Vol.2015-ARC-217 No.8
2015/10/8

畳み込みニューラルネットワークにおける数値表現と分類精度に関する調査

成子　貴洋[1,a)]　平木　敬[1,b)]

概要：畳み込みニューラルネットワーク (Convolutional Neural Network, CNN) は画像認識アプリケーションに適した機械学習モデルである．CNN は，入力画像とそれが属するクラスのラベルからなる訓練データを基に，自身のパラメータを学習する．学習が完了した CNN は画像の分類器として機能する．計算機的側面から CNN を観察すると，CNN は膨大な浮動小数演算と高い並列性を有することが分かる．この高い並列性を活用して分類に要するレイテンシを削減するために，CNN を FPGA によって高速化する研究が行われてきた．FPGA は浮動小数点数の演算器を持たないため，固定小数点数を数値表現に用いるのが適当である．固定小数点数に何ビットを割り当てるかに関して，必要な FPGA リソースと精度はトレードオフの関係にある．入力画像から分類結果を得る feed forward パスの演算に何ビットの固定小数点数が必要であるかは，先行研究で明らかになっている．しかし，学習を含めた CNN の演算に，何ビットの固定小数点数が必要であるかに関しては十分な知見が得られていない．本稿では，固定小数点数のビット数と丸め方法が CNN の分類性能に与える影響について調査する．

1. はじめに

ニューラルネットワーク (Neural Network, NN) は動物の脳の仕組みを基にした機械学習モデルである．NN は，入力 x とそれに対応する出力 $f(x)$ のペアからなる訓練データから，関数 f を近似するパラメータを学習する．NN にはいくつかのバリエーションが存在するが，畳み込み層を持つものを特に畳み込みニューラルネットワーク (Convolutional Neural Network, CNN) と呼ぶ．CNN は図 1 のような構成となっており，画像認識において高い分類性能を示すことが知られている [9]．

CNN の計算的側面の特徴として，高い並列性を有する点が挙げられる．なぜならば，同一レイヤ内の各ニューロンの出力は，それぞれ独立に計算できるからである．アクセラレータによりこの高い並列性を活用することで，CNN を高速化することが可能である．アクセラレータには，GPGPU，ASIC[2], [3] に加え，FPGA[4], [5], [12] が用いられる．FPGA は数百個の DSP ブロックを搭載しており，独立な積和演算を超並列に実行することができる．故に FPGA は，多数の独立なベクトル内積演算から成り立つ CNN と親和性が高い．

[1] 東京大学
Hongo, Bunkyo-ku, Tokyo 7-3-1, Japan
[a)] cinccinaru@is.s.u-tokyo.ac.jp
[b)] hiraki@is.s.u-tokyo.ac.jp

FPGA で NN の高速化に取り組む先行研究では，数値表現に固定小数点数が採用されている [4], [5]．FPGA は FPU を持たないため，FPGA 上で浮動小数点数を扱うのはコストが高いからである．固定小数点数は図 2 のように符号部，整数部，小数部から成り，小数点は静的に固定されている．固定小数点数の演算器は整数と同様の回路で実現できるため，DSP ブロックを効果的に利用することができる．整数部・小数部に各々何ビットを割り当てるかは設計者が自由に設定できる．より多くのビットを割り当てれば，より広い範囲・細かい粒度で数値を表現できる一方で，FPGA リソースやメモリがより多く必要となる．逆に，少ないビットを割り当てれば，ハードウェアリソースを節約できる一方で，精度が犠牲となる．よって，CNN における固定小数点数の最適なビット数は自明ではない．

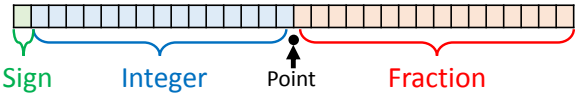

図 2　固定小数点数の構成．

固定小数点数のビット長と NN の分類精度の関係について報告した先行研究は存在する [1], [2], [6], [7], [10]．[7], [10] は小規模な多層パーセプトロンにおける評価を行ったものであり，現在用いられるような深い CNN を想定していない．学習はオフラインに double で行い，入

