量子化誤差を考慮したニューラルネットワークの学習手法

Quantization Error-aware Neural Network Training

廣瀬 一俊[1*]　安藤 洸太[1]　植吉 晃大[1]　池辺 将之[1]
浅井 哲也[1]　本村 真人[1]　高前田 伸也[1]
Kazutoshi Hirose[1], Kota Ando[1], Kodai Ueyoshi[1], Masayuki Ikebe[1],
Tetsuya Asai[1], Masato Motomura[1], and Shinya Takamaeda-Yamazaki[1]

[1] 北海道大学大学院情報科学研究科
[1] Graduate School of Information Science and Technology (IST), Hokkaido University

Abstract: Deep neural network is a widely-used technology for various machine learning applications. A training technology for both low-precision and high-accuracy is desired for low power neural network hardware. We propose a quantization-error-aware training method for higher accuracy of quantized neural networks. Our approach appends an additional regularization term, based on quantization errors of weights, to the loss function. The evaluation results on MNIST and CIFAR-10 show that the proposed approach achieves higher accuracy than the standard approach.

1　はじめに

Deep neural network (DNN) は機械学習の技術として幅広く用いられており、画像認識、音声認識 [9]、翻訳 [6] 等といった場面で使用されている。DNN は、ネットワークの大規模化や複雑化により従来の機械学習よりも高度なタスクが可能となったが、莫大な計算資源を必要とする。サーバーの処理では、推論と学習が行われ、主に GPU が使用される。高性能である反面、多くの電力を消費する。一方で、携帯端末や組み込み機器といった将来の IoT デバイスでは、限られた環境で動作しなければならない。そのため、低電力・省メモリで動作するハードウェアに特化した DNN 技術が求められている。

ハードウェア指向の DNN 技術の 1 つとして、数値の量子化が挙げられる。量子化は浮動小数で表現されている数値を固定小数 [11] や対数 [12]、バイナリ [2] といった表現で表すことである。量子化された値はメモリ量を削減し、演算を単純にすることが可能となる。例えば、バイナリ化された重み係数とアクチベーションの計算は乗算が不要になり、代わりに XNOR 演算に置き換えられる。XNOR 回路は乗算回路に比べ、非常に簡単な回路であるため、消費電力は大幅に削減される [1]。

しかし、量子化を行うと、認識精度を下げてしまうといった問題が起こる。この原因は、浮動小数で表現されている値を量子化することで発生する量子化誤差による。一般的に重み係数の表現精度の粗さと認識精度にはトレードオフの関係があり、量子化するほど認識精度が下がる傾向にある。そのため、認識精度を保ったまま、いかに重み係数の表現精度を落として量子化できるかが課題となっている。

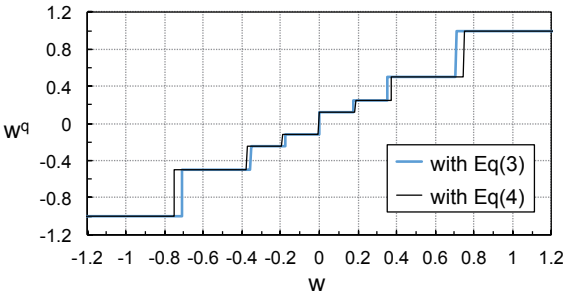

図 1: 対数量子化の適用 ($LogQuant(w, 3, 1)$ の例)

本研究では、量子化ニューラルネットワークにおいて、より認識精度を高めるために量子化誤差を考慮した学習手法を提案する。重み係数の量子化誤差は、認識精度の低下を防ぐため、小さくすることが望まれる。本手法は、量子化誤差に基づく正則化項を目的関数に取り入れて学習を進めることで量子化誤差を小さくする。

*連絡先：北海道大学大学院情報科学研究科
〒 060-0814 北海道札幌市北区北 14 条西 9 丁目
情報棟 (M棟)2F 集積アーキテクチャ研究室
E-mail: hirose@lalsie.ist.hokudai.ac.jp

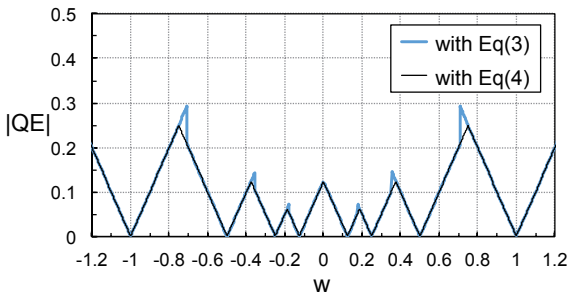

図 2: 対数量子化適用時の量子化誤差 ($|QE|$)

