(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2018-18220
(P2018-18220A)
(43)公開日　平成30年2月1日(2018.2.1)

(51)Int.Cl.			F I		テーマコード（参考）
G06N	3/10	(2006.01)	G06N	3/10	
G06F	9/50	(2006.01)	G06F	9/46	465C
G06F	9/48	(2006.01)	G06F	9/46	457

審査請求 未請求　請求項の数 10　OL　（全 35 頁）

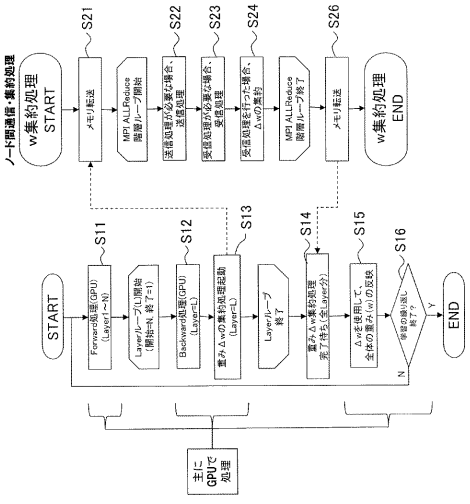
(21)出願番号　特願2016-146731 (P2016-146731)
(22)出願日　平成28年7月26日 (2016.7.26)

(71)出願人　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　100113608
弁理士　平川　明
(74)代理人　100105407
弁理士　高田　大輔
(72)発明者　山崎　雅文
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
(72)発明者　田原　司睦
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

最終頁に続く

(54)【発明の名称】並列情報処理装置、情報処理方法、およびプログラム

(57)【要約】
【課題】ノード間並列による深層学習おいて、係数演算に使用する係数の勾配情報を深層学習に反映する処理の時間を短縮する。
【解決手段】各ノードの演算部は、処理対象のデータに対する係数による演算処理を実行し、演算処理の結果を基に係数の変化量を算出し、算出した係数の変化量を処理部に転送するとともに、係数の変化量を並列情報処理装置内の他のノードとの間で授受する処理の実行を処理部に要求する。各ノードの処理部は、演算部から転送された係数の変化量を並列情報処理装置の他のノードに送信するとともに他のノードで算出された係数の変化量を受信する通信処理と、自ノードの演算部から転送された係数の変化量と他のノードで算出された係数の変化量とを積算する集約処理とを実行する。
そして、演算部および処理部の少なくとも一方が積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新する。
【選択図】図７



【特許請求の範囲】
【請求項１】
演算部と処理部とを有するノードを複数備えた並列情報処理装置において、
それぞれのノードの演算部は、処理対象のデータに対する係数による演算処理を実行し、前記演算処理の結果を基に前記係数の変化量を算出し、算出した係数の変化量を前記処理部に転送するとともに、前記係数の変化量を前記並列情報処理装置内の他のノードとの間で授受する処理の実行を前記処理部に要求し、
前記それぞれのノードの処理部は、前記演算部から転送された係数の変化量を前記並列情報処理装置の他のノードに送信するとともに前記他のノードで算出された係数の変化量を受信する通信処理と、前記演算部から転送された係数の変化量と前記他のノードで算出された係数の変化量とを積算する集約処理とを実行し、
前記演算部および前記処理部の少なくとも一方が前記積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新する並列情報処理装置。
【請求項２】
前記演算処理は、所定順序で実行される複数階層の層別処理を有し、それぞれの階層の層別処理はそれぞれの階層の前の階層から入力されるデータに前記係数による演算を実行して次の階層に出力する処理であり、
前記演算部は、それぞれの階層での前記層別処理の結果を基に前記それぞれの階層での前記係数の変化量を算出し、前記算出した係数の変化量を前記処理部に転送し、
前記処理部は、前記それぞれの階層での前記係数の変化量に対する前記集約処理を２以上並列に実行する請求項１に記載の並列情報処理装置。
【請求項３】
前記係数は、前記複数階層のそれぞれにおいて複数使用され、係数列を形成し、
前記演算部は、前記複数階層のそれぞれの前記係数列を複数の部分列に分割して部分列ごとに前記変化量を前記処理部に転送するとともに、前記部分列ごとに前記授受する処理の実行を前記処理部に要求する請求項２に記載の並列情報処理装置。
【請求項４】
前記演算部は、前記複数階層のうち、前記演算処理の実行の順序が早い階層の係数の変化量を優先して前記処理部に転送し、前記授受する処理の実行を要求する請求項２または３に記載の並列情報処理装置。
【請求項５】
前記処理部は、前記複数階層のうち前記演算処理の実行の順序が早い階層の係数を優先して前記演算部に前記次回以降の演算処理で使用される係数を更新させる請求項２から４のいずれか１項に記載の並列情報処理装置。
【請求項６】
前記演算部は、前記複数階層の層別処理を前記所定順に繰り返して実行し、前記複数階層のうち、実行の順序が先の階層で使用される係数に対して前記積算された変化量を基に前記次回以降の演算処理で使用される係数が更新された場合には、実行の順序が後の階層で使用される係数に対する前記積算された変化量の反映を待たないで、次の演算処理における前記実行順が先の階層の層別処理を開始する請求項２から５のいずれか１項に記載の並列情報処理装置。
【請求項７】
前記演算処理と前記積算された変化量を基に前記次回以降の演算処理で使用される係数を更新する処理とが複数回繰り返して実行される場合に、前記演算部は、現在の演算処理による変化量を基に前記次回以降の演算処理で使用される係数が更新される前に次の演算処理を開始し、前記現在の演算処理による変化量を基に次々回の演算処理で使用される係数が更新される請求項２から６のいずれか１項に記載の並列情報処理装置。
【請求項８】
前記係数を格納するための２組以上の記憶部を有し、
前記演算部は第１の記憶部に格納した係数を用いて前記演算処理を実行し、前記演算処

理による変化量を基に更新した係数を第２の記憶部に格納する請求項１から７のいずれか１項に記載の並列情報処理装置。
【請求項９】
演算部と処理部とを有するノードを複数備えた並列情報処理装置における情報処理方法であって、
それぞれのノードの演算部は、処理対象のデータに対する係数による演算処理を実行し、前記演算処理の結果を基に前記係数の変化量を算出し、算出した係数の変化量を前記処理部に転送するとともに、前記係数の変化量を前記並列情報処理装置内の他のノードとの間で授受する処理の実行を前記処理部に要求し、
前記それぞれのノードの処理部は、前記演算部から転送された係数の変化量を前記並列情報処理装置の他のノードに送信するとともに前記他のノードで算出された係数の変化量を受信する通信処理と、前記演算部から転送された係数の変化量と前記他のノードで算出された係数の変化量とを積算する集約処理とを実行し、
前記演算部および前記処理部の少なくとも一方が前記積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新する情報処理方法。
【請求項１０】
演算部と処理部とを有するノードを複数備えた並列情報処理装置に実行させるためのプログラムであり、
それぞれのノードの演算部に、処理対象のデータに対する係数による演算処理を実行し、前記演算処理の結果を基に前記係数の変化量を算出し、算出した係数の変化量を前記処理部に転送するとともに、前記係数の変化量を前記並列情報処理装置内の他のノードとの間で授受する処理の実行を前記処理部に要求することを実行させるプログラムと、
前記それぞれのノードの処理部に、前記演算部から転送された係数の変化量を前記並列情報処理装置の他のノードに送信するとともに前記他のノードで算出された係数の変化量を受信する通信処理と、前記演算部から転送された係数の変化量と前記他のノードで算出された係数の変化量とを積算する集約処理とを実行させるプログラムとを含み、
前記演算部および前記処理部の少なくとも一方に前記積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新させるためのプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、並列情報処理装置、情報処理方法、およびプログラムに関する。
【背景技術】
【０００２】
近年、深層学習（ｄｅｅｐ　ｌｅａｒｎｉｎｇ、ＤＬ）に関する研究が盛んである。例えば、画像や音声、文章等に対し、その内容の認識・理解といった研究領域が例示される。このような研究領域の具体的な応用（アプリケーション）としては、携帯電話における通信時の音声認識、ネットワーク上の検索、大量のログ情報からの異常検出、さらには自動運転などが例示される。このような応用のプロジェクトは、実際に動き始めており、今後もさらに広い分野への応用が進むと考えられる。
【０００３】
ところで、深層学習が導入されたシステムにおいては、学習処理は、膨大なデータを繰り返し学習させる手法が例示される。したがって、この学習処理には膨大な計算量が費やされる。例えば、画像識別等の分野では、学習用としてラベル付きの百万枚を超える静止画を、繰り返し学習する。このために、ＧＰＵのような積和演算のような学習処理で多用される演算を高速に演算可能な演算部品（以下、演算部品）を利用したシステム、あるいは、演算部品を含むノードを複数組み合わせたクラスタ環境が利用される。すなわち、学習処理には、ＧＰＵのような演算部品の利用は有効であり、さらに複数の演算部品で処理を分散し実行させることで処理の高速化が可能である。複数の演算部品で処理を分散し実行させる方法としては、ノード内並列と、ノード間並列が考えられる。

【先行技術文献】
【特許文献】
【０００４】
【特許文献１】特開２０１０－０２０４４５号公報
【特許文献２】特開２０１２－０２２５５８号公報
【特許文献３】特開２００５－１８２７８５号公報
【発明の概要】
【発明が解決しようとする課題】
【０００５】
上述のように、深層学習ついてはこれまで、ノード内に複数のＧＰＵ等の演算部品を実装し、並列に処理を行うことで、ノード内並列による高速化が行われてきた。一方、演算部品が実装されているノードを複数組み合わせたノード間並列による成果は少ない。
【０００６】
これまでノード間並列による成果が少ない理由としては、ノード数が増加するに従いノード間をまたいだ深層学習において、係数演算に使用する係数情報のノード間での集約処理、および集約された結果を深層学習に反映する処理に時間がかかることが想定できる。つまり、ノード数の増加による演算能力の向上が実行速度の増加に十分に寄与していないことが想定できる。
【０００７】
深層学習では、処理対象のデータに対する係数による演算処理と、演算処理の結果を係数に反映する処理が繰り返し実行される。そこで、１つの側面では、本実施の形態は、演算部品が実装されたノードを組み合わせて係数演算を並列に実行するときに、係数演算に使用する係数情報のノード間での処理の時間を短縮することを目的とする。
【課題を解決するための手段】
【０００８】
本発明の一側面は、並列情報処理装置によって例示される。すなわち、本並列情報処理装置は、演算部と処理部とを有するノードを複数備える。それぞれのノードの演算部は、処理対象のデータに対する係数による演算処理を実行し、演算処理の結果を基に係数の変化量を算出し、算出した係数の変化量を処理部に転送するとともに、係数の変化量を並列情報処理装置内の他のノードとの間で授受する処理の実行を処理部に要求する。
【０００９】
それぞれのノードの処理部は、演算部から転送された係数の変化量を並列情報処理装置の他のノードに送信するとともに他のノードで算出された係数の変化量を受信する通信処理と、自ノードの演算部から転送された係数の変化量と他のノードで算出された係数の変化量とを積算する集約処理とを実行する。
そして、演算部および処理部の少なくとも一方が積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新する。
【発明の効果】
【００１０】
本並列情報処理装置によれば、演算部品が実装されたノードを組み合わせて係数演算を並列に実行するときに、係数演算に使用する係数情報のノード間での処理の時間を短縮することができる。
【図面の簡単な説明】
【００１１】
【図１】ニューラルネットワークの処理を例示する図である。
【図２】フォーワード方向の処理とバックワード方向の処理を例示する図である。
【図３】並列情報処理装置の構成図を例示する図である。
【図４】比較例による処理を示す図である。
【図５】比較例による処理を例示するタイムチャートである。
【図６】実施形態１の処理を例示するタイムチャートである。

【図７】実施形態１の計算ノードの処理を例示するフローチャートである。
【図８】実施形態１の計算ノードにおけるデータフローを例示する図である。
【図９】実施形態２の計算ノードの処理を例示するフローチャートである。
【図１０】実施形態２の計算ノードにおけるデータフローを例示する図である。
【図１１】実施形態３の処理を例示するタイムチャートである。
【図１２】実施形態３の計算ノードの処理を例示するフローチャートである。
【図１３】分割重みの反映処理を起動する処理の詳細を例示するフローチャートである。
【図１４】キュー情報を例示する図である。
【図１５】実施形態４の処理を例示するタイムチャートである。
【図１６】学習処理後のメモリ転送において、層１、２が層３よりも優先される処理例のタイムチャートである。
【図１７】実施形態４の学習処理を例示するフローチャートである。
【図１８】実施形態４の処理の起動を例示するフローチャートである。
【図１９】実施形態５の処理のタイムチャートを実施形態４と対比して例示する図である。
【図２０】実施形態５における学習処理結果を集約する集約処理を例示するフローチャートである。
【図２１】実施形態６のタイムチャートを実施形態４と対比して例示する図である。
【図２２】実施形態６における集約処理および反映処理を例示するフローチャートである。

【発明を実施するための形態】

【００１２】

以下、図面を参照して一実施形態に係る並列情報処理装置について説明する。

＜深層学習の処理例＞

【００１３】

図１に、ニューラルネットワークの処理を例示する。ニューラルネットワークは、画像を認識し、識別するためのフォーワード方向の処理と、フォーワード方向の処理で使用するパラメータを決定するバックワード方向の処理（バックワードプロパゲーションともいう）を実行する。

【００１４】

図１のニューラルネットワークは、入力画像に対して、畳み込み演算を実行する畳み込み層（Convolution Layer）の処理と、サブサンプリング層（sub-sampling Layer）の処理とを実行し、画像の特徴を抽出し、画像を識別する。すなわち、図１では、フォーワード方向の処理が例示されている。

【００１５】

フォーワード方向の処理は、入力画像に対して、畳み込み層の処理と、サブサンプリング層の処理を繰り返し実行する特徴抽出部の処理と、識別結果を出力する識別部の処理を含む。特徴抽出部は、入力画像に対して、畳み込み層の処理と、サブサンプリング層の処理を繰り返し実行することで、間引かれた画像を抽出する。畳み込み層の処理は、畳み込み演算ともいう。畳み込み演算は、例えば、Ｎ個×Ｎ個の画素を有する画像の情報（第Ｎ－１層）に対して、例えば、ａ×ｂ個の重みw_{ab}（ａ，ｂ＝０，．．．，ｍ－１）のフィルタによる畳み込み演算を実行することで、次の層（第Ｎ層）の画像の情報を作る。サブサンプリング層の処理は、画像間引き処理であり、プーリング演算ともいう。

【００１６】

畳み込み層およびサブサンプリング層での演算の入力画像および演算の出力画像はフィーチャマップとも呼ばれる。図１の例では、フィーチャマップは、例えば、画像のチャンネル数、あるいは、ＲＧＢ等の色に対応して１つのニューロン層で複数作成されている。

【００１７】

図２に、フォーワード方向の認識処理および識別処理とともに、バックワード方向の処理を例示する。本実施の形態では、フォーワード方向の処理とバックワード方向の処理を

合わせて学習処理と呼ぶ。図２のニューラルネットワークにおいても、フォーワード方向の認識処理は、入力画像に畳み込み演算を実行する畳み込み層、間引き処理を実行するサブサンプリング層（図２ではｐｏｏｌｉｎｇと記載）によって実行される。また、識別結果を出力する識別処理は、全結合層（図２では、Ｆｕｌｌｙ　ｃｏｎｎｅｃｔｅｄと記載）によって実行される。フォーワード方向の畳み込み層とサブサンプリング層とを１つのニューロン層という。また、フォーワード方向の全結合層も１つのニューロン層ということができる。

【００１８】

フォーワード方向の処理の結果は、正解値と比較され、比較結果である差分値がエラーとして出力される。エラーは、バックワード方向に各ニューロン層によって処理される。バックワード方向の処理は、全結合層のエラーから、バックワード方向に順次、各ニューロン層でのエラーの評価関数（ＥＲＲＯＲ）および各ニューロン層での次の重みを計算する処理である。図２では、現在の重みとして、畳み込み層（１層）における１つの重みw_iと、全結合層（１層）における１つの重みw_jが例示されている。また、次の重みとして、畳み込み層（１層）における１つの重みw_{i+1}と、全結合層（１層）における１つの重みw_{j+1}が例示されている。

