情報処理学会論文誌 Vol.60 No.1 2–15 (Jan. 2019)

センサネットワークにおける分散型深層学習の設計と評価

福島 悠太[1,a)] 山口 弘純[1] 東野 輝夫[1]

受付日 2018年4月11日, 採録日 2018年10月2日

概要：無線センサネットワークにおいて，センサに付随するマイコンの高機能化・省電力化が進めば，従来クラウドで行っていた学習や異常検出，判定などのタスク処理をセンサネットワークにオフローディングし，データ発生場所に近い場所でそれらを効率良く行うことができる自律的な知能センサネットワークが実現できる．本研究では CNN を対象に，それをデータ発生源であるセンサ機器からなるローカルな無線センサネットワーク内で分散実行する新しいアーキテクチャを提案し，そのための分散実行プロトコルならびにアルゴリズムを提案する．提案手法はメッシュ型の無線センサネットワークが面的かつ定期的に取得するデータ（たとえば温度分布など）を対象とし，センサノードに深層学習におけるユニットの役割を割り当てる．提案手法の有効性を評価するため，1,400 m^2 超の実ラウンジスペースの 50 地点の温度データと 6×6 の赤外線センサで収集された人の動きのデータを用いた．1 台の PC 上で複数のノードによる分散学習を実行できるプログラムを仮想的に実装し，通常の CNN による学習と提案手法による分散学習におけるノードのデータ通信コストと学習精度の比較をシミュレーションで行った．その結果，十分妥当なノードのデータ通信コストのもとで，通常の CNN と遜色ない学習精度を達成できることが確認できた．

キーワード：エッジコンピューティング，センサネットワーク，深層学習，分散化

Design and Evaluation of Distributed Deep Learning in Wireless Sensor Network

Yuta Fukushima[1,a)] Hirozumi Yamaguchi[1] Teruo Higashino[1]

Received: April 11, 2018, Accepted: October 2, 2018

Abstract: If wireless sensor nodes become more powerful and more energy-efficient, data processing tasks such as classification and anomaly detection, which has been performed in cloud servers, can be offloaded to the wireless sensor network. This enables to realize an autonomous intelligent sensor network which can efficiently perform the tasks at places close to the data sources. In this paper, we propose new architecture, protocol and algorithm to distribute and execute CNN (Convolutional neural network) over wireless sensor nodes. The idea is to appropriately assign neurons of the CNN to wireless nodes, each of which has limited processing capability but can have some power when they are united. We virtually implement a program that can execute distributed learning by multiple nodes on one PC and conducted two experiments by simulation using real data; one is for anomaly detection of temperature in an over-1,400 m^2 lounge space using 50 temperature sensors to confirm the learning capability as well as communication overhead, and another is for activity recognition using a 6×6 array of thin-, energy-efficient film-type infra-red sensors with micro-processors to demonstrate our concept. As a result, it was found that the proposed method can achieve learning accuracy comparable to normal CNN with adequate communication overhead.

Keywords: edge computing, sensor network, deep learning, decentralization

1. はじめに

分散コンピューティングのフレームワークは，サーバやクライアントの通信性能や可用性，性能差などに応じ，ク

[1] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565–0871 Japan
[a)] y-fukushima@ist.osaka-u.ac.jp

ライアント指向やサーバ指向へとつねに変遷しているが，近年では，クラウドサービスの信頼性と可用性向上により，データ処理をクラウドで集約して行う傾向がある．IoTに関するサービスでは，このクラウドヘビーコンピューティングに基づいており，それによって多くのツールが利用可能となっている．一方，マイクロソフトやGoogle TensorFlow [1], [2] などによるいくつかのIoTツールは，機械学習などにより訓練された判定関数などをIoTデバイスに導入できるエッジコンピューティング機能をサポートしつつある．これにより，全センシングデータをクラウドサーバに送信するための通信リソースを確保する必要がなくなり，プライバシデータをローカルで処理することも可能となる．しかし，そういった既存ツールおよび既存アプローチのほとんどは，学習済みの判定機能の一部または全部をクラウドからエッジデバイスに移行し，それ以後の検知や判定をエッジサイドで行うことで，クラウドへのデータ量を削減することが目的である．したがって，学習段階では依然として，学習機能を有するクラウドサーバあるいはホームゲートウェイなどにすべてのデータを集約する必要がある．

一方，現在IoTデバイスと呼ばれるセンサノードは，メモリ量や処理能力という点での性能向上が著しく，センシングやそのデータ送信だけでなく，機械学習においても一部のタスクを実行することが可能であると考えられる．現状の無線センサネットワーク（WSN）は，経路制御を含むデータ集約プラットフォームとみなされることが多いが，これらのノードをシームレスに連携させることで，より多くのメモリと処理能力が利用可能となり，センシング・学習・フィードバックのプロセスのほとんどがセンサネットワーク内で完結するような，知的な局所型データ処理プラットフォームを形成できる．

そこで本研究では，WSNにおいて深層学習およびそれを用いたデータ処理を行うための新しい手法を提案する．提案手法では，WSNの各センサノードから連続的に生成されるデータを2次元や3次元の地理的データ（たとえば，温度データなど）として扱う．これに対し，多層ニューラルネットワークの中でも画像認識をはじめとする多くの分野において高い精度を達成できる畳み込みニューラルネットワーク（CNN）を対象とし，CNNの各順伝播ユニット（畳み込み処理（フィルタ）やプーリング，全結合など）を，WSN内のノード（センサノード）のいずれかに割り当てる．

また，逆伝播時のパラメータ更新処理を分散型で行う仕組みを開発することで，ユニットのパラメータ更新を分散環境で実現する．通信コストやノード処理負荷の偏りをなるべく少なくするユニット割当てならびにプロトコル設計を行うことで，各ノードの最大処理負荷や最大通信コストを抑制しながら分散環境でCNNを実現している．

提案手法の学習精度を検証するために，1台のPC上で複数のノードによる分散学習を実行できるプログラムを仮想的にpythonで実装した．そして，実環境で取得したセンシングデータを用いた2つの実験を仮想環境内で分散学習をし，シミュレーションを行った．1つ目は，1,400 m^2超の実ラウンジスペースにおいて50個の温度センサを設置し，スポット温度の異常検知を行うシナリオを対象とした実験である．その結果，提案手法（分散型CNN）の学習精度は95.570%，データをWSNで集約して学習を行う方式（通常型CNN）では97.095%となった．また，分散型CNNでは一度の学習にかかるセンシングデータの最大通信コストが396であったのに対し，通常型CNNでは2,866であった．2つ目は，6×6の薄型でエネルギー効率の高いモーションセンサアレイを用いた異常行動検知を行うシナリオを対象とした実験である．その結果，提案手法（分散型CNN）の学習精度は92.2%，データをWSNで集約して学習を行う方式（通常型CNN）では93.1%となった．また，分散型CNNでは一度の学習にかかるセンシングデータの最大通信コストが210であったのに対し，通常型CNNでは360であった．これらから，2つの実験において，通常のCNNと遜色ない学習精度を維持したまま，ノードにかかる最大通信負荷を抑えることができることが確認された．

2. 関連研究と位置づけ

近年，深層学習は多層ニューラルネットワークを学習する方法として幅広く研究されており，音声認識 [3], [4]，物体認識 [5], [6]，画像検索 [7], [8]，および強化学習 [9], [10] など様々なデータ解析において大きな成果が得られている．一般的に，深層学習はデータ量が増加すればするほど高い精度を得ることができるが，画像などの高精細データにおける訓練では数千万のパラメータと数十億の訓練データが必要となる．したがって，それだけの膨大なデータ量により訓練する場合にはデータ量に応じた処理コストが要求される．そこで，訓練にかかる処理負荷の軽減を目指した分散実行手法が提案され，また，それらの手法を容易に扱うことを可能にするTheano [11]，Torch [12]，cuda-convnet and cuda-convnet2 [13]，Decaf [14]，Overfeat [15]，Caffe [16] などの多くのフレームワークが開発されている．

2.1 GPUを用いた深層学習の並列化

文献 [17] では，大規模な深層学習のモデル学習のために，数千のコアおよび数千の計算スレッドを有するGPUと分散システムを用いる手法を提案している．GPUは数千のALUコアを搭載しているため数値演算能力が優れる一方，メモリ制限が課題となる．この問題を解決するため，マルチGPUを用いたデータ並列化，モデル並列化，およびデータモデル並列化からなる分散システムを導入してい

る．データ並列化は，各 GPU が同じモデルと異なるデータセットで学習を実行し，その後異なる GPU から学習したパラメータ勾配を同期する手法である．モデル並列化は，大規模なモデルを分割し，分割したモデルを各 GPU で担当し，各 GPU が同じデータセットを異なるモデルで学習する手法である．しかしデータ並列化では，計算ノードが多数ある場合にはスムーズに学習を行うため学習率を下げる必要があり，モデル並列化では，モデル間の通信コストが問題となる．これに対し，データモデル並列化は，全結合層と畳み込み層の特性から，畳み込み層をデータ並列化し，全結合層をモデル並列化することで，大規模な深層学習のモデルより高速な学習を実現している．ほかに深層学習の並列化として，文献 [18] では 16,000 個の CPU を用いることで数十億のパラメータを持つ大規模な深層学習モデルでの学習精度を向上させる手法が提案されている．また文献 [19] では，6,000 万のパラメータと 65 万のニューロンを持つ大規模な畳み込みニューラルネットワークにおいて，GPU を用いることによる学習高速化が報告されている．