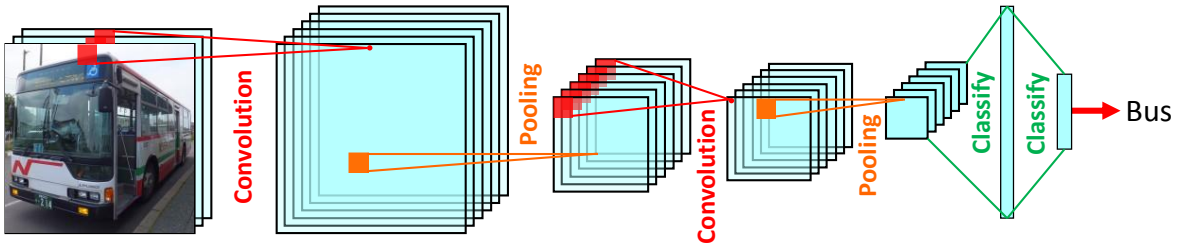

図 1　畳み込みニューラルネットワークの構成例.

力から分類結果を得る feed-forward パスの演算のみ固定小数点数で実行する場合を想定した研究には [1] が挙げられる．Chen らは，MNIST データセットを用いて実験した結果，ASIC 内部の数値表現に 16 ビットを採用している [1]．feed-forward パスに加え，学習を含めて固定小数点数で実行する場合を想定した研究には [2], [6] が挙げられる．Chen らは，MNIST データセットの学習においては，32 ビットの固定小数点数が必要であると主張している [2]．Gupta らは，丸めにおいて乱数を利用することで，固定小数点数のサイズを 16 ビットに抑えつつも，分類精度を保つ手法 (Stochastic Rounding) を提案した [6]．Stochastic Rounding は最近接丸めよりも良い分類性能を示すものの，小数点の位置はデータセット毎にアドホックに決定する必要がある．さらに，[6] で実験に用いたデータセット，トポロジ，パラメータ以外においても 16 ビットで十分であるかは分かっていない．

本稿では，固定小数点数のビット数と丸め方法が CNN の分類性能に与える影響について，[6] とは異なる条件で調査する．[6] では 12 ビットと 14 ビットの小数部について実験している．本稿の実験では，小数部を 1 ビットから 20 ビットまで 1 ビットずつ変化させながら，ビット数が分類性能に与える影響を詳細に観察する．先行研究 [6] では最近接丸めと Stochastic Rounding について調査が行われた．本稿ではこれらに加え，丸め方法として切り捨てと切り上げを対象とする．切り捨てを対象とするのは，下位ビットを捨てるだけで実装可能な最も簡単な丸めだからである．切り上げを対象とするのは，微小量が 0 に丸められるケースが排除できるからである．

2. 丸め

2 つの固定小数点数 a, b のビット列をそれぞれ A, B とする．f を小数部のビット数とすると，$A = a*2^f$, $B = b*2^f$ である．このとき，a, b の加減乗算は，それぞれ以下の様に計算できる．

$$a + b = (int)A + (int)B$$
$$a - b = (int)A - (int)B$$
$$a * b = r((int)A * (int)B) >> f$$

ただし，$r(n)$ は整数 n の下位 f ビットを丸める関数である．本稿では，以下の 4 通りの丸め関数を評価対象とする．

切り捨て

$r(n) = n$

切り上げ

$r(n) = n + (2^f - 1)$

最近接丸め

$r(n) = n + 2^{f-1}$

Stochastic Rounding[6]

$(int)A*(int)B$ の下位 f ビットを e とする．Stochastic Rounding では，確率 $p = e/2^f$ で切り上げ，確率 $1-p$ で切り捨てを行う．よって，

$r(i) = i + s$ ただし s は $[0, 2^f)$ の範囲における一様乱数.