【００１９】

勾配降下法によるニューラルネットワークの学習処理においては、エラーの評価関数（ＥＲＲＯＲ）の勾配と、学習係数イータの積が重みｗの変化量（例えば、現在の重みｗｔと次の重みｗ＋１の差分値）となる。すなわち、深層学習においては、フォーワード方向に各ニューロン層の処理が実行され、バックワード方向に、各ニューロン層でのエラーの評価関数（ＥＲＲＯＲ）が伝搬される。そして、各ニューロン層は、バックワード方向に伝搬するエラーの評価関数（ＥＲＲＯＲ）から、エラーの評価関数（ＥＲＲＯＲ）の勾配を求める。そして、各ニューロン層は、エラーの評価関数（ＥＲＲＯＲ）が小さくする方向でのエラーの評価関数（ＥＲＲＯＲ）の勾配と、学習係数イータの積から重みｗｔの変化量（勾配情報ともいう）を算出し、次回の重みｗｔ＋１を求める。ここで、現在の重みをｗｔで表し、次回の演算で使用される重みをｗ＋１で表した。また、図１で説明したように、学習処理において、重みｗは１以上の成分を有する係数列（ベクトル）である。

【００２０】

このようにして、バックワード方向に順次、各ニューロン層で、エラーの評価関数（ＥＲＲＯＲ）を小さくする方向に重みを変化させるための変化量が求められる。そして、バックワード方向に順次伝搬するエラーの評価関数（ＥＲＲＯＲ）と重みｗの変化量が算出され、最終的に、入力層に最も近い層の重みｗの変化量が算出される。重みｗｔの変化量は、各層において、次回の重みｗｔ＋１に反映され、次回の学習処理に使用される。なお、以下の説明においては、並列演算処理装置における学習処理の時間の短縮について説明するが、学習処理自体のアルゴリズムの詳細は省略する。

＜構成＞

【００２１】

図３に、並列情報処理装置１の構成図を例示する。並列情報処理装置１は、計算ノード１０－１、１０－２、１０－３、１０－４等を有する。計算ノード１０－１、１０－２、１０－３、１０－４等は、ノード間高速ネットワーク２０で接続される。以下、計算ノード１０－１等を総称する場合には、単に計算ノード１０という。本実施の形態において計算ノード１０の数に限定がある訳ではない。並列情報処理装置１は、本実施形態の情報処理方法を実行する。

【００２２】

計算ノード１０は、Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ（ＣＰＵ１１）とメモリ１２とＧｒａｐｈｉｃｓ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ（ＧＰＵ１３）とメモリ１４を有する。ＣＰＵ１１とＧＰＵ１３とはバス１５によって接続される。さらにバス１５を介して、ＣＰＵ１１とＧＰＵ１３とは、ノード間インターフェース（ノード間ＩＦ１６）に接続される。計算ノード１０はノードの一例である。

【００２３】
ＣＰＵ１１は、メモリ１２に実行可能に展開されたコンピュータプログラムにしたがって、計算ノード１０の処理、例えば、他の計算ノード１０との通信処理、あるいは、ＧＰＵ１３を制御し、管理する処理を実行する。ＣＰＵ１１は、ＭＰＵ（Microprocessor）、プロセッサとも呼ばれる。ＣＰＵ１１は、単一のプロセッサに限定される訳ではなく、マルチプロセッサ構成であってもよい。また、単一のソケットで接続される単一のＣＰＵ１１がマルチコア構成を有していても良い。上記ＣＰＵ１１の少なくとも一部の処理は、ＣＰＵ１１以外のプロセッサ、例えば、ＧＰＵ１３で実行されてもよい。ＣＰＵ１１は、処理部の一例である。メモリ１２は、ＣＰＵ１１で実行されるコンピュータプログラム、ＣＰＵ１１が処理するデータを格納する。

【００２４】
ＧＰＵ１３は、例えば、高速のＶＲＡＭ、高速の演算器を複数搭載し、積和演算機能等を高速に実行する。ＧＰＵ１３は、メモリ１４に実行可能に展開されたコンピュータプログラムにしたがって、計算ノード１０の処理のうち、例えば、学習処理を実行する。ＧＰＵ１３は、演算部の一例である。メモリ１４は、ＧＰＵ１３で実行されるコンピュータプログラム、ＧＰＵ１３が処理するデータを格納する。

【００２５】
上記ＣＰＵ１１およびＧＰＵ１３の少なくとも一部の処理は、例えば、Ｄｉｇｉｔａｌ Ｓｉｇｎａｌ Ｐｒｏｃｅｓｓｏｒ（ＤＳＰ）、数値演算プロセッサ、ベクトルプロセッサ、画像処理プロセッサ等の専用プロセッサで行われても良い。また、上記各部の少なくとも一部の処理は、集積回路（ＩＣ）、その他のディジタル回路で実行されてもよい。また、上記各部の少なくとも一部にアナログ回路が含まれても良い。集積回路は、ＬＳＩ，Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｃｉｒｃｕｉｔ（ＡＳＩＣ），プログラマブルロジックデバイス（ＰＬＤ）を含む。ＰＬＤは、例えば、Ｆｉｅｌｄ－Ｐｒｏｇｒａｍｍａｂｌｅ Ｇａｔｅ Ａｒｒａｙ（ＦＰＧＡ）を含む。

【００２６】
すなわち、ＣＰＵ１１あるいはＧＰＵ１３の処理の少なくとも一部は、プロセッサと集積回路との組み合わせであっても良い。組み合わせは、例えば、マイクロコントローラ（ＭＣＵ），ＳｏＣ（Ｓｙｓｔｅｍ－ｏｎ－ａ－ｃｈｉｐ），システムＬＳＩ，チップセットなどと呼ばれる。

【００２７】
ＢＵＳ１５は、ＣＰＵ１１およびＧＰＵ１３の例えば内部バスに接続され、ＣＰＵ１１およびＧＰＵ１３を相互に接続する。また、ＢＵＳ１５は、ＣＰＵ１１およびＧＰＵ１３をノード間ＩＦ１６に接続する。ＢＵＳ１５は、例えば、ＰＣＩ－Ｅｘｐｒｅｓｓの規格に従うバスである。

【００２８】
ノード間ＩＦ１６は、ノード間高速ネットワーク２０を介して計算ノード１０同士を接続するインターフェースである。ノード間高速ネットワーク２０は、例えば、クロスバー、インターコネクト等とも呼ばれる。なお、ノード間高速ネットワーク２０は、どのようなネットワーク構成であってもよい。例えば、ノード間高速ネットワーク２０は、トーラス構造のメッシュであってもよいし、Ｌｏｃａｌ Ａｒｅａ Ｎｅｔｗｏｒｋ（ＬＡＮ）のようなバス型のネットワークであってもよい。

＜複数ノードによる学習処理＞

【００２９】
学習処理では、まず、フォーワード方向の処理が、各ニューロン層に対して、それぞれのニューロン層が持つ重みパラメータ(w)を用いて、バッチ単位で実行され、次に、バックワード方向の処理が各ニューロン層に対して順次実行される。ここで、バッチ単位とは、学習処理の対象をまとめた処理の単位である。例えば、ニューラルネットワークが画像の認識を行う場合に、バッチ単位として、数十枚から数千枚分の画像のデータが学習処理に用いられ、画像の認識と、正解判定が繰り返し実行される。

【００３０】
図３に例示した複数の計算ノード１０がバッチ内の画像データを分担して処理することで、学習処理が並列に実行される。一度のバッチ単位での学習処理の結果としては、重みパラメータ(w)の変化量(Δw)が算出される。図１で述べたように、重みパラメータ（ｗ）は、１以上の成分を有するベクトルである。以下、重みパラメータ（ｗ）は、単に重み（ｗ）ともいう。上述のように、重み（ｗ）の変化量（Δｗ）は、エラーの評価関数（ＥＲＲＯＲ）を小さくする方向に算出される。各計算ノード１０は、次のバッチ処理に向けて、自身のバッチ単位での重み（ｗ）の変化量（Δｗ）の計算結果と、他の計算ノード１０でのバッチ単位での重み（ｗ）の変化量（Δｗ）の計算結果と相互に授受し、相互の計算結果を積算する。重み（ｗ）の変化量（Δｗ）の計算ノード１０相互の積算処理を集約処理ともいう。そして、各計算ノード１０は、相互の計算結果を集約処理した変化量（Δｗ）を用いて、重み（ｗ）の更新処理を行う。各層の重み（ｗ）を集約処理された変化量（Δｗ）を用いて更新することを、集約処理された変化量（Δｗ）を重み（ｗ）に反映する、ともいう。
【００３１】
３以上のノード数の計算ノード１０が相互に計算結果を授受する場合、計算ノード１０の１対１の通信が複数回実行される。例えば、計算ノード１０－１、１０－２、１０－３、１０－４が相互に情報をバタフライ方式（Ｒｅｃｕｒｓｉｖｅ　Ｄｏｕｂｌｉｎｇ）で授受する場合、まず、１回目の授受で、計算ノード１０－１と計算ノード１０－２が情報を授受し、計算ノード１０－３と計算ノード１０－４が情報を授受する。次に、２回目の授受で、計算ノード１０－１と計算ノード１０－３が情報を授受し、計算ノード１０－２と計算ノード１０－４が情報を授受する。以上の２回の情報の授受によって、計算ノード１０－１、１０－２、１０－３、１０－４の間での情報の授受が完了する。
【００３２】
本実施の形態で、ノード間通信アルゴリズムはＲｅｃｕｒｓｉｖｅ　Ｄｏｕｂｌｉｎｇに限定される訳ではない。例えば、ノード間通信アルゴリズムとして、Ｒｅｄｕｃｅ＋Ｂｒｏａｄｃａｓｔ（Ｂｃａｓｔ），　Ｒｅｄｕｃｅ＿ｓｃａｔｔｅｒ＋Ａｌｌｇａｔｈｅｒ等の方式を用いてもよい。このようなノード間通信処理は、ＭＰＩ　ＡｌｌＲｅｄｕｃｅ処理として、コンピュータプログラムが提供されている。なお、以下の実施の形態説明では、ＭＰＩ　ＡｌｌＲｅｄｕｃｅ処理が実装された計算ノード１０を用いて説明するが、計算ノード１０間の通信処理がＭＰＩ　ＡｌｌＲｅｄｕｃｅ処理に限定される訳ではない。また、計算ノード１０間の通信処理が実行されるネットワーク構成に限定がある訳ではなく、どのようなネットワーク構成が用いられてもよい。
＜比較例＞
【００３３】
比較例では、図２に例示したニューラルネットワークに含まれる各ニューロン層（例えば、ニューロン層１からＮ）が１つの計算ノード１０内に構築される。つまり、比較例では、各ニューロン層の処理は、計算ノード１０のコンピュータプログラムによって実行される。なお、以下の説明で用いる図中には、ニューロン層ＮをＬａｙｅｒ　Ｎのように記述する。
【００３４】
図４に、比較例による処理を示す。比較例では、それぞれの計算ノード１０が図２に例示したフォーワード処理およびバックワード処理を実行する。また、比較例では、計算ノード１０は、フォーワード方向の処理をすべてのニューロン層（ニューロン層１からＮ）において順次実行する（Ｓ３０１）。次に、計算ノード１０は、バックワード方向の処理をすべてのニューロン層（ニューロン層Ｎから１）において順次実行する（Ｓ３０２）。
【００３５】
各計算ノード１０は、各ニューロン層１～Ｎにおける重み（ｗ）の変化量（Δｗ）を相互に転送し、相互に転送した演算結果（各ニューロン層１～Ｎにおける重みwの変化量Δｗ）を積算する。上述のように、それぞれの計算ノード１０において計算された演算結果

をそれぞれの計算ノード１０で積算することを集約するともいう（Ｓ３０３）。そして、各計算ノードは、集約した各ニューロン層１～Ｎにおける重み（ｗ）の変化量（Δｗ）を各層の重み（ｗ）に反映する（Ｓ３０４）。そして、計算ノード１０は、学習処理の繰り返しを終了するか否かを判定する（Ｓ３０５）。ここで、計算ノード１０は、未学習のバッチが存在する場合には、処理をＳ３０１に戻し、次のバッチでの学習処理を実行する（Ｓ３０５でＮＯ）。一方、計算ノード１０は、すべてのバッチで学習した場合には、処理を終了する（Ｓ３０５でＹＥＳ）。

【００３６】

図５は、比較例による処理を例示するタイムチャートである。図５では、比較のため、単一ノードでの処理も例示されている。図５の左側に例示したように、単一ノードでの処理は、バッチ単位での学習処理、重み（ｗ）の更新処理、バッチ単位での学習処理の繰り返しとなる。

【００３７】

一方、図５の右側に例示したように、複数ノードでは、バッチ単位での学習処理が、計算ノード１０の数だけ並列で実行可能である。しかしながら、それぞれの計算ノード１０は、バッチ単位での学習処理が終了すると、重み（ｗ）の変化量（Δｗ）をノード間通信で授受し、集約した後に、それぞれの計算ノード１０での重み（ｗ）を更新することになる。したがって、比較例の処理では、計算ノード１０の数が増加しても、ノード間通信・集約処理、更新処理の時間が増加し、計算ノード数の増加による学習処理の時間短縮効果が十分に発揮されない結果となる。

＜実施形態１＞

【００３８】

図６は、実施形態１の処理を例示するタイムチャートである。ところで、計算ノード１０の構成要素のうち、ＧＰＵ１３は、グラフィックス処理で用いる積和演算を高速に実行する。したがって、ＧＰＵ１３は、学習処理で主体となる重み（ｗ）による演算を高速に実行可能である。しかしながら、学習処理、ノード間通信・集約処理、反映処理を演算部が主体となって処理すると、処理手順としては、図４のフローチャートと同様であり、重み（ｗ）の変化量（Δｗ）をノード間通信で授受し、集約処理、反映処理を実行する時間が無視できない。

【００３９】

そこで、実施形態１の並列情報処理装置１は、演算部（ＧＰＵ１３）と処理部（ＣＰＵ１１）を備えた計算ノード１０を複数備え、学習処理を演算部（ＧＰＵ１３）で行い、ノード間通信、集約処理、反映処理は処理部（ＣＰＵ１１）で行う。

（１）学習処理

【００４０】

学習処理は、主にＧＰＵ１３で実行される。学習処理は、ニューロン層毎にフォーワード処理とバックワード処理（ニューロン層の処理の順番はフォーワード処理の逆）を順に行う。複数の計算ノード１０がバッチ内の画像データを分担して処理することで、並列に学習処理が実行される。図６では、ニューロン層として、ニューロン層１（ＬＡＹＥＲ１）から４（ＬＡＹＥＲ４）が例示されている。ニューロン層１から４は、複数階層の一例である。各ニューロン層１から４におけるフォーワード処理およびバックワード処理は、層別処理の一例である。また、各ニューロン層１から４におけるフォーワード処理およびバックワード処理は、それぞれの階層の前の階層から入力されるデータに係数による演算を実行して次の階層に出力する処理の一例である。フォーワード処理がニューロン層１から４の順に実行され、バックワード処理がニューロン層４から１の順に実行されることは、所定順序の一例である。

（２）メモリ転送(ＧＰＵ１３からＣＰＵ１１への転送)

【００４１】

演算部（ＧＰＵ１３）は、学習処理の各ニューロン層で計算された重み（ｗ）の変化量（Δｗ）を学習処理が終わったニューロン層ごとに順次、処理部（ＣＰＵ１１）へメモリ

転送する。これによって、演算部（ＧＰＵ１３）は、ニューロン層ごとに、ノード間通信・集約処理、反映処理を処理部（ＣＰＵ１１）に開始させる。ニューロン層ごとにノード間通信・集約処理、反映処理を開始することで、次のバッチ単位での学習処理の開始を早め、高速化が実現される。

【００４２】

具体的には、各計算ノード１０において各層のバックワード処理が終わる毎に、演算部（ＧＰＵ１３）に割り当てられた学習処理用のスレッドはメモリ転送を起動するためのキューを発行する。キューは要求と呼ぶこともできる。メモリ転送(ＧＰＵ１３からＣＰＵ１１への転送)用処理スレッドは、キューを受けると転送対象のデータをＧＰＵ１３からＣＰＵ１１へ転送し、最後に集約処理のキューをＣＰＵ１１に発行する。図６では、ニューロン層として、ニューロン層４（ＬＡＹＥＲ４）から層１（ＬＡＹＥＲ１）のバックワード処理で、重みの変化量としてそれぞれ、Δ　ＷＬ４－１、Δ　ＷＬ３、Δ　ＷＬ２、Δ　ＷＬ１が算出されている。

