一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
CPSY2018-115,DC2018-97（2019-03）

車載応用向けDNNモデル軽量化の検討

平山　侑樹†　廣瀬　一俊†　早川　剛††　気屋村純一††　深谷　安利††
栗田　裕二††　安藤　洸太†　植吉　晃大†　高前田伸也†　本村　真人†

† 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
†† NEC ソリューションイノベータ株式会社　〒136–8627 東京都江東区新木場 1-18-7
E-mail: †hirayama.yuki.cp@ist.hokudai.ac.jp

あらまし　ハードウェア実装に向けた畳み込みニューラルネットワークの量子化手法について検討した。パラメータ間での量子化の度合いの関係について議論したものは少なく、高精度を保ったまま、かつ計算量を減らしたい。そこで Residual Network [1] を対象とし、活性、重み、バッチ正規化、畳み込み演算の出力それぞれに対し、個別のビット幅、スケールを設定することにより、一般にスケールの異なる値の効率的な量子化手法について提案する。独自に用意した歩行者検出のデータセットを用いてその性能を評価するとともに、CIFAR-10 による評価を行った。歩行者検出のデータセットにおいて浮動小数点での演算とほぼ変わらない精度を実現できることを示した。

キーワード　ディープニューラルネットワーク, 量子化

1. はじめに

近年、ニューラルネットワークは、誤差逆伝播法による効率的な学習法の登場などの内的要因、また計算機性能の向上などの外的要因により、様々な分野で成功を収めており、広く研究の対象となっている。中でも畳み込みニューラルネットワーク (Convolutional Neural Network;CNN) は画像のクラス分類や種別に色分けするセグメンテーション、姿勢推定、ノイズ除去、超解像、画像からの深度推定、スタイル合成など様々な分野で成功を収めている。その精度は一般にモデルの複雑さにより担保されるが、モデルの複雑さゆえ計算量は膨大なものとなる。これらの学習は高速に行列演算を行うことができる GPU を用いることで可能となった。

しかし、スマートフォン、車載向けなどのエッジコンピューティングでは、限られた計算機資源で高速に処理を行わなければならないため、モデルを単純化し計算量を少なくする必要がある。ネットワークの軽量化には量子化、重みを部分的に消去するプルーニング、圧縮など様々な手法が提案されている。本研究ではその中の量子化に焦点を当て、その学習、量子化パラメータ探索について取り上げる。量子化とは計算に使われるパラメータのビット数を削減し、計算量、メモリ消費量を削減する手法である。

車載用歩行者検出のためのベースモデルとして Residual Network(ResNet) [1] を採用し、量子化の手法を用いたモデルの軽量化を行うことが本研究の目的である。車載用であるため限られた資源の中で処理を行う必要があり、認識精度を落とさずにかつ量子化による軽量なモデルとその手法について提案する。

これまでの量子化手法においては、重みまたは活性を 2 値化したものがあるが、ここではそれらビット精度をパラメータとして扱い、ビット精度と認識精度の両立を図る。本研究では、精度を維持したままモデルを軽量化することが目的であるが、その過程においてパラメータの量子化の相互作用について明らかにする。

2. 関連研究

ニューラルネットワークの量子化として重みを 2 値化した Binary Connect [2]、重みに加え活性を 2 値化した Binarized Neural Networks [3]、重み、活性を 2 値化した上で表現力を確保するためスケーリング係数を持たせた XNOR-Net [4] などがある。重みを 2 値にすることで畳み込み層での乗算をなくすことができ、また、活性も 2 値にすることで乗算と加算を XNOR ゲートとポップカウントで実現することができる。

CNN の学習は逆伝播演算による誤差関数の重みに対する勾配計算、得られた勾配による重みの更新により行われる。しかし勾配計算において量子化を施した CNN ではその勾配はほとんどの定義域で 0 となってしまう。そこで一般に Straight Through Estimator(STE) [5] と呼ばれる手法が用いられる。STE では順伝播時の量子化関数を逆伝播演算時には区分的な線形関数とみなし、その導関数によって逆伝播演算を行うというものである。これにより勾配消失は起きず、学習が可能となる。

