人工知能学会 インタラクティブ
情報アクセスと可視化マイニング研究会(第24回)
SIG-AM-24-04

ベクトル単位での重み共有化と符号化による カプセルネットワーク軽量化の試み

Experimental study on the vector-based compression of Capsule Networks

島田 研太[1]　尾崎 知伸[1]*
Kenta Shimada[1]　Tomonobu Ozaki[1]

[1] 日本大学 文理学部
[1] College of Humanities and Sciences, Nihon University

Abstract: In this study, we discuss the compression of the capsule networks (CapsNets), which is one of extensions of the convolutional neural networks (CNNs). Focusing on the fact that the basic unit of CapsNets is a vector, we propose a vector-based quantization and Huffman coding for the compression by extending typical weight reduction methods for CNN. Experimental evaluations using MNIST and CIFAR-10 with various vector sizes and quantization granularities show that the proposed method can compression the networks with a slight loss of prediction accuracy.

1　はじめに

深層学習の代表的な手法の一つである畳み込みニューラルネットワーク（Convolutional Neural Network; CNN）は，特徴抽出を行う畳み込み層と次元圧縮を行うプーリング層とを複数接続させることで，画像認識分野において非常に高い精度を実現している．しかしその一方で，少なくとも 2 つの側面から問題点が指摘されている．

第一の問題点は，モデル容量に関するものである．一般に CNN は，膨大な数のパラメタを含み，結果としてモデルの容量が巨大になる傾向が認められる．これに対し，これまで様々な方面で改善策が提案されている．例えば文献 [1] では，訓練済みモデルから別の軽量モデルを構築する蒸留と呼ばれる技術が提案されている．また，絶対値が小さな重みをもつ連結辺を削除する枝刈り [2, 3] や，類似する重みをグループ化し代表値に縮約する重み共有とその符号化 [4] に関してもそれぞれ提案が行われており，これらの技術を組み合わせることで，精度を劣化させずに大幅なモデル圧縮が達成されることが報告されている [5].

CNN の第二の問題点は，精度に関するものであり，これまでプーリング層に起因する問題が指摘されている [6]．プーリング層は，畳み込み層で抽出された特徴を一定範囲で集約･集計することで位置や大きさに対する頑健性を CNN に与えている．しかしその反面，プーリングの結果として，抽出された特徴の向きや大きさ，特徴間の位置関係等の情報が捨てられることとなり，精度向上に対して負の影響を与えることが考えられる．この問題に対し，プーリング層を排除し，カプセルと呼ばれる新たなニューロングループを一つの単位として利用するカプセルネットワーク (Capsule Networks; CapsNet)[6] と呼ばれる手法が提案されている．詳しくは後述するが，CNN など従来の深層学習モデルでは，各ニューロンはパラメタとして重みベクトルを持ち，一つのスカラー値を出力するが，CapsNet では，各ニューロンはパラメタとして重み行列群を持ち，ベクトル値を出力する．そのため，必然的にモデル（パラメタ数）が大規模化し，大量の演算リソース（計算時間とメモリ）が必要とされることが予想される．

本論文では，CNN の 2 つの問題点，すなわちモデルの大規模化とプーリング層に起因する精度劣化を同時に解消することを目指し，CapsNet に対する軽量化手法を模索する．具体的には，CNN に対する代表的な軽量化手法である枝刈り，重み共有化，符号化を基本とするが，スカラー出力をベクトル出力に変更したという CapsNet の大きな特徴を考慮し，各技術の適用単位をスカラーではなく，ベクトルとすることを提案する．

本論文の構成は以下の通りである．2 章でカプセルネットワークの概要を示す．3 章で代表的な軽量化手法について概観し，4 章でそれらをカプセルネットワークへと応用する．5 章で実験と評価を行い，最後に 6 章でまとめを行い今後の課題を述べる．

*連絡先：日本大学文理学部情報科学科
〒 156-8550 東京都世田谷区桜上水 3-25-40
E-mail: tozaki@chs.nihon-u.ac.jp

2 カプセルネットワーク

カプセルネットワーク [6] とは，カプセルと呼ばれる，特定タイプの実体に関する（大きさや向きなどの）パラメタを表す活性化ベクトルを持つニューロンのグループを基本単位とするニューラルネットワークである．活性化ベクトルは，その名の通り実数値ベクトルであり，ベクトル長を用いて実体の存在確率を，向きを用いてパラメタ値を表す．カプセルネットワークは，複数のカプセルを連結させたネットワークであり，カプセル間でベクトル表現される情報を伝播することで学習と予測が行われる．すなわち，あるレベルで活性化したカプセルがより高いレベルのカプセルのパラメタに対して予測を行い，また複数カプセルによる予測が一致すると，より高いレベルでのカプセルが活性化されることになる．非常に大雑把に言えば，カプセルネットワークとは，CNN からプーリング層を排除し，伝播させる情報をスカラー値からベクトルに変更したものと言える．

