バイナリ型 DCNN による識別計算の高速化とモデル圧縮に関する研究

TP15011 神谷 龍司　　指導教授：藤吉 弘亘

1. はじめに

Deep Convolutional Neural Network (DCNN) は，物体認識や物体検出，セマンティックセグメンテーションといった様々なタスクにおいて非常に高い性能を達成している．しかし，DCNN は計算量やパラメータ数が多く，膨大な識別時間とメモリサイズを必要とする．また，識別精度の向上や難しいタスクを対象とする場合，ネットワークが多層化あるいは複雑化する傾向がある．モデルの複雑化に伴い，識別時間やモデルサイズの増加といった問題が発生する．組み込み機器やモバイル機器で利用するには，識別計算の高速化とモデルサイズの圧縮が必要不可欠である．そこで本研究では，これらの問題を同時に解決する Binarized-DCNN を提案する．本手法は，実数同士の内積計算を二値同士の内積計算に置き換えることで，既存の学習済みモデルの識別計算を高速化でき，モデルを圧縮することが出来る．

2. DCNN の計算量とモデルサイズ

DCNN は計算量やパラメータ数が非常に多いため，膨大な識別時間とメモリサイズを要する．DCNN のデファクトスタンダードなモデルである AlexNet[1] と VGG-Net[2], Residual Network (ResNet)[3] の計算量とモデルサイズを表 1 に示す．AlexNet は 8 層，VGG-Net は 16 層，ResNet は 152 層で構成されるネットワークモデルである．表 1 より，層数が増えることで計算量とモデルサイズの両方が増加していることがわかる．これらの問題を解決するために，識別計算の高速化には Speeding up CNN や Accelerating CNN が，モデル圧縮には Deep Compression や SqueezeNet, 高速化とモデル圧縮の同時達成には BinaryNet や XNOR-Net が提案されている．

表 1：各ネットワークの計算量とモデルサイズ

モデル	AlexNet	VGG-Net	ResNet
計算量 [GFLOPS]	1.08	15.4	11.3
モデルサイズ [MB]	238	528	229

3. 提案手法

提案手法では，DCNN の識別計算を高速化するために，畳み込み層と全結合層の重みを二値基底行列 $\mathbf{M}$ とスケール係数ベクトル $\mathbf{c}$ に分解し近似内積計算を行う．このとき，各層の重みの大半を二値に変換するため，モデルサイズの圧縮を同時に行うことができる．ただし，二値に変換した重みを用いて近似内積計算を行うには，入力特徴マップも二値である必要がある．提案手法では，Quantization sub-layer を導入することで特徴マップを二値に変換する．

3.1 ベクトル分解による重みの分解

ベクトル分解 [4][5] は，重みベクトル $\mathbf{w} \in \mathbb{R}^D$ を二値基底行列 $\mathbf{M} \in \{-1, 1\}^{D \times k}$ とスケール係数ベクトル $\mathbf{c} \in \mathbb{R}^k$ に分解する．ここで，D は入力次元数，k は基底数を表している．ベクトル分解は重みの分解に膨大な時間を要するが，近似精度の良い分解が可能である．分解アルゴリズムを Algorithm1 に示す．まず，$\mathbf{M}$ を $\{-1, 1\}$ により初期化する．次に，初期化した $\mathbf{M}$ を用いて最小二乗法により $\mathbf{c}$ を求める．今度は最適化した $\mathbf{c}$ を用いて，式 (1) を最小化するように最適な $\mathbf{M}$ を全探索する．最後に，式 (1) が収束するまで $\mathbf{c}$ と $\mathbf{M}$ の最適化を繰り返す．なお，ベクトル分解の近似精度は初期値に依存するため，初期値を L 回変更して式 (1) が最も小さくなる $\mathbf{M}$ と $\mathbf{c}$ をその基底値の分解結果とする．

$$E = ||\mathbf{w} - \mathbf{M}\mathbf{c}||_2^2 \tag{1}$$

ここで，DCNN の重みに対してベクトル分解を適用することを考える．ベクトル分解で分解できる重みはベクトルのみであるため，全結合層の結合重みはそのまま分解できる．しかし，畳み込み層の重みフィルタは行列であるため分解できない．提案手法では，重みフィルタをベクトル表現することで重みの分解を可能にしている．また，AlexNet や VGG-Net, ResNet などで用いられる重みフィルタのサイズ H は 3，5 など小さい場合が多い．そのため畳み込みの計算量と近似内積計算の計算量があまり変わらず，近似内積計算による効果が得られない．そこで，分解するフィルタを 1 枚だけではなくチャンネル方向のフィルタも利用する．これにより，ベクトル表現した次元数が $D \cdot H^2$ となり次元数が増加するため，近似内積計算の効果が得られやすくなる．また，学習済みの重みは既知であり，事前に分解を行うことが可能であるため，分解に要する時間は識別時間には影響しない．