3. 計算コストと精度の両立のためのモデル最適化手法の提案

3.1 システム全体の構成

図 1 にシステム全体の構成を示す。歩行者検出には二つのセンサーを用いる。一つは 3D LiDAR と呼ばれる、赤外線センサーによって周囲の深度マップを得るものであり、これにより路面推定、歩行者候補の検出を行う。歩行者候補の検出をもとにカメラからの画像から歩行者を抽出する。本研究では 3D LiDAR とカメラ画像により切り取られた高さ 160、幅

128 のグレースケール画像を畳み込みニューラルネットワークの入力とし、それが人物であるか否かを判定する。

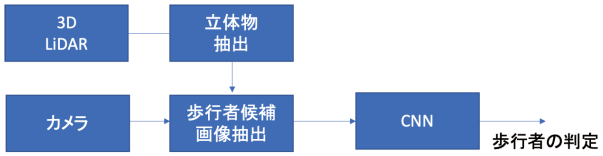

図 1　システム全体の構成

3.2　ニューラルネットワークのモデル構成

ネットワークは 14 層の ResNet をベースとした。

ResNet では Residual Block(ResBlock) と呼ばれる単位でショートカットパスが存在し、ショートカットパスの入力と出力でチャネル、特徴マップのサイズが異なる場合、畳み込み層によりこれを調整する。畳み込み層のカーネルはショートカットコネクションでは 1*1、それ以外では 3*3 とし、ストライドは block2, block3 の最初のみ 2、それ以外は 1 とした。これは活性化関数には ReLU, または符号関数を用いた。プーリングにはアベレージプーリングを用い、全結合層ののちソフトマックス関数をかけたものを出力とした。

3.3　固定小数点演算

ネットワークの軽量化のため、パラメータに量子化を施すが、畳み込みニューラルネットワークの重み、活性のなどの値のとりうる幅は異なり、同じスケールでの量子化を行うと量子化誤差が大きくなってしまう。そのため本研究では量子化を行う際にパラメータごとに異なるビット幅、最大値を設定することで効率的な量子化を実現する。このとき、これらのパラメータの異なる変数同士の演算がどのように行われるかについて説明する。

例えば活性の範囲 ±4, ビット幅 4, 重みの範囲 ±0.25, ビット幅 4, 畳み込み後の範囲 ±8, ビット幅 8 の場合の乗算加算を図 2 に示す。

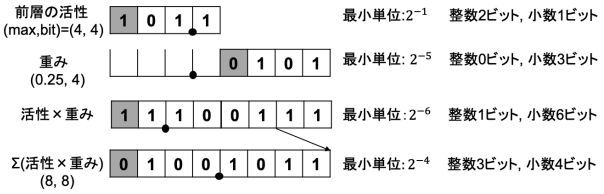

図 2　スケールの異なる固定小数点演算

図 2 のように、パラメータのビット幅、最大値によって固定小数点のビット位置、表現できる最小単位が定まる。例えば最大値を 4、ビット幅を 4 としたとき、符号ビットを除いた 3 ビットで 0 から 4 を表現することになる。そのため最下位ビットは 2^{-1} を表すことになる。このとき 4 ビットで-4 から 3.5 まで表現できることになる。畳み込み演算では乗算と加算を繰り返すことになる。前層からの活性が 4 ビットであり、重みも 4 ビットである場合、その乗算結果には 8 ビット必要となる。その小数点位置も活性と重みの小数点位置から一意に決まる。この値をアキュムレートし、畳み込み演算の出力を得る。

3.4　量子化の手法

本研究では重み、バッチ正規化のパラメータ、活性化関数にそれぞれ個別のビット幅、最大値を設定し、図 3 のようにそれぞれに量子化フィルタとして階段関数をかけることで量子化を行った。量子化のパラメータであるビット幅、最大値の設定は量子化を行わない場合 (TensorFlow における浮動小数点 32 ビット) において得られたパラメータから設定した。

