情報処理学会論文誌　データベース　Vol.9　No.2　46–54 (June 2016)

GeSdA—GPU上でのAutoencoder処理並列化による高速Deep Learningの実装

中山 浩太郎[1,a)]　松尾 豊[1,b)]

受付日 2015年12月20日, 採録日 2016年4月8日

概要：「Deep learning」は，その強力な表現学習能力により画像処理・人工知能研究などを中心に幅広い分野で多大な注目を集めている．特に画像処理の分野を中心に，畳込みニューラルネットワーク（CNN）を基盤としたモデルについては多くの研究が精力的に進められ，高速化などの最適化においても顕著な成果が出ている．しかし，CNNと比較するとSdAなどの他のモデルに対するパフォーマンスの最適化についてはいまだ大きな課題として残っていた．本論文ではGPUを利用したSdAの高速化手法とその実装「GeSdA」を提案する．また，GeSdAの有効性を示すために開発したジェスチャー認識アプリケーション「Deep Motion」の概要と実験結果を示す．

キーワード：Deep Learning，SdA，GPGPU

GeSdA—High Performane Deep Learning Implementation with Autoencoder Parallelization

Kotaro Nakayama[1,a)]　Yutaka Matsuo[1,b)]

Received: December 20, 2015, Accepted: April 8, 2016

Abstract: The Stacked denoising Auto-encoder (SdA) is one of the most popular pretraining based deep learning algorithms and its capability as a representation learner is widely known among researchers. However, in contrast to Convolutional Neural Networks (CNNs), little attention has been paid to performance enhancement of SdA. We introduce GeSdA, a high performance SdA algorithm with an efficient GPU-based pretraining approach which we refer to as cross-layer parallel pretraining (CrossPre). Experimental results show that the proposed approach is helpful in improving computational performance and classification accuracy. To prove the practical value of GeSdA, we introduce DeepMotion as an application example.

Keywords: Deep Learning, SdA, GPGPU

1. はじめに

Deep Learningが人工知能研究のブレークスルーとして大きな注目を集めるようになって久しく，AutoencoderやRBM（Restricted Boltzmann Machine）を多層化する研究[3], [10]や，Deep Q Learning[13]などの強化学習といった分野で顕著な成果が出ている．また，従来から主に画像処理の分野で活発に研究がなされていたCNN（Convolutional Neural Network）[12]も，Deep Learningの一種のアルゴリズムとして，改めて注目されるようになった．

このように，もはや「Deep Learning」という1つの言葉でまとめるには難しいほど，多様な方向性のアルゴリズムが研究されているのが現状であるが，その中でも筆者らはSdA（Stacked denoising Autoencoder）[14], [15]に着目している．SdAは，Deep Learningの代表的なアルゴリズムの1つであり，CNNやDBN（Deep Belief Network）[10]，Maxout[9]などの他のアルゴリズムに比べて比較的シンプルな原理で動作するため，汎用性と拡張性が高く，画像処理以外の用途にも活用しやすいという特徴を持っている．

本論文では，GPU（CUDA）上で高速に動作するSdAの

[1] 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo, Bunkyo, Tokyo 113–8656, Japan
[a)] nakayama@weblab.t.u-tokyo.ac.jp
[b)] matsuo@weblab.t.u-tokyo.ac.jp

実装として，「GeSdA（GPU empowered Stacked denoising Autoencoder）」を紹介し，その詳細を解説する．特に，GeSdA の高速な処理を支えるレイヤー横断事前学習「CrossPre」について，その有効性を各種の実験によって示す．また，GeSdA の実用性を検証するために開発したアプリケーションの例としてジェスチャー認識システム「DeepMotion」を紹介する．DeepMotion の評価実験では，GeSdA のグリッドサーチ機能を利用して各種パラメータで学習したところ，従来の MLP や SVM に比べてエラー率が大幅に減少するパラメータの組合せを発見できることを確認した．

2. 従来研究

2.1 Deep Learning

1970 年代に登場したパーセプトロンに起源を持つ MLP（Multi Layer Perceptron）などのニューラルネットワークは，線形分離不能な問題に適用できる機械学習のアルゴリズムとして，幅広い分野で利活用されている．ニューラルネットワークは，理論的には層を重ねることで表現する能力が高まり，より複雑な問題に適用できるはずだと考えられていたが，実際には層を重ねていくと学習が収束せず，認識精度は向上するどころか大幅に下がることが指摘されていた [8]．そのため，歴史的には 3 層程度のレイヤー数の低い MLP が現実的なネットワーク構成として採用されてきた．

ニューラルネットワークの研究は長らく顕著なブレークスルーがなかったため，研究分野としては大きな着目を集めてきたとはいいがたい状況であったが，2006 年に Hinton ら [10] が 3 層以上多層にレイヤーを重ねても，レイヤーごとに事前学習（Pre-training）を行うことで，深い層を持つネットワークでも学習ができることを証明し，再び注目されるようになった．

Deep Learning のアルゴリズムは，DBN や SdA のように教師なし学習による事前学習（Pre-training）を行った後に，一般的な MLP と同様にラベルを利用した教師あり学習（Fine-Tuning）を行うものを指す場合が多い．この方式のアルゴリズムにおいては，事前学習を行うことで，重要な特徴表現を学習していると考えられている [2]．たとえば，Le らの研究 [11] では，深い階層を持つネットワークに対し大量の画像を利用して事前学習を行うことで，下位の層（入力に近い層）ではエッジ情報などプリミティブな特徴表現が学習されたことを確認し，より上層では猫や人といった抽象度の高い特徴表現が学習できていたことを確認している．