本稿では，ハードウェアでの実現性を考慮し，乱数には Linear Feedback Shift Register (LFSR) を用いる．以下の様に生成される 32 ビット LFSR の下位 f ビットを s として利用する．

```
static unsigned int lfsr32 = 0;
unsigned int bit0 =
    ((lfsr32 >>  0) & 1) ^
    ((lfsr32 >>  1) & 1) ^
    ((lfsr32 >> 21) & 1) ^
    ((lfsr32 >> 31) & 1) ^ 1;
lfsr32 = ((lfsr32 << 1) & 0xFFFFFFFE) | bit0;
```

固定小数点数の演算においては，飽和演算を仮定する．つまり，加減乗算によりオーバーフロー・アンダーフローは発生させず，想定する固定小数点数で表現しうる最大値・最小値を計算結果とする．

FPGA の DSP は内部にアキュムレータを持つ．アキュムレータのサイズは，乗算器の入力サイズに対して十分大きい．例えば Xilinx 社の DSP48E1[11] では，乗算器の入力長が 18 ビット * 25 ビットであるのに対して，アキュムレータは 48 ビット長である．したがって，本稿においては，ベクトル内積計算の部分積・部分和の演算では丸めや飽和処理を行わず，全ての加減乗算が完了した段階でそれらの処理を行うと仮定する．つまり，$\sum_i r(x_i w_i)$ でなく，$r(\mathbf{x} \cdot \mathbf{w})$ を内積の計算時に用いる．

表 1　MNIST の学習に用いる CNN の構成．ただし，M_i, R_i, C_i, K_r, K_c, M_o, R_o, C_o はそれぞれ入力のマップ数・縦幅・横幅，カーネルの縦幅・横幅，出力のマップ数・縦幅・横幅を表す．

Type	(M_i, R_i, C_i)	(K_r, K_c)	(M_o, R_o, C_o)
INPUT			(1, 28, 28)
CONV	(1, 28, 28)	(5, 5)	(20, 24, 24)
POOL	(20, 24, 24)	(2, 2)	(20, 12, 12)
CONV	(20, 12, 12)	(5, 5)	(50, 8, 8)
POOL	(50, 8, 8)	(2, 2)	(50, 4, 4)
CLASS	(50, 4, 4)		(500, -, -)
OUTPUT	(500, -, -)		(10, -, -)

表 2　MNIST の学習に用いる CNN のパラメータ．Random Range は重みとバイアスの初期値の範囲．

Learning Rate	0.001
Batch Size	1 (Incremental Learning)
Random Range	$[-0.1, 0.1]$

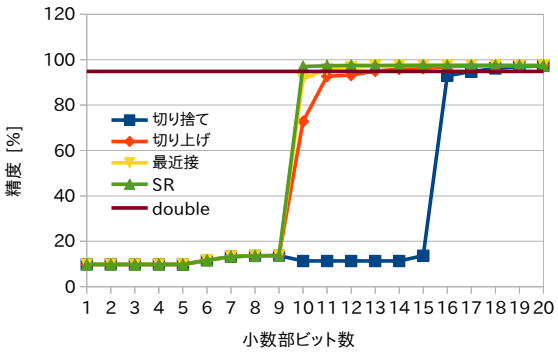

図 3　小数部のビット数，並びに丸め方法と精度の関係．

3. 評価

3.1 評価手法

評価には，C++で作成した CNN の数値型を，自作の固定小数点数クラスで置き換えたプログラムを用いる．固定小数点数の整数部・小数部のビット数は，実行時にコマンドライン引数で指定することができる．重みとバイアスの初期値は，指定された範囲の一様乱数により決定される．異なる 5 通りの乱数シードについて評価を行い，それらの分類精度の相加平均を結果とする．

現時点では，データセットに手書き数字認識の MNIST を用いる．MNIST は 60000 の訓練データと 10000 のテストデータから構成される．学習フェーズでは，訓練データを一巡だけ与える．学習後にテストデータを与え，正しく分類されたケースの割合を CNN の精度とする．CNN の構成とパラメータは表 1, 2 に示す．

