Vol.2017-HPC-158 No.25
2017/3/9

低ランク近似行列によるCNNにおける畳み込み演算の最適化

本山 義史[1,a)]　遠藤 敏夫[1]　松岡 聡[1]　横田 理央[1]　福田 圭祐[1]　佐藤 育郎[2]

概要：機械学習による画像認識の分野において, Convolutional Neural Network(CNN) を用いた優れた認識結果が報告されている. データセットが巨大であるため, 学習には非常に大きな時間がかかり, また, 必要となるメモリ量は大きくなる. そこで我々は, DL の計算におけるメモリ量の削減を図るため, 畳み込みの演算の約 7 割を占める行列積計算に対し, 低ランク近似行列を用いることを提案する. CNN アプリケーション中の行列に対し, SVD と階層型行列を適用し, 評価した. 特に, SVD を用いた時, 圧縮率と精度とのトレードオフにおいて, 認識精度をほとんど落とさず, サイズが特に大きい image 行列で最大約 9 割程のメモリ量削減に成功した.

1. はじめに

近年, Neural Network を用いた機械学習手法である Deep Learning(DL) が盛んに研究されている。画像認識の分野においては, Convolutional Neural Network(CNN) を用いた優れた認識結果が報告されている. [1] しかし、データセットが巨大であり, また学習や認識には非常に大きな時間がかかることが問題とされている. 本研究では、特に計算中の使用メモリ量の問題に着目する.

DL の計算を加速するために GPU が盛んに利用されているが、一般に GPU はホストメモリとくらべて搭載しているメモリ量が少ない. また DL の推論部分は IoT 機器や組み込みデバイス等での利用が期待される. このような搭載メモリ量が少ないデバイスにおいて DL を活用する際は, 特に推論部分の計算に必要なメモリ量が膨大であることは大きな課題である.

一方, 深層学習の演算に必要な精度は, 単精度や半精度でも十分な場合が多いことがすでに報告されており [2], 古典的な科学技術計算よりも誤差が比較的に許容されるアプリケーションと言える. しかし計算精度の低下を許容して計算を最適化する方法は, 演算精度を下げることだけではない. 近似計算の手法を用いて, ある程度の誤差を許容することによって計算の最適化を図る手法は広く行われており, 代表的なものには N 体問題で用いられる Barnes-Hut 法 [3] や FMM 法 [3] などが挙げられる.

本研究では, DL の計算に必要とされるメモリ量を削減することを主眼として, DL の計算に用いる行列と演算を近似的におこなう手法として低ランク近似を適用し評価する. 具体的には, 行列に対する低ランク近似法として, SVD と階層型行列の二種類の方法を適用し, その時の使用メモリ量の削減と学習結果の認識精度の低下のトレードオフを評価した.

一つ目の低ランク化手法は, 階層型行列を用いた行列積演算である. 階層型行列とは, 密行列を階層的な正方行列に分解し, 各小行列を低ランク行列で近似した密行列の近似圧縮表現の一種である. 密行列を有効に圧縮した元の行列に比べ, 疎な近似行列表現となっており, この近似行列表現を用いることにより, 元行列の圧縮や計算量削減を図っている. 階層型行列ライブラリが正方行列向けであるため, 深層学習アプリケーションで発生する Tall-and-skinny 行列に対しては, 行列全体を分割し階層型行列を適用する手法と 0 要素でパディングする手法を提案する.

二つ目は, 特異値分解を行列全体に適用した行列積演算である. 特異値分解を行列全体へと適用することにより, 元行列を 3 つの積で分解することができる. 特異値分解後の行列は, 特異値の値を減らすことにより, その行列のランクを落としつつも, その近似行列として表すことができる. 近似圧縮表現を深層学習アプリケーションに適用し, ランクを落としつつも認識精度が保てることを示した.

1　東京工業大学
2　デンソーアイティーラボラトリ
a)　motoyama.y.ab@m.titech.ac.jp

ⓒ 2017 Information Processing Society of Japan

2. 背景

2.1 Deep Learning

Deep Learning とは, 多層構造のニューラルネットワーク (以下, NN) を用いた機械学習の手法の一種である. NN は, 人間の脳を模倣したアルゴリズムで, パターン認識をするよう設計されている. Deep Learning では, NN を深く階層化した Deep Neural Network(以下, DNN) を用いているが学習対象によって, その構造は大きく異なる.