（３）集約処理および（４）ノード間通信

【００４３】

予め、指定数（１個から数十個）が用意されている集約処理用スレッドは、キューを受けると、まず、ノード間通信処理のためのキューを発行する。ノード間通信処理用スレッドは、ノード間通信処理のためのキューを受けるとノード間通信の Ｍｅｓｓａｇｅ　Ｐａｓｓｉｎｇ　Ｉｎｔｅｒｆａｃｅ（ＭＰＩ）リクエストを、ノンブロッキング通信を指定してＭＰＩ通信プログラムに投入する。リクエストに対応する通信が完了した時点で、ＭＰＩ通信プログラムから集約処理用スレッドへ通信完了が通知され、集約処理用スレッドにしたがい集約処理が実行される。集約処理には多数回の演算が実行されるため、集約処理は複数のスレッドを並列で実行することで高速化を実現する。すなわち、計算ノード１０に複数のＣＰＵ１１が搭載される場合には、複数のスレッドを並列で実行することで、ＣＰＵ１１による並列処理が実行される。また、単一のＣＰＵ１１がマルチコアを有する場合も同様である。

【００４４】

図６では、第１回目のノード間通信において、例えば、ニューロン層４（ＬＡＹＥＲ４）については、ノード間通信用スレッドは、ΔＷＬ４－１を他ノードに送信し、ΔＷＬ４－２を他ノードから受信する。そして、集約処理用のスレッド１は、ΔＷＬ４－１とΔＷＬ４－２を積算し、集約処理を実行する。集約処理によってΔＷＬ４－１＋ΔＷＬ４－２が得られる。

【００４５】

次に、第１回目のノード間通信において、例えば、ニューロン層４（ＬＡＹＥＲ４）については、ノード間通信用スレッドは、ΔＷＬ４－１＋ΔＷＬ４－２を他ノードに送信し、ΔＷＬ４－３＋ΔＷＬ４－４を他ノードから受信する。そして、集約処理のスレッド１は、ΔＷＬ４－１＋ΔＷＬ４－２とΔＷＬ４－３＋ΔＷＬ４－４を積算し、集約処理を実行する。図６のスレッド１から３は、一例として、それぞれの階層での係数の変化量に対する集約処理を２以上並列に実行する。

（５）メモリ転送（ＣＰＵ１１からＧＰＵ１３への転送）

【００４６】

他の全ノードと情報を授受するための回数分のノード間通信と集約処理が完了すると、ＣＰＵ１１は、メモリ転送（ＣＰＵ１１からＧＰＵ１３への転送）処理のキューを発行する。メモリ転送処理用スレッドがキューを受けて、メモリ転送（ＣＰＵ１１からＧＰＵ１３への転送）を実行する。

（６）反映処理

【００４７】

各層のメモリ転送（ＣＰＵ１１からＧＰＵ１３への転送）が完了すると、主にＧＰＵ１３側での反映処理が、メモリ転送が完了したニューロン層から順に実行される。

【００４８】

図７は、実施形態１の計算ノード１０の処理を例示するフローチャートである。図で左側のフローチャートは、主としてＧＰＵ１３が実行する学習処理と反映処理を例示する。また、右側のフローチャートは、主としてＣＰＵ１１が実行するノード間通信・集約処理を例示する。図７の処理では、まず、ＧＰＵ１１がニューロン層（例えば、ニューロン層１からＮ）について、フォーワード処理を実行する（Ｓ１１）。

【００４９】

フォーワード処理は、図１に例示したように、入力データと重み（ｗ）とによる演算処理である。演算処理は、例えば、入力データの要素ｘ（ｉ，ｊ）とａ×ｂ個の重みw_{ab}（ａ，ｂ＝０，．．．，ｍ－１）のフィルタによる畳み込み演算、サブサンプリング層のプーリング演算、全結合層の演算等である。Ｓ１１の処理は、処理対象のデータに対する係数による演算処理の一例である。

【００５０】

次に、ＧＰＵ１３は、バックワード方向にニューロン層Ｎから１のループ（ＬＡＹＥＲループ（Ｌ）、開始＝Ｎ、終了＝１）の中で、Ｓ１２、Ｓ１３の処理を実行する。Ｓ１２の処理では、ＧＰＵ１３は、バックワード方向に各ニューロン層（Ｌ）において上位の層（Ｌ＋１）でのエラーの評価関数（ＥＲＲＯＲ）から当該ニューロン層（Ｌ）におけるエラーの評価関数（ＥＲＲＯＲ）を求める。そして、ＧＰＵ１３は、当該ニューロン層（Ｌ）のエラーの評価関数（ＥＲＲＯＲ）に基づいて当該ニューロン層（Ｌ）のエラーの評価関数（ＥＲＲＯＲ）を減少させる方向の重み（ｗ）の変化量（Δｗ）を求める。Ｓ１２の処理は、演算処理の結果を基に係数の変化量を算出することの一例である。Ｓ１２の処理は、それぞれの階層での層別処理の結果を基にそれぞれの階層での係数の変化量を算出することの一例でもある。

【００５１】

また、Ｓ１３の処理は、ＣＰＵ１１に対して重みの変化量（Δｗ）の集約処理の起動を要求する処理である。Ｓ１３の処理により、ＧＰＵ１３は、Ｓ１２で求めた当該ニューロン層（Ｌ）について計算された重み（ｗ）の変化量（Δｗ）をＣＰＵ１１にメモリ転送するとともに、集約処理を実行するＣＰＵ１１のスレッドにキューを登録する（Ｓ１３）。したがって、実施形態１では、各ニューロン層（Ｌ）でバックワード処理が終了するごとに、ＣＰＵ１１に対して、重み（ｗ）の変化量（Δｗ）の集約処理の起動が要求される。Ｓ１３の処理は、算出した係数の変化量を処理部に転送するとともに、係数の変化量を並列情報処理装置内の他のノードとの間で授受する処理の実行を処理部に要求することの一例である。Ｓ１３の処理は、算出した係数の変化量を処理部に転送することの一例でもある。

【００５２】

以降、ＧＰＵ１３は、ＣＰＵ１１からの重み（ｗ）の変化量（Δｗ）の集約処理の完了を全ニューロン層数分待つ（Ｓ１４）。そして、ＣＰＵ１１で集約処理された各ニューロン層（Ｌ）の重み（ｗ）の変化量（Δｗ）が、ＣＰＵ１１からＧＰＵ１３にメモリ転送される。そして、全レイヤの集約処理が完了すると、ＧＰＵ１３は、集約処理された変化量（Δｗ）を各層の重み（ｗ）に反映する（Ｓ１５）。すなわち、ＧＰＵ１３は、次のバッチのフォーワード処理およびバックワード処理で使用される各層の重み（ｗ）を更新する。Ｓ１５の処理は、演算部が積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新することの一例である。

【００５３】

そして、ＧＰＵ１３は、学習の終わりか否かを判定する（Ｓ１６）。学習の終わりとは、例えば、計算ノード１０について用意されたすべてのバッチが終了する場合である。計算ノード１０について用意された未学習のバッチが残っている場合には、ＧＰＵ１１３は、処理をＳ１１に戻し、次のバッチを実行する。

【００５４】

Ｓ１３の処理によって、集約処理の起動が要求されると、キューがＣＰＵ１１のスレッドに登録され、キューが順次処理される。ＣＰＵ１１は、まず、メモリ転送を実行し、Ｇ

ＰＵ１３で計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を取得する（Ｓ２１）。そして、ニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を他の計算ノード１０との間で授受する。上述のように、本実施の形態では、ノード間のデータ交換の処理として、ＭＰＩ仕様のＡＬＬＲｅｄｕｃｅアルゴリズムが用いられる。ただし、本実施の形態のノード間のデータ交換の処理がＡＬＬＲｅｄｕｃｅアルゴリズムに限定される訳ではない。図７において、ＣＰＵ１１は、ＭＰＩ　ＡＬＬＲｅｄｕｃｅの階層ループにおいて、Ｓ２２からＳ２４の処理を繰り返し実行する。

【００５５】

例えば、ノード数が４であって（計算ノード１０－１～１０－４）、Ｒｅｃｕｒｓｉｖｅ　Ｄｏｕｂｌｉｎｇの場合には、以下の処理が実行される。計算ノード１０－１と１０－２の組と計算ノード１０－３と１０－４の組のそれぞれ組でＣＰＵ１１が、Ｓ２２からＳ２４の処理を実行する。すなわち、自ノードで計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を相手ノードに送信する（Ｓ２２）。Ｓ２２の処理は、演算部から転送された係数の変化量を並列情報処理装置の他のノードに送信することの一例である。

【００５６】

また、ＣＰＵ１１は相手ノードで計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を受信する（Ｓ２３）。Ｓ２３の処理は、他のノードで算出された係数の変化量を受信することの一例である。したがって、Ｓ２２とＳ２３の処理は通信処理の一例である。

【００５７】

そして、ＣＰＵ１１は自ノードで計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）と相手ノードで計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を積算する（Ｓ２４）。Ｓ２４の処理は、演算部から転送された係数の変化量と他のノードで算出された係数の変化量とを積算する集約処理の一例である。

【００５８】

さらに、計算ノード１０－１と１０－３の組と計算ノード１０－２と１０－４の組のそれぞれの組でＣＰＵ１１が、Ｓ２２からＳ２４の処理を実行する。この処理によって、計算ノード１０－１～１０－４の間でニューロン層Ｌの重み（ｗ）の変化量（Δｗ）が集約される。ニューロン層Ｌの重み（ｗ）の変化量（Δｗ）が集約されると、ＣＰＵ１１は、集約されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）をメモリ転送し、ＧＰＵ１３に戻す（Ｓ２６）。計算ノード１０は、Ｓ２１からＳ２６の処理をキューの蓄積順にすべてのニューロン層Ｌについて繰り返し実行する。

【００５９】

図８に、実施形態１の計算ノード１０におけるデータフローを例示する。計算ノード１０においては、まず、ＧＰＵ１３による学習処理では、ＧＰＵ１３による演算結果がＧＰＵ１３のメモリ１４に格納される（矢印Ａ１）。上述のように演算結果は、ニューロン層Ｌの重み（ｗ）の変化量（Δｗ）である。

【００６０】

次に、ノード間通信処理が実行される。まず、ＧＰＵ１３とＣＰＵ１１との間のメモリ転送が実行され、メモリ１４に格納されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ）がＣＰＵ１１のメモリ１２へ転送される（矢印Ａ２－１）。ここでは、メモリ１２に格納された重み（ｗ）の変化量をΔｗ１とする。そして、ノード間ＩＦを介して、メモリ１２に格納された重み（ｗ）の変化量（Δｗ１）が他の計算ノード１０に送信される（矢印Ａ２－２）。一方、ノード間ＩＦを介して、他の計算ノード１０で計算されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ２）が当該計算ノード１０で受信される（矢印Ａ２－３）。

【００６１】

さらに集約処理が実行される（矢印Ａ３）。集約処理では、ＣＰＵ１１は、メモリ12のデータ（変化量Δｗ１とΔｗ２）を加算する。ここでは、加算結果は集約された重みの変化量としてΔｗ２に保持されるとする。ノード数が３以上の場合には、矢印Ａ２－２からＡ３がノード間通信のアルゴリズムで実行される回数だけ繰り返される。

【００６２】
そして、ＧＰＵ１１は、メモリ転送でＧＰＵ１３に集約されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ２）を転送する（矢印Ａ５－１）。転送先のＧＰＵ１３は、転送された重みの変化量を変化量（Δｗ）に保存する。そして、ＧＰＵ１３は、集約された層Ｌの重み（ｗ）の変化量（Δｗ）を使用して、重み（ｗ）を更新する（Ａ５－２）。
【００６３】
以上述べたように、実施形態１の並列情報処理装置１は、複数の計算ノード１０がそれぞれのバッチによって、入力データに対する重み（ｗ）の演算を複数のニューロン層について実行するため、重み（ｗ）の学習処理が並列に実行される。そして、並列に実行された学習処理によって得られた重み（ｗ）の変化量（Δｗ）を複数の計算ノード１０間で集約し、各ニューロン層についてのすべての計算ノード１０のバッチの結果を反映した重み（ｗ）を各計算ノード１０が取得する。
【００６４】
このような処理において、各計算ノード１０は、ＧＰＵ１３が各ニューロン層の学習処理を順次実行する。すなわち、ＧＰＵ１３は、フォーワード方向にニューロン層１からニューロン層Ｎに対して、重み（ｗ）による演算を実行する。次に、ＧＰＵ１３は、バックワード方向にニューロン層Ｎからニューロン層１について、各ニューロン層Ｌの重み（ｗ）の変化量（Δｗ）を計算する処理を実行する。そして、各ニューロン層Ｌの重み（ｗ）の変化量（Δｗ）の計算が終了するごとに、ＧＰＵ１３は、計算した変化量（Δｗ）をＣＰＵ１１にメモリ転送するとともに、集約処理のキューをＣＰＵ１１のスレッドに発行し、集約処理を依頼する。
【００６５】
以上述べたように、積和演算等の重み（ｗ）による演算を高速に実行可能なＧＰＵ１３が学習処理を複数の計算ノード１０において並列に実行し、ＣＰＵ１１が重みの変化量（Δｗ）のメモリ転送、ノード間通信、および集約処理を実行する。したがって、ＧＰＵ１３は、ＣＰＵ１１との連携により、もっぱら学習処理を実行すればよく、ＧＰＵ１３の演算性能が発揮されやすい。
【００６６】
また、ＣＰＵ１１は、集約処理の依頼を受けると、キューの順に、ノード間通信を実行する。例えば、ＣＰＵ１１は、ＡＬＬＲｅｄｕｃｅアルゴリムにより、自ノードで計算した重み（ｗ）の変化量（Δｗ）を他の計算ノード１０に送信するとともに、他の計算ノード１０で得られた計算結果を受信する。そして、ＣＰＵ１１は、ニューロン層ごとに、順次重み（ｗ）の変化量（Δｗ）を集約する。したがって、比較例で例示した図４のようにバックワード方向の処理がすべてのニューロン層について完了した後に重み（ｗ）の変化量（Δｗ）の集約処理を実行する場合と比較して、各層の集約処理が早期に開始される。例えば、ＣＰＵ１１がマルチコア構成の場合には、図６のように、集約処理を複数のスレッドに分けて、異なるニューロン層の集約処理を割り当てることで、複数のニューロン層の集約処理が並列して実行される。
【００６７】
また、あるニューロン層Ｌの集約処理を実行中に、他のニューロン層Ｌ＋１のノード間通信が並列して実行可能である。また、メモリ転送用のスレッドがニューロン層Ｌの集約処理結果をＧＰＵ１３にメモリ転送中に、集約処理の複数のスレッドが複数のレイヤＬ＋１、Ｌ＋２、Ｌ＋３に対して集約処理とノード間通信処理とを並列に実行できる。図５に例示した比較例では、バッチ単位で全ニューロン層について学習処理を実行し、全ニューロン層について集約処理を実行し、全ニューロン層について次の学習処理を実行する。このような比較例の処理に対して、実施形態１の計算ノード１０は、少なくとも集約処理の処理時間が短縮される。また、次のバッチにおけるフォーワード方向の処理の開始を早めることができる。
＜実施形態２＞
【００６８】

図９および図１０により実施形態２に係る並列情報処理装置１について説明する。実施形態２の並列情報処理装置１は、図６に例示した「（６）反映処理」をニューロン層単位でＣＰＵ１１が実行する。そして、ＣＰＵ１１は、ニューロン層単位での反映処理の後に、（５）メモリ転送（ＣＰＵ１１からＧＰＵ１３）を実行する。実施形態２の他の構成および作用は実施形態１と同様である。そこで、実施形態２の並列情報処理装置１の構成要素のうち、実施形態１と同一の構成要素については、同一の符号を付してその説明を省略する。

【００６９】

図９に、実施形態２の計算ノード１０の処理を例示するフローチャートである。図９の処理は、変化量（Δｗ）を重み（ｗ）に反映する処理がＧＰＵ１３ではなく、ＣＰＵ１１によって実行される点で図７と相違する。例えば、図９では、ノード間通信・集約処理において、Ｓ２５の処理が追加されている。

【００７０】

まず、ＧＰＵ１３は、学習処理によって計算された変化量（Δｗ）を重み（ｗ）に反映する処理を起動する（Ｓ１３Ａ）。このとき、メモリ転送処理によってＧＰＵ１３からＣＰＵ１１に当該ニューロン層の重み（ｗ）の変化量（Δｗ）が送信される点は図７と同様である。すると、ＧＰＵ１３は、キューの優先順に変化量（Δｗ）のメモリ転送（Ｓ２１）、および集約処理を実行する（Ｓ２２－Ｓ２４）。そして、ＭＰＩ　ＡＬＬＲｅｄｕｃｅ階層ループが終了すると、ＣＰＵ１１は、集約処理されたあるニューロン層Ｌの重みの変化量（Δｗ）を重み（ｗ）に反映する（Ｓ２５）。Ｓ２５の処理は、処理部が積算された係数の変化量を基に次回以降の演算処理で使用される係数を更新することの一例である。