一方，CNN やその派生アルゴリズムのように事前学習を明示的に行わないアルゴリズムも存在する．そのため，事前学習によって特徴抽出を行うものを総称して Deep Learning である，と一様にまとめることは難しい．しかし，下層（入力層に近い層）で入力情報の特性を学習し，特徴表現を学習することで，深い層を持つニューラルネットワークの学習が可能になっている点は多くのアルゴリズムで共通している．そのため，本論文では，下層での表現学習を行う仕組みを持つことで特徴表現を学習し，深いレイヤー構造のニューラルネットワークの学習を可能にしているアルゴリズムを総称して Deep Learning と呼ぶこととする．

2.2 SdA

SdA（Stacked denoising Autoencoder）[15] は，Deep Learning の代表的なアルゴリズムの 1 つであり，他のアルゴリズムに比べて比較的シンプルな原理で動作するため汎用性と拡張性が高く，画像処理以外の用途にも活用しやすいという特徴を持つ．SdA は Pre-training（以降事前学習）と Fine-tuning という 2 つの学習フェーズを持ち，denoising Autoencoder を利用して事前学習を行う．**図 1** に SdA の概要を示す．

まず，SdA では事前学習フェーズで下層（入力層）から順に入力データを模倣するネットワークを学習する．この際，1 つの層（レイヤー）は，可視層（Visible Layer）と隠れ層（Hidden Layer）の 2 つのサブレイヤーから構成される．可視層から隠れ層方向へ，重み W とバイアス b に応じてデータを写像する処理を Encode，逆に隠れ層から可視層方向へ重み W' とバイアス b' に応じて再び入力層に情報を復元する処理を Decode と呼ぶ．Autoencoder は，元の入力情報と復元情報の誤差が最小になるように W，W'，b，b' を修正するように学習する．「denoising Autoencoder」は，入力情報の一部をわざと欠損させることで，ネットワークのロバスト性を向上させる手法である [14]．

事前学習フェーズでは，下から順に上述の denoising Autoencoder の処理をすべてのレイヤーに対して適用した後，通常の MLP と同様，誤差逆伝播法を利用した教師あり学習を行う Fine-tuning の学習を行う．つまり，事前学習な

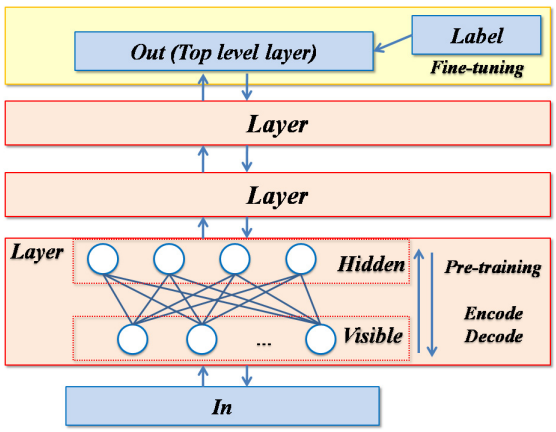


図 1　SdA の概要
Fig. 1　SdA at a glance.

しの SdA は従来の MLP と同義だと考えることができる．

2.3 Mini-batch SGD と GPU 計算

上述のとおり，SdA には事前学習と Fine-tuninig の 2 つの学習フェーズが存在するが，その両方で重要な役割を果たす学習アルゴリズムとして，Mini-batch SGD（Stochastic Gradient Descending）[4] がある．Mini-batch SGD は，入力データ集合を複数個（通常 10〜20 個程度）まとめた「ミニバッチ」を 1 つの単位として学習を行う．ここで，学習とは誤差関数を最小化するパラメータの探索を指す．事前学習フェーズでは復元した情報と元の入力情報との誤差，Fine-tuning フェーズでは，最上位レイヤーの出力情報と，教師データとの誤差を最小化するパラメータの探索に相当する．

事前学習における SGD を利用したパラメータの最適化は，Encode・Decode・誤差計算・勾配計算・パラメータの更新といった手順で行われる．具体的な手順を以下に示す．

(1) Encode（隠れ層の活性化）：

$$Y = f_\theta(X) = \phi(X \cdot W + b)$$

(2) Decode（可視層の活性化）：

$$Z = f_{\theta'}(Y) = \phi(Y \cdot W' + b')$$

(3) 誤差計算：

$$E = L(X, Z)$$

(4) 勾配計算と重みの修正：

$$\theta^{t+1} = \theta^t - \eta \nabla E(\theta^t)$$

（∇E の式は活性化関数と誤差関数に依存する）

- $\mathbf{X}$：入力ベクトルの集合（Minibatch 行列）
- $\mathbf{Y}$：隠れ層の活性化情報（Encode の結果）
- $\mathbf{Z}$：可視層の活性化情報（Decode の結果）
- $\mathbf{W}$：可視層→隠れ層の重み行列
- $\mathbf{W}'$：隠れ層→可視層の重み行列
- $\mathbf{b}$，$\mathbf{b}'$：隠れ層・可視層のバイアスベクトル
- t：学習時間
- η：学習係数（Learning Rate）
- θ：W，b のパラメータ集合
- θ'：W'，b' のパラメータ集合
- $f_\theta(X)$：パラメータ集合 θ と入力 X によるエンコード処理
- ϕ：シグモイド関数や $tanh()$ などの活性化関数
- $L(X, Z)$：損失関数