2.2 対象とする DL システム SPRINT

株式会社デンソー及びデンソーアイティーラボラトリが開発した大規模分散 DL システム SPRINT は, ILSVRC-2015 の Object localization のための学習を行うよう設計されたシステムである. SPRINT は ILSVRC-2015[4] において, 62.7409% の Localization error, 13.893% の Classication error を達成した.

2.3 低ランク近似行列

低ランク近似とは, 図 1 のように行列を積で分解し, 行列を圧縮する手法である.

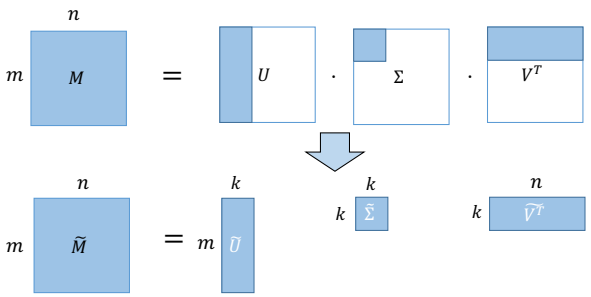

図 1 低ランク近似行列

2.3.1 階層型行列

階層型行列 (以下, Hierarchical matrix とも表記) とは, 密行列を階層的な正方行列に分解し, 各小行列を低ランク行列で近似したものである. 密行列を有効に圧縮した, 元の行列に比べて疎な近似行列表現となっている. この構造に基づき, 行列-ベクトル積や行列-行列積などの行列演算を小さなメモリ量で, 高速に行うことができる.

同じサイズ ($n \times n$ 行列) の密行列と階層型行列の行列積や LU 分解, 逆行列のような演算量の比較を表に示す.

表 1 $n \times n$ 密行列と階層型行列のデータ量と演算量の比較

	密行列	階層型行列
データ量	$O(n^2)$	$O(rnlogn)$
計算量	$O(n^3)$	$O(r^2nlogn^2)$

2.3.2 特異値分解 (SVD)

行列の特異値分解 (以下, SVD) とは, 任意の $m \times n$ 実数行列 A を以下式 1 のように 3 つの積で分解することである.

$$A = U\Sigma V^T \tag{1}$$

ただし, Σ は正数 $\sigma_1 \ldots \sigma_k$ からなる対角行列, k は行列 A のランク ($k \leq min(I, J)$) である. U, V はそれぞれ $I \times K, J \times K$ の行列で, その列は正規直交である. すなわち, 行列 U, V に対し, それぞれ $UU^T = U^TU = I, VV^T = V^TV = I$ が成立する.

以下, SVD アルゴリズムを示す.

Algorithm 1 SVD アルゴリズム

$B \leftarrow Q^TAP$ { Bidiagonalization of A to B }
$\Sigma \leftarrow X^TBY$ { Diagonalization of B to Σ }
$U \leftarrow QX$
$V^T \Leftarrow (PY)^T$ { Compute orthogonal matrices U and SVD of $A = U\Sigma V^T$ }

2.3.3 DL 中の計算中に現れる行列の特徴

深層学習アプリケーション内で大量に発生する Tall-and-skinny 行列が DL アプリケーション SPRINT 内の畳み込み演算の大部分を占める. 表 2, 3 に 1 iter 中にでてくる行列サイズを示す.

表 2 1ter 中の FeedForword に出てくる行列サイズ

$A(m, k)$	$B(k, n)$
(155236, 27)	(27, 32)
(153664, 288)	(288, 38)
(37636, 342)	(342, 54)
(36864, 486)	(486, 64)
(8836, 576)	(576, 90)
(8464, 810)	(810, 106)
(1936, 954)	(954, 150)
(1764, 1350)	(1350, 178)
(1600, 1602)	(1602, 212)
(324, 1908)	(1908, 298)
(256, 2682)	(2682, 354)
(196, 3186)	(3186, 421)
(25, 3789)	(3789, 593)
(9, 5337)	(5337, 704)
(1, 32)	(32, 6336)
(32, 1)	(1, 1)