2.2 深層学習の分散型アーキテクチャ

文献 [20] では，大規模な深層学習のモデルを学習するための手法として，SINGA という一般的な分散型の深層学習のプラットフォームを提案している．SINGA は畳み込みニューラルネットワーク（CNN），ボルツマンマシン（RBM），および再帰ニューラルネットワーク（RNN）の一般的な深層学習モデルをサポートする．SINGA は，パラメータ勾配を計算する TrainOneBatch と順伝播を行う NeuralNet からなる Worker ユニットと，結果を集約する Stub ユニット，パラメータを更新する Server ユニットの 3 つのコンポーネントを提供し，これらのユニットをユーザが設定することで分散深層学習を行える．Worker の NeuralNet では CNN や RBN，RNN のニューラルネットワークを設定でき，大規模なモデルでは，次の 4 つの並列化手法（(1) すべてのレイヤを異なるサブセットに分割，(2) 1 つのレイヤをバッチ次元によってサブレイヤに分割，(3) 1 つのレイヤを特徴次元によってサブレイヤに分割，(4) (1)〜(3) の手法の組合せ）が提供される [21]．また，TrainOneBatch では，パラメータ勾配を計算するために順伝播型ニューラルネットワークや RNN などで用いられるバックプロパゲーションとエネルギーモデルで用いられる対比分散アルゴリズムが提供されている．文献 [22] では，並列確率勾配降下アルゴリズムが提案されている．

SINGA では Worker と Server を用いた多様な同期および非同期のフレームワークを提供でき，たとえば Sandblaster [18] や AllReduce [23] といった同期フレームワークや Downpour [18], [24] や Distributed Hogwild [25] といった非同期フレームワークが知られている．同期フレームワークは，分散により 1 回の学習速度を早くすることができ，非同期フレームワークは収束率を高めることができる．学習速度と収束率はトレードオフの関係であり，異なるフレームワークを組み合わせて収束率と学習速度における最適性を得ることができる．

2.3 オープンソースアーキテクチャにおける分散深層学習

最近では，様々な分散学習のオープンソースパッケージが開発されている．All of Distributed Machine Learning Toolkit（DMTK）[26] や Distributed TensorFlow [1], ChainerMN [27] はデータ並列化によって分散学習を実行する．データの並列化は，分割されたデータセットを各 GPU 上でミニバッチとして処理することで学習速度を向上させるが，すべてのパラメータ勾配を集約してパラメータを更新する必要がある．つまり，データの並列化には，低遅延で高品質なネットワークが必要となる．

2.4 モバイル端末上での深層学習

近年では，モバイル端末による深層学習に関する研究もなされている．MoDNN [28] は，事前に学習した CNN モデルをモバイル端末上に分割することによって，CNN の高速化を行う手法を述べている．DeepX [29] や DeepMon [30] では，複数のコプロセッサで計算処理を分割することによって，モバイル端末上での CNN の実行が可能であることを示している．これらのアプローチでは，事前に学習した CNN をモバイル端末上で実行することが可能であるが，モバイル端末上で学習を実行する手法については触れられていない．

2.5 本研究の位置づけ

前述のように，深層学習の分散実行は多くの研究がなされているが，それらはいずれも膨大なデータ量を迅速に処理するために複数の計算機を用いる並列分散計算である．これに対し，本研究では IoT センサネットワークによる深層学習の新しいフレームワークを提案している．提案手法では，高性能な GPU を備えたサーバではなく，計算資源の少ない IoT センサノードを協調・連携させることにより負荷の高い学習をセンサネットワークで実現する点でこれまでのアプローチとはまったく異なる．

つまり，本研究における貢献は，センサネットワークにおける低電力センサ間通信による深層学習を行う新しい手法を提案している点，2 つの実環境のデータを用いた実験によって，分散アルゴリズムの評価を行っている点である．2 つの実験として，1 つ目は，実ラウンジスペースにおける温度データを用いたスポット温度の異常検知，2 つ目は，赤外線センサによる人の通過データを用いた異常行動検知である．

3. 提案手法

3.1 想定環境とシナリオ

本研究では，ある程度のプロセシング能力と通信機能を搭載したセンサノード（IoT ノード）が無線通信によってWSN を構成している状況を想定している．WSN を G とすると，センサノードの集合 S と，センサノード間の双方向リンク $E_G \subseteq S \times S$ を用いて，$G = (S, E_G)$ と表現できる．G は OLSR などの適切な経路制御プロトコルにより，任意のセンサノード間には必ず 1 つ以上の通信経路が存在し，互いに通信可能な状態であるとする．本研究では簡単のため，センサノードは移動しないと仮定するが，ノード間の適切なコネクティビティが経路制御機構により提供されればこれは本質的な制約ではない．また，同一 WSN に所属するセンサノードがセンサ値の取得タイミングを同期する必要はないが，各ノードは共通の時間間隔でセンサ値を取得し続けるものとする．

WSN 内で深層学習を行う典型的な例として，オフィスなどの屋内空間において温湿度や輝度などをセンシングし，省エネルギーや快適性向上のために適切な空調・照明を提供する BEMS システムがあげられる．特に個人の温熱快適性は，室温や入射日光，エアコンの位置など様々な要因に影響を受ける主観的な値であるため，温熱快適性を実現するように空調の調整を行うことは容易でない [31], [32]．実際に，6 章で用いた実験データからは，対象ラウンジスペース内の最大温度差が 3 度から 4 度程度であったことが確認され，温熱的な不快感を訴える滞在者も存在する．しかし，前述のように，どのような温度環境が人の快・不快を決定するかが分かりにくいため，たとえば温熱的不快感のクレームの有無や顧客サーベイを真値とし，WSN が快・不快を判断できるようになれば，不快状況をいち早く検知して管理者に通知する用途にも適用可能である．また，センサアレイにおける異常行動検知なども考えられる．たとえば，圧力センサアレイを内蔵したフロアマットや，安価な焦電型赤外線センサをアレイ状に構成した広範囲モーションセンサで，家庭内の高齢者の行動パターンや店舗の通行パターンなどを学習しておけば，カメラなどのプライバシに関わるデバイスを利用することなく，転倒検知や侵入検知などの異常検知に適用できる．センサノード間でこういった学習処理を分担することにより，各センサノードの処理量は十分に小さくなり，圧力発電のようなエナジーハーベスティングでの処理が可能となる．すなわち，学習処理全体がエナジーハーベスティングで実行可能となり，省エネルギーで知的な WSN の実現につながることになる．

入力データサイズはセンサノード数に依存し，センサノード数が増加すれば，入力データ量も同様のオーダで増加することを意味する．大容量のデータから効率良く学習するには，層構造が深い CNN および適切なパラメータ設定が必要となる．したがって，センサノード数が多いと，その分 CNN は深くなり，各ノードの学習処理や通信処理が増え，学習時間が長くなるなどの問題が発生する．そのため，WSN の規模としては，数十から数百のセンサノードを想定している．

表 1 CNN のパラメータ表記
Table 1 CNN notations (some are omitted).

パラメータ	詳細
(N, M)	入力データサイズ
T	隠れ層の数
$c^{(t)}$	t 番目の畳み込み層
$p^{(t)}$	$c^{(t)}$ の後のプーリング層
$c_k^{(t)}(x, y)$	(x, y) 座標の k 番目のフィルタの $c^{(t)}$ のユニット
$p^{(t)}(x, y)$	(x, y) 座標の $p^{(t)}$ のユニット
$K^{(t)}$	$c^{(t)}$ のフィルタ数
$h^{(t)}$	$c^{(t)}$ のフィルタサイズ
$s^{(t)}$	$p^{(t)}$ のプーリングサイズ

提案手法では，入力層，T 層の隠れ層（$1 \leq T$），全結合層 f，出力層 o からなる一般的な CNN を対象とする．t 層目の隠れ層（$1 \leq t \leq T$）は畳み込み層 $c^{(t)}$ およびプーリング層 $p^{(t)}$ からなり，プーリング層は任意であるものとする．各畳み込み層 $c^{(t)}$ のフィルタ数およびフィルタサイズを，それぞれ $K^{(t)}$ および $h^{(t)} \times h^{(t)}$ で表し，プーリング層がある場合のプーリングサイズは，$s^{(t)} \times s^{(t)}$ で表す．各層のユニットは XY 座標を用いて，ユニットの座標を非負の整数値で (x, y) のように表し，左下のユニットを原点 $(0, 0)$ とする．k 番目のフィルタの t 層目の畳み込み層 $c^{(t)}$，t 層目のプーリング層 $p^{(t)}$，全結合層，出力層のユニットをそれぞれ，$c_k^{(t)}(x, y)$，$p^{(t)}(x, y)$，$f(x, y)$，$o(x, y)$ と表す．畳み込み層およびプーリング層は前の層のユニット $h^{(t)} \times h^{(t)}$ または $s^{(t)} \times s^{(t)}$ 個の出力から，全結合層および出力層は前の層のユニットすべての出力から計算される．