可視層のユニット i に対する誤差を δ_i としたとき，逆伝播公式 [4] によって隠れ層のユニット j の誤差 δ_j は以下のとおりに計算される．

$$\delta_j = f'_{\theta'}(Z) \sum_i w'_{ji} \delta_i$$

ここで Theano [1] や Torch7 [5] といった，Deep Learning で一般的に利用される実装においては，行列やテンソルに関する計算を高速に実行するために GPU を利用しているが，上記の Autoencoder の処理手順や特性などを考慮して処理を最適化しているわけではない．これは，Theano や Torch7 などはそれぞれ数値計算に特化した内部ライブラリを利用しており，多くの定型的な行列処理は高速に実行されているものの，SGD の処理の一部を直列化することや，分散並列計算可能な部分を並列化するといった複雑な制御ができないことに起因する．

また，CuDNN [*1] のように，CNN（Convolutional Neural Network）における畳み込み計算を高速化させる実装も公開されているが，SdA や RBM といった方式にそのまま適用することは難しい．

一方，複数の GPU や GPU サーバを並列に動作させることで高速化する研究が行われている [7], [11]．これらの研究では，複数のサーバに異なるハイパーパラメータで処理を実行し，精度の高いハイパーパラメータの組合せを探索する方式や，学習の一部を並列に計算する手法などを提案している．たとえば，DistBelief [7] では，パラメータの現在値を管理するサーバを用意し，SGD 計算を分散処理することで高速化する方法を提案している．DistBelief のように複数の計算機を利用した計算手法はいくつか提案されているものの，単独の計算機環境内での高速化，特に SdA の特性に着目した高速化には改善の余地があるのが現状である．

3. GeSdA

3.1 Cross-Layer Parallel Pretraining（CrossPre）とメモリ最適化

本研究では，単一 GPU 環境における事前学習の高速化を目指して開発した GeSdA と，SdA において Mini-batch SGD による学習をより高速に実行するための手法「CrossPre」を提案する．まず，**図 2** に GPU の内部構造を示す．

GPU 内部では，Thread が最小の並列計算単位となり，

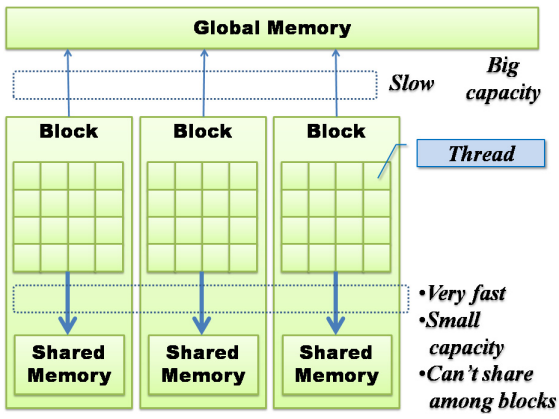


図 2　GPU の内部構造

Fig. 2　Parallel computing on a GPU.

[*1] https://developer.nvidia.com/cuDNN

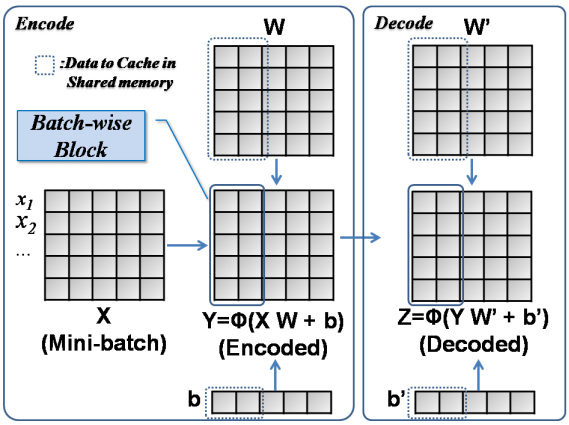


図 3　Encode-Decode 処理
Fig. 3　The encode-decode process.

複数の（計算ユニットによるが，一般的には最大 512）スレッド（Thread）をまとめたブロック（Block）という単位で「共有メモリ」（Shared Memory）と呼ばれる高速なメモリを共有する．このとき，ブロック内のスレッドどうしは共有メモリを介してデータを共有できるのに対し，ブロック間のデータのやりとりにはグローバルメモリ（Global Memory）と呼ばれる低速のメモリを介す必要があるため，高速な計算モデルを開発するためには，ブロック内の共有メモリへどんな情報を蓄えるか，どのような処理をスレッドに割り当て，ブロックに分割するか，どの処理とどの処理が並列で実行可能か，といったことを深く検討する必要がある．ここで，GPU 上へデータを受け渡し，実行する側の（CPU）環境を「ホスト」，対比として GPU を「デバイス」と呼び，ホストからデバイス上で処理の実行命令を与えることを「カーネル実行」と呼ぶ．ホストからデバイスへの情報の受け渡しやカーネル実行にはオーバヘッドが発生するため，ホストとデバイス間の制御フローの往来を最小化することも，処理速度向上のための重要な要因となる [16]．