2.3.4 本研究に使用したライブラリについて

HLIBpro

HLIBpro [6] は, 階層型行列または行列のアルゴリズムを実装するライブラリである. HLIBpro には, ACA や HCA のような密行列の近似, 利用可能な代数の完全集合, 幾何学的および代数的クラスタリングなどの様々なクラスタリング技法, ラプラス, ヘルムホルツ, マクスウェル方程式などの積分方程式を離散化するための多くの関数が含まれてい

表 3　1ter 中の Backpropagation に出てくる行列サイズ

$A(m,k)$	$B(k,n)$	$A(m,k)$	$B(k,n)$
(6336, 1)	(1, 32)	(1764, 178)	(178, 1350)
(1, 32)	(32, 6336)	(954, 1936)	(1936, 150)
(5337, 9)	(9, 704)	(1, 1936)	(1936, 150)
(1, 9)	(9, 704)	(1936, 150)	(150, 954)
(9, 704)	(704, 5337)	(810, 8464)	(8464, 106)
(3789, 25)	(25, 593)	(1, 8464)	(8464, 106)
(1, 25)	(25, 593)	(8464, 106)	(106, 810)
(25, 593)	(593, 3789)	(576, 8836)	(8836, 90)
(3186, 196)	(196, 421)	(1, 8836)	(8836, 90)
(1, 196)	(196, 421)	(8836, 90)	(90, 576)
(196, 421)	(421, 3186)	(486, 36864)	(36864, 64)
(2682, 256)	(256, 354)	(1, 36864)	(36864, 64)
(1, 256)	(256, 354)	(36864, 64)	(64, 486)
(256, 354)	(354, 2682)	(342, 37636)	(37636, 54)
(1908, 324)	(324, 298)	(1, 37636)	(37636, 54)
(1, 324)	(324, 298)	(37636, 54)	(54, 342)
(324, 298)	(298, 1908)	(288, 153664)	(153664, 38)
(1602, 1600)	(1600, 212)	(1, 153664)	(153664, 38)
(1, 1600)	(1600, 212)	(153664, 38)	(38, 288)
(1600, 212)	(212, 1602)	(27, 155236)	(155236, 32)
(1350, 1764)	(1764, 178)	(1, 155236)	(155236, 32)
(1, 1764)	(1764, 178)		

る．ただし, HLIBpro は正方行列のみを対象としている．

LAPACK

LAPACK(Linear Algebra PACKage)[12] は, 線形計算のための数値解析ソフトウェアライブラリである．BLAS をビルディングブロックとして使い, 線形方程式や線形最小二乗法, 固有値問題や本論文でも扱う特異値問題等を数値的に解くために利用されている．

MAGMA

MAGMA(Matrix Algebra for GPU Muticore Architectures)[13] は, テネシー大学を中心に Stanimire Tomov らのグループで作成されている GPU 向けの線形代数ライブラリである．CPU も同時に使用するヘテロジニアス型のライブラリであるため, GPU 単体より一般に性能がよい．また, MAGMA は LAPACK, BLAS を基に設計されており, LU 分解, QR 分解, 連立一次方程式, 固有値問題, 特異値問題などのルーチンが実装されている．

3. 関連研究

3.1 階層型行列適用による行列圧縮に関する研究

N 体問題シミュレーションや境界要素解析では, 物理要素間の相互関係を表す密行列を係数行列とする連立一次方程式を解くことが多い．N 個の物理要素の全相互関係を表す密行列の要素数は N^2 となり, これを直接扱うと N が増大するにつれ, 必要となるメモリ量・計算量は急激に増大する．そこで, N 体問題シミュレーションや境界要素解析に対し, 密行列の近似圧縮表現である階層型行列を用いて, メモリ量・計算量を抑えつつも, 精度は保ったまま, 計算をする手法が研究されてきた．[8]

本研究では, これまで N 体問題や境界要素解析への適用が主だった階層型行列を深層学習へ適用している．関連研究では, 正方行列を対象としているため, そのまま階層型行列を適用することができるが, 本研究では, Tall-and-skinny 行列と呼ばれる矩形行列を扱うため, 正方行列へと変換するため, 分割による手法とパディングによる手法を用いて, 正方行列の作成をしたのち, 各ブロックごとに階層型行列への適用をし, 行列積演算を行なっている．また, 低ランクの際には ACA 法を元にした ACA+ 法 [11] という低ランク手法を用いている．