詳細は後述するが，提案手法においては多地点センシングデータを画像のような 2 次元データとみなし，各データを有するセンサノードが周辺のセンサノードから畳み込みなどに必要なデータを受け取り CNN の順伝播を実現する．したがって，フィルタサイズ $h^{(t)}$ やフィルタ数 $K^{(t)}$，プーリングサイズ $s^{(t)}$ が通信コストに影響を与える．たとえば，フィルタサイズ $h^{(t)}$ が大きいと，畳み込み処理を行うためには，各センサノードがより離れたセンサノードからデータを集める必要があり，通信コストが増大する．したがって，提案手法ではフィルタサイズ $h^{(t)}$ は比較的小さい値（3 あるいは 5 程度）が望ましく，本稿で行った実験においては，そのような比較的小さいフィルタが適切であった．なお，一般には最適なフィルタサイズやフィルタ数，プーリングサイズはデータセットに依存する．

本稿で用いる CNN のパラメータなどの表記を**表 1** に示す．

4. 分散実行プロトコル

4.1 動作概要

本研究は，WSN のノードの処理能力および通信能力を用い，CNN の学習を WSN 内で自律分散的に実行させることが目的である．

提案手法では，CNN の各ユニットがセンサノードに対応付けられる．順伝播では．各センサノードは WSN を介して，（入力層を含む）ユニットの出力データを交換し，ユニットの処理を行う．出力層のユニットを割り当てられているセンサノードは，出力データと正解データから逆伝播処理を行う．逆伝播では，各ユニットが前のユニットの伝播のみに基づいて分散実行し，重みを更新する．したがってユニット間での重み共有は行わず，その更新はユニットごとに独立して行われる．

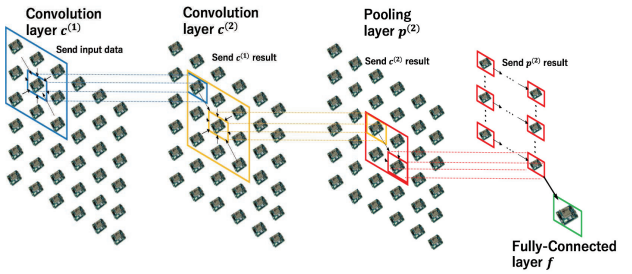
図 1 に，畳み込み 2 層，プーリング 1 層の場合における CNN の順伝播と逆伝播の処理を示す，WSN 上でこの処理を行うために，(1) センサノードへのユニットの割当て，(2) 順伝播におけるユニットの実行，(3) 逆伝播におけるユニットごとの重み更新，の処理が行われる．これらの各処理について次節以降で述べる．

4.2 ユニットの割当て

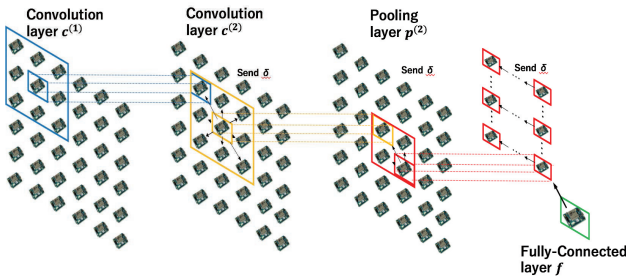
WSN 間での任意のセンサノード間の通信が保証されており，通信コストに制限がないのであれば，どのようなユニット割当てでも CNN の学習は可能である．しかし，ユニット割当てによって，各センサノードの通信コストおよび処理オーバヘッドは大きく変化する．たとえば，入力層を除いたすべてのユニットを 1 つのセンサノードに割り当てたとすると，このノードに送信されるセンシングデータは，センサ数（$N \cdot M - 1$）に比例することに加え，1 つのノードで順伝播および逆伝播のすべての処理を行う必要がある．本研究ではこの処理負荷や通信負荷をなるべくノード均等に振り分けるようにユニットを割り当てた分散型 CNN を実現することである．

ここで，理想的なセンサノードへのユニットの割当ては，CNN のユニット間リンクと WSN のノード間リンクとの対応を最大化し，かつ各センサノードに割り当てられるユニットの数を可能な限り平均化することである．このような割当ては，特定のセンサノードに過負荷を課すことなく，CNN を実行するための通信のほとんどが隣接するセンサノード間で行われる．しかし，一般にこの問題は，WSN のリンクが CNN のリンクに完全に対応するわけではないことから，そのような割当てが存在するとは限らず，CNN と WSN でできるだけ対応リンクが多い割当てを発見する問題は，部分グラフ同型判定問題 [33] に帰着され，最適解を求めるのは NP 困難問題である．このような問題の最適解を見つけることは容易ではない．

したがって，2 つのセンサノード間の物理的距離を WSN におけるそれらの間の無線リンクに対応させることによって，センサノードの座標を CNN の XY 座標に適合させる．具体的には，各センサノードの位置情報に基づいて，CNN の XY 座標平面に割り当てる簡単な手法を用いて，(x, y) の CNN の各ユニットを (x, y) のセンサノードに割り当てる．理想的には，$N \times M$ 個のセンサノードがグリッド状に配置され，センサノードが XY 座標平面に 1 対 1 のマッピングが可能であり，CNN のすべてのユニットをそれらのセンサノードに割り当てることができる状態である．しかし，実環境においては，配置の制約などからグリッド状にセンサノードを配置できないことがあり，**図 2** (a) に示すようにすべての座標平面にセンサノードが割り当てられない場合が考えられる．このような場合，センサノードが割り当てられない座標を含む状態でグリッド状に分割することになる．センサノードが割り当てられなかった座標はセンサノードの欠損として，一部データが欠けた面的なデータとして CNN のユニット処理を行う．なお，センサノードがグリッド状に割り当てられない場合として，図 2 (a) のように単にセンサノードが割り当てられない座標が存在する場合，およびセンサノードの配置に偏りがあり 1 つの座標に 2 つ以上のセンサノードが割り当てられる場合，の 2 通りが考えられる．前者はセンサノードの欠損として扱い，後者はうまくグリッド状に分割することで 1 座標に対し 1 センサノードが割り当てられるようにする．なお，後者の方法はすべてのセンサノードのセンシングデータを利用できるメリットがあるが，配置に偏りがある場合には同様に欠損ノードが発生する可能性がある．別の方法とし

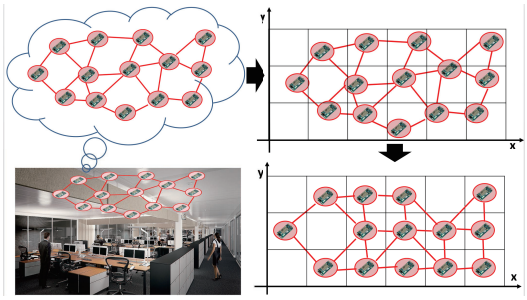


(a) 順伝播

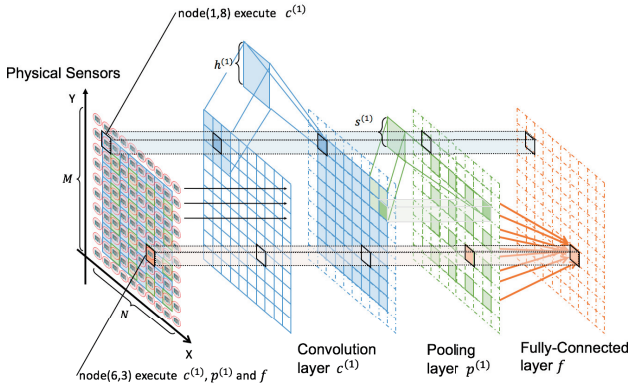


(b) 逆伝播

図 1 分散型 CNN の処理例
Fig. 1 Feedforward and backpropagation.



(a) WSN の座標系への割当



(b) CNN の座標系への割当

図 2 センサノードのユニット割当
Fig. 2 Unit assignment to sensor nodes.

