(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7453563号
(P7453563)

(45)発行日　令和6年3月21日(2024.3.21)

(24)登録日　令和6年3月12日(2024.3.12)

(51)国際特許分類
G06N 3/08 (2023.01)

FI
G06N 3/08

請求項の数　9　（全18頁）

(21)出願番号　特願2021-569687(P2021-569687)
(86)(22)出願日　令和2年1月10日(2020.1.10)
(86)国際出願番号　PCT/JP2020/000644
(87)国際公開番号　WO2021/140643
(87)国際公開日　令和3年7月15日(2021.7.15)
審査請求日　令和4年7月11日(2022.7.11)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100094525
弁理士　土井 健二
(74)代理人　100094514
弁理士　林 恒徳
(72)発明者　檀上 匠
神奈川県川崎市中原区上小田中４丁目１番１号 富士通株式会社内
審査官　渡辺 順哉

最終頁に続く

(54)【発明の名称】　ニューラルネットワークシステム、ニューラルネットワークの学習方法及びニューラルネットワークの学習プログラム

(57)【特許請求の範囲】

【請求項１】

メモリと、

前記メモリにアクセスする複数のプロセッサとを有し、

前記複数のプロセッサそれぞれは、複数回の学習それぞれにおいて、

訓練データの入力とニューラルネットワーク内のパラメータとに基づいて前記ニューラルネットワークの演算を実行して前記ニューラルネットワークの出力を算出し、前記算出した出力と前記訓練データの教師データとの差分の前記パラメータに対する勾配または前記勾配に基づく更新量を算出し、

前記勾配または前記更新量の累積が閾値未満でない第１の場合、前記複数のプロセッサが、それぞれ算出した複数の前記勾配または前記更新量の累積を、前記複数のプロセッサ内の他のプロセッサに送信して前記複数の勾配または更新量の累積を集約し、前記集約された勾配または更新量の累積を受信し、前記集約された勾配または更新量の累積で前記パラメータを更新する第１の更新処理を実行し、

前記勾配または前記更新量の累積が前記閾値未満である第２の場合、前記複数のプロセッサが、前記送信による前記複数の勾配または更新量の累積の集約を行わず、前記複数のプロセッサそれぞれが算出した前記勾配または更新量でそれぞれのパラメータを更新する第２の更新処理を実行する、ニューラルネットワークシステム。

【請求項２】

前記複数のプロセッサそれぞれは、更に、

前記第１の場合、前記第２の更新処理を連続して行った学習回数が第１の基準回数未満の場合には、前記第２の更新処理を実行する、請求項１に記載のニューラルネットワークシステム。

【請求項３】

前記複数のプロセッサそれぞれは、更に、

前記第２の場合、前記学習回数が前記第１の基準回数より多い第２の基準回数未満でない場合には、前記第1の更新処理を実行する、請求項２に記載のニューラルネットワークシステム。

【請求項４】

前記複数のプロセッサそれぞれは、更に、

前記学習回数が前記第１の基準回数未満でなく且つ前記第２の基準回数未満の場合、前記第１の場合に前記第１の更新処理を実行し、前記第２の場合に前記第２の更新処理を実行する、請求項３に記載のニューラルネットワークシステム。

【請求項５】

前記第１の場合では、前記複数のプロセッサそれぞれが算出した複数の勾配または更新量の累積のうち少なくとも１つの勾配または更新量の累積が前記閾値未満でない、請求項１に記載のニューラルネットワークシステム。

【請求項６】

前記複数の勾配または更新量の集約は、前記複数の勾配または更新量の累積を加算すること、前記複数の勾配または更新量の累積の最大値を求めることのいずれかである、請求項１に記載のニューラルネットワークシステム。

【請求項７】

前記集約された勾配または更新量の累積は、前記複数の勾配または更新量を前記第２の更新処理の回数分累積し、前記複数の勾配または更新量の累積を平均化した値である、請求項１に記載のニューラルネットワークシステム。

【請求項８】

複数のプロセッサそれぞれが、複数回の学習それぞれにおいて、

訓練データの入力とニューラルネットワーク内のパラメータとに基づいて前記ニューラルネットワークの演算を実行して前記ニューラルネットワークの出力を算出し、前記算出した出力と前記訓練データの教師データとの差分の前記パラメータに対する勾配または前記勾配に基づく更新量を算出し、

前記勾配または前記更新量の累積が閾値未満でない第１の場合、前記複数のプロセッサが、それぞれ算出した複数の前記勾配または前記更新量の累積を、前記複数のプロセッサ内の他のプロセッサに送信して前記複数の勾配または更新量の累積を集約し、前記集約された勾配または更新量の累積を受信し、前記集約された勾配または更新量の累積で前記パラメータを更新する第１の更新処理を実行し、

前記勾配または前記更新量の累積が前記閾値未満である第２の場合、前記複数のプロセッサが、前記送信による前記複数の勾配または更新量の累積の集約を行わず、前記複数のプロセッサそれぞれが算出した前記勾配または更新量でそれぞれのパラメータを更新する第２の更新処理を実行する、ニューラルネットワークの学習方法。

【請求項９】

ニューラルネットワークの学習を複数のプロセッサに実行させるニューラルネットワークの学習プログラムにおいて、

前記学習は、

複数のプロセッサそれぞれが、複数回の学習それぞれにおいて、

訓練データの入力とニューラルネットワーク内のパラメータとに基づいて前記ニューラルネットワークの演算を実行して前記ニューラルネットワークの出力を算出し、前記算出した出力と前記訓練データの教師データとの差分の前記パラメータに対する勾配または前記勾配に基づく更新量を算出し、

前記勾配または前記更新量の累積が閾値未満でない第１の場合、前記複数のプロセッサ

が、それぞれ算出した複数の前記勾配または前記更新量の累積を、前記複数のプロセッサ内の他のプロセッサに送信して前記複数の勾配または更新量の累積を集約し、前記集約された勾配または更新量の累積を受信し、前記集約された勾配または更新量の累積で前記パラメータを更新する第１の更新処理を実行し、

前記勾配または前記更新量の累積が前記閾値未満である第２の場合、前記複数のプロセッサが、前記送信による前記複数の勾配または更新量の累積の集約を行わず、前記複数のプロセッサそれぞれが算出した前記勾配または更新量でそれぞれのパラメータを更新する第２の更新処理を実行する、ニューラルネットワークの学習プログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、ニューラルネットワークシステム、ニューラルネットワークシステムの学習方法及びニューラルネットワークシステムの学習プログラムに関する。

【背景技術】

【０００２】

ニューラルネットワークは、例えば、複数の入力にそれぞれの重みを乗算し、その複数の乗算値を加算した値を出力層のニューロンの活性化関数に入力し、活性化関数の出力を出力する構成を有する。このような単純な構成のニューラルネットワークは、単純パーセプトロンと称される。さらに、上記の単純な構成を複数層有し、ある層の出力を他の層に入力するニューラルネットワークは、多層パーセプトロンと称される。また、ディープニューラルネットワークは、多層パーセプトロンのように、入力層と出力層との間に複数の隠れ層を有する。以下、ニューラルネットワークは略してNN（Neural Network）と称する。

【０００３】

NNは、大量の訓練データを用いて学習することで、前述の重みなどのパラメータを最適化する。訓練データの数が多いほどモデルの精度を高めることができるが、一方で、学習回数が増大し学習に要する演算時間が長くなるという問題がある。

【０００４】

NNの学習に要する演算時間を短くする方法として、訓練データを分割し、複数の演算ノードで並列に学習の演算を実行するデータ並列型分散学習が提案されている。データ並列型分散学習については、例えば、以下の特許文献に記載されている。

【先行技術文献】

【特許文献】

【０００５】

【文献】特開２０１８－１２０４７０号公報

【文献】特開２０１２－７９０８０号公報

【発明の概要】

【発明が解決しようとする課題】

【０００６】

データ並列型分散学習では、訓練データを演算ノードの数で分割し、複数の演算ノードがそれぞれの訓練データに基づいて学習の演算を実行して、NNのパラメータについてNNの出力の誤差関数の勾配を算出し、勾配に学習率を乗じたパラメータの更新量を算出する。その後、演算ノードは、各ノードの更新量の平均を求め、全演算ノードが、更新値の平均でパラメータを更新する。複数の演算ノードがそれぞれの訓練データで並列に学習演算を行うことで、単一の演算ノードが単一の訓練データで学習演算するよりも学習に要する演算時間を短くできる。