上記の議論をふまえ，GeSdA では Autoencoder による事前学習の際の SGD 計算時のメモリ最適化戦略およびレイヤー横断並列化手法（CrossPre）を提案する．

本手法では，まず Encode 処理を行う際，隠れ層のユニットの出力値計算を 1 つのスレッドが担当し，いくつかの隠れ層ユニットをまとめて 1 つのブロックで処理することで処理速度を向上させている．具体的には，スレッドに対応する W と b の値を共有メモリにキャッシュとして共有することで，処理を高速化する．これは，Mini-batch SGD のように複数のレコードをひとまとめにして処理単位とする計算方式においては，W や b，レコード x などのデータが何度も参照されるという特性を利用したものである．**図 3** に Encode-Decode 処理の概念を示す．

次に，δ 計算処理の効率化について説明する．可視層において，誤差情報である δ は交差エントロピーや二乗和誤差などで計算されるが，その誤差情報は前述の逆伝播公式によって隠れ層のユニットへ伝播する．このとき，W' と可視層の δ の行列積の計算が発生するが，可視層と隠れ層のユニット数が同一である場合は，Encode・Decode 時と同じスレッドがユニットに割り当てられるため，計算結果を高速な共有メモリにキャッシュしておくことで処理の高速化が図れる．

表 1　通常の Greedy layer-wise pretraining
Table 1　Traditional layer-wise pretraining.

	1st Layer	2nd Layer	3rd Layer
1	Encode		
2	Decode		
3	Calc Gradients		
4	Update Params		
5	Encode		
…	…		
n	Update Params		
1		Forward Propagate	
2		Encode	
3		Decode	
…		…	

表 2　GPU での CrossPre 計算
Table 2　Cross-layer parallel pretraining on a GPU.

	1st Layer	2nd Layer	3rd Layer
1	Encode		
2	Decode	Encode	
3	Calc Gradients	Decode	Encode
4	Update Params	Calc Gradients	Decode
5	Encode	Update Params	Calc Gradients
…	…	…	…

最後に，レイヤー横断並列化手法「Cross-layer Parallel Pretraining」（以降 CrossPre）について説明する．一般的な事前学習方式である Greedy layer-wise pretraining の計算手順を**表 1** に，CrossPre の計算手順を**表 2** にそれぞれ示す．

SdA や DBN といった事前学習フェーズを持つ Deep Learning のアルゴリズムについては，下の層（入力層）に近い層から順に学習を行い，すべてのデータを十分に学習してから次の層の学習へと移行することが一般的である．一方，CrossPre では，これとは逆に 1 つのデータ（ミニバッチ）について下の層から順に学習させ，最上位の層までの学習したら次のデータに移行する．この方法にはいくつかの利点がある．

まず 1 点目の利点は，1 つのデータに集中して多くの学習を行えることである．前述のとおり，GPU と CPU 間のデータ通信には時間がかかるため，一度 GPU 上に展開したデータは複数の処理で共有する設計になっていることが望ましい．そのため，Deep Learning に関する研究では，多く（もしくはすべて）のデータを一度 GPU 上のメモリに格納しておくことが多い．しかし，GPU のメモリは通常の計算機に搭載するメモリと比較して容量が少ないため，大規模なデータを解析する際の問題となる．一方で事前学習をレイヤー横断で並列化し，1 つのデータ（ミニバッチ）を複数のレイヤーの事前学習に利用する場合，その時点で

必要となるメモリ領域は 1 データ分だけであり，何度も読み込む必要はない．

2 つ目の利点は，事前学習の各種処理を並列計算できることである．前述のとおり，SdA における事前学習では，Encode・Decode・誤差計算・勾配計算・パラメータの更新といった処理があるが，Encode 処理以外の処理はレイヤーごとに独立しており，並列計算が可能である．さらに，いったん誤差計算が終了すれば勾配計算やパラメータ更新といった処理もそれぞれのパラメータについて並列に計算・更新可能である．そのため，これらの処理を非同期に計算させることによって処理速度を向上させることができる．

4. 実験

GeSdA の有効性を確認するために，認識精度・計算速度について検証した．以下，実験の詳細と結果を示す．

4.1 認識精度の比較

GeSdA の主な開発目的は，1）高速な事前学習，2）多機能性，3）多様なアプリケーションへ適用可能な拡張性といった特徴を持った SdA 実装の実現であり，従来のアルゴリズムを超えるような高い精度を実現することは当面の優先的な目的ではない．しかし，SdA の実装として，他の実装と同程度の精度が実現できていることを検証することは重要であるため，Deep Learning の研究でよく利用されるデータセットの 1 つ，MNIST を利用して精度を検証した．以下に実験環境を示す．

- 入力データ：MNIST（784 次元）
- 隠れ層のユニット数：784
- GPU：GeForce 780 Ti（2,880 コア）

MNIST は，0〜9 までの手書きの数値を認識するタスクのためのデータ・セットであり，訓練用データ 6 万件（うち 1 万件は検証用データ），テスト用データ 1 万件から構成される．GeSdA のグリッドサーチ機能を利用して，事前学習のエポック数（Pretrain），ノイズ率（Noise），中間レイヤー数（Layers）などのパラメータが認識精度に与える変化を検証した．実験結果を**表 3** に示す．なお，最終的にテスト用データを用いた精度計測には，検証用データに対するスコアの一番高かったネットワークを利用した．

事前学習を行わない SdA は，従来の MLP と同様であり，ノイズ項は使われないため，ノイズ項は 0.0 の 1 つだけとした．GeSdA における認識精度は，Vincent らの論文 [14] で議論された SdA の特徴と，以下の点でほぼ合致することから，SdA の実装として正しく動作していると考えられる．

- 事前学習を行うことで認識精度が向上する．
- エラー率の最小値は 1.29%である（Vincent らの報告では 1.28%）．

表 3　グリッドサーチによるエラー率検証
Table 3　Grid search for the optimum parameters.