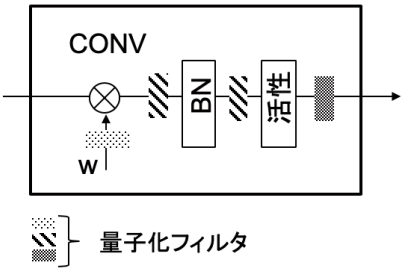

図 3　量子化概要

3.5　バッチ正規化の量子化

バッチ正規化の量子化を以下の式に示す。バッチ正規化ではバッチ間で各チャネルの平均を 0、分散を 1 にする処理と、そのあとにチャネルごとに定数倍、定数加算をする処理が続く。ここで 1 チャネルのバッチ間での平均を μ、標準偏差を σ とした。量子化するにあたり、バッチ正規化を次のように展開し、入力 x に依存する項とそうでない項に分割し、それぞれに図 4 のように量子化フィルタにかけることで量子化した。

$$
\begin{aligned}
y &= \gamma \frac{x-\mu}{\sqrt{\sigma^2}} + \beta \\
&= \frac{\gamma}{\sqrt{\sigma^2}} x + \beta - \frac{\gamma\mu}{\sqrt{\sigma^2}} \\
&= Ax + B \\
\overline{y} &= \overline{A}x + \overline{B}
\end{aligned}
$$

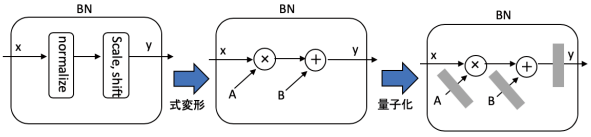

図 4　バッチ正規化層の量子化

3.6　順伝播と逆伝播

量子化を行ったときの順伝播、逆伝播は次のアルゴリズムによって行われる。学習において勾配は浮動小数点で計算され、更新が行われることに注意されたい。ここでは簡単のため畳み込み層、バッチ正規化層、活性化関数をまとめたものを 1 単位とする。

順伝播、逆伝播のアルゴリズム

w^l:l 層の重み y^l:l 層の出力 C^l:畳み込み層の出力 E:誤差関数
q():量子化フィルタ関数 act():活性化関数 $\bar{x}$:量子化した x

1. 順伝播

$\bar{w^l} = q(w^l)$
$C^{l+1} = conv(\bar{w}, y^l)$
$\bar{C}^{l+1} = q(C^{l+1})$
$A^{l+1} \equiv \frac{\gamma}{\sqrt{\sigma^2}}$
$B^{l+1} \equiv \beta - \frac{\gamma\mu}{\sqrt{\sigma^2}}$
$BN^{l+1} = q(A^{l+1}) * \bar{C}^{l+1} + q(B^{l+1})$
$y^{l+1} = act(q(BN^{l+1}))$

2. 逆伝播

$\frac{\partial E}{\partial q(BN^{l+1})} = \frac{\partial E}{\partial y^{l+1}} * \frac{\partial y^{l+1}}{\partial q(BN^{l+1})}$
ここで

$$\frac{\partial y^{l+1}}{\partial q(BN^{l+1})} = \begin{cases} 1 & (0 < q(BN^{l+1}) < max) \\ 0 & (else) \end{cases}$$

$\frac{\partial E}{\partial a}$ と $\frac{\partial E}{\partial b}$ を $\frac{\partial E}{\partial q(BN^{l+1})}$, $\bar{C}^{l+1}$ から計算
$\frac{\partial E}{\partial \bar{C}^{l+1}} = \frac{\partial E}{q(BN^{l+1})} * q(a)$
$\frac{\partial E}{\partial q(w)}$ と $\frac{\partial E}{\partial y^l}$ を $\frac{\partial E}{\partial \bar{C}^{l+1}}$ から計算
$\frac{\partial E}{\partial w}$を $\frac{\partial E}{\partial q(w)}$ により更新