【００７１】

そして、ＣＰＵ１１は、変化量（Δｗ）が反映された重み（ｗ）をメモリ転送でＧＰＵ１３に送信する（Ｓ２６Ａ）。すると、ＧＰＵ１３は、変化量（Δｗ）が反映された重み（ｗ）をメモリ転送で受信し、メモリ１４に保存する（Ｓ１４Ａ）。そして、ＧＰＵ１３は、未学習のバッチが残っている場合には（Ｓ１６でＮ）、次のバッチの学習を実行する。

【００７２】

図１０に、実施形態２の計算ノード１０におけるデータフローを例示する。図１０の処理は、学習処理（矢印Ａ１）、ノード間通信処理（Ａ２－２、Ａ２－３）、集約処理（矢印Ａ３）までは、図８と同一である。ただし、ノード間通信処理前のメモリ転送（矢印Ａ２－１）において、ＣＰＵ１１は、ＧＰＵ１３から重みの変化量（Δｗ）とともに重み（ｗ）を受信し、ｗ１としてメモリ１２に格納する。

【００７３】

そして、ＣＰＵ１１は、重みの変化量（Δｗ）の集約処理の後、集約された重みの変化量（Δｗ）を重みｗに反映し、重みｗ１としてメモリ１２に格納する（矢印Ａ５－３）。そして、ＣＰＵ１１は、重みの変化量（Δｗ）が反映された重み（ｗ１）をメモリ転送でＧＰＵに転送し、重み（ｗ）としてメモリ１４に保存する（矢印Ａ５－４）。

【００７４】

以上述べたように、実施形態２では、ＣＰＵ１１が変化量（Δｗ）を重み（ｗ）に反映する処理を実行する。この構成および手順により、ＧＰＵ１３は重みの変化量（Δｗ）の演算により専念することが可能となる。また反映処理のスレッドは、集約処理と同様にＣＰＵ１１のコア数に応じて並列処理することで、学習処理の高速処理が可能となる。

＜実施形態３＞

【００７５】

図１１から図１３により実施形態３の並列情報処理装置１について説明する。上記実施形態１では、ＣＰＵ１１が学習結果のノード間通信・集約処理を実行する際に、各ニューロン層単位で処理を分割した。すなわち、ＣＰＵ１１は、１つのニューロン層について学習結果のノード間通信・集約処理を個別に実行し、それぞれのニューロン層の重みの変化

量（Δｗ）が集約されるごとに、ＧＰＵ１３にメモリ転送した。また、実施形態２では、ＣＰＵ１１が重みの変化量（Δｗ）を重み（ｗ）に反映し、ＧＰＵ１３にメモリ転送した。しかし、実施形態１、２の処理でも、１つのニューロン層が大きなパラメータ数の重みを持つ場合には転送処理に時間がかかり、マルチコアのＣＰＵ１１が複数スレッドによって並列処理を実行する構成を有していても、並列化の効果が発揮されない場合がある。そこで、実施形態３では、ＧＰＵ１３およびＣＰＵ１１は、ノード間通信スレッド、複数の集約処理スレッド、および反映処理スレッドの実行単位をニューロン層単位よりも細かく分割して処理する。このような手順により、計算ノード１０は、各処理をパイプライン化し、高速化する。

【００７６】

例えば、あるニューロン層Ｌの重み（ｗ）がｗ＝（ｐ１、ｐ２、・・・・、ｐＸ）のようなパラメータ列であるとする。パラメータ列は、係数列の一例である。つまり、ニューロン層Ｌの重み（ｗ）は、複数使用され、係数列を形成する。そして、学習処理の結果、重みの変化量はΔｗ＝（Δｐ１、Δｐ２、・・・・、ΔｐＸ）のような多数のパラメータ列として計算されるとする。このような場合に、ＧＰＵ１３はΔｗを部分列に区切り、Δｗ１＝（Δｐ１、Δｐ２、・・・、ΔｐＸ１）、Δｗ２＝（ΔｐＸ１＋１、・・・、ΔｐＸ２）、Δｗ３＝（ΔｐＸ２＋１、・・・、ΔｐＸ３）、・・・、Δｗｘ＝（ΔｐＸ-１、・・・、ΔｐＸ）のように分割する。

【００７７】

図１１は、実施形態３の処理を例示するタイムチャートである。なお、図１１では、実施形態３の処理が適用される前のタイムチャート（「適用前」）が、実施形態３の処理が適用された場合のタイムチャートとともに例示されている。適用前の例（図１１の上側）では、ニューロン層Ｎに対するバックワード処理の終了後、ＧＰＵ１３からＣＰＵ１１へのメモリ転送が実行され、その後、スレッド１による集約処理が２回のノード間データ通信（例えば、 ＡＬＬＲｅｄｕｃｅアルゴリズム）とともに実行されている。

【００７８】

一方、適用後の例（図１１の下側）では、ニューロン層Ｎに対するバックワード処理の終了後、ＧＰＵ１３は、学習処理で計算した重みの変化量（Δｗ、パラメータ列）をΔｗ１、Δｗ２、Δｗ３、Δｗ４の部分列に分割し、ＣＰＵ１１にメモリ転送する。

【００７９】

ＣＰＵ１１はメモリ転送で分割された変化量Δｗ１、Δｗ２、Δｗ３、Δｗ４を取得し、集約処理用のスレッド１から３により順次集約処理を起動する。例えば、スレッド１が分割された変化量（Δｗ１）を受け取ると、まず、ノード間通信処理のスレッドを起動する。ノード間通信処理のスレッドは、分割された変化量（Δｗ１）を他の計算ノード１０－２に送信するとともに、計算ノード１０－２からニューロン層Ｎの分割された変化量Δｗ１を受信する。今、自ノードと他ノードとで変化量Δｗ１を区別するため、自ノードで計算されたものをΔｗ１－１とし、計算ノード１０－２で計算されたものをΔｗ１－２とする。スレッド１は、自ノードで計算され、分割された変化量（Δｗ１－１）と、ノード間通信処理で得られた他ノードで計算された変化量（Δｗ１－２）とを積算し、計算ノード１０－２との間で集約処理を実行する。このとき、スレッド１の集約処理と並行してスレッド２は、分割された変化量（Δｗ２）について、ノード間通信処理のスレッドを起動しており、スレッド２もスレッド１と同様に、ノード間通信処理と集約処理をパイプラインで実行する。スレッド３も、スレッド１、２と同様に、ノード間通信処理と集約処理をパイプラインで実行する。

【００８０】

スレッド１は、自ノードで計算された重みの変化量（Δｗ１－１）と他ノード計算された重みの変化量（Δｗ１－２）との間の集約処理が完了すると、再びノード間通信処理のスレッドを起動し、計算ノード１０－３との間で、集約処理を実行する。また、スレッド２、３についても、１回目の集約処理が終了すると、スレッド１と同様に、再びノード間通信処理のスレッドを起動し、計算ノード１０－３との間で、集約処理を実行する。

【００８１】
そして、例えば、スレッド１が分割された変化量（Δｗ１）について、他のすべての計算ノード１０との間で集約処理を完了すると、メモリ転送スレッドを起動する。メモリ転送スレッドにより、ＣＰＵ１１は、集約された変化量（Δｗ１）をＧＰＵ１３に転送する。スレッド２、スレッド３も同様である。
【００８２】
また、スレッド１は、分割された変化量（Δｗ１）についてメモリ転送スレッドのキューを発行すると、分割された次の変化量（Δｗ４）について、分割された変化量（Δｗ１）と同様の処理を実行する。このようにして、例えば、ＣＰＵ１１が複数、例えば、５つのコアを有している場合には、ＣＰＵ１１はスレッド１から３およびメモリ転送スレッド、およびノード間通信スレッドを並行して実行できる。したがって、例えば、ある分割された変化量（Δｗｋ）についてのノード間通信の時間が、別の分割された変化量（Δｗｊ）についての集約処理の時間に実行できる。また、仮に、あるニューロン層Ｌの重み（ｗＬ）のパラメータ数が他の層よりも多いものであっても、ＧＰＵ１３およびＣＰＵ１１は重み（ｗＬ）に含まれるパラメータを複数部分に分割し、複数スレッドで並行して処理できる。
【００８３】
図１２は、実施形態３の計算ノード１０の処理を例示するフローチャートである。図１２の処理は、反映処理の起動と反映処理待ちにおいて、図９の処理と相違する。すなわち、実施形態３においては、図１１で説明したように、ＧＰＵ１３は、ニューロン層のループにおいて（ニューロン層１からＮ）、各ニューロン層Ｌの重みの変化量（ΔｗＬ）を複数の部分に分割する（ΔｗＬｋ、ｋは分割された部分列に対応する数）。そして、ＧＰＵ１３は、メモリ転送を行い、各部分列ごとに集約処理、反映処理を起動する（Ｓ１３Ｂ）。そして、ニューロン層のループの終了後、ＧＰＵ１３は、分割された重みの変化量（ΔｗＬｋ）の反映処理の完了待ちとなる（Ｓ１４Ｂ）。そして、すべてのニューロン層のすべての分割された重みの変化量（ΔｗＬｋ）についての反映処理が終了すると、ＧＰＵ１３は、学習の繰り返しの終了か否かを判定し、未学習のバッチがある場合に、処理をＳ１１に戻し、次のバッチの学習を実行する。
【００８４】
なお、図１２の処理フローは、図９を変形したもので、ＣＰＵ１１が重みの変化量（ΔｗＬｋ）を基に重み（ｗＬｋ）を更新する反映処理を実行する。しかし、図７に例示したように、ＣＰＵ１１が重みの変化量（ΔｗＬｋ）をメモリ転送でＧＰＵ１３に転送し、ＧＰＵ１３が反映処理を実行してもよい。
【００８５】
図１３は、実施形態３におけるＧＰＵ１３によって分割重み（ｗＬｋ）の反映処理を起動する処理（図１２の１３Ａ）の詳細を例示するフローチャートである。この処理では、ＧＰＵ１３は、レイヤＬの重み（ｗＬ）のｋ番目の分割重みの部分列（ｗＬｋ）と重みの変化量（ΔｗＬｋ）のメモリ転送を起動する（Ｓ１３Ｂ１）。Ｓ１３Ｂ１の処理は、係数列を複数の部分列に分割して部分列ごとに変化量を処理部に転送することの一例である。
【００８６】
次に、ＧＰＵ１３は、分割された重みの部分列（ｗＬｋ）の変化量（ΔｗＬｋ）の集約処理、および、重みの部分列（ｗＬｋ）への反映処理をスレッドＳｎ（ｎ＝１～Ｎ）のキューに登録する（Ｓ１３Ｂ２）。Ｓ１３Ｂ２の処理は、部分列ごとに授受する処理の実行を処理部に要求することの一例である。
【００８７】
以上述べたように、本実施形態の並列情報処理装置１は、複数のスレッドによって、メモリ転送（ＧＰＵ１３からＣＰＵ１１）、ノード間通信、集約および反映処理、メモリ転送（ＣＰＵ１１からＧＰＵ１３）を実行できる。さらに、実施形態３では、ＧＰＵ１３はニューロン層Ｌの重みのパラメータ列（ｗＬ）を複数の部分列（ｗＬｋ、ｋ＝１、２、３、・・・）に分ける。そして、ＧＰＵ１３はそれぞれの重みの変化量の部分列（ΔｗＬｋ

、ｋ＝１、２、３、・・・）ごとに、メモリ転送、集約および反映を起動する。すると、ＣＰＵ１１は、重みの変化量の部分列（ΔｗＬｋ、ｋ＝１、２、３、・・・）ごとに、メモリ転送（ＧＰＵ１３からＣＰＵ１１）、集約および反映、メモリ転送（ＣＰＵ１１からＧＰＵ１３）を実行する。したがって、ニューロン層の重み（ｗ）に含まれるパラメータ数が多い場合であっても、メモリ転送、ノード間通信、集約処理のパイプラインを形成し、例えば、ノード間通信処理に要する時間（またはその一部）を集約処理の時間で隠すことができる。なお、重みのパラメータ列（ｗＬ）は、係数列の一例である。

＜実施形態４＞

【００８８】

図１４から図１８により実施形態４を説明する。上記実施形態１から実施形態３では、例えば、学習処理の終了順にニューロン層ごとのデータがメモリ転送され、ノード間通信処理、集約処理、反映処理が実行された。実施形態４では、各スレッドは、ニューロン層のうち、最も階層が低い層、すなわち、図２の入力画像が入力される層（例えば、ニューロン層１））の優先順位を高くし、階層が上がるほど優先順位が低くなるようにキューの発行を制御する。このような処理によって、１つのバッチが終了前にすでに、階層が低いニューロン層の重み（ｗ）に対して、変化量（Δｗ）が反映されている場合には、階層が低いニューロン層における次のバッチの開始を可能とする。

【００８９】

図１４は、Ｒｅｄｕｃｅ処理で用いられるキュー情報を例示する図である。キュー情報は、キュー情報を発行する処理（前処理、キュー情報発行スレッドともいう）から発行され、後続処理（キュー処理スレッドともいう）によって処理される。図１４では、前処理として、処理Ａ－１、処理Ａ－２等が例示されている。また、後続処理として処理Ｂ－１、処理Ｂ－２が例示されている。

【００９０】

図１４の例では、前処理（キュー発行スレッド）は、処理が終わる毎に後続処理のキューを登録する。後続処理（キュー処理スレッド）は、処理が要求されているキューが存在しない場合は何もしない。一方、処理が要求されているキューが存在する場合、後続処理（キュー処理スレッド）は、要求された処理を実行し、処理が終了すると処理完了フラグ情報を更新する。処理完了フラグ情報は、例えば、完了した処理数（または未完了の処理数）のカウンタである。なお、ある前処理が、それ以前に実行される前処理（例えば、処理Ａ－１、処理Ａ－２）に依存する場合には、処理を行う前に、依存する前処理の完了を確認してから処理を開始する。

【００９１】

以上のようにして、後続処理（キュー処理スレッド）は、登録されたキューの順に処理を実行する。以下、実施形態４では、登録されるキューの順を所定の優先順序で優先する制御、具体的には、ニューロン層のうち、階層の低いニューロン層を優先して処理を実行する制御手順を例示する。

【００９２】

図１５は、実施形態４の処理を例示するタイムチャートである。図１５では、ニューロン層として、ニューロン層１から４が想定されている。ただし、実施形態４のニューロン層が４つのニューロン層に限定される訳ではない。バックワード方向の処理がニューロン層４から１までの順でそれぞれ終了すると、この終了順にメモリ転送処理が起動され、ノード間通信処理、集約処理が実行される。さらに、各ニューロン層の集約処理が完了後、メモリ転送（ＣＰＵ１１からＧＰＵ１３）が実行される。

【００９３】

ところで、図１５の例では、ニューロン層１の集約された重みの変化量がＣＰＵ１１からＧＰＵ１３にメモリ転送可能となったときに、まだ、ニューロン層２についても、集約された変化量のメモリ転送が起動されていない。例えば、ニューロン層２のメモリ転送処理（ＣＰＵ１１からＧＰＵ１３）は、キューが登録された状態で未実行の状態となっている。実施形態４では、このような場合にニューロン層１の集約処理が終了すると、集約処

理用のスレッドは、ニューロン層２よりもニューロン層１のメモリ転送を優先する。すなわち、ＣＰＵ１１の集約処理用のスレッドは、ニューロン層１よりもニューロン層１が先に転送されるように、ニューロン層１の集約された変化量のメモリ転送のキューを登録する。そのようなキュー登録の結果、メモリ転送用スレッドはニューロン層２よりもニューロン層１の重みの変化量を先にメモリ転送する。

【００９４】

図１６は、学習処理後のメモリ転送において、層１、２が層３よりも優先される処理例のタイムチャートである。このタイムチャートでは、バックワード方向の処理において、ニューロン層４のメモリ転送中に、ニューロン層３とニューロン層２の学習が完了している。このような場合、階層が入力データに近いニューロン層２がニューロン層３よりも優先されてメモリ転送が開始される。

【００９５】

さらに、ニューロン層２のメモリ転送中に、ニューロン層１の学習処理が完了する。すると、階層が入力データに近いニューロン層１がニューロン層３よりも優先されてメモリ転送が開始される。その後、ニューロン層３のメモリ転送が開始される。