2 量子化ニューラルネットワーク

ニューラルネットの量子化は浮動小数で表現される値を固定小数やバイナリ化 (二値 ($\pm$ 1)) といった少ない情報量で表現することである。近年では、さらに対数を用いて数値を表現する対数量子化を適用したニューラルネットが提案されている。対数量子化を用いることによって、表現できる数値の領域の広さと小さい値の解像度を同時に高めることが可能となる。本研究では、重みの量子化手法として対数量子化とバイナリ化を取り上げる。ただし、アクチベーションの量子化については考慮しないこととする。

はじめに、元の実数表現の重み係数を w とすると、対数量子化は次式で表される。

$$AP2(w) \quad = \quad sign(w) \times 2^{round(log_2|w|)} \tag{1}$$

この $AP2(\cdot)$ は approximate-power-of-2 の頭文字をとったもので、最も近い 2 のべき乗に近似する演算である。この演算を用いると浮動小数で表現されている値を対数量子化することができる。さらに、1 つの値を表現するために必要なビット幅 (符号ビットを含む) を $bitwidth$、このビット幅で表現できる最大値と最小値をそれぞれ $maxV$ と $minV$ としたとき、次式を用いてビット制約をかける。

$$\begin{aligned} & LogQuant(w, bitwidth, maxV) \\ = \quad & Clip(AP2(w), minV, maxV) \end{aligned} \tag{2}$$

式 (1) には round 演算が含まれている。一般的には round 演算は以下の四捨五入が用いられる。

$$round(x) = \begin{cases} ceil(x) & (x - \lfloor x \rfloor \geq 0.5) \\ floor(x) & (x - \lfloor x \rfloor < 0.5) \end{cases} \tag{3}$$

この round 演算は中央値が 0.5 であるため実数領域での量子化誤差を最も減らすことができる。しかし、対数領域での中央値は 0.5 ではなく、$\log_2(\frac{3}{2})$ となる。そのため式 (1) では次式を用いることによって量子化時の誤差が減らすことができる。

$$round(x) = \begin{cases} ceil(x) & (x - \lfloor x \rfloor \geq \log_2(\frac{3}{2})) \\ floor(x) & (x - \lfloor x \rfloor < \log_2(\frac{3}{2})) \end{cases} \tag{4}$$

上式を用いた対数量子化時の誤差を図 2 に示す。また、バイナリ化には次の式を用いる。

$$Binarize(w) = sign(w) = \begin{cases} +1 & (if\ w \geq 0) \\ -1 & (otherwise) \end{cases} \tag{5}$$

3 量子化誤差に基づく正則化

これまでの量子化ニューラルネットワークは、数値を量子化するときに発生する量子化誤差 (QE;Quantization Error) を考慮していない。本研究では、量子化誤差を考慮したニューラルネットワークの学習手法を提案する。

QE の発生による認識精度の低下を抑えるためには、QE が小さくなるように重みを学習すれば良い。ここで QE を正則化項として目的関数に付加する。この目的関数について重みの学習を進めることで、QE と認識誤差を同時に小さくし、認識精度を保つことが可能となる。

実数表現の重みを w, 量子化 (対数量子化およびバイナリ化) 後の重みを w_q としたとき、量子化誤差は次式で定義される。

$$QE(w) = w - w^q \tag{6}$$

そして量子化誤差に基づく正則化 (QER;Quantization Error-based Regularization) 項を以下のように定義する。

$$QER(w) = \|w - w^q\|_2 \tag{7}$$

さらに認識誤差関数を $E(w)$ としたとき、目的関数に QER 項を加えて以下のように定義する。

$$L(w) = E(w) + \eta_2 QER(w) \tag{8}$$

この目的関数を次式のように最小化する方向へ学習を進めることで、認識誤差と QE が同時に小さくなる。

$$\min_w L(w) \tag{9}$$

一般的なニューラルネットワークでは、過学習を防ぐために重みの L2 ノルムという正則化項を用いることがある [5]。L2 正則化は重みが 0 に近づくように働くため、重みの発散を防ぐことができる。一方で、本提案の QER は実数表現の重みが量子化後の重みに近づくように働く。そのため、量子化誤差が小さくなり、認識誤差を抑えることが可能となる。

Algorithm 1 QER を適用した量子化ニューラルネットワークの学習

Require: a minibatch of inputs and targets (x_0, x^*), previous weights w, previous learning rate η^t.