4. 評　　価

TensorFlow を用いて Residual Network を構成し、独自に用意した歩行者検出用のデータセットを用いて学習を行い、その精度、計算量を評価した。また、ネットワークモデルを軽量にしたモデルでの評価も行なった。それぞれにおいて量子化を行わなかったモデル、つまり浮動小数点 32 ビットで学習を行ったモデルをベースラインとし、量子化をおこなったものと比較する。また、この手法のより一般的な評価を与えるため CIFAR-10 での評価を行った。

4.1 ResNet14 での歩行者検出精度

評価条件を表 1 に示す。

表 1　評 価 条 件

フレームワーク	TensorFlow
学習法	モメンタム法
学習率	0.01
モメンタム係数	0.9
誤差関数	交差エントロピー
データセット	訓練データ:14246(pos:10788, neg:3458) テストデータ:1000(pos:500, neg:500)
バッチサイズ	128

ここで学習率は指数減衰とした。重みは-0.25~0.25 での一様分布から初期値を生成した。14 層の ResNet のネットワーク構造を表 2 に示す。以下ではこれを ResNet14 と表記する。重み、活性、バッチ正規化層の量子化を表 3 のように設定したときのそれぞれの精度を図 5 に示す。filter, BN は畳み込み層後、バッチ正規化層での量子化条件を示し、kernel は重み、activation は活性の量子化条件を示している。表 3 において、float は 32 ビットの浮動小数点である。(x, y) は y ビットで $\pm x$ を表現することを示す。活性化関数は 1 ビットのときのみ符号関数、他は ReLU 関数を用いた。表に記載されている精度は最後の 5 つのテストデータでの 5 つの精度の移動平均を取ったものである。

表 2　ResNet14

Layer			Output Shape	params
n/a	n/a	input	(128, 160, 128, 1)	0
		conv1	(128, 160, 128, 16)	176
block1	unit1	conv2	(128, 160, 128, 16)	2336
block1	unit1	conv3	(128, 160, 128, 16)	2336
block1	unit2	conv2	(128, 160, 128, 16)	2336
block1	unit2	conv3	(128, 160, 128, 16)	2336
block2	unit1	conv2	(128, 80, 64, 32)	5184
block2	unit1	conv3	(128, 80, 64, 32)	9280
block2	unit2	conv2	(128, 80, 64, 32)	9280
block2	unit2	conv3	(128, 80, 64, 32)	9280
block3	unit1	conv2	(128, 40, 32, 64)	20608
block3	unit1	conv3	(128, 40, 32, 64)	36992
block3	unit2	conv2	(128, 40, 32, 64)	36992
block3	unit2	conv3	(128, 40, 32, 64)	36992
n/a	n/a	average-pooling	(128, 1, 1, 64)	0
n/a	n/a	fully-connected	(128, 1, 1, 2)	128

表 3　試行パラメータ

filter, BN	kernel	activation	accuracy
(max, bit)			%
float	float	float	99.8
(8, 8)	(1/4, 4)	(4, 4)	98.0
(8, 8)	(1/4, 4)	(1, 1)	97.8
(8, 8)	(1, 1)	(4, 4)	49.5
(8, 8)	(1, 1)	(1, 1)	71.9
(8, 4)	(1/4, 4)	(4, 4)	95.1
(8, 4)	(1/4, 4)	(1, 1)	49.9