カプセル j の出力であるベクトル v_j は，その長さを用いて入力値に対する (j が表す) 実体の存在確率を表す．このため，長さの短いベクトルはその長さを 0 に縮約し，逆に長いベクトルはその長さを 1 に近づけるような非線形変換を行う Squashing 関数と呼ばれる活性化関数が提案されている．

Squashing 関数 squash は，カプセル j への入力 s_j を引数とし，活性化ベクトル v_j を出力する関数であり，以下の様に定義される [6].

$$v_j \;=\; \mathrm{squash}(s_j) \;=\; \frac{||s_j||^2}{1+||s_j||^2}\frac{s_j}{||s_j||}$$

ここで squash への入力である s_j は，下位層のカプセル i の出力ベクトル $\mathbf{u}_i$ と重み行列 $\mathbf{W}_{ij}$ の積として得られる予測ベクトル $\hat{\mathbf{u}}_{j|i}$ の重み付き和であり，以下の様に定義される．

$$s_j \;=\; \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}, \quad \hat{\mathbf{u}}_{j|i} \;=\; \mathbf{W}_{ij}\mathbf{u}_i$$

また結合係数（結合強度）と呼ばれる重み

$$c_{ij} \;=\; \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

は，カプセル i から j への（正規化済みの）結合の強さを表し，ルーティングと呼ばれる下記の計算を反復することで学習される．

$$\left\{\begin{array}{lcl} \mathbf{c}_i & \leftarrow & \mathrm{softmax}(\mathbf{b}_i) \\ \mathbf{s}_j & \leftarrow & \sum_i c_{ij}\hat{\mathbf{u}}_{j|i} \\ \mathbf{v}_k & \leftarrow & \mathrm{squash}(\mathbf{s}_j) \\ b_{ij} & \leftarrow & b_{ij} + \hat{\mathbf{u}}_{j|i}\cdot\mathbf{v}_j \end{array}\right.$$

3 ニューラルネットワークの軽量化

ニューラルネットワークに対する汎用的な圧縮・軽量化の一つの手法として，これまでに，重みの枝刈りと共有，ハフマン符号化を用いる手法が提案されている [4].

枝刈り： ニューラルネットワーク中の各ニューロンは一般に，下位層から連結されているニューロンの出力に対する重み付き和を入力とし，何らかの演算を行う．従って，重み（の絶対値）の大きさがその影響度に相当する．枝刈りは，絶対値が 0 に近い重みを有する接続辺は影響が小さいものとして削除する操作を指す．より形式的には，閾値 ϵ を用いて，重み w の値を

$$w' \;=\; \left\{\begin{array}{ll} 0 & (|w| \le \epsilon) \\ w & (\mathrm{otherwise}) \end{array}\right.$$

と更新し，$w'=0$ に相当する辺を削除する．また，枝刈り操作と枝刈り後のネットワークに対する再学習を重みが収束するまで繰り返し，精度劣化を抑制した上での不要接続辺の排除を実現している．

重み共有： 重み共有とは，訓練済みのネットワークの各層において類似する重みをグループ化し，各重みを所属グループの代表値に置き換える操作を指す．この操作により，重みの数（種類数）がグループ数にまで削減されることとなるので，ネットワーク全体の軽量化が期待できる．

具体的には，K-平均法を用いて重みの集合 $W \;=\; \{w_1, w_2, \cdots, w_n\}$ を

$$\arg\min_C \sum_{c_i\in C}\sum_{w\in c_i} |w - c_i^*|$$

を満たす k 個のクラスタ $C \;=\; \{c_1, c_2, \cdots, c_k\}\,(k \ll n)$ にグループ化し，各重み w を所属するクラスタの代表値 c_i^* に置き換える，すなわち

$$w' \;=\; c_i^* \text{ s.t. } c_i = \arg\min_{c_i\in C} |w - c_i^*|$$

と w を更新する操作を行う．また，クラスタ c_i の代表値としては，クラスタ内平均 $c_i^* \;=\; \frac{1}{|c_i|}\sum_{w\in c_i} w$ や，セントロイド $c_i^* \;=\; \arg\min_{w\in c_i}\sum_{w_j\in c_i}|w - w_j|$ などが利用される．