【０００７】

しかしながら、複数の演算ノードそれぞれが算出した更新量の平均を求めるためには、複数の演算ノードそれぞれが算出した更新量を加算するなどして集約し、集約した加算値を複数の演算ノードで共有する必要がある。そして、集約するときと共有するときに、複

数の演算ノード間でデータの通信処理が行われる。
【０００８】
その結果、データ並列型分散学習は、訓練データを並列に演算することで学習に要する演算時間を短縮するが、学習工程毎に行われる演算ノード間の通信処理の時間により、演算時間の短縮効果が抑制される。
【０００９】
そこで、本実施の形態の第1の側面の目的は、データ並列型分散学習のスループットを向上させるニューラルネットワークシステム、ニューラルネットワークの学習方法及びニューラルネットワークの学習プログラムを提供することにある。
【課題を解決するための手段】
【００１０】
本実施の形態の第１の側面は、メモリと、前記メモリにアクセスする複数のプロセッサとを有し、前記複数のプロセッサそれぞれは、複数回の学習それぞれにおいて、
訓練データの入力とニューラルネットワーク内のパラメータとに基づいて前記ニューラルネットワークの演算を実行して前記ニューラルネットワークの出力を算出し、前記算出した出力と前記訓練データの教師データとの差分の前記パラメータに対する勾配または前記勾配に基づく更新量を算出し、
前記勾配または前記更新量の累積が閾値未満でない第１の場合、前記複数のプロセッサが、それぞれ算出した複数の前記勾配または前記更新量の累積を、前記複数のプロセッサ内の他のプロセッサに送信して前記複数の勾配または更新量の累積を集約し、前記集約された勾配または更新量の累積を受信し、前記集約された勾配または更新量の累積で前記パラメータを更新する第１の更新処理を実行し、
前記勾配または前記更新量の累積が前記閾値未満である第２の場合、前記複数のプロセッサが、前記送信による前記複数の勾配または更新量の累積の集約を行わず、前記複数のプロセッサそれぞれが算出した前記勾配または更新量でそれぞれのパラメータを更新する第２の更新処理を実行する、ニューラルネットワークシステムである。
【発明の効果】
【００１１】
第１の側面によれば、データ並列型分散学習のスループットが向上する。
【図面の簡単な説明】
【００１２】
【図１】本実施の形態におけるNNシステムの構成例を示す図である。
【図２】一般的なNNにおける学習の処理例を示す図である。
【図３】データ分割型分散学習における複数の演算ノードの処理のフローを示す図である。
【図４】データ分割型分散学習における複数の演算ノードの具体的処理を示すフローチャート図である。
【図５】Reduce処理とAllreduce処理の一般的な例を示す図である。
【図６】NNの学習をデータ分割型分散学習で行った場合のReduce処理とAllreduce処理の例を示す図である。
【図７】第１の実施の形態によるデータ並列型分散学習のフローチャート図である。
【図８】第２の実施の形態によるデータ並列型分散学習のフローチャート図である。
【図９】第２の実施の形態における学習の更新サイクルの変化例を示す図である。
【発明を実施するための形態】
【００１３】
［本実施の形態のニューラルネットワークシステム］
図１は、本実施の形態におけるNNシステムの構成例を示す図である。NNシステム１は、例えば、ハイパフォーマンスコンピュータであり、メインプロセッサ（CPU：Central Processing Unit）１０と、メインメモリ１２と、サブプロセッサモジュール１３と、ネットワークNETとのインタフェース１８とを有する。サブプロセッサモジュール１３は、例えば、４つの演算ノードモジュールを有し、各演算ノードモジュールは、サブプロセッ

サ１４とサブプロセッサがアクセスするメモリ１６とを有する。

【００１４】

更に、NNシステム１は、大容量ストレージである補助記憶装置２０－２６を有し、補助記憶装置には、NN学習プログラム２０と、NNプログラム２２と、訓練データ２４と、パラメータ２６が記憶される。NN学習プログラム２０は、プロセッサ１０、１４により実行され、訓練データを利用した学習の処理を行う。また、NNプログラム２２は、プロセッサ１０、１４により実行され、NNモデルの演算を実行する。訓練データ２４は、それぞれ入力と教師データであるラベルを有する複数のデータである。パラメータ２６は、例えば学習により最適化されるNN内の複数の重みである。

【００１５】

メインプロセッサ１０は、NN学習プログラム２０を実行し、複数の訓練データによるNN学習の演算を、複数のサブプロセッサ１４に分散して並列に実行させる。４つのサブプロセッサ１４は、プロセッサチップからなる演算ノードであり、バス２８を介して互いに通信可能に構成される。

【００１６】

NNシステム１は、ネットワークNETを介して、NNのプラットフォームをクライアント端末３０、３２に提供することができる。NNシステム１は、図１の構成以外に、複数のコンピュータを演算ノードとし、複数のコンピュータが互いに通信可能な構成であってもよい。

【００１７】

［NNの学習］

図２は、一般的なNNにおける学習の処理例を示す図である。図２のNNは、入力層IN_Lと、３つの隠れ層であるニューロン層NR_L1～NR_L3を有する。第３のニューロン層NR_L3は出力層を兼ねる。

【００１８】

NNの学習の処理の概略は次の通りである。プロセッサ１０又は１４は、ストレージ内の訓練データ２４から、入力データとラベルを有する一つの訓練データを読み出し、入力層IN_Lに入力データを入力する。プロセッサは、NN学習プログラムを実行し、入力された訓練データを使用して第１のニューロン層NR_L1の演算を実行し、その演算結果を第２のニューロン層NR_L2に入力する。更に、プロセッサは、演算結果を使用して第２のニューロン層NR_L2の演算を実行し、その演算結果を第３のニューロン層NR_L3に入力し、最後に、その演算結果について第３のニューロン層NR_L3の演算を実行し、演算結果を出力する。入力層IN_Lの入力に対して３つのニューロン層NR_L1～NR_L3が順番にそれぞれの演算を実行することは、順伝播処理FWと称される。

【００１９】

一方、プロセッサは、訓練データのラベル（正解データ）と第３のニューロン層NR_3の出力との差分E3を演算し、差分E3をニューロン層NR_L3内のパラメータｗで微分して勾配ΔE3を求める。プロセッサは、差分E3から第２のニューロン層NR_L2での差分E2を算出し、差分E2をニューロン層NR_L2内のパラメータｗで微分して勾配ΔE2を求める。そして、プロセッサは、差分E2から第１のニューロン層NR_L1での差分E1を算出し、差分E1をニューロン層NR_L1内のパラメータｗで微分して勾配ΔE1を求める。更に、出力層である第３のニューロン層NR_L3の出力と正解値のラベルとの差分E3を第２、第１のニューロン層に伝播しながら、各層NR_L3、NR_L2、 NR_L1で勾配ΔE3、ΔE2、ΔE1を順番に演算することは、逆伝播処理BWと称される。

【００２０】

一般に、各ニューロン層の演算処理では、各層への入力と複数のパラメータｗとでそれぞれの演算が行われる。教師あり学習では、各学習で、訓練データの入力データに基づいてNNが推定した出力とラベル（教師データ）との差分が最小になるよう、勾配法により、複数のパラメータｗを更新する。パラメータの更新量は、差分Eをパラメータｗで微分して求めた勾配ΔEに学習率ηを乗じて算出される。