て，1 座標に複数のセンサノードが割り当てられることを許容して可能な限り均等にグリッド状に分割する方法も考えられる．1 座標に割り当てられた複数のセンサノードはほぼ同様のセンシングデータを出力すると考え，それらのうちの 1 つのみを用いる．これは欠損の数を抑制できる反面，センサノード総数は減少し，入力データのサイズが小さくなる．どの割当て方がよいかは通信コストや欠損による精度への影響を評価する必要があるが，これは今後の課題としたい．

各センサノードが XY 座標平面に割り当てられると，図 2 (b) に示すように，座標 (x, y) のセンサノードは，CNN においてその座標系に割り当てられた各層のユニット $c_k^{(t)}(x,y)$，$p^{(t)}(x,y)$，$f(x,y)$，$o(x,y)$ の処理を行う．

4.3 順伝播処理

説明の簡略化のため，この章では，WSN 内の $N \times M$ 個のセンサノードが $N \times M$ のアドレス空間に割り当てられたとする．センサノードが欠損している場合については 5 章で説明する．つまり，この章では，センサノードとユニットが 1：1 に割り当てられているものとする．また，各センサノードは他のセンサノードの座標およびセンサノード間のリンクが既知であるとする．したがって，このとき，他のセンサノードの座標および WSN のリンクから送信先のノードへの送信経路を以下の方法によって導出する．隣接ノードから順番に幅優先探索で最短経路を探索する．幅優先探索を用いる際に，ランダムに隣接ノードを選択することで，選択する経路が集中しないようにし，これによってデータ送信経路によるデータの集中を防ぐ．この節では，各センサノードの順伝播処理について説明する．

順伝播処理では，センサノード間の通信によって，通常の CNN と同様に行われる．具体的には，センサノード (x, y) は，割り当てられた入力層のユニット (x, y)，畳み込み層のユニット $c_k^{(t)}(x,y)$，プーリング層のユニット $p^{(t)}(x,y)$，全結合層のユニット $f(x,y)$，出力層のユニット $o(x,y)$ を実行する．つまり CNN において (x, y) に対応するすべてのユニットは，センサノード (x, y) で実行される．

畳み込み層のユニット $c_k^{(t)}(x,y)$ を実行するために，センサノードは前の層（入力層，畳み込み層またはプーリング層）のユニットから入力データを取得する．そのために，オフセット O_x, O_y を導入する．これは周囲のノードからデータを取得する際に，そのノードまでの距離を表したものである．オフセット O_x, O_y はプーリング処理を行う度に，以下の式によって加算される．

$$(O_x^{(t)}, O_y^{(t)}) = (O_x^{(t-1)} + s^{(t-1)}, O_y^{(t-1)} + s^{(t-1)}) \quad (1)$$

ただし，前の層にプーリング層がなければ，$s^{(t-1)} = 0$ とする．畳み込み層のユニット $c_k^{(t)}(x,y)$ は，前の層のユニット $p^{(t-1)}(x + i \cdot O_x, y + j \cdot O_y)$ に対応するセンサノードと通信する． ただし，i，j は $\underline{\rho}_k^{(t)} \le i, j \le \overline{\rho}_k^{(t)}$ とし，$\underline{\rho}_k^{(t)}$，$\overline{\rho}_k^{(t)}$ は

$$\underline{\rho}_k^{(t)} = -\lfloor \frac{h^{(t)}}{2} \rfloor, \quad \overline{\rho}_k^{(t)} = \lfloor \frac{h^{(t)}}{2} \rfloor - d. \quad (2)$$

とする．このとき，フィルタサイズ $h^{(t)}$ が奇数のとき $d = -1$，偶数のとき $d = 0$ である．

そして以下の式によって畳み込み処理が行われる．f は活性化関数（ここでは ReLU）であり，センサノード (x, y) での重みおよびバイアスをそれぞれ w_{ij} および b とする．

$$c_k^{(t)}(x,y) = f\left(\sum_{\underline{\rho}_k^{(t)} \le i,j \le \overline{\rho}_k^{(t)}} w_{ij} \cdot p_{ij} + b \right) \quad (3)$$

ただし，$p_{ij} = p^{(t-1)}(x + i \cdot O_x, y + j \cdot O_y)$ とする．入力データとして，前の層にプーリング層が存在しない場合は，入力層および畳み込み層のデータを用いて計算される，

プーリング層では，以下の式によってプーリング層のユニット $p^{(t)}(x,y)$ の処理が行われる．

$$p^{(t)}(x,y) = max_{\underline{\rho}_k^{(t)} \le i,j \le \overline{\rho}_k^{(t)}} c_{ij} \quad (4)$$

ただし，$c_{ij} = c_k^{(t)}(x + i \cdot O_x^{(t)}, y + j \cdot O_y^{(t)})$ とし，$\underline{\rho}_k^{(t)}$，$\overline{\rho}_k^{(t)}$ の導出の際には，フィルタサイズ $h^{(t)}$ の代わりに，プーリングサイズ $s^{(t)}$ を用いる．

4.4 逆伝播処理

逆伝播処理における微分の計算は，CNN に沿って順伝播と逆方向に実行されるプロセスである．しかし，通常の逆伝播アルゴリズムでは，逆伝播によって求められた各層のすべてのユニットの微分結果を用いて，その層におけるユニットの重みを更新するという集中型の処理である．つまり，この重み更新を行うためにはすべてのセンサノードと通信を行う必要がある．これを避けるために，ユニットが後の層のユニットから微分結果を取得し，その結果のみを用いて重みを更新する分散アルゴリズムを設計する．したがって，提案手法では，各ユニットは他のユニットと重みを共有しないものとする．

出力層のユニットが実行され，正解データを取得すると，微分 δ を計算し，その結果を前の層のユニットへ送信する．微分 δ を受け取ったユニットは，δ および入力データを用いて，パラメータ勾配および前の層のユニットに送信するための δ を計算する．このとき，パラメータ勾配を用いて重みの更新も行う．

ここで出力層の δ は，出力結果を z，正解データを t とすると，以下の式によって求められる．

$$\delta = z - t \tag{5}$$

後の層から δ を受け取った後，前の層へ送信するための δ は，以下の式によって求められる．このとき．$w_i^{(t)}$ および $\delta_i^{(t)}$ は，$T-1$ 層での i 番目のユニットから T 層での j 番目のユニットへの重みおよび微分を表すとする．

$$\delta_i^{(t)} = \sum_j \delta_j^{(t+1)} (w_j^{(t+1)} f'(u_j^{(t)})) \tag{6}$$

また，以下の式によってパラメータ更新を行う．

$$w_i^{(t+1)} = w_i^{(t)} - \epsilon \delta_i^{(t)} z_i^{(t)} \tag{7}$$

ϵ は学習率で $z^{(t)}$ は前の層の出力結果を表す．

5. ノード欠損時の処理

4 章での設計は，$N \times M$ の座標すべてにセンサノードが割り当てられていることを前提としている．しかし，実環境を考慮すると，必ずしもすべての座標にセンサノードが割り当てられているとは限らない．たとえば，センサノードが 11 個しか配置されていない場合，4×3 の空間を網羅することはできない．また，センサノードの動作が必ず保証されているわけではなく，センサノードが故障する場合も考えられる．このようにセンサノードが欠損している場合でも学習処理を可能にするために，センサノード近辺の代替処理を定義する．

センサノードがセンサデータを生成するため，センサノードの欠損は，CNN の入力データを失うことを意味する．そこで，欠損部分の入力データを必要とするユニットは，その部分を 0 で補間する．この補間による入力データは，畳み込み処理の線形結合性より無視される．また，センサノードの欠損は入力データの欠損だけでなく，畳み込み層の出力結果を失うことでもある．そのため，畳み込み層の出力を必要とするユニットは，同様に 0 として補間する．この補間データは，次の層の畳み込み層およびプーリング層の処理によって無視される．次の層が畳み込み層の場合は，上述と同様に線形結合性により無視され，プーリング層の場合は，活性化関数 ReLU により入力データが非負の値であり，それに MAX プーリング処理を行うため，0 は無視される．プーリング層におけるユニットの欠損も同様である．

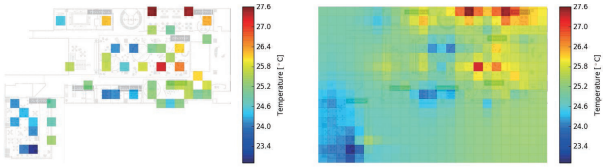
全結合層および出力層の場合は，処理を 1 つのノードに集約して行うため，このセンサノードの欠損は，順伝播および逆伝播処理を行うことができないということである．したがって，この場合には，代替ノードを設け，そのノードが処理を行うものとする．

6. 温度センシングによる異常検知と評価

6.1 学習に使用したデータセット

まず，深層学習の実行にあたり用いたデータセットについて述べる．大阪府内の 1,400 m^2 超の実ラウンジスペースに設置した 50 個の温度センサが取得したデータを学習のためのデータセットとして使用した．ラウンジの空間を 17×25 のメッシュ状の小領域に分割し，各小領域に取得された室温をマッピングした画像状のデータの集合として構成する．温度センサとして，工場の温度管理などで使用される信頼性の高い無線温度センサ RTR-503 を用いた．また設置場所に制約があり，17×25 のすべてにセンサを準備することができなかったため，センサの存在しない位置の温度データは実際に取得できた位置の温度データを用いて補間し，面的な温度分布データを作成した．異常温度を示したスポット以外では，比較的近いスポットどうしでも急激な温度変化は見られず，なめらかな温度変化を示すのが一般的である．そこで同様の傾向を表現するための補間方法を用いた．補間の方法として，50 個の温度センサから取得した実温度データに対し，補間位置から実温度データまでのマンハッタン距離の逆数を重みとして与え，すべての実温度データに対して重み平均をとることで，温度データの補間を行った．この結果，補間されたデータはすべて正常に分類されるため，異常検知性能に影響を与えることはないと考える．一例として，ある時刻における実際に取得できた計測値のみから構成した室温マップと，補間処理によって作成した室温マップをそれぞれ**図 3** (a) と図 3 (b) に示す．本研究では 2016 年 8 月 26 日から 10 月 27 日までの間に 30 分間隔で記録された 2,961 個の室温マップを深層学習に使用した．

このような二次元の温度データに対し，本研究では最終



(a) 補間前 (b) 補間後

図 3 ラウンジの温度データ一例：(a) 補間前，(b) 補間後

Fig. 3 Temperature Sensing in Lounge; (a) Raw Data, (b) Complemented Data temperature.