データセットに CIFAR-10[8] を用いた評価は準備中である．

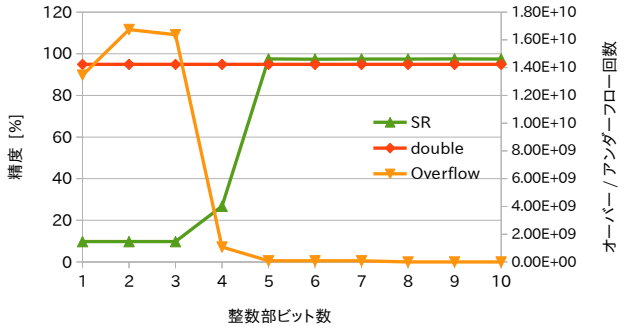

図 4　丸めに Stochastic Rounding を用いた場合の，整数部のビット数と精度の関係．

3.2 MNIST における結果

小数部のビット数を変化させた時の結果を，図 3 に示す．各丸め方法における結果に加え，ベースラインとして double を用いた場合の結果を示す．整数部は 32 ビットで固定した．小数部のビット数は，表現できる数値の粒度を決定する．

いずれの丸めにおいても，小数部のビット数が増加するのに伴い，分類精度が向上する傾向が分かる．小数部が 9 ビット以下の範囲では，いずれの丸め方法を利用した場合であっても，学習が正しく行えていないことが読み取れる．小数部が 10 ビットの場合には，最近接と Stochastic Rounding (SR) において double と互角の性能が得られている．最近接の 91%に対し，SR は 97%の精度が得られており，SR の方がより丸め誤差に強い手法であるといえる．切り上げについては，10 ビットでは十分な性能は得られないものの，9 ビットの場合と比較して大きな精度向上が確認できる．切り下げを用いる場合には，最低 16 ビット必要である．

整数部のビット数を変化させた時の結果を，図 4 に示す．このビット数には符号ビットが含まれる．丸め方法には Stochastic Rounding を用いる．小数部分は 32 ビットで固定した．整数部のビット数は，表現できる数値の範囲を決定する．

グラフより，整数部は最低 5 ビット必要であることが分かる．逆三角の線は学習フェーズにおいて発生したオーバーフロー・アンダーフローの回数を表す．これらの回数が初めて 0 になるのは，7 ビットの整数部においてである．5 ビット・6 ビットの整数部は double と同等の分類性能を示すものの，学習において 8.57×10^7 回のオーバーフロー・アンダーフローが発生している．

以上の議論より，本実験で用いた CNN については，Stochastic Rounding を用いる場合に必要な固定小数点数のフォーマットは $< 5, 10 >$ である．ただし，$< i, f >$ は整数部，小数部がそれぞれ i, f ビットの固定小数点数を表す．Gupta らが MNIST データセットについて行った実験では，ニューロンの出力に $< 6, 10 >$，その他の数値表現に $< 2, 14 >$ が必要であった [6]．本実験の結果は，先行研

究 [6] の結果とほぼ一致したといえる．一方で，最近接丸めを用いた場合の結果については，本実験においては 11 ビットで収束が見られたのに対し，先行研究 [6] では 14 ビットでも収束に至らない．この結果から，固定小数点数に必要なビット数は，用いる CNN のトポロジやパラメータにより変動すると考えられる．

4. まとめと今後の課題

本稿では，数値表現に用いる固定小数点数のビット数と丸め方法が，CNN の分類精度に与える影響について調査した．本研究では，先行研究 [6] で対象とされた最近接丸めと Stochastic Rounding に加え，切り下げと切り上げを丸め方法の評価対象とした．さらに，1 ビット刻みで整数部・小数部のビット数を変化させ，分類精度の調査を行った．実験結果から，MNIST データセットに対しては，整数部に 5 ビット，小数部に 10 ビット，それぞれ必要であることが確かめられた．さらに，Stochastic Rounding は，本稿で扱った 4 つの丸め方法の中で最も良い分類性能を示すことが確かめられた．切り捨ては下位ビットを捨てることで実現できるため，最も実装が簡単な丸めであるが，性能が大きく劣ることが明らかとなった．本実験においては，最近接丸めを用いた場合は，11 ビットの小数部で十分であった．しかし，この結果は 14 ビットの小数部では収束しないとする先行研究の結果 [6] に反する．このことは，想定する CNN の構成やパラメータにより数値表現に求められる精度にばらつきが存在することを示唆している．

