ とする. 式 (6) により分割された入力データ対する階層型ニューラルネットワークの出力は, 式 (1), (2) の代わりに次式 (10), (11) を用いることで導出することができる.

$$y_i(\boldsymbol{x}) = \frac{1}{1+\exp\left(-\left(\sum_{j=0}^{n}\Pi_{q=1}^{Q} w_{ij}^{k(q)} x_j^{(q)}\right)\right))} \tag{10}$$

y_i を計算後, $y_i = \Pi_{q=1}^{Q} y_i^{(q)} (i\in Z_P^*)$ と分割し, 各サーバに送る. ここで, $\Pi_{q=1}^{Q} y_0^{(q)} = 1$ とする. 各サーバで重みを乗じてクライアントで以下の $h_s(\boldsymbol{x})$ を計算する.

$$h_s(\boldsymbol{x}) = \frac{1}{1+\exp\left(-\left(\sum_{i=0}^{P}\Pi_{q=1}^{Q} v_{si}^{k(q)} y_i^{(q)}\right)\right)} \tag{11}$$

その後, $h_s(\boldsymbol{x}) = \Pi_{q=1}^{Q} h_s^{(q)}(\boldsymbol{x})$ と分割し, 各サーバに送る.

このとき, データ集合 X に対する平均二乗誤差は次式により求められる.

$$E(X) = \frac{1}{2L}\sum_{l=1}^{L}\sum_{s=1}^{S}\sum_{q=1}^{Q}(d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l))^2 \tag{12}$$

BP 法については, 式 (4), (5) の代わりに次式 (13), (14) を用いることで, 分割された重み $w_{ij}^{(q)} (i\in Z_P, j\in Z_P^*)$, $v_{si}^{(q)} (s\in Z_S, i\in Z_P^*)$ を更新することができる [5].

$$\begin{aligned} w_{ij}^{(q)} = w_{ij}^{(q)} + \ & \alpha \sum_{s=1}^{S}\sum_{q=1}^{Q}(d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l))(1-h_s^{(q)}(\boldsymbol{x}^l)) \\ & \times \Pi_{q=1}^{Q} v_{si}^{(q)} h_s^{(q)}(\boldsymbol{x}^l)(1-h_s(\boldsymbol{x}^l)) \\ & \times (\Pi_{q=1}^{Q} w_{ij}^{(q)} x_j^{l(q)})/w_{ij}^{(q)} \end{aligned} \tag{13}$$

$$\begin{aligned} v_{si}^{(q)} = v_{si}^{(q)} + & \alpha\sum_{q=1}^{Q}(d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l))h_s(\boldsymbol{x}^l) \\ & \times(1-h_s(\boldsymbol{x}^l))(\Pi_{q=1}^{Q} y_i^{(q)} v_{si}^{(q)})/v_{si}^{(q)}) \end{aligned} \tag{14}$$

ここに, 式 (13) と (14) の右辺の更新式において $1/w_{ij}^{(q)}$ と $1/v_{si}$ に注意する.

3.2 データ分割の更新の提案

文献 [5] において, 従来法の SMC に対する BP 法のアルゴリズムにおいては, 分割された入力データ $x_j^{l(q)} (j\in Z_n, l\in Z_L, q\in Z_Q)$ および出力データ $d_s^{(q)}(\boldsymbol{x}^l) (l\in Z_L, s\in R, q\in Z_Q)$ は各サーバ内部において, 値が変更されることなく保持されている.

本稿では, サーバ内部に保持されたデータ $x_j^{(q)}$ および $d_s^{(q)}(\boldsymbol{x}^l)$, すなわち各データの分解を学習中に変更することで, データの安全性を高める BP 法のアルゴリズムを提案する.

時刻 u における分割された入力データを $x_j^{l(q)}(u) (j\in Z_n, l\in Z_L, p\in Z_Q)$ とおく. このとき, 時刻 u における入力データ x_j^l は次のように表される.

$$x_j^l = \Pi_{q=1}^{Q} x_j^{l(q)}(u) \tag{15}$$

ここで, 時刻 u において以下に示すランダムな値 $O_1^{(q)}(u) (q\in Z_Q)$ を生成する.

$$\Pi_{q=1}^{Q} O_1^{(q)}(u) = 1 \tag{16}$$

このとき, 次式が成立する.

$$\begin{aligned} x_j^l &= x_j^l \times 1 \\ &= \Pi_{q=1}^{Q} O_1^{(q)} x_j^{l(q)}(u) \end{aligned} \tag{17}$$

そこで, サーバ q 内部に格納されたデータ $x_j^{l(q)}(u)$ を次式に基づき更新する.

$$x_j^{l(q)}(u+1) = O_1^{(q)}(u) \times x_j^{l(q)}(u) \tag{18}$$

このとき, 式 (17) より, 次式が成立する.

$$\Pi_{q=1}^{Q} x_j^{l(q)}(u+1) = x_j^l \tag{19}$$

つまり, 式 (18) を用いることで, サーバ q 内部に格納されたデータ $x_j^{l(q)}(u)$ を, 式 (6) の性質を維持しつつ, 別の値に変更することができる.