Pretrain	Noise	Layers	Corrects	Error
0	0	3	9,844	1.56%
		4	9,830	1.70%
		5	9,808	1.92%
		6	9,783	2.17%
30	0	3	9,844	1.56%
		4	9,833	1.67%
		5	9,838	1.62%
		6	9,833	1.67%
	0.1	3	9,838	1.62%
		4	9,846	1.54%
		5	9,857	1.43%
		6	9,856	1.44%
	0.2	3	9,841	1.59%
		4	9,850	1.50%
		5	9,871	1.29%
		6	9,868	1.32%
60	0	3	9,828	1.72%
		4	9,830	1.70%
		5	9,833	1.67%
		6	9,825	1.75%
	0.1	3	9,852	1.48%
		4	9,861	1.39%
		5	9,856	1.44%
		6	9,866	1.34%
	0.2	3	9,851	1.49%
		4	9,859	1.41%
		5	9,865	1.35%
		6	9,867	1.33%

- データにノイズを加えた場合，レイヤー構造が深いネットワークではエラー率が下がる．
- データにノイズを加えない場合，レイヤーを深くしてもエラー率は下がらない．

4.2 計算速度の比較

次に，CrossPre の有効性を確認するために，GeSdA の実行速度に関する実験を行った．なお，GPU などの実験環境は認識精度の比較実験と同様とする．

本実験では，ベースラインの実装として SGD の処理速度が高速である Theano を利用した．「(CUDA)」の表示があるものが GPU を利用して計算したものであり，「(BLAS)」の表示があるものが CPU を利用した場合の計測結果である．本実験では，CPU 上で数値計算を高速に行うための BLAS の代表的な実装である ATLAS（Automatically Tuned Linear Algebra Software）[*2] を利用して行列の処理に利用した．隠れ層のユニット数は入力と同じ 784 とした．**表 4** にベンチマークの結果を示す．

まず，表から分かるとおり，CrossPre を適用した GeSdA の実行速度が最小となった．GeSdA において CPU（BLAS）

[*2] http://math-atlas.sourceforge.net

表 4　処理速度ベンチマーク（μs）
Table 4　Benchmark simulations (in μs).

	Encode, Decode	Delta	Gradient	Total
GeSdA (CUDA) layer-wise	425	298	596	1319
GeSdA (CUDA) CrossPre	288	256	452	996
SdA + Blas	3,188	375	1,713	5,276
Theano (CUDA) with fastmath	1,427			1,427
Theano (BLAS)	6,361			6,361

を利用した計算方法と比較すると，GPU を利用することで 4 倍程度，CrossPre を導入すると 5.2 倍程度高速化できることが分かる．これは，事前学習の並列化や，メモリ最適化戦略が有効に動作していることを意味している．また，参考値として SGD の挙動が高速である Theano との比較も行ったが，CrossPre を導入した手法が Theano と比較して 40%程度高速であることが分かる．ただし，Theano と GeSdA では内部構造や計算方法が異なるため，Theano と比較して高速であるという結論を出すものではない．また，Caffe や Torch7 においても Autoencoder の実装があるが，CrossPre を導入することが可能であり，事前学習を高速化することが可能であると考えられる．

次に，事前学習の並列化（CrossPre）に関する精度の検証も行った．前述のとおり，事前学習を並列化することで計算時間の面では多くの利点が存在するが，問題は通常レイヤーごとに実施する事前学習を並列化することによる精度への影響である．この点について，どのような影響があるか以下のとおり実験した．1）精度への影響を調べるために MNIST データセットを利用して，並列化する場合としない場合（通常の SdA）における分類精度（エラー率）への影響を調べる，2）事前学習の回数（エポック数）を 0–20 へ変動させ，より詳細に事前学習への影響を調査する，3）問題を簡素化するために事前学習の回数以外のパラメータ（学習率・教師あり学習のエポック数・ノード数）を固定した．**図 4**（4 レイヤー）と**図 5**（5 レイヤー）に事前学習の並列化がエラー率に与える影響を調べる実験結果を示す．なお，CrossPre のあり・なしに加え，SGD の最適化手法として有名な AdaDelta[*3] と Momentum [6] を加えた場合の変化も観測した．

まず，事前学習を並列化する場合，しない場合（通常の SdA）両方において，事前学習のエポック数の増加とともに，最終的な分類のエラー率が下がることを確認した．これは事前学習によって画像内の重要な特徴表現が抽出され，最終的な分類精度の向上に寄与していることを示している [8]．