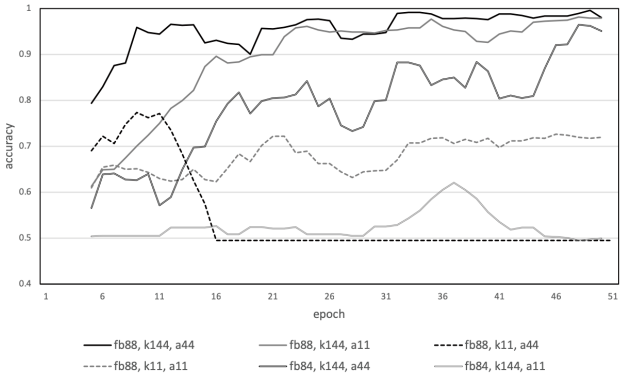

図 5　ResNet14 での精度

量子化の際のパラメータは次の 3 つの構成に従い設定し、評価を行った。

（1） 畳み込み演算後とバッチ正規化層を最大値 8、ビット幅 8、重みを最大値 1/4、ビット幅 4 とし、活性の量子化を変

えた場合。

（2） 畳み込み演算後とバッチ正規化層を最大値 8、ビット幅 8、重みを ±1 とし、活性の量子化を変えた場合。

（3） 畳み込み演算後とバッチ正規化層を最大値 8、ビット幅 4、重みを最大値 1/4、ビット幅 4 とし、活性の量子化を変えた場合。

(1) の構成において

活性が 4 ビットのときの認識精度の低下は浮動小数点で学習を行ったときと比べ、1.8% にとどまった。さらに活性化関数として符号関数を用い、±1 にしたところ、2% の悪化にとどまった。2 クラス分類であれば活性が 1 ビットであっても、その特徴マップの大きさにより十分な表現力を保っていたと考えられる。

(2) の構成において

畳み込み演算後とバッチ正規化層での量子化条件はそのままに、重みの精度を ±1 に固定し、活性の精度を 4 ビットから 1 ビットへと下げた。ここで注目すべきは二つの精度を比べると活性が 4 ビットでは学習が進まず、1 ビットの方が精度の良い結果となったことである。活性が 4 ビットの場合では、全結合層において出力が片方に張り付いたために入力側へと勾配が全く伝播されず、重みの更新がほとんど見られなかった。この条件では重みの最大値は 1、活性の最大値は 4 であり、その乗算の和が容易に最大値である 8 に達しやすい。全結合層の出力の最大値を 16 にしたところ学習が進むことが確認された。活性が 1 ビットのときは畳み込み演算後の分布は広がりを持ったものであった。活性の最大値が 1 であり、またその値は正に限らないことから活性を 4 ビットとしたときの問題は起こらず、学習が進んだと考えられる。

(3) の構成において

最後に畳み込み演算後、バッチ正規化の精度を落とし、4 ビット、表現範囲を ±8 とした。活性が 4 ビットのときは 95.1% と高い精度であった。バッチ正規化が 8 ビットの場合と比べると、3% の悪化にとどまっている。活性が ±1 の場合はほとんど学習が進まず、49.9% にとどまった。この場合は活性が 4 ビットのときと異なり、バッチ正規化が 8 ビットのときから大幅に精度が悪化している。この結果から活性の精度が十分である場合はバッチ正規化、畳み込み層の精度による認識精度の悪化は小さいことがわかる。

以上の結果から構成 1 と構成 3 で活性が 4 ビットの場合において認識精度を 95% 以上に保ちつつ、かつモデルを軽量とする上で可能となる量子化限界であることがわかる。また、量子化の度合いが大きすぎる場合は全くモデルの学習が行われない、または著しく精度が悪化することがわかった。また全結合層では畳み込み層と異なりその加算回数の多さから、最大値に張り付いてしまうことがあった。

4.2 ResNet8 での歩行者検出精度

ResNet14 での評価から、このネットワーク構造でもある程度の量子化に耐えうることがわかった。そこでネットワークの構造を見直し、さらなるモデルの軽量化が可能であるかを実験した。ネットワークの構造を表 4 に示す。このネットワークを表 5 のパラメータに従い、その精度を評価した。精度を図 6 に示す。これは先ほどのネットワークである ResNet14 の畳み込み層を半分に、特徴マップのチャネル数を半分にしたものである。層は合計で 8 層となっているため、以下では ResNet8 と表記する。

表 4　ResNet8 の構成

Layer			Output Shape	params
n/a	n/a	input	(128, 160, 128, 1)	0
n/a	n/a	conv1	(128, 160, 128, 8)	88
block1	unit1	conv2	(128, 160, 128, 8)	592
block1	unit1	conv3	(128, 160, 128, 8)	592
block2	unit1	conv2	(128, 80, 64, 16)	1312
block2	unit1	conv3	(128, 80, 64, 16)	2336
block3	unit1	conv2	(128, 40, 32, 32)	5184
block3	unit1	conv3	(128, 40, 32, 32)	9280
n/a	n/a	average-pooling	(128, 1, 1, 32)	0
n/a	n/a	fully-connected	(128, 1, 1, 2)	64