3.2 DNN における低ランク近似行列適用に関する研究

Preetum Nakkiran[9] らは, DNN のサイズを縮小するため, DNN の入力層の重みを低ランク近似する手法を提案している．この手法では, 個々の隠れ層ノードで学習されたフィルタのレベルで適用され, 入力における自然な 2 次元の時間周波数構造を利用している．これらの手法は, 小さなフットプリントの DNN ベースのキーワードスポッティングタスクに適用することで, パフォーマンスを損なうことなくモデルサイズをベースラインに対し, 75 %削減することに成功した．さらに Preetum Nakkiran らによって提案された手法では, 同等の数のパラメータで評価されたときの他の一般的な次元削減技術と比較して, モデル性能を改善する上でより効果的であることが明らかとなっている．

画像認識の分野以外にも, ディープニューラルネットワーク (DNN) は, 自動音声認識に非常に有効であることが証明されている．しかし, パラメータの数が多く, 計算コストが高くなる傾向がある．DNN モデルのサイズを縮小するために, SVD を利用し計算された重み行列の低ランク近似は以前より適用済みだが, ノイズのないクリーンな音声のみの適用であった．そこで Tachioka Y. ら [10] は, ノイズの多い音声の追加の変動はモデルの縮小を困難にする可能性があると考え, SVD の騒音残響音声に対する有効性を検討し, 低ランク近似とシーケンス識別訓練を組み合わせることで, 元の DNN が識別基準を使用して構築されたとしても DNN のパフォーマンスがさらに向上することを示した．また, さらにこの実験では, 低ランク近似が騒音の発声に有効であることを示し, 識別訓練とモデル減少との最も効果的な組み合わせは, まず低次元近似をベースモデルに適用し次に低ランクモデルで差別訓練を行うことであり, この低ランクの弁別的に訓練されたモデルは, 完全に弁別的に訓練されたモデルよりも優れている結果となった．

4. 提案

4.1 CNN における畳み込み演算とメモリ使用量削減のための提案

CNN アプリケーション中の畳み込み演算では, 図 2 のように, image 行列と weight 行列の内積計算の繰り返しにより, 計算を更新していく.

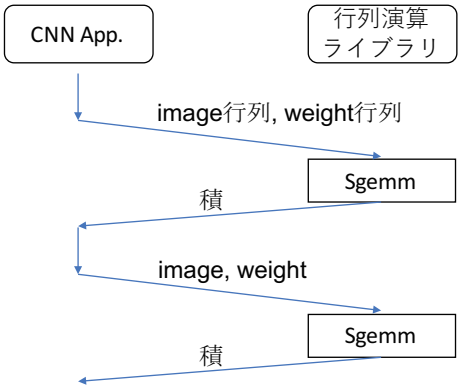

図 2 CNN への低ランク近似適用の流れ

CNN アプリケーションから image 行列と weight 行列の内積計算をする際には, cublas などといった行列演算ライブラリを用いて, 行列演算を行い, その演算結果を CNN アプリケーションに返すということを繰り返し行なっている. CNN アプリケーション中では, Tall-and-skinny 行列と呼ばれる, $m \times n$ 行列の一辺に対し, もう一辺の大きさが非常に大きい行列 ($m >> n$ または $m << n$) 同士の行列演算を行うことから, 我々はこの演算部分に対し, 低ランク近似行列を適用することにより, メモリ量使用の削減を提案する.

4.2 CNN への低ランク近似行列への適用

本研究では, メモリ量と精度のトレードオフの評価に着目し, 元アプリケーションに対する変更量を少なく抑えるため, Sgemm 関数呼び出し部分を置き換え, 低ランク化・低ランク行列積, 逆変換を行なった.

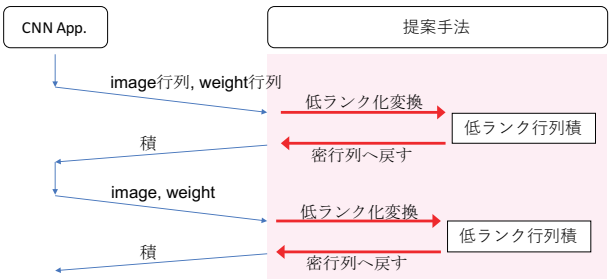

図 3 CNN アプリケーションの畳み込み計算の流れ

次節以降では, 具体的な低ランク近似手法の実装説明に移る.