また, 時刻 u における分割された出力データを $d_s^{(q)}(\boldsymbol{x}^l)(u) (s\in Z_S, l\in Z_L, q\in Z_Q)$ とおく. このとき, データ $d_s(\boldsymbol{x}^l)$ について, 次式が成立する.

$$d_s(\boldsymbol{x}^l) = \Pi_{q=1}^{Q} d_s^{(q)}(\boldsymbol{x}^l)(u) \tag{20}$$

ここで, 時刻 u において次式を満たすランダムな値 $O_0^{(q)}(u) (q\in Z_Q)$ を生成する.

$$\Pi_{q=1}^{Q} O_0^{(q)}(u) = 0 \tag{21}$$

このとき, 次式が成立する.

$$\begin{aligned} d_s(\boldsymbol{x}^l) &= d_s(\boldsymbol{x}^l) + 1 \\ &= \sum_{q=1}^{Q}\left(O_0^{(q)}(u) + d_s^{(q)}(\boldsymbol{x}^l)(u)\right) \end{aligned} \tag{22}$$

そこで, サーバ q 内部に格納されたデータ $d^{(q)}(\boldsymbol{x}^l)(u)$ を次式に基づき更新する.

$$d_s^{(q)}(\boldsymbol{x}^l)(u+1) = O_0^{(q)}(u) + d_s^{(q)}(\boldsymbol{x}^l)(u) \tag{23}$$

このとき, 式 (22) より, 次式が成立する.

$$\sum_{q=1}^{Q} d_s^{(q)}(\boldsymbol{x}^l)(u+1) = d_s(\boldsymbol{x}^l) \tag{24}$$

つまり, 式 (23) を用いることで, サーバ q 内部に格納されたデータ $d_s^{(q)}(\boldsymbol{x}^l)(u)$ を, 式 (7) の性質を維持しつつ, 別の値に変更することができる.

そこで, 本稿では, BP 法の学習中に, サーバ内部において式 (18), (23) に基づきデータの更新を行うことで, データの安全性を高める学習アルゴリズムの提案を行う.

	サーバ 0	サーバ q (q∈Z_Q)
初期化		$\{x_j^{l(q)}\|l\in Z_L, j\in Z_n\}$, $\{d_s^{(q)}(\boldsymbol{x}^l)\|l\in Z_L, s\in Z_R\}$ を記憶 $\{w_{ij}^{(q)}\|i\in Z_P, j\in Z_n^*\}$, $\{v_{si}^{(q)}\|s\in Z_R, i\in Z_P^*\}$ を初期化
Step 1	t←0	
Step 2	データ番号 l∈Z_L をランダムに選択し, 各サーバに送る.	
Step 3		$w_{ij}^{(q)}x_j^{l(q)}(i\in Z_P, j\in Z_n^*)$ を計算し, サーバ 0 に送る.
Step 4	式 (10) により $y_i(\boldsymbol{x}^l)$ を計算し, $y_i=\Pi_{q=1}^Q y_i^{(q)}$ と分割する. $y_i^{(q)}(q\in Z_Q)$ を各サーバに分けて送る.	
Step 5		$v_{si}^{(q)}y_i^{(q)}(s\in Z_R, i\in Z_P^*)$ を計算し, サーバ 0 に送る.
Step 6	式 (11) により $h_s(\boldsymbol{x}^l)(s\in Z_S)$ を求める. $h_s(\boldsymbol{x}^l)=\Pi_{q=1}^Q h_s^{(q)}(\boldsymbol{x}^l)$ と分割する. $h_s^{(q)}(\boldsymbol{x}^l)(q\in Z_Q)$ を各サーバに分けて送る.	
Step 7		$d_s^{(q)}(\boldsymbol{x}^l)-h_s^{(q)}(\boldsymbol{x}^l)(s\in Z_R)$ を計算し, サーバ 0 に送る.
Step 8	$p_{1(ij)}=\sum_{s=1}^R\sum_{q=1}^Q(d_s^{(q)}(\boldsymbol{x}^l)-h_s^{(q)}(\boldsymbol{x}^l))(1-h_s^{(q)}(\boldsymbol{x}^l))$ $\times v_{si}y_i(\boldsymbol{x}^l)(1-y_i(\boldsymbol{x}^l))(\Pi_{q=1}^Q w_{ij}^{(q)}x_j^{l(q)})$, $p_{2(si)}=\sum_{q=1}^Q(d_s^{(q)}(\boldsymbol{x}^l)-h_s^{(q)}(\boldsymbol{x}^l))h_s(\boldsymbol{x}^l)(1-h_s(\boldsymbol{x}^l))$ $\times(\Pi_{q=1}^Q y_i^{(q)}v_{si}^{(q)})$ を計算し, 各サーバに送る.	
Step 9		次式に基づき $\{w_{ij}^{(q)}\|i\in Z_P, j\in Z_j^*\}$, $\{v_{si}^{(q)}\|s\in Z_R, i\in Z_P^*\}$ を更新 $w_{ij}^{(q)}$←$w_{ij}^{(q)}+\alpha p_{1(ij)}/w_{ij}^{(q)}$ $v_{si}^{(q)}$←$v_{si}^{(q)}+\alpha p_{2(i)}/v_{si}^{(q)}$
Step 10	式 (12) に基づき $E(X,W,V)$ を求める.	
Step 11	$E<\theta$ または $t<T$ の場合は学習終了. それ以外の場合は t←$t+1$ として Step 2 へ.	