【００９６】

入力データが入力されるニューロン層１を最も優先し、ニューロン層１に近い層の順に優先してメモリ転送を実行することで、その後のノード間通信、集約、反映処理は、ニューロン層１を最も優先し、ニューロン層１に近い層の順に優先する結果となる。したがって、現在のバッチの学習終了後、次のバッチでは、現在のバッチで学習結果がニューロン層１から順に優先して重みwに反映される。したがって、現在のバッチのすべてのニューロン層の処理が完了する前であっても、ＧＰＵ１３は次のバッチでニューロン層１から学習を開始でき、次のバッチ全体の開始時期が早まる。

【００９７】

図１５、図１６のように、階層が低いニューロン層に対する処理の優先順位を高くするため、処理順序の変更は、ＭＰＩ　ＡＬＬＲｅｄｕｃｅ階層ループの単位、もしくは実施例３における重みのパラメータ細分化後の部分列単位で実行される。各処理スレッドは、次のスレッドへのキューの登録時、通常Ｆｉｒｓｔ　Ｉｎ　Ｆｉｒｓｔ　Ｏｕｔ（ＦＩＦＯ）方式でキューを登録する。一方、実施形態４では、各処理スレッドは、処理順序の変更条件（キューが優先順でない状態）が検知された場合には、優先順の位置にキューを登録する。

【００９８】

１つのノードの処理順序の変更により、処理順序が変更されたノードの処理順序が他ノードの処理順序とずれるとノード間転送がロックするため、計算ノード１０同士が同期をとる。同期をとる手法としては、処理順序の変更を検知した計算ノード１０がすべてのノードに処理順序の変更を配信し、各ノードは他ノードでの処理順序の変更に対して、同様に処理の順番を組み直す。

【００９９】

図１７は、実施形態４の学習処理を例示するフローチャートである。この処理では、ＧＰＵ１３は、ニューロン層１からＮについて、フォーワード方向の処理を実行する（Ｓ１１Ｃ）。ただし、Ｓ１１Ｃの処理は、前のバッチにおける全層についての学習処理が終了していなくても開始される点で実施形態１から３と相違する。そして、全層についてのフォーワード方向の処理が終了すると、ＧＰＵ１３は、バックワード方向にニューロン層Ｎから１のループ（ＬＡＹＥＲ　ループ（Ｌ）開始＝Ｎ、終了＝１）の中で、Ｓ１２、Ｓ１３の処理を実行する。Ｓ１２の処理は、実施形態１から３と同様である。

【０１００】

Ｓ１３の処理では、ＧＰＵ１３は、ニューロン層のうち、入力側に近いニューロン層を優先して、ＣＰＵ１１にメモリ転送するとともに、集約処理を実行するＣＰＵ１１のスレッドにキューを登録する（Ｓ１３Ｃ）。Ｓ１３Ｃの処理は、複数階層のうち、演算処理の実行の順序が早い階層の係数の変化量を優先して処理部に転送することの一例である。

【０１０１】
したがって、実施形態１では、ＧＰＵ１３は、各ニューロン層（Ｌ）でバックワード方向の処理が終了するごとに、優先順の制御を実行する。すなわち、ＧＰＵ１３は、バックワード方向の処理が終了したニューロン層（Ｌ）より上位のニューロン層（Ｌ＋ｋ）で、メモリ転送および集約処理が未実行のニューロン層がキューに残っていないか否かを判定する。そして、バックワード方向の処理が終了したニューロン層（Ｌ）より上位のニューロン層（Ｌ＋ｋ）がキューに残っている場合には、ＧＰＵ１３は、入力側に近い下位のニューロン層（Ｌ）を優先してキューを登録する。なお、このように、下位のニューロン層を優先するキューの登録は、ＣＰＵ１１がノード間通信およびメモリ転送（ＣＰＵ１１からＧＰＵ１３）のキューを登録する場合も同様である。

【０１０２】
そして、ＧＰＵ１３は、ＣＰＵ１１からの重み（ｗ）の変化量（Δｗ）の集約処理の完了を待つ。ただし、実施形態４では、ＧＰＵ１３は、ニューロン層１層ずつ、集約処理の完了を待つ（Ｓ１４Ｃ）。

【０１０３】
その後、ＣＰＵ１１で集約処理された各ニューロン層（Ｌ）の重みの変化量（Δｗ）が、ＣＰＵ１１からＧＰＵ１３にメモリ転送される。あるニューロン層（Ｌ）の集約処理が完了すると、ＧＰＵ１３は、当該ニューロン層において集約処理された変化量（Δｗ）を重み（ｗ）に反映する（Ｓ１５Ｃ）。すなわち、ＧＰＵ１３は、次のバッチのフォーワード処理およびバックワード処理で使用されるニューロン層（Ｌ）の重み（ｗ）を更新する。

【０１０４】
そして、ＧＰＵ１３は、全層の集約処理が完了したか否かを判定する（Ｓ１６）。全層の集約処理が完了していない場合、ＧＰＵ１３は、次のバッチのニューロン層Ｌのフォーワード処理の開始が可能か否かを判定する（Ｓ１７）。次のバッチのニューロン層Ｌのフォーワード処理の開始が可能でない場合、ＧＰＵ１３は、制御をＳ１４Ｃに戻し、次のニューロン層の集約処理の完了を待つ。

【０１０５】
一方、次のバッチのニューロン層Ｌのフォーワード処理の開始が可能である場合、ＧＰＵ１３は、次のバッチのニューロン層Ｌのフォーワード処理を開始させる（Ｓ１８）。Ｓ１７の判定で、フォーワード処理の開始が可能との判定される場合は、複数階層のうち、実行の順序が先の階層で使用される係数に対して積算された変化量を基に次回以降の演算処理で使用される係数が更新された場合の一例である。Ｓ１６からＳ１８の処理を実行することは、実行の順序が後の階層で使用される係数に対する積算された変化量の反映を待たないで、次の演算処理における実行順が先の階層の層別処理を開始することの一例である。

【０１０６】
次のバッチのニューロン層Ｌのフォーワード処理の開始が可能である場合とは、例えば、次のバッチのニューロン層１について、重みの変化量（Δｗ）が集約処理され、重み（ｗ）への反映が完了している場合をいう。また、例えば、次のバッチのニューロン層１からＬ－１のフォーワード方向の処理が終了し、ニューロン層Ｌについて、重みの変化量（Δｗ）が集約処理され、重み（ｗ）への反映が完了している場合をいう。このような場合には、ＧＰＵ１３は、現在処理中のバッチについて、全層の処理が終了していなくても、次のバッチのフォーワード方向の処理を開始させる。そして、ＧＰＵ１３は、処理をＳ１４Ｃに戻す。

【０１０７】
一方、全レイヤの集約処理が完了すると、ＧＰＵ１３は、学習の終わりか否かを判定する（Ｓ１９）。計算ノード１０について用意された未学習のバッチが残っている場合には、ＧＰＵ１３は、処理をＳ１１Ｃに戻し、次のバッチを実行する。ただし、次のバッチにおけるニューロン層については、フォーワード処理は、Ｓ１８による処理開始によって、

すでに開始されているか、実行が完了しているものがあり得る。したがって、次のバッチでのＳ１１Ｃの処理は、前のバッチについての全層についての学習処理が終了していなくても開始され、当該バッチでは、未実行のニューロン層から開始される。

【０１０８】

なお、図１７では、反映処理は、Ｓ１５ＣでＧＰＵ１３が実施したが、実施形態２のようにＣＰＵ１１が反映処理を実行してもよい。また、図１７の処理は、ニューロン層ごとに実行されたが、実施形態３のように、ニューロン層の重みｗのパラメータ列を部分列に分割し、部分列ごとに実行されるようにしてもよい。

【０１０９】

図１８は、実施形態４の起動処理を例示するフローチャートである。この処理は、学習処理後のメモリ転送（ＧＰＵ１３からＣＰＵ１１）、ＣＰＵ１１での集約処理、ノード間通信処理、反映処理、集約処理後のメモリ転送（ＣＰＵ１１からＧＰＵ１３）を起動するときのキューの登録において適用可能である。なお、反映処理自体は、実施形態１のようにＧＰＵ１３によって実行されてもよいし、実施形態２のようにＣＰＵ１１によって集約処理とともに実行されてもよい。図１８の処理の主体はＧＰＵ１３またはＣＰＵ１１である。また、この処理は、図１４で説明した前処理（キュー発行スレッド）の処理である。そこで、以下の説明はキュー発行スレッドを主体として説明する。

【０１１０】

キュー発行スレッドは、キュー発行対象のニューロン層と処理対象データを取得する（Ｓ４１）。例えば、キュー発行スレッドは、キュー発行スレッドの処理が完了したときに、キュー発行対象のニューロン層と処理対象データを取得することになる。

【０１１１】

次に、キュー発行スレッドは、現在登録済みのキューを読む（Ｓ４２）。そして、キュー発行スレッドは、優先順位変更の要否を判定する（Ｓ４３）。例えば、現在登録済みのキューのニューロン層がいずれも、キュー発行対象のニューロン層よりも、入力側に近い層（下位の層）があれば（Ｓ４３でＮ）、キュー発行スレッドは、最後尾の位置にキュー発行対象のニューロン層のキューを登録する（Ｓ４４）。

【０１１２】

一方、例えば、現在登録済みのキューに、キュー発行対象のニューロン層よりも、入力側から遠い層（上位の層）があれば（Ｓ４３でＹ）、キュー発行スレッドは、当該上位の層より優先してキュー発行対象のニューロン層のキューを登録する（Ｓ４５）。Ｓ４３からＳ４５の処理は、演算部が複数階層のうち、演算処理の実行の順序が早い階層の係数の変化量を優先して処理部に転送することの一例である。Ｓ４３からＳ４５の処理は、授受する処理の実行を要求することの一例でもある。Ｓ４３からＳ４５の処理は、処理部が、複数階層のうち演算処理の実行の順序が早い階層の係数を優先して演算部に次回以降の演算処理で使用される係数を更新させることの一例でもある。

そして、キュー発行スレッドは、処理順序の変更をＭＰＩ　ＡＬＬＲｅｄｕｃｅのアルゴリズムで他の計算ノード１０に通知する（Ｓ４６）。

【０１１３】

以上述べたように、実施形態４によれば、入力側に近いニューロン層が優先して処理されるように、処理順序が変更される。１つのニューロン層Ｌの重みのパラメータ列（ｗＬ）が複数の部分列（ｗＬｋ）に分割されて処理される実施形態３場合も同様である。このような処理順序の変更により、処理順序の変更が実施されたバッチの次のバッチにおいて、入力側に近い、階層の低いニューロン層が優先されて前のバッチの学習結果が重みに反映されることになる。すなわち、次のバッチでの入力データに近いニューロン層で使用される重みの更新を早めることができる。

【０１１４】

そして、Ｓ１６からＳ１８のように、全層の集約処理が完了していない場合であっても、次のバッチにおいて下位のニューロン層のフォーワード処理の開始が可能である場合、ＧＰＵ１３は、次のバッチのニューロン層Ｌのフォーワード処理を開始させる。したがっ

て、一部のニューロン層の重みに対して学習結果が反映されていなくても、次のバッチでの入力データに近いニューロン層での学習が早期に開始可能となる。

＜実施形態５＞

【０１１５】

図１９および図２０を参照して、実施形態５を説明する。実施形態１から４では、１つのバッチにおいて、学習、集約、ノード間通信、反映処理が完了した後に、次のバッチが開始された。実施形態５においては、現在のバッチ（Ｎ番目のバッチ）の学習処理が完了すると、集約、ノード間通信、反映処理が実行される前に、次のバッチ（Ｎ＋１番目のバッチ）の学習処理が起動される。そして、現在のバッチ（Ｎ番目のバッチ）の学習処理の結果は、次のさらに次のバッチ（Ｎ＋２番目のバッチ）の前に重みに反映される。実施形態５におけるこのような手順以外の手順および構成要素は、実施形態１から４と同様である。そこで、実施形態５の構成要素のうち、実施形態１から４と同一の構成要素については、同一の符号を付してその説明を省略する。

【０１１６】

図１９に、実施形態５の処理のタイムチャートを実施形態４と対比して例示する。図１９では、上側は実施形態４のタイムチャートであり、下側が実施形態５のタイムチャートである。実施形態５では、ニューロン層１から４までが想定されている。また、フォーワード方向のニューロン層１から４の学習処理はＦ１からＦ４のラベルで示されている。一方、バックワード方向のニューロン層４から１の学習処理はＢ４からＢ１のラベルで示されている。

【０１１７】

図１９のように、実施形態５では、Ｎ番目の学習処理（バッチ処理（Ｎ番目））が終了すると、Ｎ－１番目のバッチの学習処理の結果（集約済みの重みの変化量Δｗ）が重みｗに反映される。そして、Ｎ＋１番目のバッチに対する学習処理（バッチ処理（Ｎ＋１番目））が開始する。図１９のように、バッチ処理（Ｎ番目）に続いてバッチ処理（Ｎ＋１番目）の学習処理が実行されることは、演算処理と積算された変化量を基に次回以降の演算処理で使用される係数を更新する処理とが複数回繰り返して実行される場合の一例である。

【０１１８】

なお、実施形態２で説明したように、Ｎ＋１番目の学習処理が開始するまでに、Ｎ－１番目のバッチよる学習処理の結果が重みｗに反映されるようにすればさらに時間は短縮できる。また、Ｎ＋１番目の各ニューロン層の学習処理の開始までに、Ｎ－１番目のバッチによる各層（ｋ）の学習処理の結果（集約済みのΔｗ（Ｌｋ））が各層の重みに反映されるようにすればさらに時間は短縮できる。なお、実施形態６とは異なり、実施形態５では、重み（ｗ）を格納するバッファが１面だけ使用されるので、ＧＰＵ１３はバッチ処理（Ｎ番目）の学習処理後、直ちにバッチ処理（Ｎ＋１番目）を開始できない。すなわち、ＧＰＵ１３は、バッチ処理（Ｎ＋１番目）を開始する前に、学習処理の結果（集約済みのΔｗ（Ｌｋ））が各層の重みに反映する時間を要する。また、実施形態２のように、ＣＰＵ１１が学習処理の結果が各層の重みに反映する場合には、ＧＰＵ１３は、バッチ処理（Ｎ＋１番目）を開始する前に、学習処理の結果が反映された重みをメモリ１４に保持する時間を要する。

【０１１９】

以上の処理の結果、実施形態５では、実施形態４と比較して、学習処理の結果の反映がバッチ１つ分遅れることになる。しかし、学習処理終了時に学習処理の結果を重みに反映しないため、実施形態４と比較して早期に次のバッチを開始できる。すなわち、実施形態４に対して、概ね少なくとも学習処理の結果を集約する時間が節約される。

【０１２０】

なお、図１９の処理は、例えば、図７において、Ｓ１４、Ｓ１５の処理を実行しないで、Ｓ１６において、未処理のバッチの有無を判定し、次のバッチの学習処理を実行することで実行される。図１９でＧＰＵ１３がＮ＋１番目の学習処理が終了すると、Ｎ＋２番目

のバッチに対する学習処理を開始することは、演算部が現在の演算処理による変化量を基に次回以降の演算処理で使用される係数が更新される前に次の演算処理を開始することの一例である。

【０１２１】

図２０は、実施形態５におけるＣＰＵ１１の学習処理結果の集約処理のフローチャートを例示する。図２０の集約処理は、例えば、Ｎ番目のバッチでの学習処理が終了後に、Ｎ＋１番目の学習処理と並列に実行される。この処理では、まず、ＣＰＵ１１は、バッチが２番目より後のバッチか否かを判定する（Ｓ５１）。バッチが１番目または２番目のバッチの場合、ＣＰＵ１１は、処理を終了する。

【０１２２】

一方、バッチが２番目より後のバッチの場合、ＣＰＵ１１は、メモリ転送を実行し、Ｎ番目のバッチでの学習処理の結果を取得する（Ｓ５２）。そして、メモリ転送したバッチの学習結果である（Δｗ）を集約する（Ｓ５３）。そして、ＣＰＵ１１は、集約された（Δｗ）のＧＰＵ１３へのメモリ転送を起動する（Ｓ５４）。Ｓ５４のメモリ転送を受け、Ｎ＋２番目のバッチに対する学習処理の開始前に、ＧＰＵ１３は、集約された（Δｗ）を重み（ｗ）に反映する。Ｓ５２からＳ５４の処理は、現在の演算処理による変化量を基に次々回の演算処理で使用される係数が更新されることの一例である。

【０１２３】

なお、変化量（Δｗ）の集約と重み（ｗ）への反映は、実施形態２のように、ＣＰＵ１１において行ってもよい。つまり、ＧＰＵ１３は、集約された重み（Δｗ）が反映済みの重み（ｗ）をメモリ転送で受け取るようにしてもよい。この場合には、反映処理は、単に、変化量（Δｗ）が反映済みの重み（ｗ）をＧＰＵ１３のメモリ１４に保存する処理ということができる。