表 5　試行パラメータ

Filter, BN	kernel	activation	accuracy
(max, bit)			%
float	float	float	99.2
(8, 8)	(1/4, 4)	(4, 4)	98.9
(8, 8)	(1/4, 4)	(1, 1)	96.2
(8, 8)	(1, 1)	(4, 4)	87.8
(8, 8)	(1, 1)	(1, 1)	81.8
(8, 4)	(1/4, 4)	(4, 4)	86.5
(8, 4)	(1/4, 4)	(1, 1)	83.0

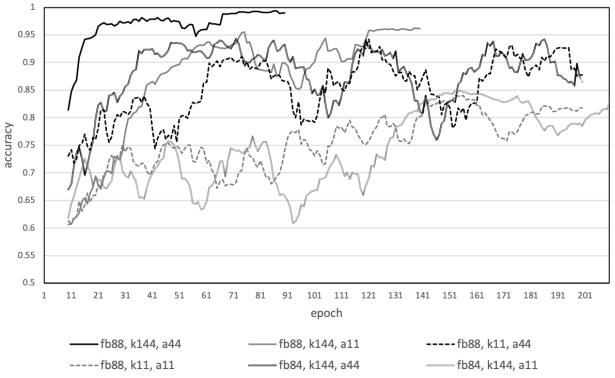

図 6　ResNet8 での精度

全体の精度の推移の傾向を見ると、ResNet14 と比べて収束に時間がかかっている。学習回数は ResNet14 と比べ、4 倍となっていることに注意されたい。また、学習が安定していない

ことがわかる。精度が収束していないうちに学習を打ち切っているように見えるが、これは訓練データ集合での精度が飽和したためである。

ここでも ResNet14 のときと同様に (1),(2),(3) に分けてその精度について考察する。

(1) の構成において

活性が最大値 4、ビット幅 4 のときは 98.9% と ResNet14 とほぼ変わらない精度であった。また活性が 1 ビットであっても 96.2% の精度を保っていた。他の条件時と比べ、これらは精度の収束が見られた。

これらの結果から ResNet8 であってもこのデータ集合に対しては十分な表現力を持っていると言える。

(2) の構成において

活性が最大値 4、ビット幅 4 では 87.8%、1 ビットでは 81.8% と ResNet14 のときより良い精度となった。この条件では ResNet14 の場合は学習が進まなかった。活性が 4 ビットの場合、ResNet14 で見られた全結合層での最大値への張り付きはある程度見られたものの、ほとんどの値は最大値の中に収まっており、学習に支障は生じなかった。これは全結合層でのパラメータ数が減ったためであると考えられる。全結合層での重みは一様分布により生成され、量子化フィルタによって ±1 となる。ここで確率 1/2 で 1、確率 1/2 で-1 となる確率変数 X を考えると、N 回の試行での和 Y の期待値は 0、分散は N となる。

$$V[Y] = V[\sum_{i}^{N} w_i] = N$$

ここで w_i は独立に生成されることを使った。N が小さいときは分散は小さく、N が大きいときは分散は大きくなる。ここでの N は全結合層のパラメータの数にあたり、ResNet14 では 128、ResNet8 では 64 に相当する。出力はパラメータの線型結合で表されるが、また活性化関数は ReLU であり、正の値のみをとるため、N が大きいときの出力は上式により正負どちらかに大きく振れてしまうと考えられる。

活性が 1 ビットの場合では、ResNet14 よりも良い結果となった。ヒストグラムの観察から畳み込み演算の出力が少なからず最大値に張り付いていることが確認された。このとき、バッチ正規化層への入力は多峰の分布となり、バッチ正規化層での平均を 0、分散を 0 にする処理は効果が薄くなってしまう。バッチ正規化層の出力は変わらず多峰形の分布となり、活性化層である符号関数ではバッチ正規化の恩恵を受けられない。このため ResNet14 では学習に時間がかかり、その結果、ResNet8 での精度に及ばなかったと考えられる。