Algorithm 1 分解アルゴリズム

Require: $\mathbf{w}$, k, L
  for i to L **do**
    $\mathbf{M}_i$ を $\{-1, 1\}$ の乱数により初期化
    $\mathbf{c}_i$ を実数の乱数により初期化
    repeat
      $\mathbf{c}_i = (\mathbf{M}_i^{\mathrm{T}}\mathbf{M}_i)^{-1}(\mathbf{M}_i^{\mathrm{T}}\mathbf{w})$
      $\mathbf{M}_i = \underset{\mathbf{M}_i \in \{-1,1\}^{D \times k}}{\arg\min} ||\mathbf{w} - \mathbf{M}_i\mathbf{c}_i||_2^2$
    until 式 (1) が収束
  end for
  $\hat{\mathbf{M}}, \hat{\mathbf{c}} = \underset{\mathbf{M},\mathbf{c}}{\arg\min} ||\mathbf{w} - \mathbf{M}\mathbf{c}||_2^2$
  return $\hat{\mathbf{M}}$, $\hat{\mathbf{c}}$

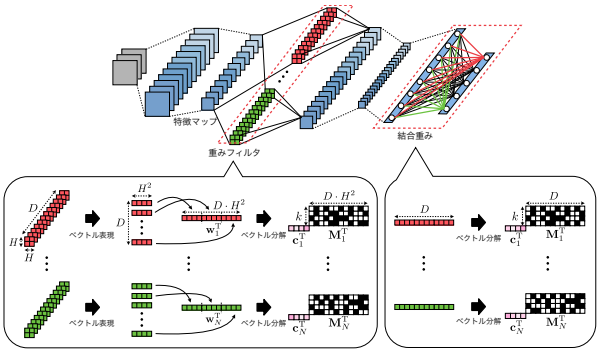

図 1：各層におけるベクトル分解の流れ

3.2 Quantization sub-layer

DCNN の特徴マップは負値を含んだ実数値であるため，近似内積計算を行うためにこれを二値 $\in \{0, 1\}$ の特徴マップに変換する．しかし，特徴マップは入力サンプル毎に異なるため，事前に二値化することができない．そのため，識別計算時に二値化する必要があり，二値化に要する時間が識別時間に直接影響する．そこで，負値を含んだ実数値を二値に変換できる Quantization sub-layer を導入する．Quantization sub-layer は，特徴マップの最大値と最小値の間を量子化することで量子化幅を可変にすることができる．これにより負値を含んだ実数値に対しても量子化が可能となる．まず，特徴マップ $\mathbf{h}$ の最大値及び最小値より量子化幅 Δd を求める．Δd は特徴マップの最大値と最小値に依存するため，特徴マップ毎に異なる値を持つ．次に，$\mathbf{h}$ の最小値が 0 になるように $\mathbf{h}$ をシフトさせる．$\mathbf{h}$ をシフトさせることで負値に対応した量子化が可能となる．最後にシフトさせた $\mathbf{h}$ を量子化し Q ビットの二値にする．

3.3 Inference 処理

DCNN の計算は，実数値の入力特徴マップ $\mathbf{h}$ と重みフィルタ $\mathbf{w}$ の積和を計算することで出力が得られる．本手法では，実数値による積和計算を二値に置き換えて出力を計算する．入力特徴マップ $\mathbf{h}$ と重み $\mathbf{w}$ の内積計算は式 (2) で近似計算することができる．ここで，$\hat{\mathbf{M}}$ と $\hat{\mathbf{c}}$ はベクトル分解により得られた二値基底行列とスケール係数ベクト