Ensure: updated weights w^{t+1}, updated learning rate η^{t+1}

1. Forward propagation

for $l = 1$ to L **do**

$w_l^q \Leftarrow \mathrm{Quantize}(w_l)$

$u_l \Leftarrow x_{l-1} \cdot w_l$

if $l < L$ **then**

$x_l \Leftarrow \mathrm{ReLU}(u_l)$

end if

end for

2. Backward propagation

Compute $\frac{\partial E}{\partial u_L}$ knowing u_L and x^*

for $l = L$ to 1 **do**

$\frac{\partial E}{\partial u_{l-1}^q} \Leftarrow \frac{\partial E}{\partial u_l} \cdot w_l^q$

$\frac{\partial E}{\partial w_l} \Leftarrow \frac{\partial E}{\partial u_l}^T \cdot u_{l-1}^q$

end for

3. Accumulating the parameter gradients

for $l = 1$ to L **do**

$\frac{\partial QER(w_l)}{\partial w_l} \Leftarrow QE(w_l)$

$w_l^{t+1} \Leftarrow w_l - \eta_1^t \cdot \frac{\partial E}{\partial w_l} - \eta_2^t \cdot \frac{\partial QER(w_l)}{\partial w_l}$

$\eta^{t+1} \Leftarrow \lambda \eta^t$

end for

4 評価

本節では、QER 項を用いず通常の学習を行った場合と、QER 項を目的関数に付加して学習を行った場合の評価を行った。評価には機械学習フレームワーク TensorFlow を用いた。使用した数値表現は上記までと同様に、対数量子化とバイナリ化である。対数量子化では、すべての層における重みの量子化を $LogQuant(w, 4, 1)$ とした。学習は Algorithm 1 に則って行う。ここでの学習率 η_1 は 0.001 とした。また η_2 は初期値を 0.00001 とし、10 エポック毎に 1.2 倍と設定した。この係数を用いることで、学習開始直後は認識誤差を重視し、学習終盤は量子化誤差を重視するように重みが更新される。学習の最適化には Adam[7] を使用した。また、学習に使用したデータセットは次の 2 つである。

1. MNIST[10] は 28×28 のグレースケール画像で構成されており、0 から 9 までの手書き数字画像のデータセットである。学習用の画像 6000 枚と評価用の画像 10000 枚の 70000 枚がある。データアーギュメントは使用していない。学習に使用したネットワークは隠れ層 2 層含む multi-layer perceptron を使用しており、以下のものである。ここでの FC は全結合層を示す。

   FC728-256, FC256-256, FC256-10

2. CIFAR-10[8] は 32×32 のカラー画像で構成されており、10 クラス分類の画像のデータセットである。学習用の画像 50000 枚と評価用の画像 10000 枚の 60000 枚がある。データアーギュメントは使用していない。学習に使用したネットワークは CNN(convolutional neural network) を使用しており、以下のものである。ここでの C3-X は 3×3 フィルターを用いた X チャネル出力の畳込み層、MP2 は max-pooling 層を示す。

   C3-64, MP2, C3-64, MP2,
   FC4096-384, FC384-192, FC192-10

表 1: 各手法の認識精度

		MNIST	CIFAR-10
float		0.9777	0.6941
LogQuantize (4bit)	w/o QER	0.9773	0.6844
	w/ QER	0.9783	0.7031
Binarize (1bit)	w/o QER	0.9664	0.6724
	w/ QER	0.9709	0.6839

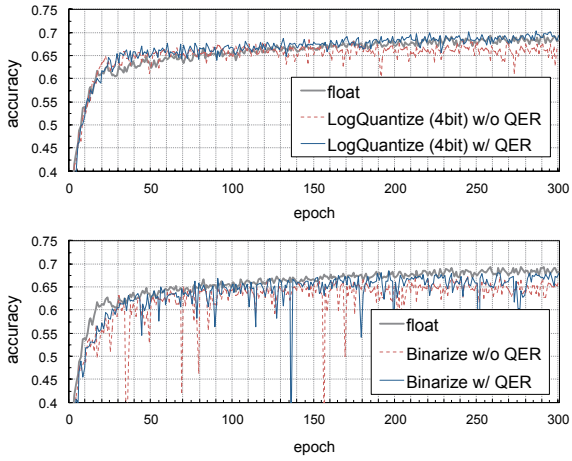

図 3: CIFAR-10 での認識精度の収束

表 1 に認識精度の結果を示し、図 3 にそれらの収束状況を示す。MNIST と CIFAR-10 の両方のベンチマークで、対数量子化、バイナリ化のどちらの量子化法においても QER 項を含めて学習をすることで認識精度が向上した。特に LogQuantize(4bit) の評価では QER を適用することで float の認識精度を上回る結果となった。