ハフマン符号化： 各重み w をビットで符号化することで，モデル全体の圧縮を行う．具体的には，重み共有後にハフマン符号化を適用する．ハフマン符号化により，頻度の高い情報に短いビットが，頻度の低い情報に長いビットが割り当てられ，モデル全体としての圧縮が期待できる．

4 カプセルネットワークの軽量化

2 章で概観した通り，カプセルネットワークでは，2 つのカプセル i と j の連結に対し，結合強度を表す c_{ij} に加えて重み行列 $\mathbf{W}_{ij}$ が付与される．そのため一般的なニューラルネットワークと比較し，パラメタ数（重み総数）が膨大となる．特に c_{ij} がスカラーであるのに対し，$\mathbf{W}_{ij}$ は行列であり，含まれるパラメタ数も大きい．従って，カプセルネットワークの圧縮においては，重み行列の圧縮が重要となる．

本研究では，重み行列に対し，枝刈り処理（絶対値が 0 に近い重みの値を 0 にする処理）を施した後，要素，行，列のそれぞれの単位での重み共有化とハフマン符号化を適用することを提案し，各単位での効果を実験的に比較する．

対象とする重み行列の全体集合を $\mathcal{W}$ とし，各重み行列 $\mathbf{W}_{ij} \in \mathcal{W}$ を

$$\mathbf{W}_{ij} = \begin{pmatrix} w_{ij}^{1,1} & \cdots & w_{ij}^{1,n} \\ \vdots & \ddots & \vdots \\ w_{ij}^{m,1} & \cdots & w_{ij}^{m,n} \end{pmatrix}$$

と表記する．このとき，行列に含まれる要素，行ベクトル，列ベクトルの全体集合をそれぞれ

$$\mathcal{W}^{r,c} = \{w_{ij}^{k,l} \mid 1 \le k \le m, 1 \le l \le n, \mathbf{W}_{ij} \in \mathcal{W}\}$$

$$\mathcal{W}^{r,*} = \{\langle w_{ij}^{k,1} \cdots w_{ij}^{k,n} \rangle \mid 1 \le k \le m, \mathbf{W}_{ij} \in \mathcal{W}\}$$

$$\mathcal{W}^{*,c} = \{\langle w_{ij}^{1,l} \cdots w_{ij}^{m,l} \rangle \mid 1 \le l \le n, \mathbf{W}_{ij} \in \mathcal{W}\}$$

と表記する．各集合 $\mathcal{W}^{r,c}$，$\mathcal{W}^{r,*}$，$\mathcal{W}^{*,c}$ を重み集合と捉え，それぞれクラスタリングを通じた重み共有とハフマン符号化を適用する．なお，要素を単位とする場合は 3 章で示した一般的な圧縮方法と等価となる．また，行ベクトルまたは列ベクトルを単位とすることで，複数の重みが一塊として扱われることとなり，より一層の圧縮が期待できる．

5 評価実験

5.1 データセットと実験設定

下記に示す 2 つのデータセットを対象にモデルを構築し，実験を行った．

MNIST： MNIST データセット [1] は，一桁の手書き数字画像に関するデータセット（データ数 70,000，クラス数 10）である．実験では，60,000 事例を訓練例とし，10,000 事例をテスト例として利用した．また文献 [6] と同様に，畳み込み層，畳み込み層＋カプセルネットワーク，全結合層＋カプセルネットワークの 3 部でモデルを構成した．なお，カプセル間の重み行列は 8 行 × 16 列である．

表 1: 実験結果：モデル容量（単位は MByte）

k	共有化			符号化		
	$\mathcal{W}^{r,c}$	$\mathcal{W}^{r,*}$	$\mathcal{W}^{*.c}$	$\mathcal{W}^{r,c}$	$\mathcal{W}^{r,*}$	$\mathcal{W}^{*.c}$
MNIST：枝刈り前 50.78，枝刈り後 44.63						
32	40.81	41.37	41.55	14.47	11.11	11.10
64	40.90	41.55	41.60	14.55	11.20	11.20
128	40.89	41.63	41.92	14.82	11.22	11.21
CIFAR-10：枝刈り前 99.89，枝刈り後 99.20						
32	90.20	92.14	92.22	88.45	87.46	87.45
64	90.50	92.40	92.45	88.56	87.50	87.45
128	90.75	92.88	92.93	88.90	87.58	87.49