【００２１】
［データ分割型分散学習］
ＮＮ、とりわけディープＮＮ（以下ＤＮＮと称する）は、訓練データの数が多いほど、そして、訓練データを使用した学習の回数が多いほど、ＮＮやＤＮＮの精度を向上させることができる。しかし、訓練データの数が膨大になると、それに対応してＮＮシステムによる学習時間が長くなる。そこで、学習時間を短縮するためには、複数の演算ノードで学習処理を分散して行うデータ並列型分散学習が有効になる。
【００２２】
データ並列型分散学習では、複数の訓練データそれぞれによる学習を複数の演算ノードに分散して実行させる。すなわち、複数の演算ノードが、それぞれの訓練データを使用して順伝播処理ＦＷと逆伝播処理ＢＷを実行して、複数の演算ノードそれぞれのＮＮの差分Ｅのパラメータｗに対応する勾配ΔＥを算出する。そして、複数のノードでそれぞれの勾配ΔＥまたは勾配に学習率を乗じたパラメータの更新量Δｗを求め、複数のノードで共有する。更に、複数のノードが、勾配ΔＥの平均または更新量Δｗの平均を取得し、複数のノードがそれぞれのＮＮのパラメータｗを更新量Δｗの平均で更新する。
【００２３】
上記の学習方法のように、複数のノードそれぞれが１つの訓練データを使用して順伝播処理と逆伝播処理を実行して勾配または更新量を求め、それらの平均に基づいて各ノードのパラメータｗを更新する処理は、ノード数の訓練データをミニバッチ単位とするミニバッチ法に対応した処理である。複数のノードそれぞれが複数のプロセスでプロセスの数の訓練データを使用して順伝播処理と逆伝播処理を実行する場合は、ノード数と各ノードのプロセス数を乗算した数の訓練データをミニバッチとするミニバッチ法に対応する。
【００２４】
上記の勾配または更新量の平均を求める処理は、並列コンピューティングの標準規格であるＭＰＩ（Ｍessage Ｐassing Ｉnterface）に含まれるＲｅｄｕｃｅ処理とＡｌｌｒｅｄｕｃｅ処理を含む。このＲｅｄｕｃｅ処理とＡｌｌｒｅｄｕｃｅ処理では、複数の演算ノードそれぞれのパラメータを集約（例えば加算）し、集約した値を全ての複数の演算ノードが取得する。この処理には、複数の演算ノード間でのデータの通信が必要となる。
【００２５】
図３は、データ分割型分散学習における複数の演算ノードの処理のフローを示す図である。また、図４は、データ分割型分散学習における複数の演算ノードの具体的処理を示すフローチャート図である。これらの例では、４つの演算ノードＮＤ_１～ＮＤ_４が、それぞれ、１つの訓練データを使用して学習を行い、ある演算ノードが４つの演算ノードがそれぞれ算出したパラメータｗの更新量Δｗを集約し、各演算ノードが集約した更新量の平均値でそれぞれのパラメータを更新する。図３、図４を参照して、データ分割型分散学習の概略を以下説明する。
【００２６】
４つの演算ノードは、例えば、図１の４つのサブプロセッサ１４に対応する。ＮＮシステムが４つのコンピュータにより構成される場合は、４つの演算ノードは、４つのコンピュータのプロセッサに対応する。
【００２７】
４つの演算ノードＮＤ_１～ＮＤ_４は、ＮＮ学習プログラムを実行して、以下の処理を行う。最初に、４つの演算ノードＮＤ_１～ＮＤ_４それぞれは、訓練データのデータＤ１～Ｄ４のうちそれぞれに対応するデータを入力する（Ｓ１０）。データＤ１～Ｄ４は、各演算ノードの第１のニューロン層ＮＲ_Ｌ１に入力される。そして、各演算ノードは、順伝播処理ＦＷを実行し、各ニューロン層の演算を実行する（Ｓ１１）。図３中、データＤ１、Ｄ２は、手書き文字「６」「２」である。
【００２８】
次に、各演算ノードは、それぞれのＮＮの出力ＯＵＴと教師データであるラベルＬＢの差分Ｅ１～Ｅ４を算出する。演算ノードＮＤ_１、ＮＤ_２の出力ＯＵＴは「５」「３」であり、ラベルＬ

Bは「６」「２」である。ここで、差分E1～E3は、出力OUTとラベルLBの差分の二乗和である。より具体的には、NNは、手書きの数字を推定するモデルであり、出力層である第３のニューロン層は、入力データの数字が数字０～９に該当する確率をそれぞれ出力する。この１０個の確率の出力に対し、数字０～９の確率の教師データは、ラベルLBが示す数字の確率が「１」、ラベルLBと異なる数字の確率が「０」とする。そして、演算ノードは、それぞれの確率の差分の二乗和を差分Eとして算出する。

【００２９】

そして、各演算ノードは、それぞれの差分Eを各ニューロン層に伝播し（S13）、各ニューロン層のパラメータwで伝播された差分Eを微分して勾配ΔEを演算する。更に、各演算ノードは、勾配ΔEに学習率ηを乗じてパラメータwの更新量Δwを算出する（S14）。

【００３０】

ここで、４つの演算ノードND_1～ND_4は、それぞれの更新量Δwを演算ノード間のバス２８を介して演算ノード間で通信し、ある演算ノードが、全ノードのパラメータw1～w4の更新量Δw1～Δw4を集約するreduce処理を行う。集約は、例えば加算（または最大値の抽出）である。そして、４つの演算ノードが、集約した加算値Δw_adをバス２８を介して受信し、全演算ノードで共有するAllreduce処理を行う（S15）。

【００３１】

次に、各演算ノードは、集約した加算値Δw_adを演算ノード数４で除算して更新量Δwの平均値Δw_ad/4を算出し、既存のパラメータw1～w4に加算してそれぞれのパラメータを更新する（S16）。これにより、ミニバッチによる１回の学習が終了する。学習が終了すると、各演算ノードのNNのパラメータは同じ値に更新される。そして、各演算ノードは、処理S10に戻り、次の学習を実行する。

【００３２】

次に、Reduce処理とAllreduce処理について説明する。

【００３３】

図５は、Reduce処理とAllreduce処理の一般的な例を示す図である。図５の例では、４つの演算ノードND_1～ND_4がそれぞれ値y1～y4を所有している。Reduce処理では、各演算ノードがそれぞれの値y1～y4をバス２８を介して通信し、例えば、一つの演算ノードND_1が全ての値y1～y4を受信し、所定の関数ｆで４つの値を演算して集約値ｆ（y1,y2,y3,y4）を算出する。

【００３４】

次に、Allreduce処理では、演算ノードND_1が集約値ｆ（y1,y2,y3,y4）を、他の演算ノードND_2～ND_4にバス２８を介してそれぞれ送信し、全演算ノードで集約値を共有する。

【００３５】

図５の例では、演算ノードND_2～ND_4がそれぞれの値y2～y4を演算ノードND_1に送信している。但し、演算ノードND_4が演算ノードND_3に値y4を送信し演算ノードND_3が加算値y3＋y4を算出し、更に、演算ノードND_2が演算ノードND_1に値y2を送信し演算ノードND_1が加算値y1＋y2を算出し、そして、演算ノードND_3が加算値y3+y4を演算ノードND_1に送信し、演算ノードND_1が集約値ｆ（y1,y2,y3,y4）＝y1+y2+y3+y4を算出してもよい。

【００３６】

また、別の処理方法としては、４つの演算ノードND_1～ND_4がそれぞれ配列データ（w1,x1,y1,z1）、（w2,x2,y2,z2）、（w3,x3,y3,z3）、（w4,x4,y4,z4）を所有している場合、各演算ノードが、データwは演算ノードND_1に、データxは演算ノードND_2に、データyは演算ノードND_3に、そして、データzは演算ノードND_4にそれぞれ送信する。そして、

演算ノードND_1が集約値ｆ（w1,w2,w3,w4）＝w1+w2+w3+w4を算出し、

演算ノードND_2が集約値ｆ（x1,x2,x3,x4）＝x1+x2+x3+x4を算出し、

演算ノードND_3が集約値ｆ（y1,y2,y3,y4）＝y1+y2+y3+y4を算出し、

演算ノードND_4が集約値ｆ（z1,z2,z3,z4）＝z1+z2+z3+z4を算出する。

【００３７】

ここまではReduce処理である。次に、各演算ノードが、それぞれ算出した集約値を他の演算ノードにバス２８を介してそれぞれ送信し、集約値を全演算ノードで取得し共有する。この処理はAllreduce処理である。