5 関連研究

重み係数を量子化してハードウェアに適した形に圧縮する手法が提案されている。Shin らは LUT(Look Up Table) を使うため重みを圧縮した [13]。Gysel らはハードウェア指向の固定小数表現の重みにするためのファインチューニング技術を提案した [3]。これらの手法は限られた数値表現での認識精度を高めるために重みを最適化している。我々の研究は、量子化誤差を考慮しているという点でこれらの研究とは異なる。しかし、これらの手法と同時に使用することが可能である。

Loss-aware binarization[4] はバイナリ化された重みに対する損失を直接最小にするために、Diagonal Hessian Approximation を用いた proximal Newton algorithm を採用している。我々の研究は量子化時の影響を考慮しているという点で同じである。我々は、正則化項を用いて認識精度を上げることを目的としているが、バイナリ化以外の量子化にも適用可能である。

6 まとめ

本研究では、ニューラルネットのハードウェア実装に向け、量子化誤差を考慮した学習手法を提案した。量子化誤差に基づく正則化項を目的関数に付加することで、量子化誤差と認識誤差を小さくなるように学習を行うことが可能となる。

今後の課題として、より大規模なネットワークへの適用や、対数量子化やバイナリ化といった量子化法だけでなく、線形量子化への適用をし、評価を行う必要がある。また、正則化項の導入には係数を用いるため、この係数の動的最適化技術が考えられる。

謝辞

本研究は JST ACCEl 及びテクノバの助成を受けたものである。

参考文献

[1] Ando, K., Orimo, K., Ueyoshi, K., Yonekawa, H., Sato, S., Nakahara, H., Ikebe, M., Asai, T., Takamaeda-Yamazaki, S., Kuroda, T., Motomura, M.: BRein Memory: A 13-layer 4.2 K neuron/0.8 M synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm cmos. In: 2017 IEEE Symposium on VLSI Circuits (VLSI-Circuits). pp. C24–C25. Kyoto, Japan (2017)

[2] Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. ArXiv e-prints (Feb 2016)

[3] Gysel, P., Motamedi, M., Ghiasi, S.: Hardware-oriented Approximation of Convolutional Neural Networks. ArXiv e-prints (Apr 2016)

[4] Hou, L., Yao, Q., Kwok, J.T.: Loss-aware Binarization of Deep Networks. ArXiv e-prints (Nov 2016)

[5] Janocha, K., Czarnecki, W.M.: On Loss Functions for Deep Neural Networks in Classification. ArXiv e-prints (Feb 2017)

[6] Johnson, M., Schuster, M., Le, Q.V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., Dean, J.: Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. ArXiv e-prints (Nov 2016)

[7] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. ArXiv e-prints (Dec 2014)

[8] Krizhevsky, A., Nair, V., Hinton, G.: Cifar-10 (canadian institute for advanced research) `http://www.cs.toronto.edu/ kriz/cifar.html`

[9] LeCun, Y., Bengio, Y., Hinton, G.: Nature. Nature (2016)

[10] LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), `http://yann.lecun.com/exdb/mnist/`

[11] Lin, D.D., Talathi, S.S., Sreekanth Annapureddy, V.: Fixed Point Quantization of Deep Convolutional Networks. ArXiv e-prints (Nov 2015)

[12] Miyashita, D., Lee, E.H., Murmann, B.: Convolutional Neural Networks using Logarithmic Data Representation. ArXiv e-prints (Mar 2016)

[13] Shin, D., Lee, J., Lee, J., Yoo, H.J.: 14.2 dnpu: An 8.1tops/w reconfigurable cnn-rnn processor for general-purpose deep neural networks. In: 2017 IEEE International Solid-State Circuits Conference (ISSCC). pp. 240–241 (Feb 2017)