次に，並列化する場合としない場合での傾向を比較した場合，事前学習の効果がほぼ同等かむしろ改善される傾向

[*3] AdaDelta は，その有効性が広く知られているにもかかわらず，学術論文としての発表がない．詳しくは http://arxiv.org/abs/1212.5701 を参照．

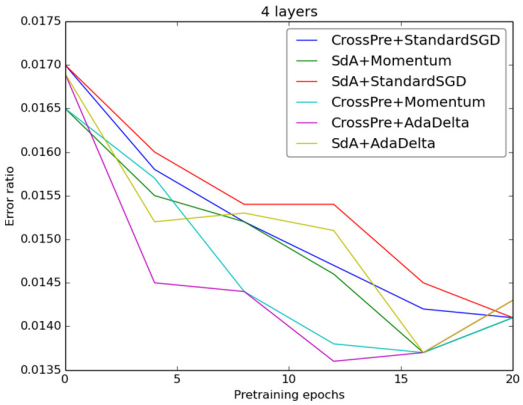


図 4　CrossPre とエラー率（4 レイヤー）
Fig. 4　Comparison in 4 layers network.

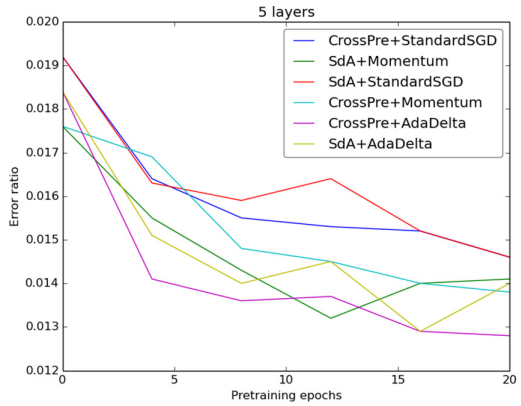


図 5　CrossPre とエラー率（5 レイヤー）
Fig. 5　Comparison in 5 layers network.

があることが分かる．これは，並列化された事前学習も，通常の事前学習と同様の効果が同程度のエポック数で得られることを示唆している．また，AdaDelta や Momentum を適用することでさらにエラー率が下がる傾向にあり，各種の最適化手法との組合せが可能であることを示している．本現象について，MNIST での検証のみであるため，他のデータセットでの検証など，さらなる検証を要するが，少なくとも並列化することで事前学習の計算時間を短縮した上で精度を保持できる可能性を示唆している．

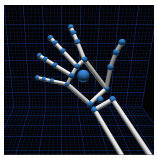
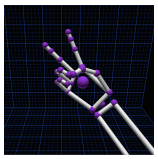
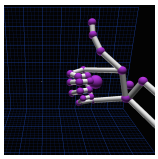
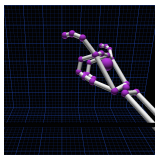
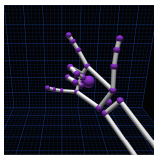
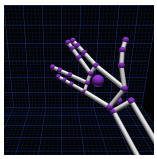
5. DeepMotion

SdA の特徴は，その簡易な構造に基づく拡張性の高さである．GeSdA のアプリケーション例として，ハンドサイン認識システム「DeepMotion」を開発し，その評価実験を行った．以下，その詳細を説明する．

5.1 概要

DeepMotion は人間の指の動きからハンドサインを認識するシステムである．DeepMotion で識別対象とする 6 つのハンドサインを**表 5** に示す．システムの動作手順としては，まず表 5 に示すハンドサインに応じた学習用のデー

表 5　ハンドサイン一覧
Table 5　Hand signs used in the experiment.

Open	Peace	Thumb	Fox	Yo	Spoch
					
					

タ・セットを用意しておく．GeSdA を利用して各ジェスチャーを学習しておき，最後に学習済みのネットワークのデータを利用することでリアルタイムにハンドサインの認識を行う．なお，実際のユーザが利用する際には，1 つのジェスチャーにつき 1 分程度の学習を行うを行うことを想定している．

5.2　内部構造

DeepMotion では，指情報を取得するために，LeapMotion（TM）[*4]という高解像度・高フレームレートのステレオ画像解析システムを利用した．LeapMotion（TM）は，指情報の取得に特化したセンサの一種であり，ステレオ画像から物体の奥行き情報を取得し，データを解析するドライバを介して，指の位置や方向といったベクトル情報に変換する．具体的には，高解像度の画像情報を 30 フレーム/秒で解析し，センサ情報としてアプリケーション側に送信する．この結果，学習用のセンサ情報として膨大な量のデータが蓄積されるが，この膨大なデータを高速に解析する技術が重要となる．特に，実運用を考えた際には，ユーザによるジェスチャ登録から学習終了までの時間をいかに短くできるかが重要なアプリケーションであるため，GeSdA のように GPU で高速な学習が可能な手法が有効であると考えられる．

本システムの構成を**図 6** に示す．

5.3　評価実験

本システムの実用性を評価するために，以下の条件で事前学習ありと事前学習なし（Fine-tuning のみ）の場合に分けて精度を計測した．この際，グリッドサーチにより様々なハイパーパラメータを適用し，その変化を観察した．

[*4] https://leapmotion.com

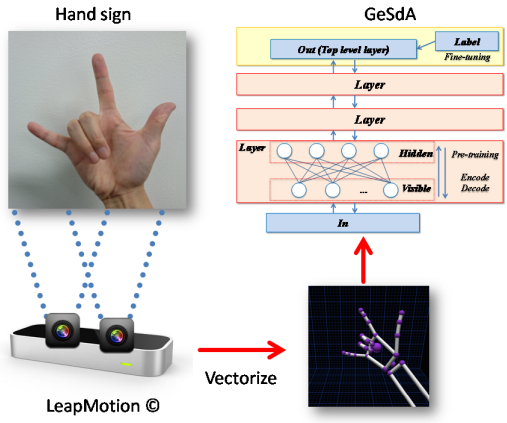


図 6　DeepMotion 構成図
Fig. 6　DeepMotion at the glance.