4.3 階層型行列による実装

本研究で提案する階層型行列の概要を示す. 本研究で使用した階層型行列ライブラリ H-LIBpro[6] は正方行列を対象としている. それ故, 本研究では, Tall-and-skinny 行列を正方行列へ変換するため, 分割による手法と 0 要素のパディングの 2 種類を行い, 正方行列を作り出し比較・調査をしている. 以下では, 具体的に行列を分割により, 正方行列にしたのちに階層型行列へ変換する手法の説明と 0 要素パディングにより正方行列へ変換したのち, 階層型行列へ変換する手法の説明に移る. なお, 以下で示す image 行列と weight 行列は畳み込み演算時の内積計算で用いる行列のことであり, (m, n, k) は Sgemm 計算時のパラメータ ($m \times k$ と $k \times n$ の行列積) を意味する.

4.3.1 行列分割による実装

(1) image 行列, weight 行列それぞれに対し, $\min(m, n, k)$ で分割サイズを決定.
(2) image 行列, weight 行列ともに $\min(m, n, k)$ のブロック正方行列の作成.
(3) 正方行列を作成できない一番端のブロックについては, 0 要素をパディングし, 正方行列とする.
(4) 各ブロックごとに階層型行列変換を行う.
(5) 階層型行列変換後のブロック同士の行列積演算を行う.

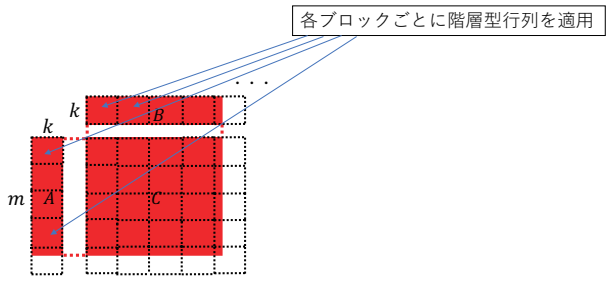

図 4 行列分割による階層型行列適用

4.3.2 0 要素パディングによる実装

(1) image 行列, weight 行列に対し, 0 パディング後に新たに設定される正方行列の行数・列数を p とし, image 行列, weight 行列の行数・列数を (n, m, k) を用いて, $p = \max(m, n, k)$ と決定する.
(2) image 行列, weight 行列を 0 パディングを用いて, 前段で決定した新しい行列サイズへ拡張する.
(3) image 行列, weight 行列に対し, 階層型行列への変換を行う.
(4) 階層型行列変換後同士の行列積を実行する
(5) 計算された積を密行列へ逆変換する

ⓒ 2017 Information Processing Society of Japan

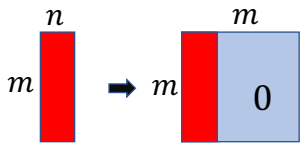

図 5　パディングによる階層型行列適用

4.4　SVD による実装

本研究で提案する SVD を用いた, DL への低ランク近似行列適用の概要を示す.

(1) image 行列, weight 行列それぞれに対し, SVD の適用により, 3 つの行列の積へと分解する.
(2) それぞれ特異値の数を減らし, 低ランク近似をする. 特異値分解後の最大特異値を基準とし, α%以下の特異値を削るという作業で, 図 6 のように特異値を削り, ランクを落としていく. 閾値 α の決め方は, 認識精度とのトレードオフにより, 決めている.
(3) ランクを落とした行列同士の行列積演算を行う.

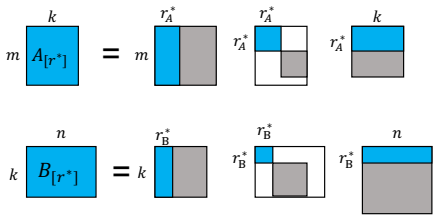

図 6　SVD 適用によるランク数削減 (行列 A のランク数 $r \to r_A$, 行列 B のランク数 $r \to r_B$)

5.　評価

提案する低ランク近似手法を評価するために, SPRINT を表 4 の評価環境上で実行・評価を行なった. 評価の全体像として, SVD と階層型行列適用による行列の削減具合, 計算時間, 認識精度の 3 つの観点で比較, 評価を行う.