(3) の構成において

活性が最大値 4、ビット幅 4 では 86.5%、1 ビットでは 83.0% となった。活性が 4 ビットでは ResNet14 から 10% 近くの大きな精度悪化が見られた。活性が 1 ビットのときでは ResNet14 のときより良い精度となった。この条件では ResNet14 の場合は学習が進まなかった。

これらの結果から ResNet8 においては畳み込み後とバッチ正規化の量子化を最大値 8、ビット幅 8、重みの量子化を最大値 1/4、ビット幅 4、活性の量子化を ±1 で行った場合が量子化限界であることがわかる。また、2 値分類においては 8 層のネットワークであっても十分な精度が得られ、量子化も可能であることがわかった。

4.3 CIFAR-10 による評価

これまでは歩行者検出のための独自のデータセットを用いて量子化された ResNet の評価を行ってきた。これは精度を落とさず、かつモデルを軽量化できる限界を探るためのものであった。ここではこの手法についてより一般的な評価を与えるため、広く使われているデータセットである CIFAR-10 による評価を行った。CIFAR-10 は 10 クラス分類問題のデータセットであり、1 枚の画像は 32*32 のカラー画像である。

ネットワークの構成は先述の ResNet14 とし評価を行った。評価条件は表 6 の通りである。試行パラメータとその精度を表 7 に示す。また、それぞれの学習経過を図 7 に示す。プリプロセスとして画像のランダムなトリミング、水平反転、明るさのランダムな調整、コントラストのランダムな調整、画像の正規化を行った。

表 6　評 価 条 件

フレームワーク	TensorFlow
学習法	モメンタム法
学習率	0.01
モメンタム係数	0.9
誤差関数	交差エントロピー
データセット	CIFAR-10 訓練データ: 50000(各クラス 10000) テストデータ: 10000(各クラス 1000)
バッチサイズ	128
エポック数	150

表 7　試行パラメータ

Filter, BN	kernel	activation	accuracy
(max, bit)			%
float	float	float	83.9
(8, 8)	(1/4, 4)	(4, 4)	78.2
(8, 8)	(1/4, 4)	(1, 1)	55.5
(8, 8)	(1, 1)	(4, 4)	10
(8, 8)	(1, 1)	(1, 1)	12.6
(8, 4)	(1/4, 4)	(4, 4)	63.1
(8, 4)	(1/4, 4)	(1, 1)	28.3

グラフ全体の傾向を見ると量子化の条件により顕著に精度の差が現れていることがわかる。これは単にタスクが 2 値分類から 10 クラス分類へと難しくなったためであると考えられる。

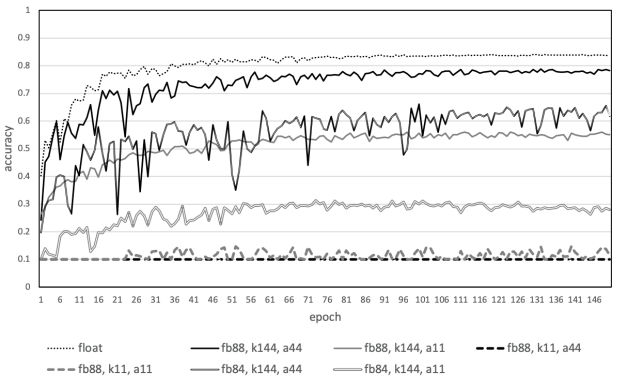

図 7　CIFAR-10 での精度

(1) の構成において

float の次に精度が良いのは fb(8,8), k(1/4,4), a(4,4) のときであり、これは最もビット精度が高い条件であるため妥当である。活性が 1 ビットのとき、精度は大幅に悪化し、55.5% となった。