【０１２４】

また、メモリ転送（ＧＰＵ１３からＣＰＵ１１）、変化量（Δｗ）の集約処理、ノード間通信、重み（ｗ）への反映処理、および、メモリ転送（ＣＰＵ１１からＧＰＵ１３）は、実施形態２のように、ニューロン層単位で行ってもよい。また、これらの処理は、実施形態３のように、ニューロン層単位よりも細かく分割したパラメータの部分列の単位で行ってもよい。

【０１２５】

以上述べたように、実施形態５では、Ｎ番目バッチの学習処理が終了すると、Ｎ＋１番目のバッチに対する学習処理と並行してＮ番目バッチの学習処理結果の集約処理が実行される。したがって、図１９のように、実施形態１から４の場合と比較して、集約処理の時間が短縮される。

【０１２６】

また、上記集約処理ともに、実施形態２と同様に、ＣＰＵ１１が反映処理を行った場合には、ＧＰＵ１３は、Ｎ＋１番目のバッチの学習処理が開始するまでに、集約されたΔｗを反映済みの重みをメモリ１４に保存する処理を実行すればよい。この場合には、実施形態１から４の場合と比較して、集約処理および反映処理の時間が短縮される。

＜実施形態６＞

【０１２７】

図２１および図２２を参照して実施形態６を説明する。実施形態５では、計算ノード１０は、Ｎ番目の学習処理の結果をＮ＋２番目のバッチの学習開始までに集約し、重み（ｗ）に反映した。このような処理によって、計算ノード１０は、Ｎ番目の学習処理の終了後、直ちにＮ＋１番目の学習処理を開始できた。実施形態６では、計算ノード１０には、重み（ｗ）を格納するバッファが複数、例えば、２面設けられる。すなわち、計算ノード１０は、学習結果である重みの変化量（Δｗ）が反映された重み（ｗ）を格納するバッファを２面有することで、実施形態５と同様に、第Ｎ番目のバッチが終了した後、直ちに、第Ｎ＋１番目のバッチの学習処理を開始できる。

【０１２８】

図２１に、実施形態６のタイムチャートを実施形態４と対比して例示する。図２１のように、実施形態４では、バッファｗａに格納した重みを用いた学習処理とバッファｗｂに格納した重みを用いた学習処理が交互に実行される。例えば、奇数番目のバッチの学習終了後に、次の偶数番目のバッチの学習処理と並行して集約処理と反映処理が実行される。そして、奇数番目のバッチの学習処理の結果である重みの変化量（Δｗ）が反映された重み（ｗ）がバッファｗａに格納される。このとき、偶数番目のバッチの学習処理では、バッファｗｂに格納された重みが使用される。

【０１２９】

一方、偶数番目のバッチの学習終了後に、次の奇数番目のバッチの学習処理と並行して集約処理と反映処理が実行される。そして、偶数番目のバッチの学習処理の結果である重みの変化量（Δｗ）が反映された重み（ｗ）がバッファｗｂに格納される。このとき、奇数番目のバッチ学習処理では、バッファｗａに格納された重みが使用される。

【０１３０】

したがって、図２１のように、バッファｗａに格納した重みによる第Ｎ番目のバッチの学習処理終了後、直ちに、バッファｗｂに格納した重みによる第Ｎ＋１番目のバッチの学習処理が開始される。したがって、実施形態４の場合と比較して、実施形態６では、学習処理終了後の学習処理の結果である重みの変化量（Δｗ）の集約処理と反映処理が次ぎのバッチの学習処理と並行して実施できる。実施形態６の場合も、実施形態５と同様、第Ｎ番目のバッチの学習処理の結果を反映した重みは第Ｎ＋２番目のバッチの学習に使用される。図２１のバッファｗａ、ｗｂは、係数を格納するための２組以上の記憶部の一例である。

【０１３１】

図２２に、実施形態６における集約処理および反映処理のフローチャートを例示する。図２２では、学習処理、集約反映処理、および格納処理の３つの処理が連携して実行される。ＧＰＵ１３が学習処理と格納処理を実行し、ＣＰＵ１１が集約反映処理を実行する。ここでは、Ｎ番目のバッチの学習処理が実行されるものとして説明する。

【０１３２】

まず、ＧＰＵ１３は、Ｎ番目のバッチが奇数番目のバッチか否かを判定する（Ｓ６０）。Ｎ番目のバッチが奇数番目のバッチの場合、ＧＰＵ１３は、バッファｗａに格納した重みによる学習処理を実行する（Ｓ６１）。一方、Ｎ番目のバッチが偶数番目のバッチの場合、ＧＰＵ１３は、バッファｗｂに格納した重みによる学習処理を実行する（Ｓ６２）。Ｓ６１、Ｓ６２の処理は、第１の記憶部に格納した係数を用いて演算処理を実行することの一例である。そして、ＧＰＵ１３は、メモリ転送をＣＰＵ１１に要求するとともに、集約反映処理のキューを登録する。そして、ＧＰＵ１３は当該バッチの学習処理を終了する。そして、ＧＰＵ１３はＮ＋１番目のバッチの学習処理を実行する。

【０１３３】

ＣＰＵ１１は、Ｎ番目のバッチの学習結果である重みの変化量（Δｗ）に対する集約処理と反映処理（以下、単に集約反映処理）のキューを受け付け、集約反映処理を実行する。ＣＰＵ１１による集約反映処理は、ＧＰＵ１３によるＮ＋１番目のバッチの学習処理と並行して実行される。

【０１３４】

まず、ＣＰＵ１１は、ＧＰＵ１３による学習結果である重みの変化量（Δｗ）をメモリ転送で取得する（Ｓ６３）。そして、ＣＰＵ１１は、重みの変化量（Δｗ）を集約し、重み（ｗ）に反映する（Ｓ６５）。Ｓ６５の処理は、実施形態２（図１２）のＳ２２からＳ２６と同様である。そして、ＣＰＵ１１は、集約した重みの変化量（Δｗ）が反映された重み（ｗ）をＧＰＵ１３にメモリ転送するＳ６６）。

【０１３５】

ＧＰＵ１３は、メモリ転送を受けると、バッチが奇数番目のバッチか否かを判定する（Ｓ６７）。バッチが奇数番目のバッチの場合、ＧＰＵ１３は、バッファｗｂに重みを格納する（Ｓ６８）。一方、バッチが偶数番目のバッチの場合、ＧＰＵ１３は、バッファｗｂ

に重みを格納する（Ｓ６９）。Ｓ６８、Ｓ６９の処理は、演算処理による変化量を基に更新した係数を第２の記憶部に格納することの一例である。なお、Ｓ６７からＳ６９の処理は、次のさらに次のバッチ（Ｎ＋２番目のバッチ）の学習処理が開始されるまでに実行される。

【０１３６】

以上述べたように、実施形態６においては、図２１のように、バッファｗａに格納した重みによる第Ｎ番目のバッチの学習処理終了後、直ちに、バッファｗｂに格納した重みによる第Ｎ＋１番目のバッチの学習処理が開始できる。

＜記録媒体＞

【０１３７】

コンピュータその他の機械、装置（以下、コンピュータ等）に上記いずれかの機能を実現させるプログラムをコンピュータ等が読み取り可能な記録媒体に記録することができる。そして、コンピュータ等に、この記録媒体のプログラムを読み込ませて実行させることにより、その機能を提供させることができる。

【０１３８】

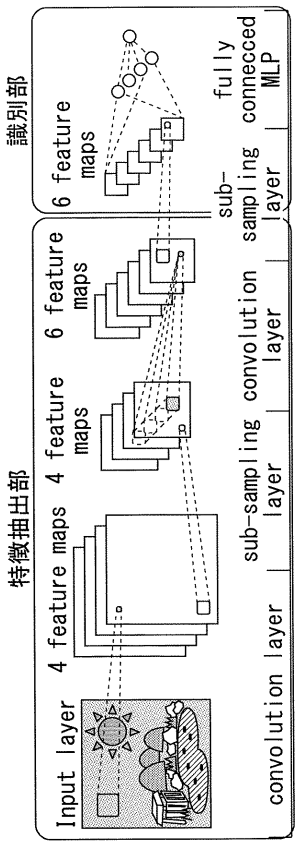
ここで、コンピュータ等が読み取り可能な記録媒体とは、データやプログラム等の情報を電気的、磁気的、光学的、機械的、または化学的作用によって蓄積し、コンピュータ等から読み取ることができる記録媒体をいう。このような記録媒体のうちコンピュータ等から取り外し可能なものとしては、例えばフレキシブルディスク、光磁気ディスク、Compact Disc（ＣＤ）－Read Only Memory（ＲＯＭ）、ＣＤ－Recordable（Ｒ）、Digital Versatile Disk（ＤＶＤ）、ブルーレイディスク、Digital Audio Tape（ＤＡＴ）、８ｍｍテープ、フラッシュメモリなどのメモリカード等がある。また、コンピュータ等に固定された記録媒体としてハードディスク、ＲＯＭ（リードオンリーメモリ）等がある。さらに、Solid State Drive（ＳＳＤ）は、コンピュータ等から取り外し可能な記録媒体としても、コンピュータ等に固定された記録媒体としても利用可能である。

【符号の説明】

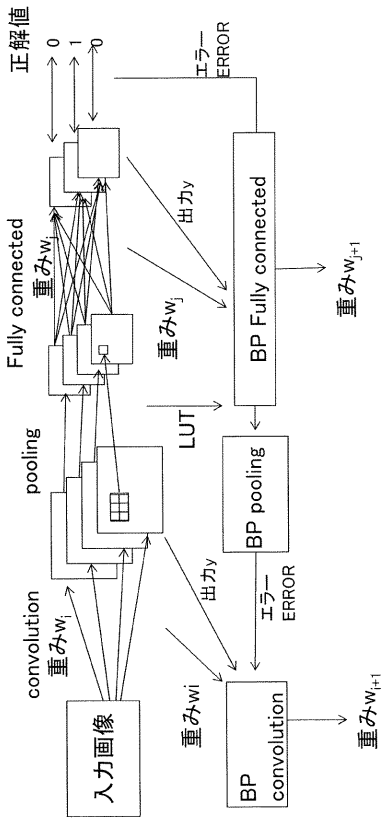
【０１３９】

１　並列情報処理装置
１０　計算ノード
１１　ＣＰＵ
１２、１４　メモリ
１３　ＧＰＵ
１４　バス
１５　ノード間インターフェース

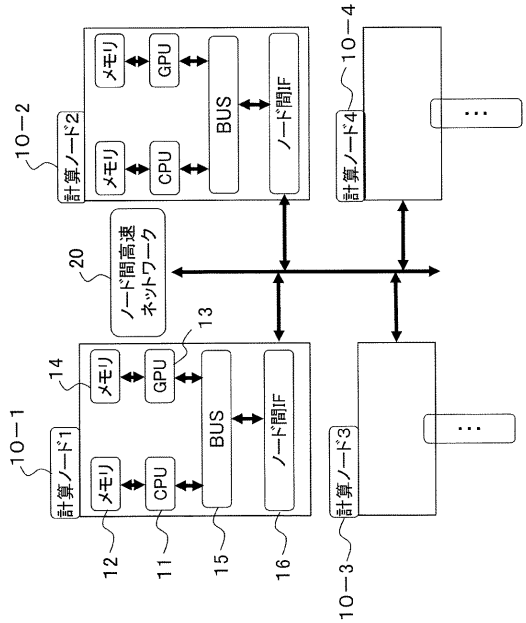
【図１】


識別部
特徴抽出部
Input layer
4 feature maps
convolution layer
4 feature maps
sub-sampling layer
6 feature maps
convolution layer
6 feature maps
sub-sampling layer
fully connecced MLP

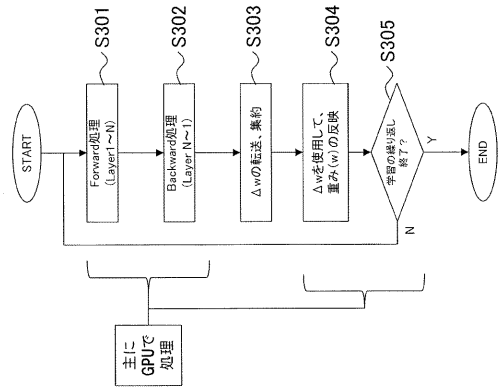
【図２】


正解値
エラー ERROR
Fully connected
重みwj
出力y
pooling
LUT
BP Fully connected
重みwj+1
convolution
重みwi
入力画像
出力y
BP pooling
エラー ERROR
重みwi
BP convolution
重みwi+1

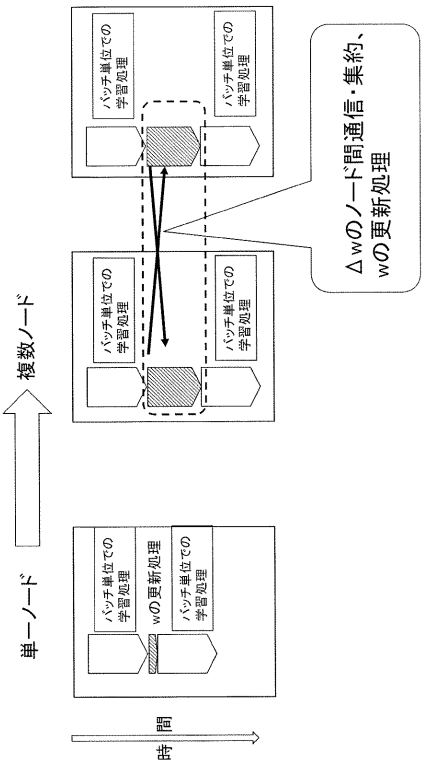
【図３】


10−1
計算ノード1
メモリ
CPU
GPU
メモリ
BUS
ノード間IF
12
11
14
13
15
16
10−2
計算ノード2
10−3
計算ノード3
10−4
計算ノード4
20
ノード間高速ネットワーク

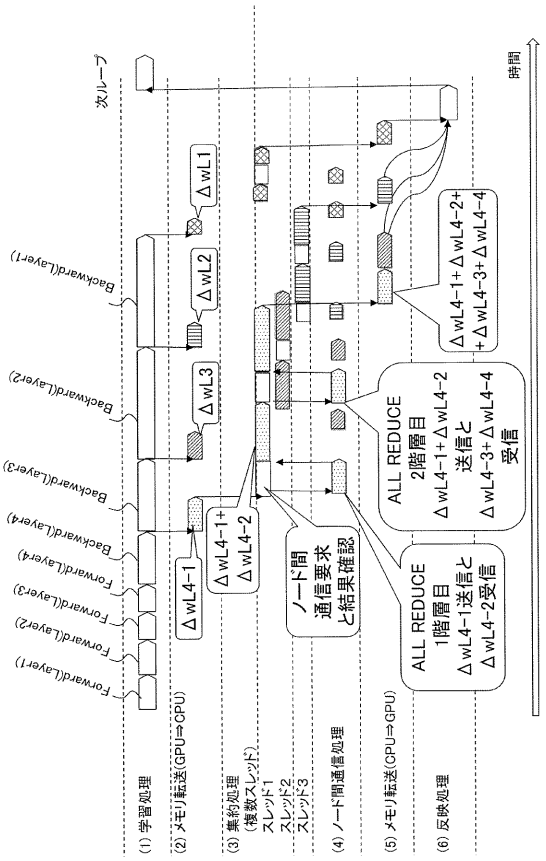
【図４】


START
S301
Forward処理 (Layer1〜N)
S302
Backward処理 (Layer N〜1)
S303
Δwの転送、集約
S304
Δwを使用して、重み(w)の反映
S305
学習の繰り返し終了?
Y
N
END
主にGPUで処理

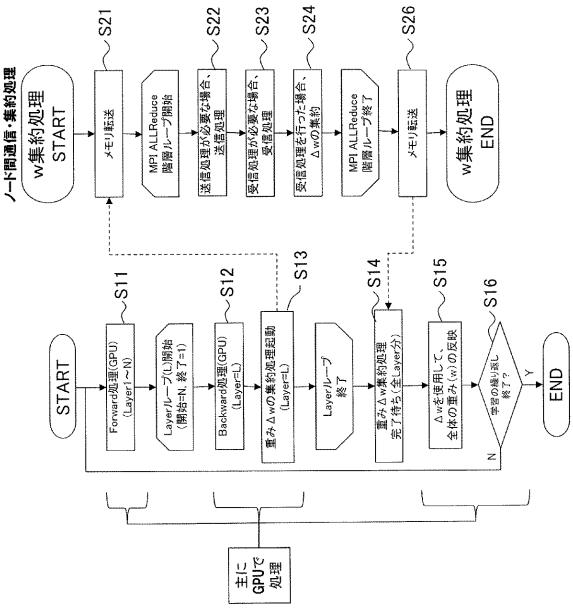
【図５】


単一ノード
複数ノード
時間
バッチ単位での学習処理
wの更新処理
バッチ単位での学習処理
バッチ単位での学習処理
バッチ単位での学習処理
バッチ単位での学習処理
Δwのノード間通信・集約、wの更新処理