表 2 SMC に対する BP 法のアルゴリズム

3.3 セキュアマルチパーティ計算をもちいた BP 学習法

表 2 のアルゴリズムについて説明する [5]. 基本的な考え方として, 学習データを Q 個に分割し, 各サーバにおいて分割されたデータに対する出力の部分計算を行い, それらを集めてサーバ 0 で統合計算を行う. これを入力層から出力層に向けて繰り返す. 次に得られたネットワークの出力に対する誤差も, 各サーバで分割して計算されたものを, サーバ 0 で統合して計算する. これを出力層から入力層に向けて繰り返す. これを 1 回の学習として, 終了条件を満足するまで複数回, 繰り返す.

あらかじめ学習データ $\boldsymbol{x}^l$, 教師データ $\boldsymbol{d}(\boldsymbol{x}^l)$, 重みパラメータ W, V を, Q 個のサーバに分割して記憶する. ステップ 3 で, l 番目のデータに対する部分計算を行い, 結果をサーバ 0 に送る. ステップ 4 では, サーバ 0 ではネットワークの 1 層目の出力 $y_i(i\in Z_P)$ を計算する. また, y_i を Q 個に分割し, 各サーバに送る. ステップ 5 では, 各サーバで 2 層目の出力に対する部分計算を行い, サーバ 0 側に送る. ステップ 6 では, ステップ 5 の結果を使って, サーバ 0 で l 番目の入力データに対するネットワークの出力 $h_s(\boldsymbol{x}^l)(s\in Z_R)$ を計算する. ステップ 8 では, 各サーバの誤差を使って重みパラメータ w_{ij} と v_{si} の更新量 $p_{1(ij)}, p_{2(si)}$ を計算し, 各サーバに送る. ステップ 9 では, 重みパラメータの各成分 $w_{ij}^{(q)}$ と $v_{si}^{(q)}$ を更新 (保管) する. ステップ 10 では, 終了条件のチェックのためにネットワークの誤差 E(X,W,V) を計算する. ステップ 11 で, 誤差が十分小さいか, 最大学習回数を越えれば, 学習を終了する. それ以外は, 学習を繰り返す. 表 2 のアルゴリズムにおいて, 各サーバには復号化された学習データやパラメータが記憶されることはないことに注意する.

表 3 の提案アルゴリズムについて説明する. 表 2 との違いは, ステップ 9 で T_u 回目の学習毎にパラメータの更新後に, ステップ 10 と 11 において, 学習データの再分割を行うことである. すなわち, ステップ 11 において, 現在の分割データ $x_j^{l(q)}$, $d_s^{(q)}(\boldsymbol{x}^l)(l\in Z_L, j\in Z_n, s\in Z_R)$ とランダムな係数 O_1, O_0 を使って, 新しい分割データを計算する. この分割データを使って, 同様に学習ステップを繰り返す.