【００３８】

図６は、NNの学習をデータ分割型分散学習で行った場合のReduce処理とAllreduce処理の例を示す図である。図４で説明したとおり、各演算ノードがそれぞれのパラメータw1～w4の更新量Δw1～Δw4を算出した段階で、各演算ノードがReduce処理を行うと、全演算ノードの更新量Δw1～Δw4が、例えば一つの演算ノードND_1に送信され、演算ノードND_1が加算関数ｆにより更新量Δw1～Δw4の加算値Δw_adを集約値として算出する。そして、演算ノードND_1が加算値Δw_adを他の演算ノードND_2～ND_4に送信して、全演算ノードが加算値を取得し共有する。

【００３９】

次に、データ分割型分散学習では、各演算ノードND_1～ND_4それぞれが、平均化処理Averageで、加算値Δw_adを演算ノード数４で除算して更新量Δｗの平均値Δw_avを算出する。そして、各演算ノードND_1～ND_4それぞれが、更新処理Updateで、更新量の平均値Δw_avを既存のパラメータw1～w4に加算する。

【００４０】

ミニバッチ法では、上記のデータ並列型分散学習により、ミニバッチの複数の訓練データを複数の演算ノードに分散し、複数の演算ノードがパラメータの勾配ΔEまたは更新量Δｗまで並列に演算を行い、複数の訓練データでそれぞれ算出した複数の更新量Δw1～Δw4の平均値で各演算ノードのパラメータｗを更新する。したがって、ある訓練データで算出された更新量Δｗが、他の訓練データで算出された更新量から大きく乖離した例外的な値であっても、複数の更新量の平均値で全演算ノードのNNのパラメータｗを更新するので、例外的な更新量による学習への悪影響を抑制することができる。

【００４１】

その一方で、各学習で演算ノード間の通信処理を含むReduce処理とAllreduce処理が実行されることにより、通信処理による学習の処理時間が長くなるという問題が生じる。

【００４２】

［第１の実施の形態によるデータ並列型分散学習］

図７は、第１の実施の形態によるデータ並列型分散学習のフローチャート図である。図７には、図４のフローチャートにおける、演算ノードND_1の学習処理のフローチャートが示され、残りの演算ノードND_2～ND_4の学習処理のフローチャートは省略されている。本分散学習では、図７に示した演算ノードND_1の学習処理のフローチャートS10～S16、S16A、S20～S23が、残りの演算ノードND_2～ND_4でも同様に実行される。そして、４つの演算ノードの学習処理のフローチャートは、図４の各演算ノードの学習処理のフローチャートを、図７の演算ノードND_1の学習処理のフローチャートに置き換えることで得られる。

【００４３】

ここでも前提として、４つの演算ノードそれぞれが、１つの訓練データを使用して学習を行い、各演算ノードは、それぞれのNN内の１つのパラメータwを最適化するものとする。一般に、NNは多数のパラメータwを有するが、まず、NNの１つのパラメータwの例で説明し、その後、NNが有する複数のパラメータwに対してどのように処理されるかを説明する。

【００４４】

本分散学習では、複数の演算ノードそれぞれが算出した勾配ΔEまたは更新量Δｗについて、全学習でReduce処理とAllreduce処理を実行することはない。つまり、プロセッサは、複数の演算ノードが算出した勾配または更新量が閾値未満の場合、Reduce処理とAllreduce処理を実行せず、各演算ノードで算出した勾配または更新量を使用してそれぞれのパ

ラメータwを更新する。また、プロセッサは、複数の演算ノードが算出した勾配または更新量が閾値未満でない場合、Reduce処理とAllreduce処理を実行し、集約した勾配または更新量の平均値を使用してそれぞれのパラメータwを更新する。

【００４５】

これにより、本実施の形態のNNシステムは、ミニバッチ法による例外的な値の勾配や更新量による弊害を抑制しつつ、Reduce処理とAllreduce処理による通信処理を時々スキップして、全学習の処理時間を短縮する。

【００４６】

但し、Reduce処理とAllreduce処理が実行されない学習が連続して行われた後、Reduce処理とAllreduce処理を行って更新量の平均値で複数の演算ノードのNNのパラメータwを更新するとき、全演算ノードのパラメータwを同じ値にリセットする必要がある。そこで、各演算ノードが、Reduce処理及びAllreduce処理を実行しない学習での勾配または更新量をそれぞれ累積しておき、Reduce処理及びAllreduce処理が行われたときに、その更新量の累積値で各演算ノードのパラメータを更新する。そのため、各演算ノードは、学習したときに算出した勾配Eまたは更新量Δwの累積ErまたはΔwrを記憶しておく。

【００４７】

以下の説明では、勾配または更新量は、簡単化して更新量Δwとする。但し、更新量Δwに代えて、勾配ΔEについてReduce処理とAllreduce処理を行ってもよい。

【００４８】

図７のフローチャートに沿って、各演算ノードの学習処理を説明する。図７には示していないが、学習の開始直後に、演算ノードND_1～ND_4は、それぞれの累積更新量Δwr1～Δwr4を０にリセットする。次に、演算ノードND_1～ND_4は、訓練データの入力データを入力し、順伝播処理、差分E1の算出、逆伝播処理、及びパラメータwの更新量Δw1～Δw4の算出を実行する（S10～S14）。そして、各演算ノードは、累積更新量Δwr1～wr4に算出した更新量Δw1～Δw4をそれぞれ加算する（S20）。学習開始直後は、全演算ノードの累積更新量は全て０であるので、累積更新量Δwr1～wr4は初回学習での更新量Δw1～Δw4と等しい。

【００４９】

次に、各演算ノードは、それぞれの累積更新量Δwr1～wr4が閾値TH未満か否かを判定する（S21）。全演算ノードでそれぞれの累積更新量Δwr1～wr4が閾値TH未満の場合（S21のYES）、各演算ノードは、それぞれのパラメータw1～w4にそれぞれの更新量Δw1～Δw4を加算してパラメータを更新する（S16A）。その結果、各演算ノードは、それぞれ算出した更新量Δw1～Δw4でそれぞれのパラメータw1～w4を更新する。

【００５０】

一方、全演算ノードでそれぞれの累積更新量Δwr1～wr4が閾値TH未満とはならない場合（S21のNO）、演算ノードND_1～ND_4が、それぞれ算出した累積更新量Δwr1～Δwr4を送受信し、全演算ノードの累積更新量Δwr1～Δwr4を加算するなどして集約し（Reduce処理）、その集約した累積更新量Δwr_adを全ノードで共有する（Allreduce処理）（S15）。具体的には、演算ノードND_1～ND_4のうち一つの演算ノード、例えば演算ノードND_1が、他の演算ノードND_2～ND_4から累積更新量Δwr2～Δwr4を受信し、加算し、その加算した集約累積更新量Δwr_adを他の演算ノードND_2～ND_4に送信する。上記判定S21のNOの場合とは、全累積更新量Δwr1～wr4が閾値TH未満にならないこと、少なくとも１つの累積更新量が閾値TH未満ではないことを意味する。上記判定S21のYESの場合とは、全累積更新量Δwr1～wr4が閾値TH未満になることを意味する。

【００５１】

そして、各演算ノードND_1～ND_4が、集約累積更新量Δwr_adを演算ノードの数「４」で除算して各累積更新量の平均値Δwr_ad/4を求め、それぞれのパラメータw1～w4の更新量を累積し始める前のパラメータw1～w4の値に、累積更新量の平均値Δwr_ad/4を加算して、各パラメータを共通の値に更新する（S16）。それと共に、各演算ノードが、それぞれの累積更新量Δwr1～Δwr4を０にリセットする（S22）。