CIFAR-10： CIFAR-10 データセット [2] は，物体認識を意図した画像データセット（データ数 60,000，クラス数 10）である．実験では，50,000 事例を訓練例とし，10,000 事例をテスト例として利用した．またモデルは，VGG-16[7] を基本とし，全結合層 3 層をカプセルネットワーク 2 層で置き換えた構成とした．MNIST データの場合と同様，カプセル間の重み行列は 8 行 × 16 列である．

5.2 実験結果と考察

提案する軽量化手法の効果を確認するため，枝刈りに関する閾値を $\epsilon = 0.25 \times$ 重み標準偏差，重み共有化のためのクラス数を $k \in \{32, 64, 128\}$ として実験を行い，圧縮手法適用前後での識別精度およびモデル容量を計測した．なおモデル容量は，圧縮対象としたカプセルネットワーク中の重み行列（$\mathbf{W}_{ij}$）と結合係数（c_{ij}）のみを対象に計測を行っている．また各パラメタ設定毎に，訓練例とテスト例を変化させながら 10 回の試行を行い，その平均を算出した．実験結果を表 1 および表 2 に示す．両表において，$\mathcal{W}^{r,c}$，$\mathcal{W}^{r,*}$，$\mathcal{W}^{*.c}$ はそれぞれ要素，行，列を単位として軽量化手法を適用した場合を表す．

実験結果より，両データセットにおいても，クラスタ数 k の値が大きいほど，符号化時の圧縮率が高いことが分かる．また条件が同じ場合，$\mathcal{W}^{r,c}$ と比較して $\mathcal{W}^{r,*}$ と $\mathcal{W}^{*.c}$ の方が圧縮効率が高く，提案した行，列を単位とする軽量化が有効に働いていることが確認できる．一方で，精度に関しては条件が同じ場合に $\mathcal{W}^{r,c}$ が優れ

[1] http://yann.lecun.com/exdb/mnist/

[2] https://www.cs.toronto.edu/ kriz/cifar.html

表 2: 実験結果：精度（単位は%）

k	クラスタ内平均			セントロイド		
	$\mathcal{W}^{r,c}$	$\mathcal{W}^{r,*}$	$\mathcal{W}^{*.c}$	$\mathcal{W}^{r,c}$	$\mathcal{W}^{r,*}$	$\mathcal{W}^{*.c}$
MNIST：枝刈り前 99.65，枝刈り後 99.62						
32	82.25	81.25	70.58	99.53	89.45	78.81
64	88.54	89.78	87.86	99.54	97.37	90.31
128	88.45	85.54	84.41	99.53	98.78	96.13
CIFAR-10：枝刈り前 63.26，枝刈り後 64.36						
32	53.54	53.21	49.54	64.33	55.57	44.71
64	45.54	50.05	42.58	64.41	53.34	49.27
128	56.87	53.85	45.89	64.38	56.61	53.88

ており，精度とモデル圧縮との間にトレードオフがあることが分かる．また，重み共有化における代表値として，クラスタ内平均を用いる場合とセントロイドを用いる場合とを比較すると，すべての場合においてセントロイドを用いた場合が精度が高いことが確認できる．これは, 平均値は外れ値に影響を受けやすく，結果として精度が低下したからではないかと考察できる．

実験結果からも，モデルの圧縮率と精度にはトレードオフの関係が認められる．ここで，両者のバランスを考慮した手法の評価を行うため，容量削減率と精度保持率を導入する．枝刈り適用前のモデルサイズを M_{org}，軽量化手法 x 適用後のモデルサイズを M_x と表記する．一方，枝刈り適用前の精度を A_{org}，x 適用後の精度を A_x と表記する．このとき，x による容量削減率 MRR_x と精度保持率 ARR_x をそれぞれ

$$MRR_x = 1 - M_x/M_{org} \qquad ARR_x = A_x/A_{org}$$

と定義する．

表 3 に各手法による容量削減率と精度保持率，両者の調和平均とその順位を示す．また図 1 に，容量削減率と精度保持率とのプロット図を示す．

表 3 より，容量削減率と精度保持率の調和平均に着目すると，提案した行，もしくは列単位での重み共有化と符号化が有効に働いていることが分かる．また図 1 より，提案手法で得られる結果の多くが極大である，すなわち容量削減率と精度保持率の両方が他の手法より劣ることはないということが確認できる．

6 まとめと今後の課題

本研究では，カプセルネットワークを対象とした軽量化の一手法として，ベクトル単位での重み共有化と符号化を提案した．また 2 つの実データを用いた実験により，その効果を確認した．