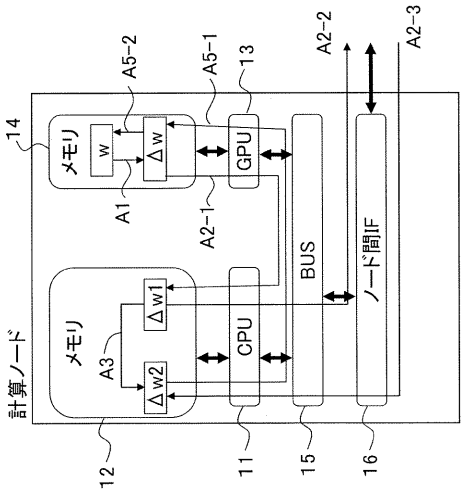
【図６】


次ループ
時間
Backward(Layer1)
Backward(Layer2)
Backward(Layer3)
Backward(Layer4)
Forward(Layer4)
Forward(Layer3)
Forward(Layer2)
Forward(Layer1)
ΔwL1
ΔwL2
ΔwL3
ΔwL4-1
ΔwL4-1+ΔwL4-2
ノード間通信要求と結果確認
ALL REDUCE 1階層目 ΔwL4-1送信とΔwL4-2受信
ALL REDUCE 2階層目 ΔwL4-1+ΔwL4-2 送信と ΔwL4-3+ΔwL4-4 受信
ΔwL4-1+ΔwL4-2+ΔwL4-3+ΔwL4-4
(1) 学習処理
(2) メモリ転送(CPU⇒GPU)
(3) 集約処理 (複数スレッド)
スレッド1
スレッド2
スレッド3
(4) ノード間通信処理
(5) メモリ転送(CPU⇒GPU)
(6) 反映処理

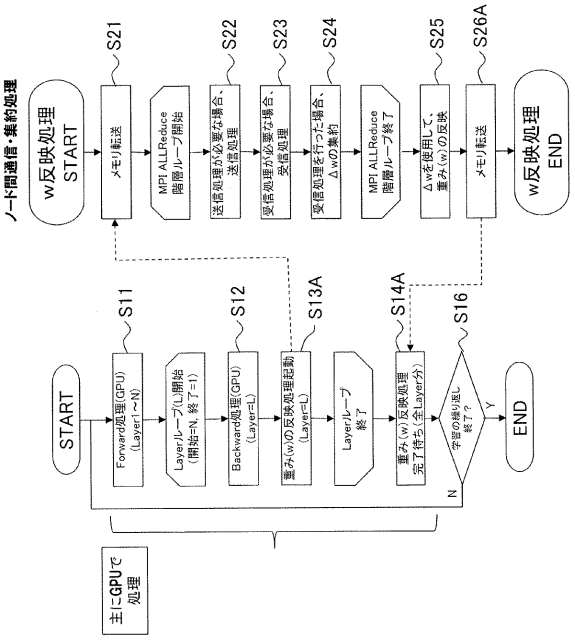
【図７】


ノード間通信・集約処理
w集約処理 START
S21
メモリ転送
MPI ALLReduce 階層ループ開始
S22
送信処理が必要な場合、送信処理
S23
受信処理が必要な場合、受信処理
S24
受信処理を行った場合、Δwの集約
MPI ALLReduce 階層ループ終了
S26
メモリ転送
w集約処理 END
START
S11
Forward処理(GPU) (Layer1～N)
Layerループ(L開始 (開始=N, 終了=1)
S12
Backward処理(GPU) (Layer=L)
S13
重みΔwの集約処理起動 (Layer=L)
Layerループ 終了
S14
重みΔw集約処理 完了待ち(全layer分)
S15
Δwを使用して、全体の重み(w)の反映
S16
学習の繰り返し 終了？
N
Y
END
主にGPUで処理

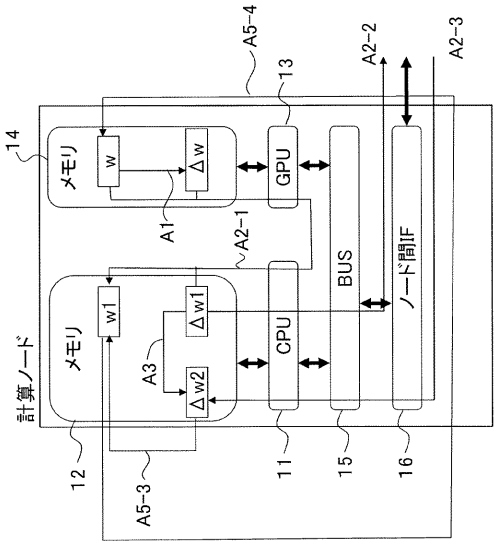
【図８】


計算ノード
メモリ
w
Δw
A5-2
A5-1
13
GPU
A2-2
A2-3
14
A1
A2-1
メモリ
A3
Δw1
Δw2
CPU
BUS
ノード間IF
12
11
15
16