【００５２】
各演算ノードは、全体の学習回数がN未満の間、上記の学習処理を繰返す（S23）。
【００５３】
上記の学習処理によれば、各演算ノードND_1～ND_4が算出したパラメータの累積更新量Δwr1～Δwr4が全て閾値未満の場合（S21のYES）、演算ノードND_1～ND_4はReduce処理とAllreduce処理を行わないので、両処理の演算ノード間の通信に要する時間がなくなり、全学習に要する時間を短くできる。そして、各演算ノードがReduce処理とAllreduce処理を行わず、それぞれのパラメータw1～w4をそれぞれの更新量Δw1～Δw4で更新することが連続する場合、各演算ノードが累積更新量Δwr1～Δwr4を算出して記録しておく。
【００５４】
それにより、各演算ノードND_1～ND_4が算出したパラメータの累積更新量Δwr1～Δwr4が全て閾値未満とはならない場合（S21のNO）、各演算ノードが、それぞれの累積更新量Δwr1～Δwr4を集約し、集約した累積更新量Δwr_addを共有し、その平均値Δwr_add/4で累積前のパラメータw1～w4を更新する。全演算ノードの累積更新量が閾値未満の場合、各演算ノードでの累積更新量のばらつきが比較的小さく、Reduce処理とAllreduce処理を省略しても、各演算ノード間のパラメータの値が大きく乖離することはない。しかし、全演算ノードの累積更新量が閾値未満ではなく少なくとも一つの累積更新量が閾値以上の場合、パラメータの値の乖離が大きくなるので、Reduce処理とAllreduce処理を行って、各演算ノードの累積更新量を集約しその平均値で全演算ノードの累積前のパラメータを更新してリセットする。
【００５５】
上記の学習では、前提として、各演算ノードが、それぞれのNN内の１つのパラメータwを最適化した。この場合、各演算ノードのパラメータw1～w4の累積更新量Δwr1～Δwr4をそれぞれ閾値と比較した。しかし、パラメータの更新量Δw1～Δw4は正の場合と負の場合があるので、望ましくは、パラメータの累積更新量Δwr1～Δwr4の絶対値をある閾値TH（THは正）と比較する。
【００５６】
次に、NNが有する複数のパラメータwの最適化がどのように処理されるかを説明する。第１の方法としては、判定工程S21で、各演算ノードがそれぞれのNNの複数のパラメータの累積更新量を個別に閾値THと比較し、それぞれのNNの全てのパラメータの累積更新量が閾値TH未満か否かを判断すると共に、全演算ノードで全て閾値TH未満か否かを判断する。
【００５７】
第２の方法として、各演算ノードが、それぞれのNNの複数のパラメータを、NN内の各層の複数パラメータ$w_1, w_2, ...w_n$にグループ化し、判定工程S21で、各層の複数パラメータ$w_1, w_2, ...w_n$の累積更新量の絶対値の最大値が閾値TH未満か否かを判断する。そして、各演算ノードが、それぞれのNNの複数の層の判定S21が全て閾値TH未満か否かを判断すると共に、全演算ノードで全て閾値TH未満か否かを判断する。最大値のみ閾値THと比較するので、判定工程S21のスループットが向上する。
【００５８】
第３の方法として、各演算ノードが、それぞれのNNの複数のパラメータを、NN内の各層の複数パラメータ$w_1, w_2, ...w_n$にグループ化し、判定工程S21で、各層の複数パラメータ$w_1, w_2, ...w_n$の累積更新量の絶対値のLpノルム（pは正の整数）が閾値TH未満か否かを判断する。そして、各演算ノードが、それぞれのNNの複数の層の判断が全て閾値TH未満か否かを判断すると共に、全演算ノードで全て閾値TH未満か否かを判断する。
【００５９】
例えば、以下の数式に示すとおり、L1ノルムは、複数パラメータ$w_1, w_2, ...w_n$の累積更新量の絶対値の和であり、L2ノルムは、複数パラメータ$w_1, w_2, ...w_n$の累積更新量の絶対値の二乗和の平方根である。Lpノルムは、複数パラメータ$w_1, w_2, ...w_n$の累積更新

量の絶対値のp乗和の-p乗である。

【００６０】

【数１】

$$\Delta \mathrm{wr} = (\Delta \mathrm{wr}_1, \Delta wr_2, \ldots, \Delta \mathrm{wr}_n)$$

$$\Delta \mathrm{wr}のLpノルム = \sqrt[p]{|\Delta wr_1|^p + |\Delta wr_2|^p + \cdots + |\Delta wr_n|^p}$$

$$\Delta \mathrm{wr}のL1ノルム = |\Delta wr_1| + |\Delta wr_2| + \cdots + |\Delta wr_n|$$

$$\Delta \mathrm{wr}のL2ノルム = \sqrt{|\Delta wr_1|^2 + |\Delta wr_2|^2 + \cdots + |\Delta wr_n|^2}$$

【００６１】

第３の方法では、各層の複数のパラメータそれぞれの累積更新量をそのL1ノルムやL2ノルムに変換した値を閾値と比較するので、判定工程S21のスループットが向上する。

【００６２】

［第２の実施の形態によるデータ並列型分散学習］

図８は、第２の実施の形態によるデータ並列型分散学習のフローチャート図である。図８には、図４のフローチャートにおける、演算ノードND_1の学習処理のフローチャートが示され、残りの演算ノードND_2～ND_4の学習処理のフローチャートは省略されている。本分散学習では、図８に示した演算ノードND_1の学習処理のフローチャートS30、S10～S16、S20～S23、S31～S33が、残りの演算ノードND_2～ND_4でも同様に実行される。

【００６３】

一般に、学習工程の開始直後は、パラメータの勾配ΔEが大きく更新量Δw1～Δw4も大きい。一方、学習工程の終了近くでは、パラメータの勾配ΔEが小さく更新量Δw1～Δw4も小さい。そのため、第1の実施の形態のデータ並列型分散学習では、学習工程の開始直後は、判定工程S21で、毎回、累積更新量が閾値TH未満でないという判断となり、毎回Reduce処理とAllreduce処理が行われる場合がある。一方、学習工程の終了に近づくにつれて、判定工程S21で、毎回、累積更新量が閾値TH未満という判断となり、Reduce処理とAllreduce処理が全く行われないという場合がある。

【００６４】

上記のような問題点を緩和するために、第２の実施の形態では、次の処理を行う。即ち、

（１）全部でN回の学習のうち初めからD-1回（Dは正の整数）は、各演算ノードは、閾値THとの比較判定にかかわらず、Reduce処理及びAllreduce処理を行わず、それぞれのNNのパラメータをそれぞれの更新量で更新する。

（２）そして、D回からU-1回（UはDより大きい正の整数）までの間は、第１の実施の形態のように累積更新量Δwr1～Δwr4が閾値TH未満の場合、Reduce処理及びAllreduce処理を行わず、閾値TH未満でない場合、Reduce処理及びAllreduce処理を行って、それぞれのNNパラメータを累積更新量の平均値で更新する。

（３）さらに、U回になるまで累積更新量Δwr1～Δwr4が閾値TH未満であるためReduce処理及びAllreduce処理が連続して行われなかったら、U回目で、各演算ノードは、閾値THとの比較判定にかかわらず、Reduce処理及びAllreduce処理を行い、それぞれのNNパラメータを累積更新量の平均値で更新する。

（４）上記の（１）～（３）のパラメータの更新サイクルを、全学習Nに達するまで、各演算ノードが繰返す。

【００６５】

上記の処理によれば、第１に、学習工程の開始直後でも、（１）～（３）の更新サイクル内の最初のD-1回では、各演算ノードは、Reduce処理及びAllreduce処理を行わないの

で、通信回数を減らすことができる。また、更新サイクル内のD回以上では、パラメータの累積更新量と閾値THとの比較判定に基づいて、累積更新量が小さいほどReduce処理及びAllreduce処理が連続して行われない回数が多くなり、逆に累積更新量が大きいほどReduce処理及びAllreduce処理が連続して行われない回数が少なくなる。

【００６６】

一方、第２に、学習工程の終了に近づいたとき、（１）～（３）の更新サイクル内で、Reduce処理及びAllreduce処理が連続して行われない学習回数がU回に達すると、各演算ノードは、ある意味、強制的にReduce処理とAllreduce処理が行われて、全演算ノードの全NNの対応するパラメータが同じ累積更新量の平均値で同じ値に更新される。