4. 数値実験

ここでは, 表 4 に示すような Iris, Wine, Sonar, BCW の 4 種類のデータ [9] に対して, 従来法と提案法によりデータの分類を行う. ここでは, 式 (1), (2) において $P=10$, Iris と Wine の場合は $S=3$, Sonar と BCW の場合は $S=2$ とする 3 層ニューラルネットワークを用いた. なお, ここでは 5-fold cross-validation による評価を行う. 実験の条件としては, 最大学習回数を $T=50000$, 学習係数を $K_w=0.01$, $K_v=0.01$ とする. また, 提案手法については, 重みの更新回数が 100 回ごとに分割されたデータの更新を行う, つまり, $T_u=100$ とする. 実験においては, 学習用データに対する平均二乗誤差 (Mean Square Error : MSE) がしきい値 θ を下回る, または重みの更新回数が最大学習回数となれば

	サーバ 0	サーバ q ($q \in Z_Q$)
初期化		$\{x_j^{l(q)} \mid l \in Z_L, j \in Z_n\}$, $\{d_s^{(q)}(\boldsymbol{x}^l) \mid l \in Z_L, s \in Z_R\}$ を記憶 $\{w_{ij}^{(q)} \mid i \in Z_P, j \in Z_n^*\}$, $\{v_{si}^{(q)} \mid s \in Z_R, i \in Z_P^*\}$ を初期化
Step 1	$t \leftarrow 0$	
Step 2	データ番号 $l \in Z_L$ をランダムに選択し, 各サーバに送る.	
Step 3		$w_{ij}^{(q)} x_j^{l(q)} (i \in Z_P, j \in Z_n^*)$ を計算し, サーバ 0 に送る.
Step 4	式 (10) により $y_i(\boldsymbol{x}^l)$ を計算し, $y_i = \Pi_{q=1}^{Q} y_i^{(q)}$ と分割する. $y_i^{(q)} (q \in Z_Q)$ を各サーバに分けて送る.	
Step 5		$v_{si}^{(q)} y_i^{(q)} (s \in Z_R, i \in Z_P^*)$ を計算し, サーバ 0 に送る.
Step 6	式 (11) により $h_s(\boldsymbol{x}^l) (s \in Z_S)$ を求める. $h_s(\boldsymbol{x}^l) = \Pi_{q=1}^{Q} h_s^{(q)}(\boldsymbol{x}^l)$ と分割する. $h_s^{(q)}(\boldsymbol{x}^l) (q \in Z_Q)$ を各サーバに分けて送る.	
Step 7		$d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l) (s \in Z_R)$ を計算し, サーバ 0 に送る.
Step 8	$p_{1(ij)} = \sum_{s=1}^{R} \sum_{q=1}^{Q} (d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l))(1 - h_s^{(q)}(\boldsymbol{x}^l))$ $\times v_{si} y_i(\boldsymbol{x}^l)(1 - y_i(\boldsymbol{x}^l))(\Pi_{q=1}^{Q} w_{ij}^{(q)} x_j^{l(q)})$, $p_{2(si)} = \sum_{q=1}^{Q} (d_s^{(q)}(\boldsymbol{x}^l) - h_s^{(q)}(\boldsymbol{x}^l)) h_s(\boldsymbol{x}^l)(1 - h_s(\boldsymbol{x}^l))$ $\times (\Pi_{q=1}^{Q} y_i^{(q)} v_{si}^{(q)})$ を計算し, 各サーバに送る.	
Step 9		次式に基づき $\{w_{ij}^{(q)} \mid i \in Z_P, j \in Z_j^*\}$, $\{v_{si}^{(q)} \mid s \in Z_S, i \in Z_P^*\}$ を更新 $w_{ij}^{(q)} \leftarrow w_{ij}^{(q)} + \alpha p_{1(ij)} / w_{ij}^{(q)}$ $v_{si}^{(q)} \leftarrow v_{si}^{(q)} + \alpha p_{2(i)} / v_{si}^{(q)}$
Step 10	もし $t \bmod T_u$ が 0 であれば, $1 = \Pi_{q=1}^{Q} O_1^{(q)}$, $0 = \sum_{q=1}^{Q} O_0^{(q)}$ と分割し, $O_1^{(q)}$, $O_0^{(q)} (q \in Z_Q)$ を各サーバに分けて送る.	
Step 11		もし $t \bmod T_u$ が 0 であれば, 次式に基づき $\{x_j^{l(q)} \mid j \in Z_n, l \in Z_L\}$, $\{d^{(q)}(\boldsymbol{x}^l) \mid l \in Z_L\}$ を更新 $x_j^{l(q)} \leftarrow O_1^{(q)} \times x_j^{l(q)}$ $d^{(q)}(\boldsymbol{x}^l) \leftarrow O_0^{(q)} + d^{(q)}(\boldsymbol{x}^l)$
Step 12	式 (12) に基づき $E(X, W, V)$ を求める.	
Step 13	$E < \theta$ または $t < T$ の場合は学習終了. それ以外の場合は $t \leftarrow t + 1$ として Step 2 へ.	