表 3: 容量削減率と精度保持率 (単位は%)

	k	MMR	ARR	F1	Rank
MNIST					
$\mathcal{W}^{r,c}$	32	71.50	99.88	83.34	6
	64	71.35	99.89	83.24	7
	128	70.82	99.88	82.87	8
$\mathcal{W}^{r,*}$	32	78.12	89.76	83.54	5
	64	77.94	97.71	86.72	2
	128	77.90	99.13	87.24	1
$\mathcal{W}^{*.c}$	32	78.14	79.09	78.61	9
	64	77.94	90.63	83.81	4
	128	77.92	96.47	86.21	3
CIFAR-10					
$\mathcal{W}^{r,c}$	32	11.45	101.69	20.59	7
	64	11.34	101.82	20.41	8
	128	11.00	101.77	19.86	9
$\mathcal{W}^{r,*}$	32	12.44	87.84	21.80	1
	64	12.40	84.32	21.63	4
	128	12.32	89.49	21.66	3
$\mathcal{W}^{*.c}$	32	12.45	70.68	21.18	6
	64	12.45	77.88	21.47	5
	128	12.41	85.17	21.67	2

今後の課題としては，より大規模かつ多様なデータを用いた提案手法の評価や，重み共有のためのクラスタリング手法と代表値算出手法の改善に加え，拡張されたカプセルネットワーク（例えば [8, 9, 10]）への対応などが挙げられる．

謝辞： 本研究の一部は，JSPS 科研費 17K00315 の助成を受けたものです．

参考文献

[1] G. Hinton, O. Vinyals, and J. Dean : Distilling the knowledge in a neural network, *NIPS 2014 Deep Learning Workshop*, pp.1–9, 2014.

[2] Y. Cheng, F. X. Yu, R. S. Feris, S. Kumar, A. Choudhary, and S. Chang : An Exploration of Parameter Redundancy in Deep Networks with Circulant Projections, *Proc.of the 2015 IEEE International Conference on Computer Vision*, pp. 2857-2865, 2015.

[3] S. Han, J. Pool, J. Tran, and W. J. Dally : Learning both weights and connections for efficient neural networks, *Proc. of the 28th Inter-*

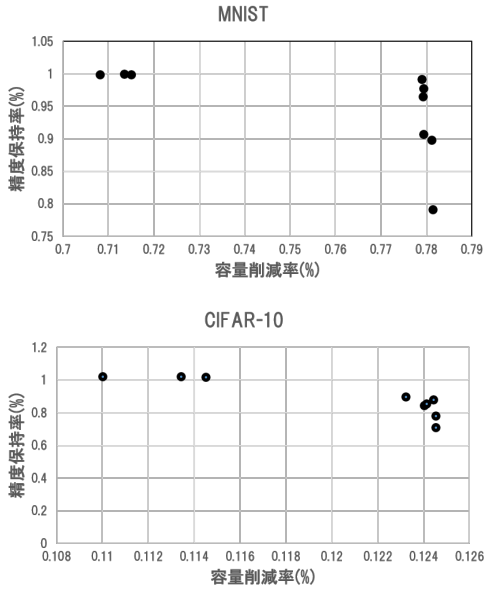

図 1: 削減率と精度のプロット図

national Conference on Neural Information Processing Systems, Vol.1, pp.1135-1143, 2015.

[4] Y. Gond, L. Liu, M. Yang, and L. Bourdev : Compressing deep convolutional networks using vector quantization, *International Conference on Learning Representations 2015*, 2015.

[5] S. Han, H. Mao, W. J. Dally : Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, *International Conference on Learning Representations 2016*, 2016.

[6] S. Sabour, N. Frosst, G. E. Hinton : Dynamic Routing Between Capsules, *Advances in Neural Information Processing Systems 30*, pp.3856–3866, 2017.

[7] K. Simonyan and A. Zisserman : Very Deep Convolutinal Networks for Large-Scale Image Recognition, *International Conference on Learning Representations 2015*, 2015.

[8] T. Jeong, Y. Lee, and H. Kim : Ladder Capsule Network, *Proc. of Machine Learning Research*, Vol.97, pp.3071–3079, 2019.

[9] S. Verma and Z.-L. Zhang : Graph Capsule Convolutional Neural Networks, *Proc. of Joint ICML and IJCAI Workshop on Computational Biology*, 2018.

[10] Z. Xinyi and L. Chen : Capsule Graph Neural Network, *International Conference on Learning Representations 2019*, 2019.