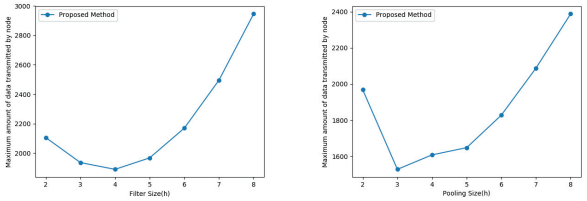
的に人の快・不快，機器の故障といった温度分布からでは判別が困難な対象を深層学習により識別することを目的としている．しかしながら，現在我々のデータセットでは温度情報のみが得られており，その空間での通常/異常状態を示す正解ラベルは付与されていない．したがって，本評価では，太陽光や壁，空調設備からの距離などにより周囲と比較して温度差が大きいスポットが発生したときに居住者が不快感を持つと仮定し，それを異常状態と定義したうえで，異なるハイパーパラメータ設定において異常状態を正しく検知できるかどうかを評価する．本研究では，収集した 2,961 セットの室温マップに対し，室温の不均衡が人の快適度に影響を及ぼすという想定の下，5 × 5 の部分空間内の温度データの分散が 1.0 以上，すなわち局所的に温度分布の偏りが存在する場合は異常，そうでない場合は正常とした正解ラベルを機械的に付与した．この設定のもとでは，2,961 個のデータセットに対し，およそ 17.46%のデータが異常として判定される．

6.2 分散学習によるノードのデータ通信コストの変化

まず，ノードのデータ通信コストという点において，ハイパーパラメータを変更したときにデータ通信量がどのように変化するのかを示す．ノードのデータ通信コストは，1 つのセンシングデータを送信または受信したときを 1 とする．

これにあたり用いたネットワークは．25 × 17 の入力層，畳み込み 2 層，プーリング 1 層，全結合層 1 層，出力層 1 層からなるネットワークである．学習回数および畳み込みのフィルタ数は，ノードのデータ通信コストの増加率に影響しないため一定とする．ノードのデータ通信コストの変化をみるために，1 台の PC 上で 25 × 17 のノードによる分散学習を実行できるプログラムを仮想的に python で実装した．この仮想環境内では，データ通信をプログラム内でのデータの受け渡しによって表現している．そして，これらのセンサを仮想環境内で分散学習をし，シミュレーションを行った．

その結果，フィルタサイズおよびプーリングサイズを変更したときのノードのデータ通信コストの変化はそれぞれグラフ図 4 のようになった．このことから，フィルタサイズおよびプーリングサイズを大きくすると，データ通信先のノード数が増加するため，ノードのデータ通信コストは増加することが分かった．



(a) フィルタサイズ $h^{(t)}$ (b) プーリングサイズ $s^{(t)}$

図 4 フィルタサイズ $h^{(t)}$ およびプーリングサイズ $s^{(t)}$ を変更したときの最大データ通信コストの変化

Fig. 4 Maximal Communication cost of Sensor Node (vs. filter and pooling size).

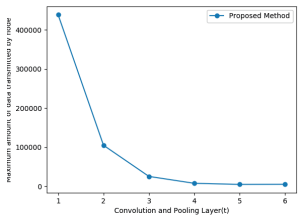


図 5 隠れ層 T を変更したときの最大データ通信コストの変化

Fig. 5 Maximum Communication Cost of Sensor Node (vs. the number T of layers).

次に，ノードのデータ通信コストという点において，ニューラルネットワークを変更したときにノードのデータ通信コストがどのように変更するかを示す，図 5 に隠れ層 T の数を変化させたときのノードのデータ通信コストの変化を示す．この結果から，畳み込みやプーリングによってニューラルネットワークが深くなるほど，出力層へのデータサイズが縮小されるため，ノードのデータ通信コストは減少することが分かる．

6.3 学習精度の評価

提案手法の課題として，ノードのデータ通信コストと学習精度をできるだけ最適にするような CNN のハイパーパラメータを選択する必要があるということがあげられる，また，提案手法では，通常の CNN と異なり，畳み込みのフィルタのパラメータ更新をユニットごとに行うため，それによる学習精度の低下が懸念される．そこで通常の CNN との学習精度の比較を行った．

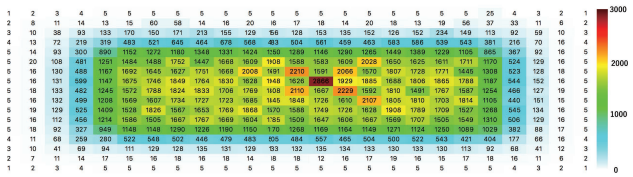
学習精度の評価を行うにあたり，データセットのうち 9 割をトレーニングデータ，1 割をテストデータとした．そしてハイパーパラメータはそれぞれ下記のように設定した．提案手法による分散学習の場合，フィルタ数：$K^{(1)} = K^{(2)} = 2$，フィルタサイズ：$h^{(1)} = h^{(2)} = 3$，プーリングサイズ：$s^{(2)} = 2$，パディングサイズ：0，全結合層のユニット数：150，バッチサイズ：3，学習回

数：7 とした．これに対し，通常の CNN の場合，フィルタ数：$K^{(1)} = K^{(2)} = 10$，フィルタサイズ：$h^{(1)} = h^{(2)} = 5$，プーリングサイズ：$s^{(2)} = 2$，パディングサイズ：0，全結合層のユニット数：300，バッチサイズ：3，学習回数：10 とし，学習精度が最適となるよう設定した．ハイパーパラメータを決定するにあたっての評価指標として，学習精度，通信コスト，実行時間などがあげられる．通常の CNN のハイパーパラメータでは，学習精度が最適となるように設定した．分散型の CNN のハイパーパラメータでは，学習精度と通信コストの指標を用いて，通信コストを削減しつつ，学習精度の低下を抑えるように設定した．この際，フィルタサイズ，プーリングサイズは分散型の実装の都合上，上記のパラメータでしか実装できていないため，この設定とした．ユニット数は全結合層のユニットに集約後に関わる値であり，またバッチサイズは一度にまとめてデータを送信するか否かの違いであるために通信コストは変わらない．したがって，学習精度が最適となるように設定した．フィルタ数，および学習回数は値の増加が直接通信コストの増加に結び付くため，この値を通信コストと学習精度の観点から設定する．学習回数については，7 回までは精度が向上するが，それ以降の学習精度の向上率が低いため，精度はほとんど変化することなく，通信コストを削減できる値として 7 に設定した．フィルタ数については，精度が大幅に低下しない（たかだか 2%程度）かつ分散型の場合のパラメータと比較して十分な通信コストを削減できる値に設定した．また，ハイパーパラメータのほかに，CNN 構造も学習精度や通信コストに影響を与える要因となるが，今回は評価の複雑化を回避するために固定とした．また，学習精度を比較する際，訓練データ数が比較的少ないため，学習開始時にランダムに設定する各層の重みの初期値によって学習精度は計算ごとに異なる．そこで，各条件で 10 回ずつ学習と評価を行った結果を平均化したものを学習精度とした．

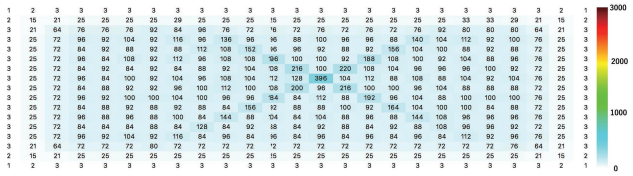
その結果，提案手法は 95.570%，通常の CNN は 97.095%となった．また，WSN のノード間のリンクが直交と斜め 4 方向のノード間で通信接続されたトポロジを用いたときの各センサノードの通信コストの評価を行った．

分散学習において，データを中心に集約し，最適なハイパーパラメータの選択を行った場合，各センサノードの通信コストは増加し，**図 6** (a) で示されるように，赤色で強調される中心が最も通信コストが増加し，そのコストは 2,866 となった．一方，提案手法におけるハイパーパラメータの選択を行った場合，図 6 (b) で示されるように，最大通信コストにかなりの減少がみられ，そのコストは 396 となった．

次に，提案手法と通常の CNN の学習において，同じハイパーパラメータの選択をした場合の学習精度の比較を行った．その結果，提案手法では 95.570%，通常の CNN では 95.6%となり，ほぼ同じ結果となった．このことから，提案手法における，重み共有を行わない畳み込みフィルタのパラメータ更新でも，通常の CNN と同程度の学習精度を達成可能であることが確認できた．



(a) 最適なパラメータ選択の場合



(b) 制約の中でのパラメータ選択の場合

図 6 各センサノードのデータ通信コスト（温度データ）
Fig. 6 Communication Costs of Sensor Nodes (Temperature).