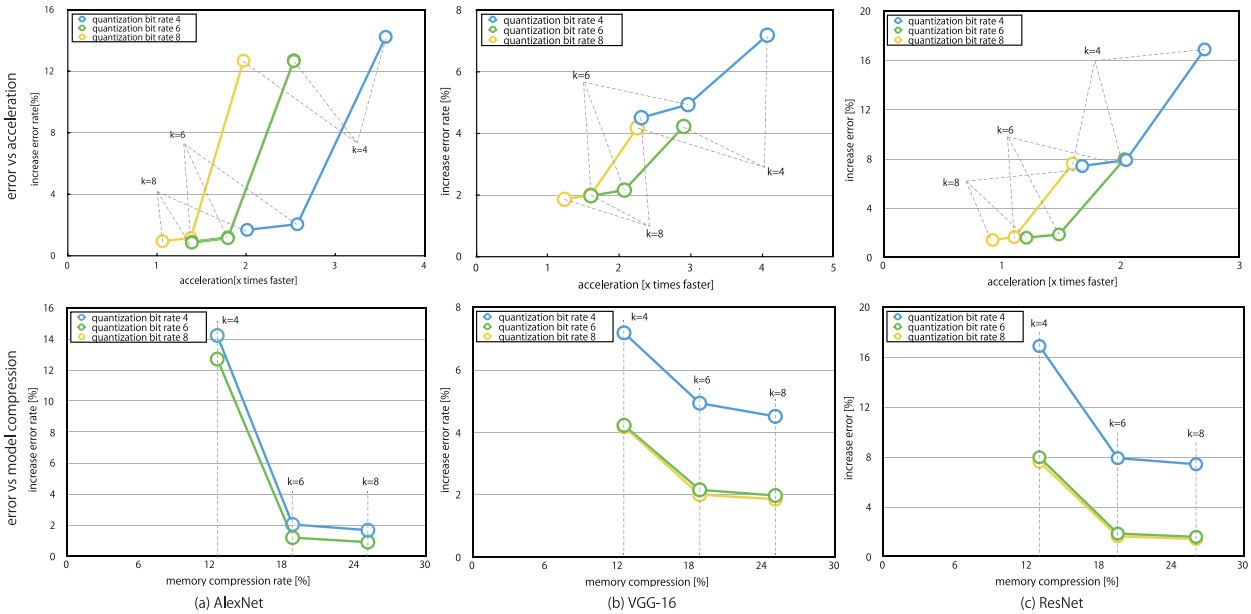

図 2：各ネットワークモデルの評価結果

ル，$\mathbf{B}$ と $\mathbf{r}$ は Quantization sub-layer により得られた二値行列と復元係数ベクトル，$\min(\mathbf{h})$ は量子化する際に発生したオフセットである．

$$\begin{aligned}\mathbf{w}^{\mathrm{T}}\mathbf{h} &\approx \hat{\mathbf{c}}^{\mathrm{T}}\hat{\mathbf{M}}^{\mathrm{T}}\left(\mathbf{B}\mathbf{r}+\min(\mathbf{h})\mathbf{1}\right)\\ &\approx \hat{\mathbf{c}}^{\mathrm{T}}\hat{\mathbf{M}}^{\mathrm{T}}\mathbf{B}\mathbf{r}+\min(\mathbf{h})\,\hat{\mathbf{c}}^{\mathrm{T}}\hat{\mathbf{M}}^{\mathrm{T}}\mathbf{1}\end{aligned} \tag{2}$$

このとき，$\hat{\mathbf{M}}^{\mathrm{T}} \in \{-1,1\}^{D\times k}$ と $\mathbf{B} \in \{0,1\}^{Q\times D}$ は二値であるため，式 (3) を用いて論理演算とビットカウントで計算することができる．ビットカウントは Streaming SIMD Extensions (SSE) 4.2 に実装されている POPCNT 関数を使用することで高速に演算が可能である．

$$\hat{\mathbf{M}}^{\mathrm{T}}\mathbf{B} = 2\times \mathrm{POPCNT}(\mathrm{AND}(\hat{\mathbf{M}}^{\mathrm{T}},\mathbf{B})) - \|\mathbf{h}\| \tag{3}$$