表 4　計測環境

	環境 1(階層型行列)	環境 2(SVD)
CPU	Intel Core i7-3930K	Intel Core i7-6700K
動作周波数	3.2GHz	4.00GHz
コア数	6	4
L1D cache size	32kB	32kB
last level cache size	12MB	8MB
GPU	Tesla K20c	Tesla K40c
単精度 FLOPS/s	3.52 TFlop/s	4.29 TFlop/s
コア数	2496	2880
メモリ	5120 MB, GDDR5	12 GB, GDDR4
メモリバンド幅	208.0 GB/s	288 GB/s

実験に使用したコンパイラ・ライブラリ等のバージョンを表 5 に示す.

表 5　コンパイラ等のバージョン

種類	バージョン
C コンパイラ	gcc 4.8.5 20150623
CUDA	CUDA 7.5
HLIBpro	hlibpro-2.4
MAGMA	magma-2.1.0

ImageNet Large Scale Visual Recognition Challenge(ILSVRC)[5] は, 2010 年から始まった大規模画像認識の競技会であり, 参加しているチームのほとんどが Deep Learning を使用し, 画像認識 Deep Learning の大きな競技会といえる. 学習には用いないが, Validation と呼ばれる汎化性能を検証するために用いられるために用いられるデータセットは, 予測した上位 1 位のラベル中に正解のラベルが含まれない確率の平均である top-1 error, 上位 5 位のラベルの中に正解のラベルが含まれない確率の平均である top-5 error を指標として用いている. 本研究でも評価指標に用いている.

5.1　行列データのサイズ削減量

5.1.1　階層型行列

階層型行列を用いる場合, 行列要素や要素相関関係に行列の圧縮が大きく依存する. 分割手法を用いる場合でも, 0 要素のパディングする手法でも, 大幅な圧縮が効く場合と, 逆に行列のサイズが大きくなる場合が混在した. SPRINT で実際に学習時に発生する行列の例を用いて, 結果に基づいたその考察をする.

以下の 2 つの図は, SPRINT 内の行列に対し, 同じパラメータ条件の下, 階層型行列へ変換した結果である. 赤のブロックが密なブロックとなっており, 緑の部分が疎なブロックである. 数字はそのブロックのランク数である. 図 7 のように, 対角成分に密行列が集中し, 密要素がほぼ対角に集まっていることがわかるが, 図 8 のように, 対角以外に密行列が現れると, 圧縮がかからなくなる.

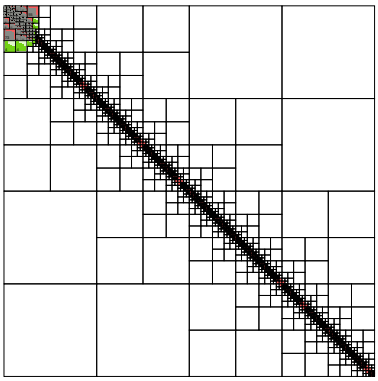

図 7　圧縮率 <100%

圧縮率は, 階層型行列へ変換後の行列に対する階層型行列への変換前の値とする. 故に 100%以下であると, その行

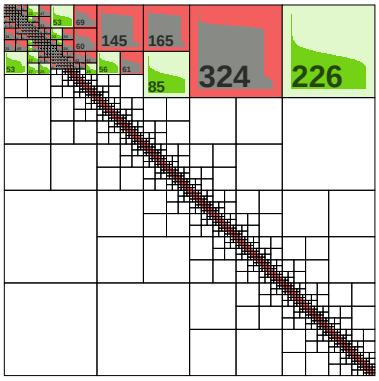

図 8　圧縮率 >100%

列は圧縮に成功したといえる.

5.1.2　SVD

SVD を用いる手法では, ランク数の削減について, 自身で設定でき, メモリ量削減を行いやすい. image 行列, weight 行列それぞれについて, 対最大特異値を基準として, 特異値を削りメモリ量を削減した行列と認識精度とのトレードオフを表した図を 5.3 節の図 9, 10 に示す.

5.2　計算時間

低ランク近似行列へ変換する際にかかる時間について, 表 6 に示す. 階層型行列の分割においては, 前章 4.3.1 の (1)-(4) までの時間, パディングにおいては, 前章 4.3.2 の (1)-(3) までの時間, SVD については, 行列の特異値分解にかかる時間を表にまとめる実行時間とする.