6.4 センサ NW のノード欠損による影響

5 章で述べたように，実環境を考慮すると，センサノードの故障に対処できるようにすることは重要な課題となる．センサノードの故障によるノードの欠損は，CNN のユニットが欠損した不完全な状態での学習となる，この状態での学習を定量化するために，25 × 17 個のセンサノードから意図的に一部のセンサノードを欠如させる．ノードが欠損した場合のユニットの出力としては，5 章で述べた方法を用いる．

25 × 17 個のセンサノードのうち，ランダムに 1 割（42 個）のノードを欠如させた場合，学習精度は 94.9%となった．同様にランダムに 2 割のノードを欠如させた場合の学習精度は 94.4%であった．また，実環境における欠損として，災害などによって一定範囲内のセンサノードが局所的に欠損する場合も考えられる．そこで，25 × 17 個のセンサノードのうち，座標の左上，右上，左下，右下の 4 隅の 9 × 9 個のノードを局所的に欠如させた場合の評価を行った．その結果，左上の欠損は 94.595%，右上の欠損は 93.581%，左下の欠損は 94.932%，右下の欠損は 96.014%となった．左隅の欠損はランダムの欠損と比較して同等の精度を示しており，右上の欠損はランダムと比較して精度が低下し，右下の欠損は精度が上昇した結果となった．図 3 (b) から分かるとおり，異常スポットは右上に多く発生している．そのため，右上の欠損では，重要な特徴量の欠損につながり，ランダムの欠損と比較して精度が低下したと考える．また，右下では，温度変化がなく学習にあまり影響を与えない特徴のないデータであったため，精度が向上したと考える．局所的欠損における 4 隅の精度の平均をとったとき 94.780%となり，ランダムの欠損と同程度であることが分

かった．欠損による影響はランダムにおいても局所的においても重要な特徴量を持つデータがいかに欠損しているかが最も影響を与える要因であると考えられる．

また，ノード間リンクが直交と斜めのトポロジであるWSN内のセンサノード20%，そしてそのうちのリンクの15%をノード間リンクが途切れないように一様に欠損させる．このときのWSNを**図 7**に示す．このWSNでの分散学習において，最適なハイパーパラメータの選択を行った場合，各センサノードの通信コストは**図 8** (a) で示され，最大通信コストは2,572となった．一方，提案手法におけるハイパーパラメータの選択を行った場合，各センサノードの通信コストは図 8 (b) で示され，最大通信コストは329となった．この結果から，センサノードやノード間リンクの欠損のある実環境に近い状況においても，最大通信コストにかなりの減少がみられ，大幅に学習精度が低下することはないといえる．これはセンサノードの入力の欠損は，学習精度にそれほど影響をあたえるものではなく，欠損した入力データが，特徴量の決定的な部分ではない限り，その入力データを必要とする周囲のユニットの出力には大きく影響しない．また，リンク欠損によって，通信経路が長くなる分，途中経路のノードの通信コストは増えるが，最大通信コストはノード欠損による集約データ量が減少する分，欠損がない場合と比較してそれほど変わりがないといえる．

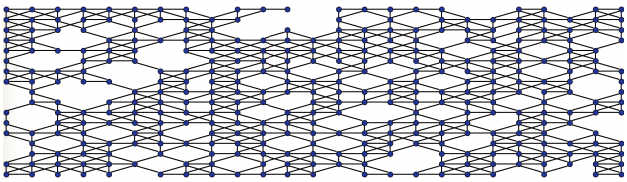
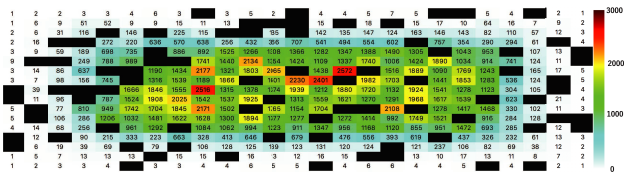
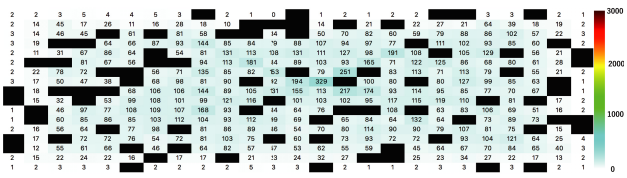


図 7　センサノードの欠損を含むWSN
Fig. 7　WSN with sensor node holes.



(a) 最適なパラメータ選択の場合



(b) 制約の中でのパラメータ選択の場合

図 8　欠損のあるWSNにおける各センサノードのデータ通信コスト（温度データ）
Fig. 8　Communication costs of sensor nodes with holes (Temperature).

7. モーションセンサアレイを用いた異常行動検知と評価

7.1 システムアーキテクチャ

焦電型の赤外線センサを備えたセンサノードを壁面にメッシュ状に配置し，無線通信を介してデータを収集した．そのデータを元に，壁面の前を通過する人が転倒した場合，これを検知するシステムを実データを入力データとし，転倒とそれ以外の二値分類の判定を行う深層学習を，6章と同様，1台のPC上で6 × 6のノードを想定した分散学習をPythonで実装し，その性能の評価を行った．

7.1.1 システムの概要

システムの概要を**図 9**に示す．現在は実装中であるが，CNNの分散実行処理を行うセンサノードとしてIntel Edisonを36台使用し，6 × 6のグリッド状に配置することを想定している．センサノードに接続される赤外線センサは，Edisonと有線接続することも可能であるが，設置場所の自由度を考慮し，無線で接続するものとした．各センサノードはグリッド上で，自身が配置された位置の上下左右，斜め方向に隣接する周囲8台のEdisonとデータの交換を行い，CNNによる学習を自律分散的に実行する．

7.1.2 センサアレイの実装

センサノードが処理するデータ源として，本研究では検知範囲内の熱源の変化をアナログ電圧として出力する焦電型の赤外線センサ素子を使用した．使用した焦電型赤外線センサ素子は検知範囲の熱源の増減に反応する特性を有する．具体的にはセンサの検知範囲内の熱源配置に変化がない場合，センサ素子の出力電圧は一定であるが，検知範囲内を人が通過すると出力電圧は大きく変化し，時間経過とともに定常状態の出力電圧へと振動しながら収束する．例として，1つのセンサの前を人が歩行しながら通過した際の出力電圧の時間変化を**図 10**に示す．使用した赤外線センサ素子をモジュール化したものを**図 11**に示す．このセンサは2.4 GHz帯無線を使用した無線マイコンであるTWELITEを使用したデータ送信機能を備えている．このモジュールは，赤外線センサ素子がアナログ出力する

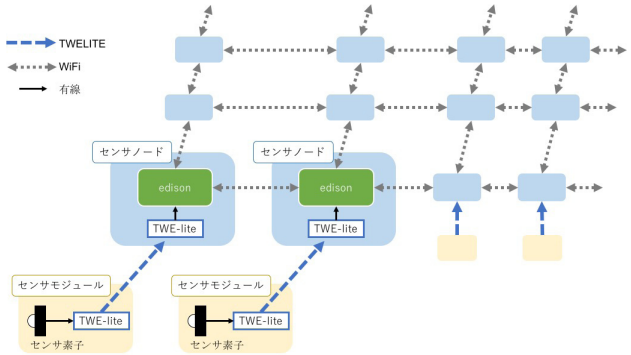


図 9　システムの概要
Fig. 9　System architecture.

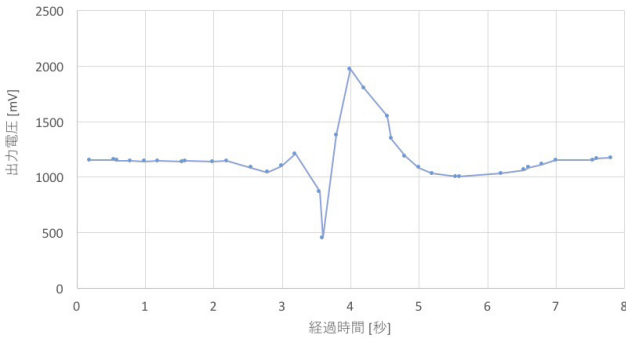


図 10 人が通過する際のセンサの出力電圧
Fig. 10 Output voltage of sensor When a person pass through.