【００６７】

図８のフローチャートについて具体的に説明する。図８でも前提として、４つの演算ノードそれぞれが、１つの訓練データを使用して学習を行い、各演算ノードは、それぞれのNN内の１つのパラメータwを最適化するものとする。

【００６８】

そして、第２の実施の形態では、全演算ノードが、共通の学習回数カウンタ値iと連続非通信カウンタ値jをカウントする。また、第１の実施の形態と同様に、各演算ノードが、学習のたびに算出した各パラメータの更新量を累積加算して累積更新量Δwr1～Δwr4を記憶する。そして、各演算ノードは、図７のフローチャートの処理に加えて、処理S30、S31-S32、S33を実行する。これらの処理を主に説明する。

【００６９】

各演算ノードは、初期化処理として、学習回数カウンタ値ｉと連続非通信カウンタ値ｊを「０」に、各演算ノードのパラメータの累積更新量Δwr1～Δwr4を「０」に、それぞれリセットする。次に、各演算ノードは、訓練データのデータ入力、順伝播処理、逆伝播処理を行ってそれぞれのパラメータの更新量Δw1～Δwr4を算出する（S10-S14）。そして、各演算ノードは、カウンタ値ｉ，ｊをそれぞれ１加算し、パラメータの累積更新量Δwr1～Δwr4に算出した更新量Δw1～Δwr4をそれぞれ加算して累積更新量を更新する（S31）。

【００７０】

（１）連続非通信カウンタ値jが第１の基準回数D未満の場合（S32のYES）、各演算ノードは、それぞれのパラメータw1～w4をそれぞれの更新量Δw1～Δwr4で更新する（S16A）。各演算ノードは、連続非通信カウンタ値jが第１の基準回数D未満でなくなるまで、上記処理S10-S14、S31-S32及びS16Aを繰返す。第１の基準回数Dが、例えば、D=2の場合、（１）～（３）の更新サイクル内の一回目の学習では、必ず、演算ノードはReduce処理とAllreduce処理を行わない。

【００７１】

（２）連続非通信カウンタ値jが第１の基準回数D未満でなくなると（S32のNO）、各演算ノードは、全演算ノードでパラメータの累積更新量Δwr1～Δwr4が全て閾値TH未満か否か判定する。

【００７２】

閾値TH未満の場合（S21のYES）、連続非通信カウンタ値jが第２の基準回数U（＞D）未満であれば（S33のYES）、各演算ノードは、それぞれのパラメータw1～w4をそれぞれの更新量Δw1～Δwr4で更新する（S16A）。

【００７３】

閾値TH未満でない場合（S21のNO）、演算ノードND_1～ND_4が、Reduce処理とAllreduce処理を実行し（S15）、それぞれのパラメータw1-w4を累積更新量の平均値Δwr_add/4で更新する（S16）。そして、演算ノードが、連続非通信カウント値jを０に、累積更新量Δwr1～Δwr4を０にそれぞれリセットする（S22A）。この場合、（１）～（３）の更新サイクルがリセットされる。

【００７４】

（３）閾値TH未満の場合（S21のYES）、連続非通信カウンタ値jが第２の基準回数U（

＞D）未満でなくなると（S33のNO）、各演算ノードは、Reduce処理とAllreduce処理を実行し（S15）、パラメータを累積更新量の平均値で更新し（S16）、連続非通信カウント値jと累積更新量を０にリセットする（S22A）。これで更新サイクルがリセットされる。

【００７５】

各演算ノードのNNの複数のパラメータwを更新する場合については、第１の実施の形態と同様に、判定工程S21では、演算ノードは、各パラメータwの累積更新量の絶対値が閾値TH未満か否か、各層の複数パラメータの累積更新量の絶対値の最大値が閾値TH未満か否か、各層の複数パラメータの累積更新量の絶対値のL1ノルムやL2ノルムが閾値TH未満か否か、などの判定を行っても良い。

【００７６】

図９は、第２の実施の形態における学習の更新サイクルの変化例を示す図である。図９（１）は、本実施の形態の更新サイクルを行わない場合の変化例である。１回の学習は、訓練データのデータに対する順伝播処理FW、逆伝播処理BW、Reduce処理及びAllreduce処理CM、パラメータの更新処理UPを含む。図９（１）での更新処理UP1は、全演算ノードのNNのパラメータの更新量の平均値Δw_ad/4でパラメータを更新する処理である。図９（１）の場合、各演算ノードは、全学習で、Reduce処理及びAllreduce処理CMとパラメータの更新処理UP1を実行する。

【００７７】

図９（２）は、第２の実施の形態における学習の更新サイクルの変化例である。図９（２）の１回目～４回目の学習は、前述の更新サイクル内の学習に該当する。

【００７８】

図９（２）での更新処理UP2は、各演算ノードがそれぞれのNNのパラメータの更新量Δwでそれぞれのパラメータを更新する処理（S16A）である。

【００７９】

図９（２）での更新処理UP3は、各演算ノードがそれぞれのNNのパラメータの累積更新量Δwrの平均値Δwr_ad/4でそれぞれのパラメータを更新する処理（S16）である。

【００８０】

図９（２）の例では、D=2の例であり、各演算ノードは、１回目の学習では、Reduce処理及びAllreduce処理CMを実行せず、更新処理UP2を実行する。２回目の学習では、j=Dとなり、各演算ノードは、Reduce処理及びAllreduce処理CMを実行し、更新処理UP3を実行する。３回目の学習と４回目の学習は、それぞれ１回目の学習と２回目の学習と同じである。

【００８１】

第２の実施の形態によれば、演算ノードが全ての学習でReduce処理及びAllreduce処理CMを実行することはないので、同処理を実行しないことにより、学習全体の演算時間を抑制することができる。

【００８２】

上記の実施の形態では、Reduce処理とAllreduce処理でパラメータの累積更新量Δwrを集約し、その平均値Δwr_ad/4で各NNのパラメータを更新した。しかし、パラメータの更新量に代えて、差分の勾配ΔEについてReduce処理とAllreduce処理を行っても良い。パラメータの更新量Δwは、差分の勾配ΔEに学習率ηを乗算して算出され、従って、累積更新量Δwrは累積した勾配に学習率を乗算して算出できるからである。その場合、各演算ノードが、Reduce処理とAllreduce処理を行わない場合、差分の勾配ΔEの累積を更新しておき、Reduce処理とAllreduce処理を行った場合、各演算ノードでの累積勾配ΔErを集約し、累積勾配ΔErの集約値（加算値）を全演算ノードで共有し、累積勾配ΔErの集約値（加算値）の平均値ΔEr_ad/4に学習率ηを乗じた累積更新量の平均値Δwr_ad/4で累積前のパラメータwを更新する。

【００８３】

上記の実施の形態では、各学習で、各演算ノードが１つの訓練データについてNNの演算

を実行する例を説明した。しかし、各学習で、各演算ノードが複数の訓練データについてNNの演算を複数のプロセスで実行しても良い。その場合、１バッチの訓練データの数は、各演算ノードの複数の訓練データに演算ノードの数（上記例では４）を乗じた数になる。そして、各演算ノードは、複数のプロセスでそれぞれ算出した複数の差分Eの勾配ΔEまたはパラメータの更新量Δwの平均値を使用して、各演算ノードのNNのパラメータを更新する。また、Reduce処理とAllreduce処理では、複数の演算ノードがそれぞれの勾配または更新量の累積を集約し、集約した値の平均を全演算ノードで共有し、集約した値の平均でそれぞれのNNのパラメータを更新する。

【００８４】

上記の実施の形態は、単純パーセプトロンや多層パーセプトロン等のNN、階層が深いNNであるディープNNなどの学習に適用できる。ディープNNには、例えば、複数の畳込み層とプーリング層及び全結合層を有するコンボリュージョンNN、入力層と出力層が同じサイズのノードを持つオートエンコーダNN、リカレントNNなどが含まれる。