(2) の構成において

重みが 1 ビットの場合の精度は 10% 程度にとどまった。ResNet14 の歩行者検出のデータセットで見られた全結合層での最大値への張り付きが同様に見られた。この現象を防ぐには全結合層での和における項数に応じたスケールを重みに与える手法、または最大値を個別に設定する手法が考えられる。入力ユニット数に関わらず最大値を設定してしまうと、入力ユニットが多い場合にその和が最大値に張り付きやすいことは先述の通りである。またバッチ正規化層の精度を浮動小数点 32 ビットで学習を行った場合でも 50% を下回っていた。

(3) の構成において

fb(8,4), k(1/4,4), a(4,4) の精度が fb(8,8), k(1/4,4), a(1,1) の精度よりも 7% 程度上回っていた。これらの結果からバッチ正規化、畳み込み層の精度は活性の精度と比べ、相対的に重要でないことが予想される。歩行者検出のデータセットでは、後者の精度の方が良かったことから簡単なタスクであれば活性が 1 ビットであっても表現できたが、難しいタスクでは 1 ビットでは情報量として足りないと解釈できる。

これらの結果から、CIFAR-10 では歩行者検出データセットと比べ、量子化による精度の悪化が大きい事がわかる。それでも浮動小数点での学習時と比べ、5% の悪化にとどまっている。また活性の精度を下げるよりもバッチ正規化の精度を下げた方が精度が良いが、この場合では固定小数点の乗算が必要となるため計算量は相対的に多くなってしまう。以上のように歩行者検出データセットと比べ、量子化の度合いによる精度の悪化に新たな傾向が見られた。

4.4　計算量・パラメータ数の見積もり

表 8 に各ネットワークの畳み込み演算、バッチ正規化層、全結合層でのパラメータ数、計算量の見積もりを示す。

表 8　計算量・パラメータ数の見積もり

	ResNet14	ResNet8	MovileNet_v2_35_160 [6]
パラメータ数 [k]	174.256	19.896	1,660.000
計算量 [M]	530	62.5	30

ResNet14 では 530[million] であったのに対し、ResNet8 では 62.5[million] とおよそ 9 分の 1 まで計算量が削減される。活性を 1 ビットにした場合、すべての畳み込み層での乗算がなくなるため、さらに計算量は削減され、ResNet8 では 86 万回の乗算が必要となる。

5. ま と め

本研究では車載向け歩行者検出のためのニューラルネットワークの軽量化に取り組んだ。量子化する手法として畳み込み層、活性化層、バッチ正規化層に個別のビット精度を設定する手法を提案し、その評価を行った。車載用歩行者検出のための軽量かつ精度の高いネットワークを構築する上で、徐々に量子化の度合いを強めるべくパラメータごとの量子化に取り組んだが、その過程において量子化を行うパラメータ間での影響を見た。ResNet8 であっても量子化を加えた上で 95% 以上と高い精度で分類ができ、実用に耐えうる精度が得られることが示された。活性が 1 ビットであれば畳み込み演算の際の乗算の必要はなくなり、大幅な演算量の削減が見込まれる。

今後の展望として、層ごとに異なる量子化条件を適用する手法などの検討、またアルゴリズムだけでなく、高速化のためのアーキテクチャ検討の必要がある。

謝辞

本研究の一部は NEC ソリューションイノベータとの共同研究で行われたものである.

文　　献

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” ArXiv e-prints, pp.••–••, Dec. 2015.

[2] M. Courbariaux, Y. Bengio, and J.-P. David, “BinaryConnect: Training Deep Neural Networks with binary weights during propagations,” ArXiv e-prints, pp.••–••, Nov. 2015.

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

[4] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, “XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks,” ArXiv e-prints, pp.••–••, March 2016.

[5] Y. Bengio, “Estimating or propagating gradients through stochastic neurons,” CoRR, vol.abs/1305.2982, pp.••–••, 2013. http://arxiv.org/abs/1305.2982

[6] M. Sandler, A.G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, “Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation,” CoRR, vol.abs/1801.04381, pp.••–••, 2018. http://arxiv.org/abs/1801.04381