- Fine-tuning の最大エポック数は 100 回
- 事前学習の最大エポック数は 60 回
- ただし，Fine-tuning，事前学習ともに検証用データを利用して，学習の収束を調べ，エラー率が最も低くなった時点のネットワークを採用
- 6 種類のジェスチャーそれぞれについて，約 600 フレームのデータ（サブセット）を 10 セット収録
- そのうちランダムに選んだ 2 つのセットをそれぞれ検証用とテスト用に，残りの 8 セットを訓練用に割り当て，学習，検証，テストに利用
- テストセットの総フレーム数は 3,674 フレームであった
- レイヤー数（layers）は，3 層～7 層までの変化した場合を調査（3 の場合，入力・中間・出力の 3 層の MLP と同一の構造となる）

評価結果を**表 6**（事前学習あり）と**表 7**（事前学習なし）に示す．

まず，事前学習なしの場合，ネットワークの深さが深く

表 6　事前学習ありグリッドサーチ
Table 6　Xxxx xxxx xxxx xxxx xxxx.

	Noise	LR	Layers	Correct	Error
With pre-training	0.1	0.05	3	3,209	12.7%
			4	2,603	29.2%
			5	2,478	32.6%
			6	3,572	**2.8%**
			7	3,251	11.5%
		0.1	3	2,854	22.3%
			4	2,749	25.2%
			5	3,077	16.2%
			6	3,190	13.2%
			7	2,994	18.5%
	0.2	0.05	3	2,619	28.7%
			4	2,634	28.3%
			5	2,711	26.2%
			6	2,824	23.1%
			7	1,853	49.6%
		0.1	3	2,608	29.0%
			4	2,602	29.2%
			5	3,370	**8.3%**
			6	3,228	12.1%
			7	2,234	39.2%

表 7　事前学習なしグリッドサーチ
Table 7　Xxxx xxxx xxxx xxxx xxxx.

	Noise	LR	Layers	Correct	Error
Without pre-training	0.1	0.05	3	2,533	31.1%
			4	1,861	49.3%
			5	1,969	46.4%
			6	1,333	63.7%
			7	650	82.3%
		0.1	3	2,641	28.1%
			4	1,923	47.7%
			5	1,857	49.5%
			6	2,004	45.5%
			7	1,333	63.7%

なるにつれてエラー率が上昇していることが分かる．これは従来のニューラルネットワークの研究で繰り返し指摘されていたとおり，階層が深くなることで誤差情報が下層に伝播されずに学習が進まないことに起因する．逆に，事前学習を行うことで，全体的に精度が大幅に向上していることが分かる．これは，事前学習が入力に近い層で重要な特徴表現を学習した結果，分類精度が向上するという Deep Learning 研究の知見と合致する結果であるといえる [3], [10], [14], [15]．また，事前学習を行った場合は，5 層や 6 層といった深い階層を持つネットワークにおいて，非常に低いエラー率を持つネットワークが学習できていることが確認できる．ただし，7 層のネットワークではすべての条件でエラー率が上昇している．また，同じ 5 層・6 層を持つネットワークとしても，学習率（LR）の値が 0.05 かつノイズ率が 0.2 の場合は非常に高いエラー率を持つネットワークが学習されるなど，パラメータの組合せによって精度に大きなばらつきが生じる現象を確認した．

学習済みのネットワークを利用して認識精度について，個別に検証したところ，事前学習なしの場合，Open と Spoch，Yo と Fox といった差異が少ないジェスチャーどうしを分類するのが困難であったのに対し，事前学習をしたネットワークではこれらの微小な違いにも反応し，多くの場合正しい分類ができていたことを確認した．また，参考情報として，同じデータを利用して SVM で学習した結果を**表 8** に示す．まず，SVM でグリッドサーチを行わなかった場合，エラー率は 16.1%となった．次に，最適なパラメータをグリッドサーチによって探索した結果，エラー率を 8.7%まで下げることができたが，GeSdA の最高値 2.8%には届かなかった．

表 8　SVM との比較
Table 8　Xxxx xxxx xxxx xxxx xxxx.