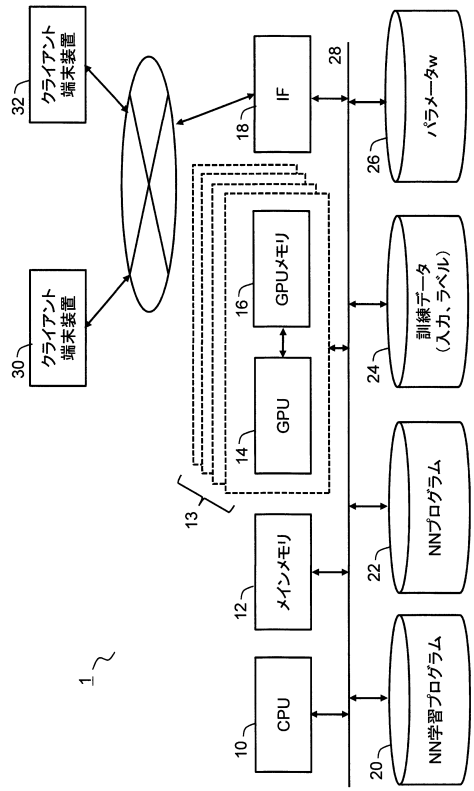
【符号の説明】

【００８５】

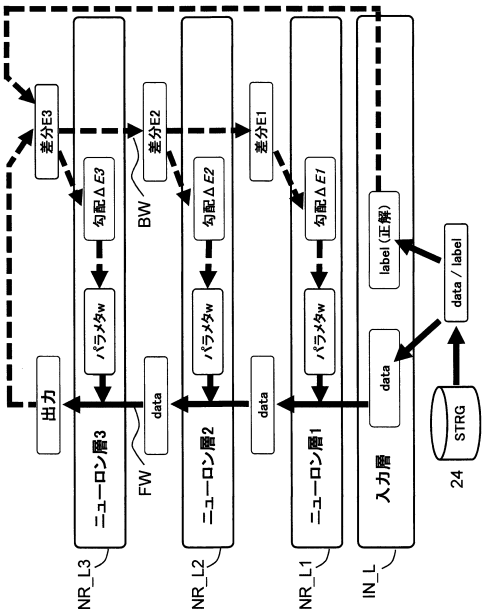
１：ニューラルネットワークシステム、NNシステム
１０：メインプロセッサ
１３：サブプロセッサモジュール
１４：サブプロセッサ、演算ノード
２０：ニューラルネットワーク学習プログラム
２２：ニューラルネットワークプログラム
２４；訓練データ
２６：パラメータw
ND_1～ND_4：演算ノード
w1～w4：パラメータ
Δw1～Δw4：パラメータの更新量
Δwr1～Δwr4：累積更新量
Δwr_add：累積更新量の集約値
Δwr_add/4：累積更新量の平均値

【図面】

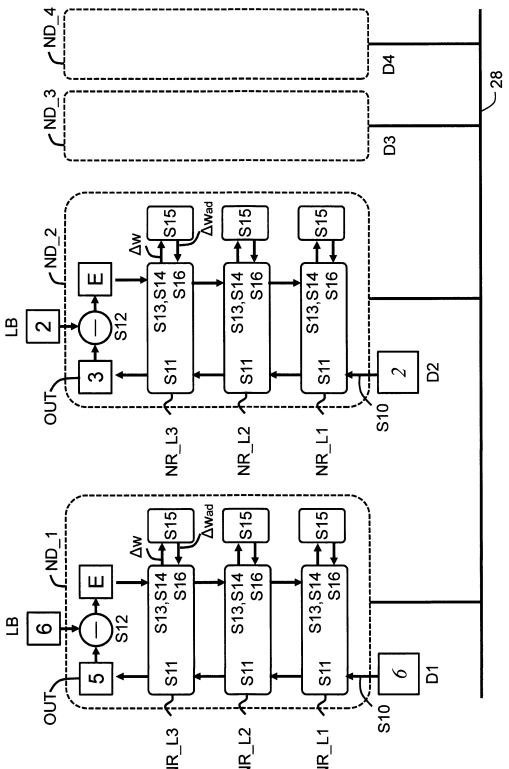
【図１】


1
10 CPU
12 メインメモリ
13
14 GPU
16 GPUメモリ
18 IF
20 NN学習プログラム
22 NNプログラム
24 訓練データ（入力、ラベル）
26 パラメータw
28
30 クライアント端末装置
32 クライアント端末装置

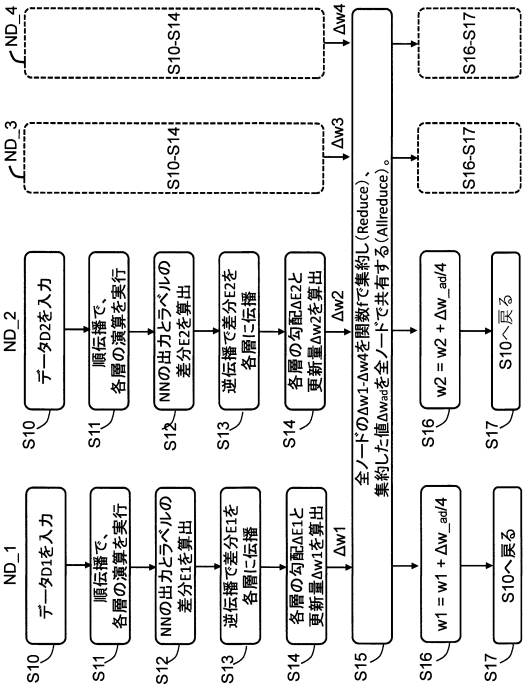
【図２】


出力
差分E3
勾配ΔE3
パラメタw
BW
差分E2
勾配ΔE2
差分E1
勾配ΔE1
label（正解）
data / label
data
FW
STRG
24
NR_L3
ニューロン層3
NR_L2
ニューロン層2
NR_L1
ニューロン層1
IN_L
入力層