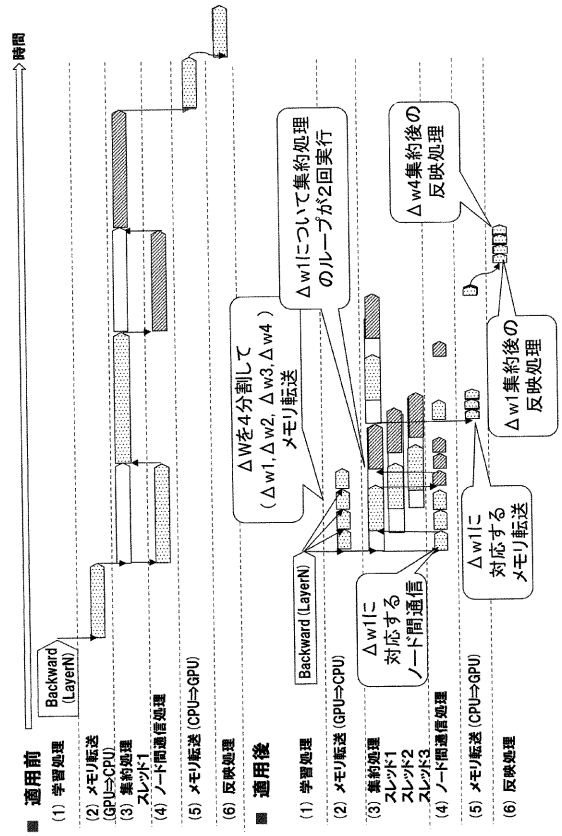
【図９】



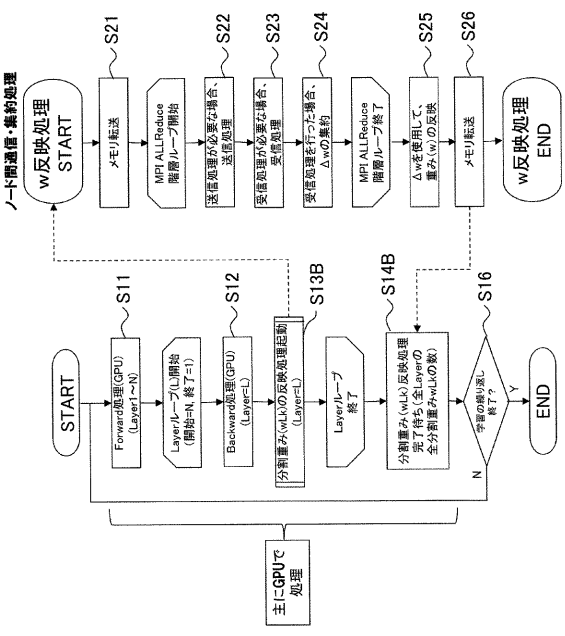
【図１０】



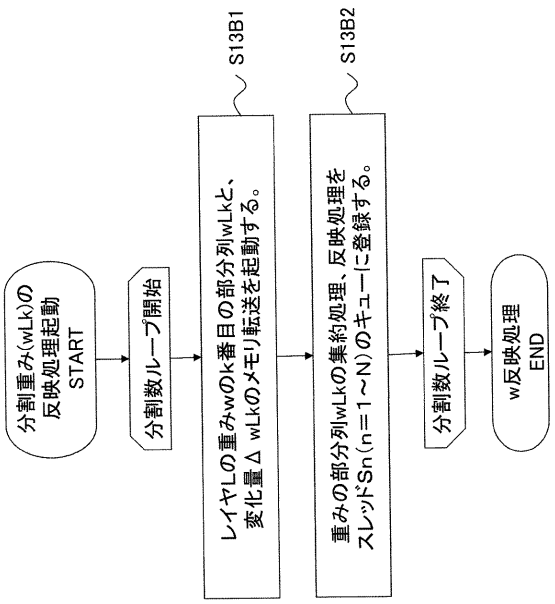
【図１１】



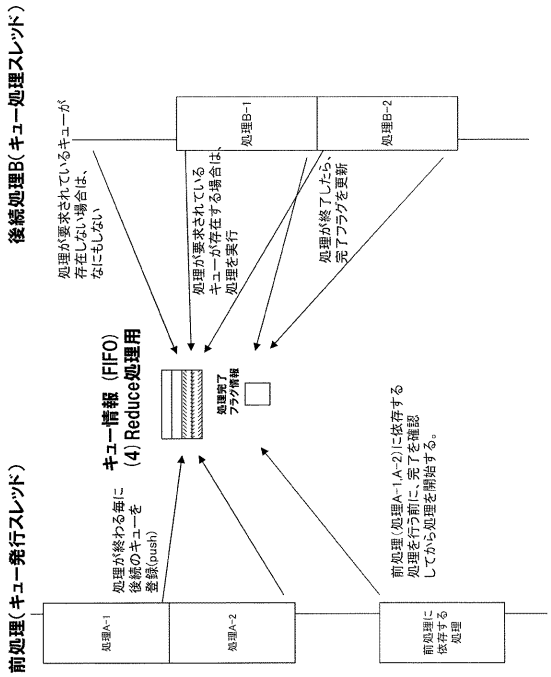
【図１２】



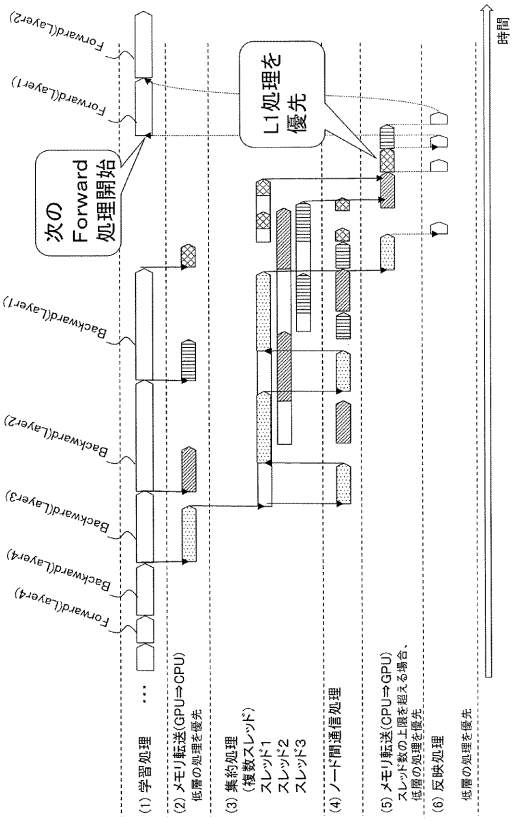
【図１３】



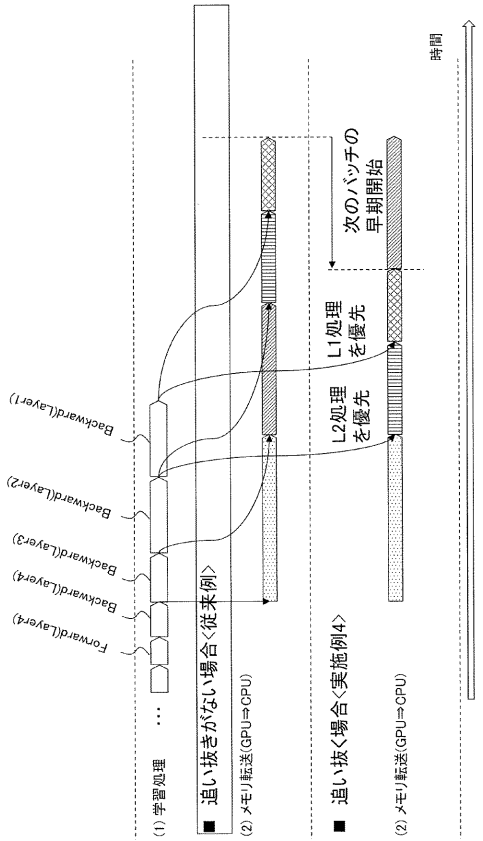
【図１４】



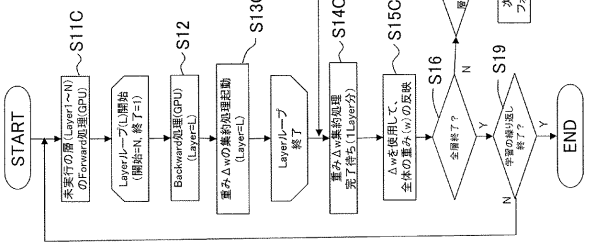
【図１５】



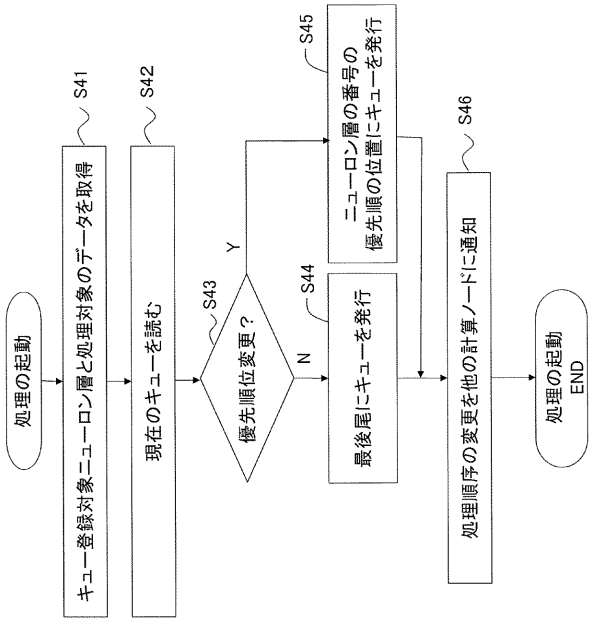
【図１６】



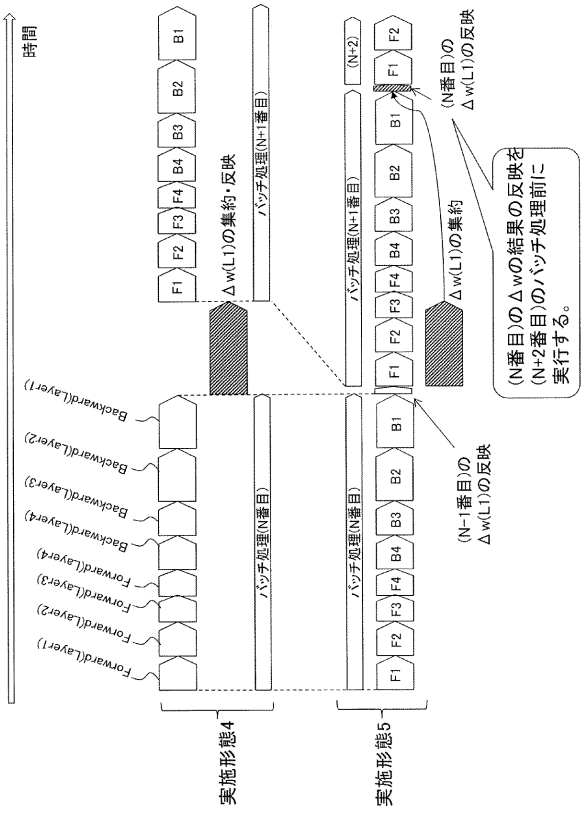
【図１７】



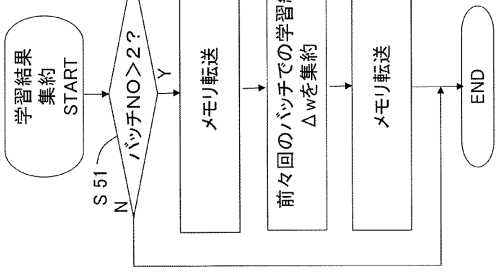
【図１８】



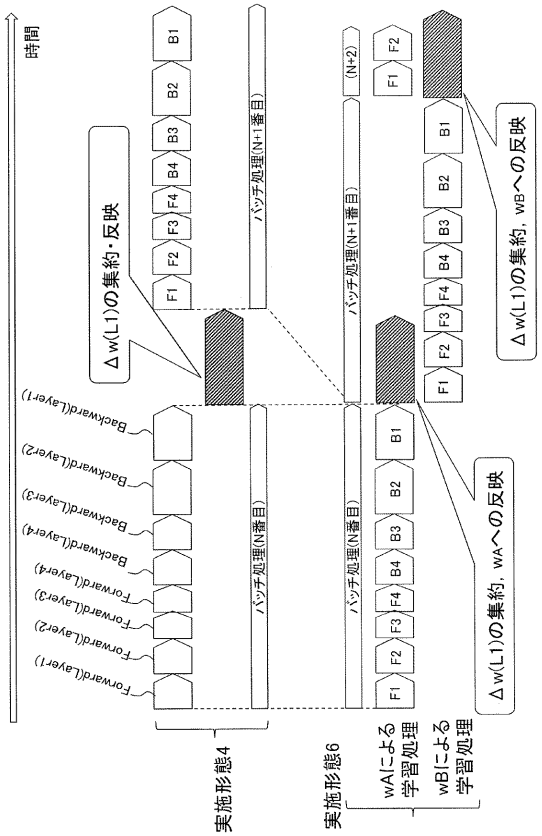
【図１９】



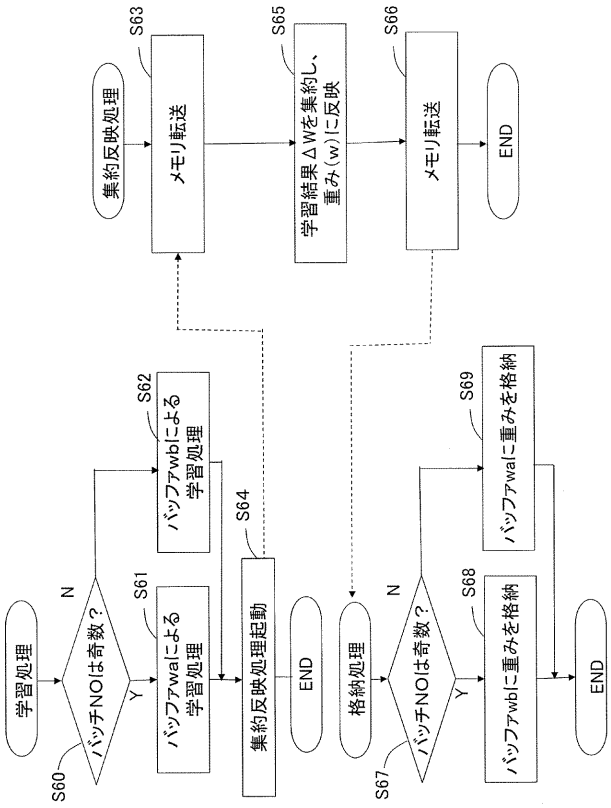
【図２０】



【図２１】



【図２２】



【手続補正書】
【提出日】平成29年6月22日(2017.6.22)
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】００１５
【補正方法】変更
【補正の内容】
【００１５】
フォーワード方向の処理は、入力画像に対して、畳み込み層の処理と、サブサンプリング層の処理を繰り返し実行する特徴抽出部の処理と、識別結果を出力する識別部の処理を含む。特徴抽出部は、入力画像に対して、畳み込み層の処理と、サブサンプリング層の処理を繰り返し実行することで、間引かれた画像を抽出する。畳み込み層の処理は、畳み込み演算ともいう。畳み込み演算は、例えば、Ｎ個×Ｎ個の画素を有する画像の情報（第Ｎ－１層）に対して、例えば、ｍ×ｍ個の重みw_{ab}（ａ，ｂ＝０，．．．，ｍ－１）のフィルタによる畳み込み演算を実行することで、次の層（第Ｎ層）の画像の情報を作る。サブサンプリング層の処理は、画像間引き処理であり、プーリング演算ともいう。
【手続補正２】
【補正対象書類名】明細書
【補正対象項目名】００１９
【補正方法】変更
【補正の内容】
【００１９】
勾配降下法によるニューラルネットワークの学習処理においては、エラーの評価関数（ＥＲＲＯＲ）の勾配と、学習係数イータの積が重みｗの変化量（例えば、現在の重みｗｔと次の重みｗｔ＋１の差分値）となる。すなわち、深層学習においては、フォーワード方

向に各ニューロン層の処理が実行され、バックワード方向に、各ニューロン層でのエラーの評価関数（ＥＲＲＯＲ）が伝搬される。そして、各ニューロン層は、バックワード方向に伝搬するエラーの評価関数（ＥＲＲＯＲ）から、エラーの評価関数（ＥＲＲＯＲ）の勾配を求める。そして、各ニューロン層は、エラーの評価関数（ＥＲＲＯＲ）が小さくする方向でのエラーの評価関数（ＥＲＲＯＲ）の勾配と、学習係数イータの積から重みｗｔの変化量（勾配情報ともいう）を算出し、次回の重みｗｔ＋１を求める。ここで、現在の重みをｗｔで表し、次回の演算で使用される重みをｗ＋１で表した。また、図１で説明したように、学習処理において、重みｗは１以上の成分を有する係数列（ベクトル）である。

【手続補正３】
【補正対象書類名】明細書
【補正対象項目名】００４８
【補正方法】変更
【補正の内容】
【００４８】
図７は、実施形態１の計算ノード１０の処理を例示するフローチャートである。図で左側のフローチャートは、主としてＧＰＵ１３が実行する学習処理と反映処理を例示する。また、右側のフローチャートは、主としてＣＰＵ１１が実行するノード間通信・集約処理を例示する。図７の処理では、まず、ＧＰＵ１３がニューロン層（例えば、ニューロン層１からＮ）について、フォーワード処理を実行する（Ｓ１１）。

【手続補正４】
【補正対象書類名】明細書
【補正対象項目名】００４９
【補正方法】変更
【補正の内容】
【００４９】
フォーワード処理は、図１に例示したように、入力データと重み（ｗ）とによる演算処理である。演算処理は、例えば、入力データの要素ｘ（ｉ，ｊ）とｍ×ｍ個の重みw_{ab}（ａ，ｂ＝０，．．．，ｍ－１）のフィルタによる畳み込み演算、サブサンプリング層のプーリング演算、全結合層の演算等である。Ｓ１１の処理は、処理対象のデータに対する係数による演算処理の一例である。

【手続補正５】
【補正対象書類名】明細書
【補正対象項目名】００６２
【補正方法】変更
【補正の内容】
【００６２】
そして、ＣＰＵ１１は、メモリ転送でＧＰＵ１３に集約されたニューロン層Ｌの重み（ｗ）の変化量（Δｗ２）を転送する（矢印Ａ５－１）。転送先のＧＰＵ１３は、転送された重みの変化量を変化量（Δｗ）に保存する。そして、ＧＰＵ１３は、集約された層Ｌの重み（ｗ）の変化量（Δｗ）を使用して、重み（ｗ）を更新する（Ａ５－２）。

【手続補正６】
【補正対象書類名】明細書
【補正対象項目名】００７６
【補正方法】変更
【補正の内容】
【００７６】
例えば、あるニューロン層Ｌの重み（ｗ）がｗ＝（ｐ１、ｐ２、・・・・、ｐＸ）のようなパラメータ列であるとする。パラメータ列は、係数列の一例である。つまり、ニューロン層Ｌの重み（ｗ）は、複数使用され、係数列を形成する。そして、学習処理の結果、重みの変化量はΔｗ＝（Δｐ１、Δｐ２、・・・・、ΔｐＸ）のような多数のパラメータ

列として計算されるとする。このような場合に、ＧＰＵ１３はΔｗを部分列に区切り、Δｗ１＝（Δｐ１、Δｐ２、・・・、ΔｐＸ１）、Δｗ２＝（ΔｐＸ１＋１、・・・、ΔｐＸ２）、Δｗ３＝（ΔｐＸ２＋１、・・・、ΔｐＸ３）、・・・、Δｗｘ＝（ΔｐＸ３＋１、・・・、ΔｐＸ）のように分割する。

【手続補正７】
【補正対象書類名】明細書
【補正対象項目名】００８２
【補正方法】変更
【補正の内容】
【００８２】
また、スレッド１は、分割された変化量（Δｗ１）についてメモリ転送スレッドのキューを発行すると、分割された次の変化量（Δｗ４）について、分割された変化量（Δｗ１）と同様の処理を実行する。このようにして、例えば、ＣＰＵ１１が複数、例えば、５つのコアを有している場合には、ＣＰＵ１１はスレッド１から３およびメモリ転送スレッド、およびノード間通信スレッドを並行して実行できる。したがって、例えば、ある分割された変化量（Δｗｋ）についてのノード間通信の処理が、別の分割された変化量（Δｗｊ）についての集約処理の時間に実行できる。また、仮に、あるニューロン層Ｌの重み（ｗＬ）のパラメータ数が他の層よりも多いものであっても、ＧＰＵ１３およびＣＰＵ１１は重み（ｗＬ）に含まれるパラメータを複数部分に分割し、複数スレッドで並行して処理できる。

【手続補正８】
【補正対象書類名】明細書
【補正対象項目名】００９３
【補正方法】変更
【補正の内容】
【００９３】
ところで、図１５の例では、ニューロン層１の集約された重みの変化量がＣＰＵ１１からＧＰＵ１３にメモリ転送可能となったときに、まだ、ニューロン層２についても、集約された変化量のメモリ転送が起動されていない。例えば、ニューロン層２のメモリ転送処理（ＣＰＵ１１からＧＰＵ１３）は、キューが登録された状態で未実行の状態となっている。実施形態４では、このような場合にニューロン層１の集約処理が終了すると、集約処理用のスレッドは、ニューロン層２よりもニューロン層１のメモリ転送を優先する。すなわち、ＣＰＵ１１の集約処理用のスレッドは、ニューロン層２よりもニューロン層１が先に転送されるように、ニューロン層１の集約された変化量のメモリ転送のキューを登録する。そのようなキュー登録の結果、メモリ転送用スレッドはニューロン層２よりもニューロン層１の重みの変化量を先にメモリ転送する。

【手続補正９】
【補正対象書類名】明細書
【補正対象項目名】００９９
【補正方法】変更
【補正の内容】
【００９９】
図１７は、実施形態４の学習処理を例示するフローチャートである。この処理では、ＧＰＵ１３は、ニューロン層１からＮについて、フォーワード方向の処理を実行する（Ｓ１１Ｃ）。ただし、Ｓ１１Ｃの処理は、前のバッチにおける全層についての学習処理が終了していなくても開始される点で実施形態１から３と相違する。そして、全層についてのフォーワード方向の処理が終了すると、ＧＰＵ１３は、バックワード方向にニューロン層Ｎから１のループ（ＬＡＹＥＲ　ループ（Ｌ）開始＝Ｎ、終了＝１）の中で、Ｓ１２、Ｓ１３Ｃの処理を実行する。Ｓ１２の処理は、実施形態１から３と同様である。

【手続補正１０】

【補正対象書類名】明細書
【補正対象項目名】０１００
【補正方法】変更
【補正の内容】
【０１００】
Ｓ１３Ｃの処理では、ＧＰＵ１３は、ニューロン層のうち、入力側に近いニューロン層を優先して、ＣＰＵ１１にメモリ転送するとともに、集約処理を実行するＣＰＵ１１のスレッドにキューを登録する（Ｓ１３Ｃ）。Ｓ１３Ｃの処理は、複数階層のうち、演算処理の実行の順序が早い階層の係数の変化量を優先して処理部に転送することの一例である。
【手続補正１１】
【補正対象書類名】明細書
【補正対象項目名】０１０１
【補正方法】変更
【補正の内容】
【０１０１】
したがって、実施形態４では、ＧＰＵ１３は、各ニューロン層（Ｌ）でバックワード方向の処理が終了するごとに、優先順の制御を実行する。すなわち、ＧＰＵ１３は、バックワード方向の処理が終了したニューロン層（Ｌ）より上位のニューロン層（Ｌ＋ｋ）で、メモリ転送および集約処理が未実行のニューロン層がキューに残っていないか否かを判定する。そして、バックワード方向の処理が終了したニューロン層（Ｌ）より上位のニューロン層（Ｌ＋ｋ）がキューに残っている場合には、ＧＰＵ１３は、入力側に近い下位のニューロン層（Ｌ）を優先してキューを登録する。なお、このように、下位のニューロン層を優先するキューの登録は、ＣＰＵ１１がノード間通信およびメモリ転送（ＣＰＵ１１からＧＰＵ１３）のキューを登録する場合も同様である。
【手続補正１２】
【補正対象書類名】明細書
【補正対象項目名】０１２３
【補正方法】変更
【補正の内容】
【０１２３】
なお、変化量（Δｗ）の集約と重み（ｗ）への反映は、実施形態２のように、ＣＰＵ１１において行ってもよい。つまり、ＧＰＵ１３は、集約された変化量（Δｗ）が反映済みの重み（ｗ）をメモリ転送で受け取るようにしてもよい。この場合には、反映処理は、単に、変化量（Δｗ）が反映済みの重み（ｗ）をＧＰＵ１３のメモリ１４に保存する処理ということができる。
【手続補正１３】
【補正対象書類名】明細書
【補正対象項目名】０１２８
【補正方法】変更
【補正の内容】
【０１２８】
図２１に、実施形態６のタイムチャートを実施形態４と対比して例示する。図２１のように、実施形態６では、バッファｗａに格納した重みを用いた学習処理とバッファｗｂに格納した重みを用いた学習処理が交互に実行される。例えば、奇数番目のバッチの学習終了後に、次の偶数番目のバッチの学習処理と並行して集約処理と反映処理が実行される。そして、奇数番目のバッチの学習処理の結果である重みの変化量（Δｗ）が反映された重み（ｗ）がバッファｗａに格納される。このとき、偶数番目のバッチの学習処理では、バッファｗｂに格納された重みが使用される。
【手続補正１４】
【補正対象書類名】明細書

【補正対象項目名】０１３２
【補正方法】変更
【補正の内容】
【０１３２】
まず、ＧＰＵ１３は、Ｎ番目のバッチが奇数番目のバッチか否かを判定する（Ｓ６０）。Ｎ番目のバッチが奇数番目のバッチの場合、ＧＰＵ１３は、バッファｗａに格納した重みによる学習処理を実行する（Ｓ６１）。一方、Ｎ番目のバッチが偶数番目のバッチの場合、ＧＰＵ１３は、バッファｗｂに格納した重みによる学習処理を実行する（Ｓ６２）。Ｓ６１、Ｓ６２の処理は、第１の記憶部に格納した係数を用いて演算処理を実行することの一例である。そして、ＧＰＵ１３は、メモリ転送をＣＰＵ１１に要求するとともに、集約反映処理のキューを登録する（Ｓ６４）。そして、ＧＰＵ１３は当該バッチの学習処理を終了する。そして、ＧＰＵ１３はＮ＋１番目のバッチの学習処理を実行する。
【手続補正１５】
【補正対象書類名】明細書
【補正対象項目名】０１３５
【補正方法】変更
【補正の内容】
【０１３５】
ＧＰＵ１３は、メモリ転送を受けると、バッチが奇数番目のバッチか否かを判定する（Ｓ６７）。バッチが奇数番目のバッチの場合、ＧＰＵ１３は、バッファｗｂに重みを格納する（Ｓ６８）。一方、バッチが偶数番目のバッチの場合、ＧＰＵ１３は、バッファｗａに重みを格納する（Ｓ６９）。Ｓ６８、Ｓ６９の処理は、演算処理による変化量を基に更新した係数を第２の記憶部に格納することの一例である。なお、Ｓ６７からＳ６９の処理は、次のさらに次のバッチ（Ｎ＋２番目のバッチ）の学習処理が開始されるまでに実行される。

フロントページの続き

(72)発明者　笠置　明彦
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