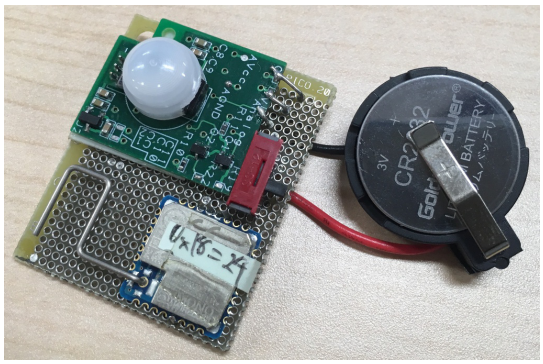
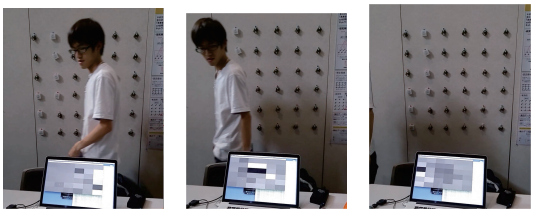


図 11 作成したセンサモジュール
Fig. 11 Sensor module.

図 12 動作検知のための IR センサアレイ
Fig. 12 Prototyped IR-sensor array for motion detection.

0–2.4 V の範囲の電圧を物理的に離れた場所に設置されたセンサノードに向け，毎秒約 5 回送信する．なお，このセンサモジュールは無線送信部を含めて 3.3 V のボタン電池 1 個で駆動し，大きさは縦 4.2 cm，横 3.5 cm となっている．本研究では，センサノードと同数である 36 個の赤外線センサモジュールを作成し，**図 12** に示すように横 30 cm，縦 20 cm の間隔で縦横 6 個ずつグリッド状に配置しセンサアレイを構築した．なお，本来であれば 1 個の赤外線センサモジュールは 1 対 1 で対応するセンサノードに向けてデータを送信する．しかし，後述の評価実験においては，CNN を用いた分散実行を仮想的に 1 つのマシンでシミュレートした．作成したシミュレータは全センサモジュールからのデータを単一のマシンに集中させたうえで処理している．処理を簡略化するため，本実験のセンサデータは 36 個のモジュールは同一の集約先に向けてデータを送信している．

図 13 試作したセンサノード
Fig. 13 Sensor node.



(a) (b) (c)

図 14 センサからの入力データの視覚化
Fig. 14 Visualization of inputs from sensor array.

7.1.3 センサノードの試作

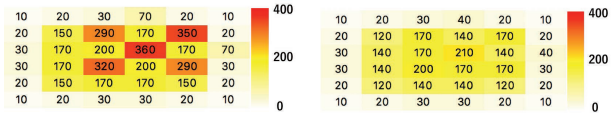
実装中のシステムにおいては，収集したデータのプロセッシング処理を行うセンサノードとして Intel Edison を使用する．Breakout Board を介した Edison の GPIO と TWELITE のチップを接続したセンサノードの試作品を**図 13** に示す．

7.2 学習に使用したデータセット

このセンサアレイを用いて，5 人の被験者から 55 回の歩行サンプルを取得した．センサアレイの前を実際に人が通過した際の出力電圧の変化を PC に表示させた様子を**図 14** に示す．このうち 23 回は検知対象である転倒のサンプルとなる．そして転倒のサンプルには異常のラベル付けを，それ以外のサンプルには正常のラベル付けを行った．取得したデータは毎秒約 5 フレームの 6×6 の動画としてみることはでき，今回は，2 秒で人の通過を検知できるとして，10 フレームを 1 データとした．また，1 フレームごとにスライディングさせてデータの増強を行うことで，$6 \times 6 \times 10$ のデータ，1,600 個を CNN の入力データとした，そのうち 1,070 個は正常，530 個は異常データ（転倒）とした．

7.3 学習精度の評価

学習精度の評価を行うにあたって，畳み込み層 1 層，プーリング層 1 層，全結合層 1 層，出力層 1 層のネットワークを用いた．CNN のハイパーパラメータとしてはそれぞれ以下のように設定した．提案手法による分散学習の場合，



(a) 最適なパラメータ選択の場合 (b) 制約の中でのパラメータ選択の場合

図 15 各センサノードのデータ通信コスト（人の動きデータ）
Fig. 15 Communication cost of sensor node (Motion).

フィルタ数：$K^{(1)} = 5$，フィルタサイズ：$h^{(1)} = 3$，プーリングサイズ：$s^{(1)} = 2$，パディングサイズ：0，全結合層のユニット数：50，バッチサイズ：1，学習回数：10 とした．これに対し，最適なハイパーパラメータとする通常の CNN の場合，フィルタ数：$K^{(1)} = 20$，フィルタサイズ：$h^{(1)} = 3$，プーリングサイズ：$s^{(1)} = 2$，パディングサイズ：0，全結合のユニット数：70，バッチサイズ：4，学習回数：10 とした，ハイパーパラメータの設定として 6 章と同様に，フィルタサイズおよびプーリングサイズは実装上，上記の値で固定され，ユニット数およびバッチサイズは学習精度が最適となるように設定した．学習回数については，少なくすると，学習精度が安定しないため，最適なものと同じ設定とした．フィルタ数については，大幅に学習精度が下がらない範囲で分散型の場合のパラメータと比較して通信コストを削減できる値に設定した．6 章と同様に，10 回ずつ学習と評価を行った結果を平均化したものを学習精度とし，WSN のノード間リンクが直交と斜め 4 方向のノード間で通信接続されたトポロジを用いた．

その結果，最適なハイパーパラメータを設定した場合において，学習精度は 91.875%，各センサノードの通信コストは**図 15** (a) で示され，最大通信コストは 360 となった．一方，提案手法において，学習精度は 89.7275%，各センサノードの通信コストは図 15 (b) で示され，最大通信コストは 210 となった．したがって，提案手法の性能としては，通常の最適化された CNN と比較して，学習精度の低下をおよそ 2%に抑えたまま，ノードのデータ通信コストをおよそ 40%削減できることが確認された．

また，通常の CNN において提案手法と同じハイパーパラメータを設定した場合，学習精度は 90.4775%となり，提案手法における重み共有を行わないパラメータ更新でも，精度の差は 1%未満であり，通常の CNN と同程度の学習精度を達成可能であることが確認できた．

今回の実験では，提案手法においても依然として高い精度を達成しており，分散学習を実行するにあたって，十分な精度を保ちつつ．6 章と比較して通信コスト削減率が低くなっているものの，通信コストを削減することができることが確認された．

7.4 センサ NW のノードの欠損による影響

6 章と同様に，一部のセンサが欠損している状態での評価についても行った．6 × 6 のセンサノードのうちランダムに 1 割（3 個）および 2 割（7 個）のセンサノードが欠損している場合について評価を行った．その結果，1 割のノード欠損では 89.7075%，2 割のノード欠損では 89.03%となった．また，6 章と同様に 4 隅の 3 × 3 個のノードを局所的に欠如させた場合の評価を行った．その結果，左上の欠損は 92.0%，右上の欠損は 90.563%，左下の欠損は 87.153%，右下の欠損は 88.0%となった．場所によって精度に差があるが，4 隅の精度の平均は 89.429%とランダムの欠損と同程度の結果となった．局所的欠損の場所によって，欠損する特徴量が異なるため，このような結果となったと考えられる．

また，特徴量の決定的な部分の欠損として，意図的にプーリング層の処理を行っている 4 つのノードのうち 3 つを欠損させた場合，学習精度は 81.125%となった．これは重要な特徴が欠損しすぎることによって，大幅な学習精度の低下となった．しかし，このような欠損の状況は非常に稀であり，また，このような最悪な状況でも学習精度はある程度維持できているといえる．

今回の実験では，6.4 節と 7.4 節でランダム欠損および 4 隅の欠損についての評価を行ったが，今後としては，どこの場所あるいはどのような欠損が学習精度の影響を与えるかを評価するためにも，様々な欠損において精度の最低値や最高値，分散などから統計的な分析をしていきたい．

8. 考察

提案手法では，入力データが 2 次元の長方形のであり，CNN によって，隣接されたデータが畳み込みされることによりデータサイズが圧縮が可能であると予想される．WSN 上で，このデータサイズの圧縮を実現するために，マルチホップ通信をできる限り回避することが可能であるように，センサノード間が地理的に十分に隣接している必要がある．つまり，高密度な接続を満たすネットワークが望ましい．

最近では，より深い CNN が主流となってきている．たとえば，AlexNet [34] や VGG [35]，GoogLeNet [36]，ResNet [37] は，画像分類において，高いパフォーマンスが示されている．しかし，これらは深いネットワーク構造であり，パラメータ数が多く，大量の計算負荷がかかる．このように，ネットワーク構造が深いと，モバイル端末上では，学習を行うのは難しく，学習を除いた CNN の実行で精一杯だといわれている [28], [30]．一方，本研究による提案手法では，ネットワーク構造が深いほど，通信コストを削減することができることが分かっている．これは，モバイル端末上でも深いネットワークの学習ができる可能性の発見であるといえる．

しかし，実際のデバイスに提案手法による分散学習を適用するのには，依然として大きな課題がある．今回は，あ

くまでシミュレーションによる評価であり，実際には，通信品質やエネルギー消費，プロセス間の同期などの影響についても考慮する必要がある，本研究による最終的な目標は，この分散学習システムをエナジーハーベスティングなセンサに導入し，WSN 内でエネルギー効率の高い深層学習を実現することである．

9. おわりに

本研究では，無線センサネットワーク内で深層学習を実行するプロトコルを提案した．提案手法では，CNN のユニットを無線センサノードに割り当て，センサノード間の通信によって CNN の学習を分散実行する．1,400 m^2 超の実ラウンジスペースでの温度データセットおよび 6×6 の赤外線センサで収集された人の動きのデータセットを用いた性能評価実験を行った結果，提案手法は，通常の CNN と同程度の学習精度を維持しながら，十分妥当な通信コストのもとで，WSN 内で CNN を分散実行可能であることを確認した．また，一部のノードが故障した場合にも，学習精度に大きな影響を与えることなく深層学習を分散実行することが可能であることも示した．

提案手法による分散学習を実環境に適用する場合，通信品質やエネルギー消費，プロセス間の同期などの影響についても考慮する必要がある．したがって，今後の計画としては，小型のエナジーハーベスティングなセンサノードに分散学習を実装し，それによる消費電力などの影響をを測定していくことがあげられる．最終的には，この分散学習システムをエナジーハーベスティングなセンサに導入し，WSN 内でエネルギー効率の高い深層学習を実現していきたいと考えている．

謝辞 本研究成果の一部は，国立研究開発法人情報通信研究機構（NICT）の委託研究「未来を創る新たなネットワーク基盤技術に関する研究開発」ならびに JSPS 科研費 15K12019 の助成を受けたものです．

参考文献

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467 (2016).