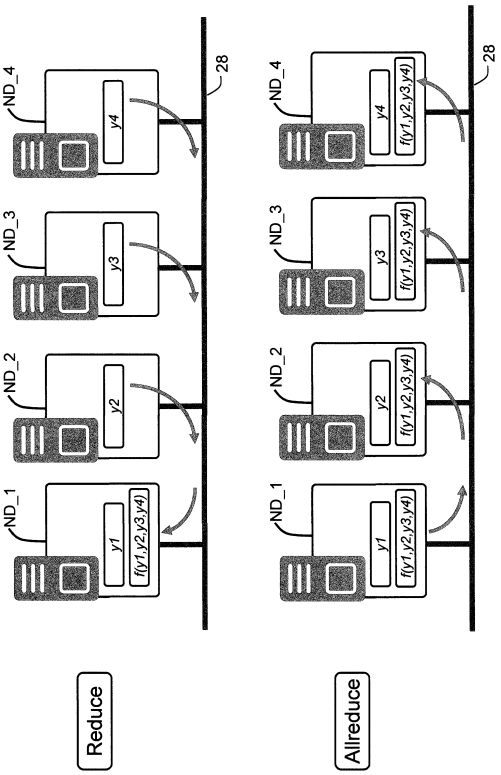
【図３】


LB
OUT
ND_1
ND_2
ND_3
ND_4
E
S11
S12
S13,S14
S15
S16
Δw
Δwad
NR_L3
NR_L2
NR_L1
S10
D1
D2
D3
D4
28

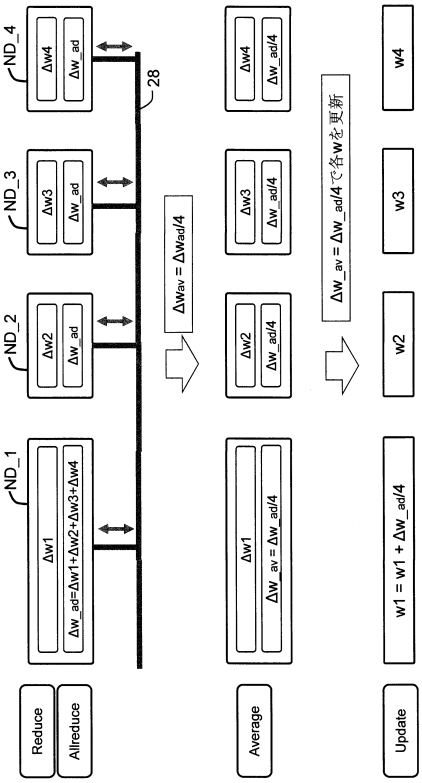
【図４】


ND_1
ND_2
ND_3
ND_4
S10 データD1を入力
S11 順伝播で、各層の演算を実行
S12 NNの出力とラベルの差分E1を算出
S13 逆伝播で差分E1を各層に伝播
S14 各層の勾配ΔE1と更新量Δw1を算出
S15 全ノードのΔw1-Δw4を関数で集約し（Reduce）、集約した値Δwadを全ノードで共有する（Allreduce）。
S16 w1 = w1 + Δw_ad/4
S17 S10へ戻る
S10 データD2を入力
S11 順伝播で、各層の演算を実行
S12 NNの出力とラベルの差分E2を算出
S13 逆伝播で差分E2を各層に伝播
S14 各層の勾配ΔE2と更新量Δw2を算出
S15
S16 w2 = w2 + Δw_ad/4
S17 S10へ戻る
S10-S14
S16-S17
Δw1
Δw2
Δw3
Δw4

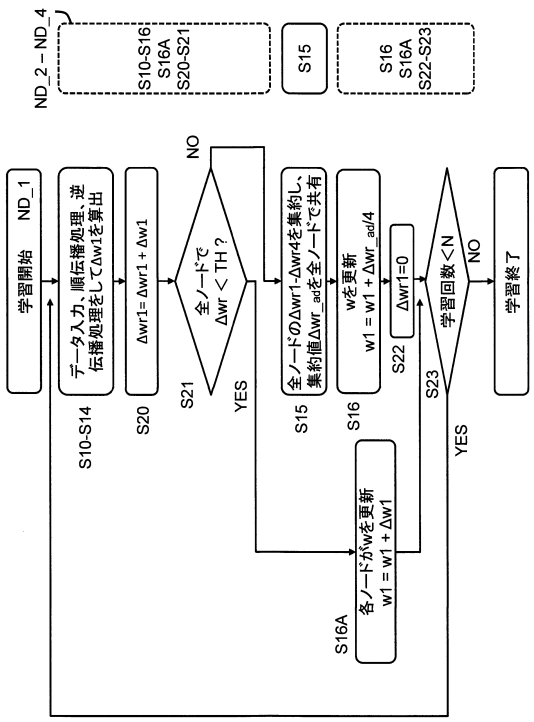
【図５】


ND_1
ND_2
ND_3
ND_4
y1
y2
y3
y4
f(y1,y2,y3,y4)
28
Reduce
Allreduce

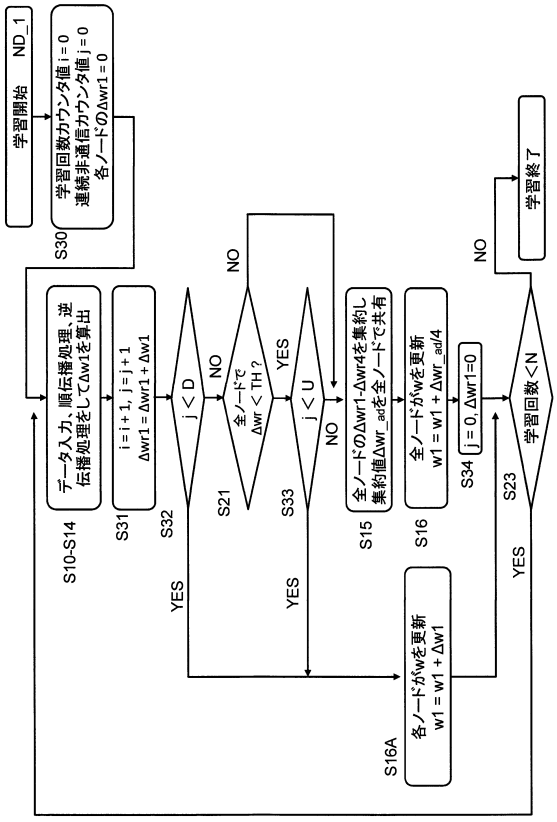
【図６】


ND_1
ND_2
ND_3
ND_4
Δw1
Δw_ad=Δw1+Δw2+Δw3+Δw4
Δw2
Δw3
Δw4
Δw_ad
28
Reduce
Allreduce
Δw_av = Δw_ad/4
Δw_ad/4
Average
Δw_av = Δw_ad/4で各wを更新
w1 = w1 + Δw_ad/4
w2
w3
w4
Update

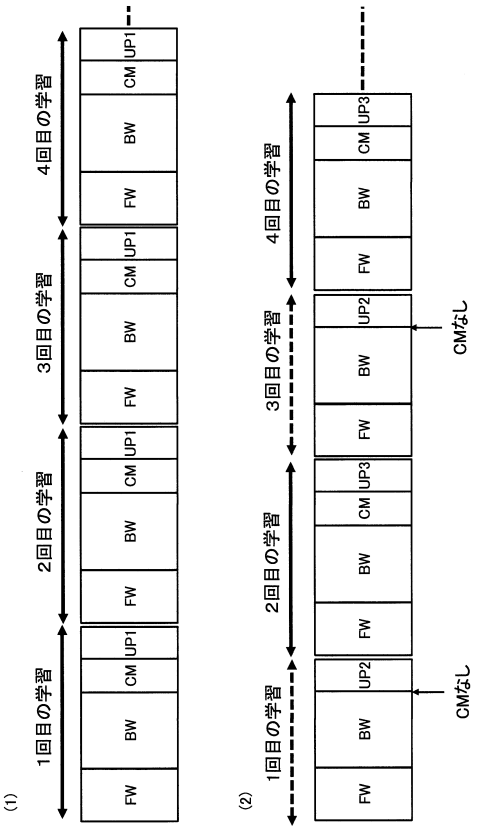
【図７】


ND_2 – ND_4
S10-S16
S16A
S20-S21
S15
S16
S16A
S22-S23
学習開始 ND_1
データ入力、順伝播処理、逆伝播処理をしてΔw1を算出
S10-S14
Δwr1= Δwr1 + Δw1
S20
全ノードで Δwr < TH ?
S21
NO
YES
全ノードのΔwr1-Δwr4を集約し、集約値Δwr_adを全ノードで共有
S15
wを更新 w1 = w1 + Δwr_ad/4
S16
Δwr1=0
S22
学習回数<N
S23
NO
YES
学習終了
各ノードがwを更新 w1 = w1 + Δw1
S16A

【図８】


学習開始 ND_1
学習回数カウンタ値 i = 0
連続非通信カウンタ値 j = 0
各ノードのΔwr1 = 0
S30
データ入力、順伝播処理、逆伝播処理をしてΔw1を算出
S10-S14
i = i + 1, j = j + 1
Δwr1 = Δwr1 + Δw1
S31
S32
j < D
YES
NO
全ノードで Δwr < TH ?
S21
YES
NO
j < U
S33
YES
NO
全ノードのΔwr1-Δwr4を集約し集約値Δwr_adを全ノードで共有
S15
全ノードがwを更新 w1 = w1 + Δwr_ad/4
S16
j = 0, Δwr1=0
S34
学習回数<N
S23
YES
NO
学習終了
各ノードがwを更新 w1 = w1 + Δw1
S16A

【図９】


(1)
1回目の学習
2回目の学習
3回目の学習
4回目の学習
FW
BW
CM
UP1
(2)
1回目の学習
2回目の学習
3回目の学習
4回目の学習
FW
BW
CM
UP2
UP3
CMなし

フロントページの続き

(56)参考文献 特開２０１９－２１２１１１（ＪＰ，Ａ）
特開２０１９－１０９８７５（ＪＰ，Ａ）
欧州特許出願公開第０３５９１５８３（ＥＰ，Ａ１）

(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０６Ｎ　３／００－９９／００