表 6　image 行列 (行列サイズ 465708 × 27) の低ランク近似に要する時間

手法名	実行時間 (sec)
階層型行列 (分割)	308.3083
階層型行列 (padding)	1.1212
SVD	1.1726

表 7　image 行列 (行列サイズ 460992 × 288) の低ランク近似に要する時間

手法名	実行時間 (sec)
階層型行列 (分割)	250.6730
階層型行列 (padding)	1.1986
SVD	8.2791

行列の分割により, 階層型行列を適用する手法をとると, 各ブロックごとに階層型行列を適用するので, 生成されたブロックの数に応じて, 階層型行列への変換, つまり低ランク近似に要する時間が大きくなる. 例えば表 6 では, 約 18000 個の小ブロックを生成後, 各小ブロックに対し, 階層型行列の変換を行なっているので, その部分がボトルネックとなる. パディングによる手法では, 0 パディングにより, 行列サイズは大きくなるものの, 階層型行列への変換は一回で済むので, 低ランク近似に要する時間はさほど変化しない. SVD に関しては, SVD そのものにかかる計算量は $m \times n$ 行列の場合, $\min(mn^2, m^2n)$ であるため, 行列サイズの変化に伴い, 計算時間も大幅に増えてしまう.

5.3　認識精度

image 行列と weight 行列に対し, SVD を適用し, 下位の特異値から落としていき, メモリ使用量を削減した. 以下, SVD 使用時のデータ量削減率とエラー率を図 9, 10 に示す.

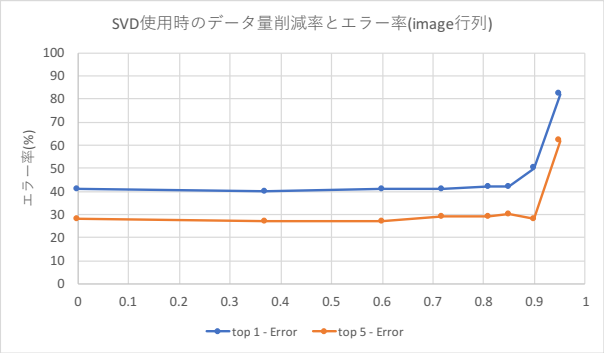

図 9　SVD 使用時のデータ量削減率とエラー率 (image 行列)

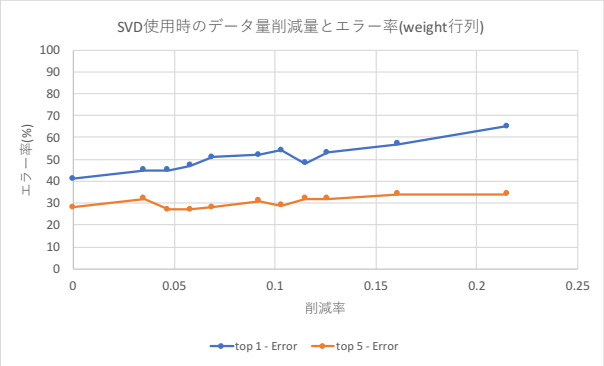

図 10　SVD 使用時のデータ量削減率とエラー率 (weight 行列)

SVD 使用時にデータ量とエラー率のトレードオフを図りつつ, ランクを落としていった結果, image 行列が最大で, 約 9 割程のメモリ削減が効いたのに対し, weight 行列では, image 行列ほどのメモリ量削減が見込まれなかった. そこで, この結果を考察するため, Validation 時の行列について, 特異値の分布について説明する.

特異値の分布として, image の方が上位数%の特異値 weight が比較的特異値はなだらかに落ちていくことがわかる. (図 11, 12) image 行列がサイズの割に情報量が少ない行列であり, 圧縮が効きやすい行列であったといえる. そのため, 最大特異値を基準とし, 特異値を削っていった場合に, image 行列の方が対最大特異値による他の特異値削減に影響をもたらす.