4. 評価実験

ILSVRC2012 の分類タスクで使用された ImageNet を用いて複数のネットワークモデルに提案手法を適用した際の認識精度と処理時間及びモデルサイズを評価する．検証サンプル 50,000 枚を使用し，認識精度は Top-5 accuracy により評価する．Top-5 accuracy は教師信号と同じクラスの推定確率が上位 5 位以内であれば認識成功とする方法である．ネットワークモデルには AlexNet と VGG-Net，ResNet を用いる．各モデルのパラメータには公開されている学習済みモデルを使用する．近似計算に用いる量子化ビット数 Q は 4, 6, 8，基底数 k も同様に 4, 6, 8 を用いる．使用するプロセッサは Intel Core i7–4770 3.40GHz である．

4.1 モデル 1: AlexNet

AlexNet における近似内積計算による認識精度と処理時間，モデルサイズの比較を図 2(a) に示す．図 2(a) より，基底数あるいは量子化ビット数を大きくすることで認識精度が向上する傾向が見られる．ただし，同基底数の量子化ビット数 6 と 8 を比較すると，ほとんどエラー増加率が変化していない．このことから，より計算量の少ない量子化ビット数 6 が最適であると言える．同基底数の量子化ビット数 4 と 6 においてメモリ圧縮率が変化していないことから，量子化ビット数はメモリ圧縮率に影響しない．量子化ビット数 6，基底数 6 のとき，約 1.79 倍の高速化とモデルサイズを約 80%(約 237.91MB から約 44.85MB) 圧縮できた．このとき，エラー増加率は約 1.20%である．

4.2 モデル 2: VGG-Net

VGG-Net における近似内積計算による認識精度と処理時間，モデルサイズの比較を図 2(b) に示す．層数が増加しても AlexNet と同程度の高速化倍率とモデル圧縮率が得られていることが確認できる．量子化ビット数 6，基底数 6 のとき，約 2.07 倍の高速化とモデルサイズを約 81%(527.74MB から 99.26MB) 圧縮できた．このとき，エラー増加率は 2.16%である．

4.3 モデル 3: ResNet

ResNet による近似内積計算における認識精度と処理時間，モデルサイズの比較を図 2(c) に示す．図 2(c) より，152 層のネットワークモデルである ResNet に対しても高速化できていることがわかる．AlexNet や VGG-Net は全結合層がモデルサイズの 90%以上を占めているが，ResNet では畳み込み層が 90%以上を占めている．ResNet のように畳み込み層が非常に多いモデルに対してもモデル圧縮の効果が得られることが確認できる．量子化ビット数 6，基底数 6 のとき，約 1.77 倍の高速化とモデルサイズを約 81%(約 229.08MB から約 44.71MB) 圧縮できた．このとき，エラー増加率は約 1.86%である．

5. おわりに

本稿では，既存のネットワークモデルに対して再学習なしに識別計算の高速化とモデルを圧縮する方法を提案した．本手法は，二値同士の演算に置き換えることで識別計算の高速化とモデルサイズの圧縮を同時に行う．量子化ビット数 6，基底数 6 のとき，約 1.5 から 2.0 倍の高速化とモデルサイズを約 80%の圧縮を実現した．

参考文献

[1] A. Krizhevsky, *et al.*, “ImageNet Classification with Deep Convolutional Neural Networks”, In NIPS, 2012.

[2] K. Simonyan, *et al.*, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, In ICLR, 2015.

[3] K. He, *et al.*, “Deep Residual Learning for Image Recognition”, In CVPR, 2016.

[4] S. Hare, *et al.*, “Efficient online structured output learning for keypoint-based object racking”, In CVPR, 2012.

[5] Y. Yamauchi, *et al.*, “Distance computation between binary code and real vector for efficient keypoint matching”, IPSJ Transactions on CVA, vol.5, pp.124–128, 2013.

研究業績

[1] 神谷龍司, 山下隆義, 濱走秀人, 山内悠嗣, 藤吉弘亘, 川出雅人,“Deep Convolutional Neural Network による文字認識”, 画像センシングシンポジウム, 2015.

[2] 神谷龍司, 山下隆義, 安倍満, 佐藤育郎, 山内悠嗣, 藤吉弘亘, “Binarized-DCNN による識別計算の高速化とモデル圧縮”, パターン認識・メディア理解研究会, 2016.

(他 学会発表 2 件)