表 3 提案手法のアルゴリズム

表 4 数値実験で扱うデータ

	Iris	Wine	Sonar	BCW
#data : L	150	178	208	683
#input : n	4	13	60	9
#output : R	3	3	2	2

学習終了とする. なお, しきい値は Iris, Wine については $\theta = 3.0 \times 10^{-2}$, Sonar, BCW については $\theta = 4.0 \times 10^{-2}$ を用いた.

学習終了後, 各手法について, 学習用データおよびテスト用データに対する誤分類率を比較する.

実験の結果を表 5 に示す. ここで, A は通常の BP 法, B は表 2 による BP 法, C は提案手法を示す. また, RM(Learning), RM(Test) はそれぞれ学習用データおよびテスト用データに対する誤分類率 (%) を, #data:L はデータ数が L 個であることを意味する. 表中の値はそれぞれ 20 回試行の平均値である.

表 5 に示す結果より, 提案手法は従来の BP 法, および従来の BP 法と組み合わせた BP 法とほぼ同等の精度となっている. このことは, データ分割の複数回の導入の影響が

表 5 数値実験の結果

		Iris	Wine	Sonar	BCW
A	RM(Learning)(%)	1.64	0.23	0.61	2.31
	RM(Test)(%)	3.20	1.97	18.17	2.82
B	RM(Learning)(%)	1.77	0.78	1.98	2.01
	RM(Test)(%)	2.51	2.63	15.60	2.66
C	RM(Learning)(%)	1.83	0.70	1.86	2.01
	RM(Test)(%)	4.43	4.03	18.74	3.07

学習精度に対して限定的であることを示している.

5. おわりに

本稿では, SMC によるエッジシステム上の BP 学習法の提案を行った. 特に, 簡易秘匿計算を任意のステップで導入する学習法への一般化されたアルゴリズムを提案した. また, 数値実験において, 提案手法は従来の BP 法, および SMC と組み合わせた BP 法において分割されたデータの更新を行わない場合と同程度の成果を示している. ここでは, エッジシステムに対する学習法として提案したが, クラウドシステムでも同様の適用ができる.

今後の課題として, ニューラルネットワーク以外のモデルに対して, SMC と組み合わせた機械学習アルゴリズムの提案, およびそれらの手法について, 分割されたデータを学習の途中で更新するアルゴリズムの提案とその有効性の検証を行いたい.

参考文献

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, : *Internet of Things : A Survey on Enabling Technologies, Protocols, and Applications*, IEEE Communication Surveys & Tutorials, Vol.17, No.4, pp.2347-2376 (2015).
[2] J. Chen, X. Ran, : *Deep Learning with Edge Computing: A Review*, Proc. of the IEEE, vol. 107, no. 8 (2019).
[3] M.G. Sarwar Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayana, and F. Hussain, : *Machine Learning at the Network Edge: A Survey*, arXiv:1908.00080 [cs.LG] (2019).
[4] Q. Yang, Y. Li, T. Chen, and Y. Tong, *Federated Machine Learaning : Concept and Applications*, ACM Trans. Intell. Syst. Technol., Vol.10, No.2, Article 12 (2019).
[5] H. Miyajima, N. Shigei, H. Miyajima, Y. Miyanishi, S. Kitagami, N. Shiratori: *New Privacy Preserving Back Propagation Learning for Secure Multiparty Computation*, IAENG International Journal of Computer Science, Vol.43, No.3, pp.270-276 (2016).
[6] C. C. Aggarwal, and P. S. Yu, : *Privacy Preserving Data Mining: Models and Algorithms*, ISBN 978-0-387-70991-8, Springer-Verlag (2009).
[7] 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 (2014).
[8] M. M. Gupta, L. Jin, N. Honma: *Static and Dynamic Neural Networks*, IEEE Pres, Wiley-Interscience (2003).
[9] UCI Repository of Machine Learning Databases and Domain Theories, https://archive.ics.uci.edu/ml/datasets.php.