ⓒ 2017 Information Processing Society of Japan

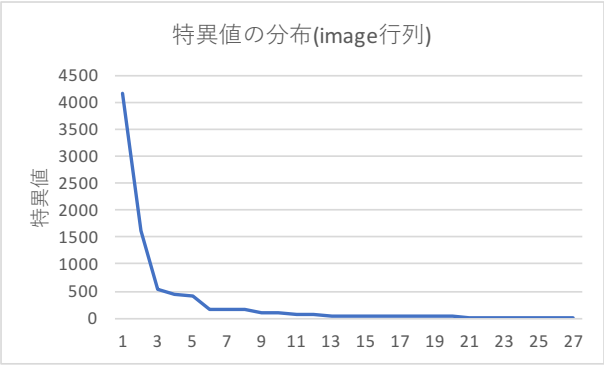

図 11 特異値の分布 (image 行列)

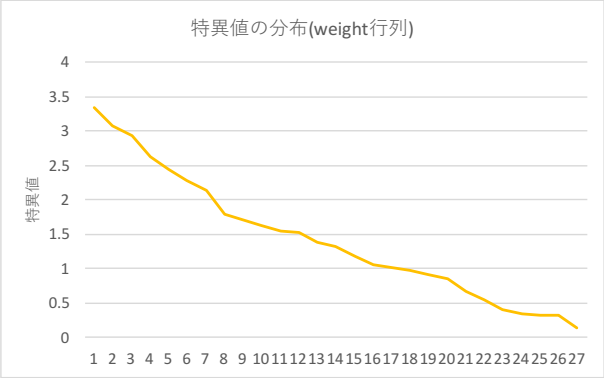

図 12 特異値の分布 (weight 行列)

6. まとめと今後の課題

本報告では, DL システム SPRINT の畳み込み計算時に低ランク近似を適用し, DL の計算に必要とされるメモリ量を削減す ることを主眼とした. 本研究では, 2 種類の低ランク近似行列を用いて, 行列積演算時のメモリ削減を図り, 特に, SVD を用いた時, 圧縮率と精度とのトレードオフにおいて, 認識精度をほとんど落とさず, サイズが特に大きい image 行列で最大約 9 割程のメモリ量削減に成功した.

今後の課題として, 現在ボトルネックとなっている, 低ランク近似行列への変換時間について, アプリケーション全体で初めから低ランク近似行列を用いるように書き換え等を行い, 密行列から低ランク近似行列変換を省略可能とすることが必要となってくると考えられる. また, 本研究では, SVD において認識精度をほぼ落とさず, 大幅なメモリ削減を達成したが, 他の低ランク近似手法の検討やより高速なライブラリ等の実装も含めた, アプリケーション全体での高速化について, 議論していく必要がある.

謝辞 本研究の一部は, JST, CREST の支援を受けたものである.

参考文献

[1] Krizhevsky, I. Sutskever, and G. Hinton.: ImageNet classification with deep convolutional neural networks, In NIPS, 2012.
[2] Harris, M.: New Features in CUDA 7.5. https://devblogs.nvidia.com/parallelforall/new-features-cuda-7-5/.
[3] G. Blellcoh and G. Narlikar.: A practical comparison of n-body algorithms, Technical report, Wright Laboratory, 1991.
[4] UNC Chapel Hill Vision Lab. ILSVRC2015 Results. http://image-net.org/challenges/LSVRC/2015/results
[5] ILSVRC2016: http://image-net.org/challenges/LSVRC/2016/
[6] HLIBpro: http://www.hlibpro.com/
[7] Steffen Börm, Lars Grasedyck, Wolfgang Hackbusch.: Hierarchical Matrices, April 21, 2005.
[8] R. Yokota, Fast Multipole Method as a Matrix-free Hierarchical Low-rank Approximation, International Workshop on Eigenvalue Problems, Tsukuba, Japan, 14-16 September, 2015.
[9] Preetum Nakkiran, Raziel Alvarez, Rohit Prabhavalkar, Carolina Parada.: Compressing Deep Neural Networks using a Rank-Constrained Topology, Proc. of Interspeech, 2015, pp. 1473-1477.
[10] Tachioka Y,Watanabe S, Le Roux J, Hershey J.: Sequence Discriminative Training for Low-Rank Deep Neural Networks.
[11] Mario Bebendorf.: Approximation of boundary element matrices, Numetishe Mathematik, 86(2000), pp. 565-589.
[12] LAPACK: http://www.netlib.org/lapack/
[13] MAGMA: http://icl.cs.utk.edu/magma/software/

ⓒ 2017 Information Processing Society of Japan