[2] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: TensorFlow: A system for large-scale machine learning, *Proc. 12th USENIX Symposium on Operating Systems Design and Implementation* (*OSDI*), pp.265–283 (2016).

[3] Abdel-Hamid, O., Mohamed, A.R, Jiang, H. and Penn, G.: Applying Convolutional Neural Networks concepts to hybrid NN-HMM model for speech recognition, *Proc. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp.4277–4280, IEEE (online), DOI: 10.1109/ICASSP.2012.6288864 (2012).

[4] Sainath, T.N., Kingsbury, B., Mohamed, A.R, Dahl, G.E., Saon, G., Soltau, H., Beran, T., Aravkin, A.Y. and Ramabhadran, B.: Improvements to Deep Convolutional Neural Networks for LVCSR, *Proc. 2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp.315–320, IEEE (online), DOI: 10.1109/ASRU.2013.6707749 (2013).

[5] Scherer, D., Müller, A. and Behnke, S.: Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition, *Proc. 20th International Conference on Artificial Neural Networks: Part III*, *ICANN'10*, pp.92–101, Berlin, Heidelberg, Springer-Verlag (online), available from ⟨http://dl.acm.org/citation.cfm?id=1886436.1886447⟩ (2010).

[6] Huang, F.J. and LeCun, Y.: Large-scale Learning with SVM and Convolutional for Generic Object Categorization, *Proc. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (*CVPR'06*), Vol.1, pp.284–291 (online), DOI: 10.1109/CVPR.2006.164 (2006).

[7] Babenko, A., Slesarev, A., Chigorin, A. and Lempitsky, V.: Neural codes for image retrieval, *Proc. European Conference on Computer Vision*, pp.584–599, Springer (2014).

[8] Ge, T., Ke, Q. and Sun, J.: Sparse-Coded Features for Image Retrieval, *BMVC* (2013).

[9] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M.: Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602 (2013).

[10] Wang, X.: Deep Reinforcement Learning (2016).

[11] Unsupervised Feature Learning and Deep Learning, available from ⟨http://deeplearning.stanford.edu/wiki/index.php/Neural_Networks⟩.

[12] Tu, J.V.: Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes, *Journal of Clinical Epidemiology*, Vol.49, No.11, pp.1225–1231 (1996).

[13] Cuda-convnet2, available from ⟨https://code.google.com/p/cuda-convnet2/⟩.

[14] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E. and Darrell, T.: DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, *ICML*, Vol.32, pp.647–655 (2014).

[15] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks, arXiv preprint arXiv:1312.6229 (2013).

[16] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T.: Caffe: Convolutional architecture for fast feature embedding, *Proc. 22nd ACM International Conference on Multimedia*, pp.675–678, ACM (2014).

[17] Buyya, R., Calheiros, R.N. and Dastjerdi, A.V.: *Big Data: Principles and Paradigms*, Morgan Kaufmann (2016).

[18] Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Senior, A., Tucker, P., Yang, K., Le, Q.V., et al.: Large scale distributed deep networks, *Advances in Neural Information Processing Systems*, pp.1223–1231 (2012).

[19] Schmidhuber, J.: Deep learning in neural Networks: An overview, *Neural Networks*, Vol.61, pp.85–117 (2015).

[20] Ooi, B.C., Tan, K.-L., Wang, S., Wang, W., Cai, Q.,

Chen, G., Gao, J., Luo, Z., Tung, A.K., Wang, Y., et al.: SINGA: A distributed deep learning platform, *Proc. 23rd ACM International Conference on Multimedia*, pp.685–688, ACM (2015).
[21] Wang, W., Chen, G., Dinh, A.T.T., Gao, J., Ooi, B.C., Tan, K.-L. and Wang, S.: SINGA: Putting deep learning in the hands of multimedia users, *Proc. 23rd ACM International Conference on Multimedia*, ACM, pp.25–34 (2015).
[22] Zinkevich, M., Weimer, M., Li, L. and Smola, A.J.: Parallelized stochastic gradient descent, *Advances in Neural Information Processing Systems*, pp.2595–2603 (2010).
[23] Wu, R., Yan, S., Shan, Y., Dang, Q. and Sun, G.: Deep image: Scaling up image recognition, arXiv preprint arXiv:1501.02876, Vol.7, No.8 (2015).
[24] Agarwal, A. and Duchi, J.C.: Distributed delayed stochastic optimization, *Advances in Neural Information Processing Systems*, pp.873–881 (2011).
[25] Recht, B., Re, C., Wright, S. and Niu, F.: Hogwild: A lock-free approach to parallelizing stochastic gradient descent, *Advances in Neural Information Processing Systems*, pp.693–701 (2011).
[26] GitHub: DMTK, available from ⟨https://github.com/Microsoft/DMTK⟩.
[27] GitHab: ChainerMN, available from ⟨https://github.com/chainer/chainermn⟩.
[28] Mao, J., Chen, X., Nixon, K.W., Krieger, C. and Chen, Y.: MoDNN: Local distributed mobile computing system for Deep Neural Network, *2017 Design, Automation & Test in Europe Conference & Exhibition* (*DATE*), pp.1396–1401, IEEE (2017).
[29] Lane, N.D., Bhattacharya, S., Georgiev, P., Forlivesi, C., Jiao, L., Qendro, L. and Kawsar, F.: Deepx: A software accelerator for low-power deep learning inference on mobile devices, *Proc. 15th ACM/IEEE International Conference on Information Processing in Sensor Networks* (*IPSN*), pp.1–12, IEEE (2016).
[30] Huynh, L.N., Lee, Y. and Balan, R.K.: DeepMon: Mobile GPU-based Deep Learning Framework for Continuous Vision Applications, *Proc. 15th Annual International Conference on Mobile Systems, Applications, and Services*, pp.82–95, ACM (2017).
[31] Rabbani, A. and Keshav, S.: The SPOT* Personal Thermal Comfort System, *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments, BuildSys '16*, pp.75–84, ACM (online), DOI: 10.1145/2993422.2993578 (2016).
[32] Chiguchi, M., Yamaguchi, H., Higashino, T. and Shimoda, Y.: Human thermal comfort estimation in indoor space by crowd sensing, *2016 IEEE International Conference on Smart Grid Communications* (*SmartGridComm*), pp.45–50 (online), DOI: 10.1109/SmartGridComm.2016.7778736 (2016).
[33] Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C.: *Introduction to Algorithms, 3rd Edition*, The MIT Press (2009).
[34] Krizhevsky, A., Sutskever, I. and Hinton, G.E.: Imagenet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, pp.1097–1105 (2012).
[35] Simonyan, K. and Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).
[36] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A.: Going deeper with convolutions, *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–9 (2015).
[37] He, K., Zhang, X., Ren, S. and Sun, J.: Deep residual learning for image recognition, *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp.770–778 (2016).

福島 悠太 （学生会員）

平成 29 年大阪大学基礎工学部情報科学科卒業．同年同大学大学院情報科学研究科博士前期課程進学．センサーネットワークにおける分散型深層学習に関する研究に従事．

山口 弘純 （正会員）

平成 6 年大阪大学基礎工学部情報工学科卒業．平成 10 年同大学大学院基礎工学研究科博士後期課程修了．平成 19 年より同大学大学院情報科学研究科准教授．博士（工学）．モバイルコンピューティングに関する研究に従事．電子情報通信学会，IEEE 各会員．

東野 輝夫 （正会員）

昭和 54 年大阪大学基礎工学部情報工学科卒業．昭和 59 年同大学大学院基礎工学研究科博士後期課程修了．同年同大学助手．現在，同大学大学院情報科学研究科教授．工学博士．分散システム，通信プロトコル，モバイルコンピューティング等の研究に従事．電子情報通信学会，ACM 各会員．IEEE Senior Member．本会フェロー．