故に，様々なパラメータやデータセットについて評価を行い，知見を得ることが重要である．現在，CIFAR-10 を用いた評価を準備中である．

参考文献

[1] Chen, T., Du, Z., Sun, N., Wang, J., Wu, C., Chen, Y. and Temam, O.: DianNao: A Small-footprint High-throughput Accelerator for Ubiquitous Machine-learning, *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, New York, NY, USA, ACM, pp. 269–284 (online), DOI: 10.1145/2541940.2541967 (2014).

[2] Chen, Y., Luo, T., Liu, S., Zhang, S., He, L., Wang, J., Li, L., Chen, T., Xu, Z., Sun, N. and Temam, O.: DaDianNao: A Machine-Learning Supercomputer, *Microarchitecture (MICRO), 2014 47th Annual IEEE/ACM International Symposium on*, pp. 609–622 (online), DOI: 10.1109/MICRO.2014.58 (2014).

[3] Du, Z., Fasthuber, R., Chen, T., Ienne, P., Li, L., Luo, T., Feng, X., Chen, Y. and Temam, O.: ShiDianNao: Shifting Vision Processing Closer to the Sensor, *Proceedings of the 42Nd Annual International Symposium on Computer Architecture*, ISCA '15, New York, NY, USA, ACM, pp. 92–104 (online), DOI: 10.1145/2749469.2750389 (2015).

[4] Farabet, C., Martini, B., Corda, B., Akselrod, P., Culurciello, E. and LeCun, Y.: NeuFlow: A runtime reconfigurable dataflow processor for vision, *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pp. 109–116 (online), DOI: 10.1109/CVPRW.2011.5981829 (2011).

[5] Gomperts, A., Ukil, A. and Zurfluh, F.: Development and Implementation of Parameterized FPGA-Based General Purpose Neural Networks for Online Applications, *Industrial Informatics, IEEE Transactions on*, Vol. 7, No. 1, pp. 78–89 (online), DOI: 10.1109/TII.2010.2085006 (2011).

[6] Gupta, S., Agrawal, A., Gopalakrishnan, K. and Narayanan, P.: Deep Learning with Limited Numerical Precision, *CoRR*, Vol. abs/1502.02551 (online), available from ⟨http://arxiv.org/abs/1502.02551⟩ (2015).

[7] Holi, J. and Hwang, J.-N.: Finite precision error analysis of neural network hardware implementations, *Computers, IEEE Transactions on*, Vol. 42, No. 3, pp. 281–290 (online), DOI: 10.1109/12.210171 (1993).

[8] Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images (2009).

[9] Krizhevsky, A., Sutskever, I. and Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks, *Advances in Neural Information Processing Systems 25* (Pereira, F., Burges, C., Bottou, L. and Weinberger, K., eds.), Curran Associates, Inc., pp. 1097–1105 (online), available from ⟨http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf⟩ (2012).

[10] Savich, A., Moussa, M. and Areibi, S.: The Impact of Arithmetic Representation on Implementing MLP-BP on FPGAs: A Study, *Neural Networks, IEEE Transactions on*, Vol. 18, No. 1, pp. 240–252 (online), DOI: 10.1109/TNN.2006.883002 (2007).

[11] Xilinx Inc.: 7 Series DSP48E1 Slice User Guide, `http://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf` (2014).

[12] Zhang, C., Li, P., Sun, G., Guan, Y., Xiao, B. and Cong, J.: Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks, *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '15, New York, NY, USA, ACM, pp. 161–170 (online), DOI: 10.1145/2684746.2689060 (2015).