	Grid Search	Correct	Error
SVM	with Grid search (c=2.0, g=2.0)	3,216	**8.7%**
	without Grid search	3,082	16.1%

DeepMotion 開発における技術的課題は，即時性の高い解析と認識精度であった．即時性の問題に対しては GeSdA の GPU 計算機能，特に CrossPre によってより高速な解析・学習が可能となった．今回の実験では，1 ジェスチャーに付きデータ取得に 1 分程度の時間を要し，学習には 6 つのジェスチャーすべてを学習するために，全体で数分程度の学習時間が必要であった．これは，ユーザが各自の環境で新しいジェスチャーなどを登録したとしても，リアルタイムで学習できる程度の時間であると考えられる．しかし，ジェスチャーを学習するのに必要なデータを取得するのにどの程度のデータ数（キャプチャ時間）が必要か，という点についてはより詳細な実験が必要と思われる．

6. まとめ

本論文では，GPU を利用した高速な SdA 実装として GeSdA を提案し，各種の実験によりその有効性を示した．GeSdA で採用したレイヤー横断の事前学習並列化手法「CrossPre」は，Autoencoder の各種の処理を GPU 上で並列計算し，高速化することが可能であることが示された．また，実験結果から，CrossPre は通常の SGD による事前学習と比較してさらにエラー率の低下に貢献することや収束に必要な学習数を減少させることが可能であることを示しており，AdaDelta や Momentum などの各種最適化手法と組み合わせることが可能なことも示された．今後は他のテストセットや各種の条件下でより詳細に検証することによって，その特性がさらに明確になることが期待される．

GeSdA を利用したアプリケーション「DeepMotion」は，SdA が画像などのラスタデータだけでなくベクトル情報を扱うためにも適していることを示唆している．今後は天気予報のベクトルデータ解析や時系列データの解析など，より多様なアプリケーションへ適用し，GeSdA の有効性を示す予定である．

参考文献

[1] Bengio, J.B., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D. and Yoshua: Theano: A CPU and GPU math compiler in Python, *9th Python Sci. Conf.*, pp.3–10 (2010).

[2] Bengio, Y., Courville, A. and Vincent, P.: Representation learning: A review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.35, No.8, pp.1798–1828 (2013).

[3] Bengio, Y. and Lamblin, P.: Greedy layer-wise training of deep networks, *Neural Inf. Process. Syst.*, pp.153–160 (2007).

[4] Bishop, C.: Evaluation of error-function derivatives, *Pattern Recognit. Mach. Learn.*, p.244 (2006).

[5] Collobert, R.: Torch7: A matlab-like environment for machine learning, *Conf. Neural Inf. Process. Syst. Work.*, pp.1–6 (2011).

[6] David E. Rumelhart, Geoffrey E. Hinton, R.J.W.: Learning representations by back-propagating errors, *Nature*, pp.533–536 (1986).

[7] Dean, J., Corrado, G. and Monga, R.: Large scale distributed deep networks, *Neural Inf. Process. Syst.*, pp.1–11 (2012).

[8] Erhan, D., Bengio, Y. and Courville, A.: Why does unsupervised pre-training help deep learning?, *J. Mach. Learn. Res.*, Vol.11, pp.625–660 (2010).

[9] Goodfellow, I.J., Warde-farley, D. and Courville, A.: Maxout Networks, *Proc Int. Conf. Mach. Learn.*, pp.1319–1327 (2013).

[10] Hinton, G.E. and Salakhutdinov, R.R.: Reducing the Dimensionality of, *Science (80-.)*, Vol.313, No.July, pp.504–507 (2006).

[11] Le, Q.V., Ranzato, M.A., Devin, M., Corrado, G.S. and Ng, A.Y.: Building High-level Features Using Large Scale Unsupervised Learning, *Proc. Int. Conf. Mach. Learn.* (2012).

[12] LeCun, Y., Bottou, L. and Bengio, Y.: Gradient-Based Learning Applied to Document Recognition, *Proc IEEE*, Vol.86, No.11, pp.2278–2324 (1998).

[13] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.a., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D.: Human-level control through deep reinforcement learning, *Nature*, Vol.518, No.7540, pp.529–533 (online), DOI: 10.1038/nature14236 (2015).

[14] Vincent, P.: Stacked Denoising Autoencoders : Learning Useful Representations in a Deep Network with a Local Denoising Criterion, *J. Mach. Learn. Res.*, Vol.11, pp.3371–3408 (2010).

[15] Vincent, P. and Larochelle, H.: Extracting and Composing Robust Features with Denoising Autoencoders, *Proc. Int. Conf. Mach. Learn.*, pp.1096–1103 (2008).

[16] Volkov, V. and Demmel, J.W.: Benchmarking GPUs to Tune Dense Linear Algebra, *Proc. Int. Conf. Supercomput.*, Vol. Nov, No.November, pp.1–11 (2008).

中山 浩太郎　（正会員）

2007 年大阪大学大学院情報科学研究科博士号取得．博士（情報科学）．（株）関西総合情報研究所代表取締役，大阪大学研究員，東京大学知の構造化センター特任助教等の勤務を経て，2014 年より，東京大学大学院工学系研究科技術経営戦略学専攻特任講師．情報処理学会山下記念研究賞，情報処理学会 CS 専攻賞，日本データベース学会論文賞等受賞．専門は，人工知能，Deep Learning，スケーラビリティ，Web Mining．

松尾 豊　（正会員）

1997 年東京大学工学部卒業．2002 年同大学院博士課程修了．博士（工学）．産業技術総合研究所，スタンフォード大学を経て，2007 年より，東京大学大学院工学系研究科技術経営戦略学専攻准教授．2012 年より人工知能学会理事・編集委員長，2014 年より倫理委員長．人工知能学会論文賞，情報処理学会長尾真記念特別賞，ドコモモバイルサイエンス賞等受賞．専門は，Web 工学，Deep Learning，人工知能．

（担当編集委員　小林 大）