OrthCaps: An Orthogonal CapsNet with Sparse Attention Routing and Pruning

Supplementary Material

1. Symbols and abbreviation used in this paper

Symbol	Description
OrthCaps	Orthogonal Capsule Network
OrthCaps-S	Shallow network variant of OrthCaps
OrthCaps-D	Deep network variant of OrthCaps
x	Input image
l	Layer index
Φ^0	Features from initial convolutional layer
$u_{l,i}/(U_l)$	Capsules/ (matrix) at layer l
$v_{l+1,i}/(V_{l+1})$	Capsules/ (matrix) at layer $l+1$
n	Capsule count in layer l
m	Capsule count in layer $l+1$
d	Capsule dimension
W	Feature map width
H	Feature map height
B	Batch size
u_{flat}	Flattened capsules
$u_{ordered}$	Capsules ordered by their L_2-norm
u_{pruned}	Pruned capsules
$m_i/(M)$	Mask column/matrix for pruned capsule layer
$t_{ij}/(T)$	Cosine similarities/ (matrix) for pruned capsule layer
θ	Threshold for pruned capsule layer
Q, K, V	Attention routing components: Query, Key, Value
W_Q, W_K, W_V	Weight matrices for Q, K, V
$c_{i,j}/(C)$	Coupling coefficients/ (matrix) for attention routing
$s_{l+1,j}/(S_l)$	Votes/ (matrix) for attention routing
W	Weight matrix for simplified attention routing
g	Activation function
H	Householder matrix
a_i	Unit vector in Householder matrix formulation
b_i	Learnable vector in Householder matrix

2. experimental Setups

2.1. hyperparameters

Hyperparameter	Value
Batchsize	512 (4 parallelled)
Learning rate	5e-3
Weight decay	5e-4
Optimizer	AdamW
Scheduler	CosineAnnealingLR and 5-cycle linear warm-up
Epochs	300
Data augmentation	RandomHorizonFlip, RandonClip with padding of 4
Dropout	0.25
m^+	0.9
m^-	0.1
λ	0.5
θ	0.7
d	16

2.2. Setups

In this section, we describe the necessary Python library and corresponding version for the experiments in the main paper.

Library	Version
pytorch	1.12.1
numpy	1.24.3
opencv-python	4.7.0.72
pandas	2.0.2
pillow	9.4.0
torchvision	0.13.1
matplotlib	3.7.1
icecream	2.1.3
seaborn	0.12.0

3. HouseHolder Orthogonalization

3.1. Proof of Lemma 1

Assumption:

$V_{l+1} = \{v_{l+1,1}, v_{l+1,2}, \ldots, v_{l+1,m}\}$ is a set of m capsule vectors in layer $l+1$ of the network. W is an orthogonal matrix, i.e. $W^TW = I$.

Proof:

we aim to prove that the cosine similarity t_{ij} between any two capsules $v_{l+1,i}$ and $v_{l+1,j}$ remains constant after multiplied by W. For an orthogonal matrix W, the dot product and vector lengths are preserved. Let $\tilde{v}_{l+1,i} = Wv_{l+1,i}$, $\tilde{v}_{l+1,j} = Wv_{l+1,j}$ denote the transformed vectors. Thus, we have:

$$\begin{aligned} \tilde{v}_{l+1,i} \cdot \tilde{v}_{l+1,j} &= (Wv_{l+1,i})^T(Wv_{l+1,j}) \\ &= v_{l+1,i}^T W^T W v_{l+1,j} \\ &= v_{l+1,i}^T v_{l+1,j} \end{aligned} \tag{1}$$

and

$$\begin{aligned} \|\tilde{v}_{l+1,i}\|_2 &= \sqrt{(Wv_{l+1,i})^T(Wv_{l+1,i})} \\ &= \sqrt{v_{l+1,i}^T W^T W v_{l+1,i}} \\ &= \sqrt{v_{l+1,i}^T v_{l+1,i}} \\ &= \|v_{l+1,i}\|_2 \end{aligned} \tag{2}$$

Thus, we obtain transformed cosine similarity $\tilde{t}_{ij}$ in Eq. (3):

$$\begin{aligned} \tilde{t}_{ij} &= \frac{(\tilde{v}_{l+1,i}) \cdot (\tilde{v}_{l+1,j})}{\|\tilde{v}_{l+1,i}\|_2 \|\tilde{v}_{l+1,j}\|_2} \\ &= \frac{v_{l+1,i} \cdot v_{l+1,j}}{\|v_{l+1,i}\|_2 \|v_{l+1,j}\|_2} \\ &= t_{ij} \end{aligned} \tag{3}$$

Therefore, orthogonal transformations preserve the cosine similarity between any two vectors.

3.2. Proof of Lemma 3

Proof:

Let W represent one of W_Q,W_K,W_V as W can be expressed as

$$W = H_0 H_1 \dots H_{d-1} \tag{4}$$

where $H_i = I - 2a_i a_i^T$. We have

$$W^T W = H_{d-1}^T \dots H_1^T H_0^T H_0 H_1 \dots H_{d-1} \tag{5}$$

We demonstrate that H_i is orthogonal, i.e. $H_i^T H_i = I$. This is obvious, as

$$\begin{aligned} H_i^T H_i &= (I - 2a_i a_i^T)^T (I - 2a_i a_i^T) \\ &= I - 4a_i a_i^T + 4a_i a_i^T = I \end{aligned} \tag{6}$$

Therefore, Equation (5) can be written as $W^T W = \underbrace{I \dots I}_{d} = I$.

3.3. Householder as a Regularization Technique

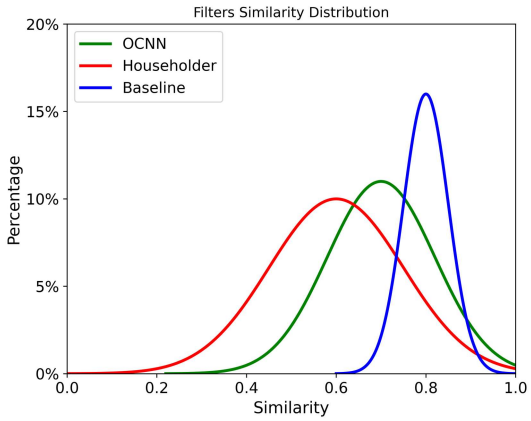

Figure 1. The normalized histogram of pairwise filter similarities in standard ResNet34 with different regularizers. HouseHolder orthogonalization method shows the best performance of descending filter similarity.

We demonstrate Householder's role as a regularization technique for neural networks. For ResNet18, we flatten and concatenate convolutional kernels into a matrix W, and orthogonalize it to minimize off-diagonal elements, which reduces channel-wise filter similarity and redundancy. To quantify these properties, we used Guided Backpropagation to dynamically visualize the activations[38]. Compared to directly computing the covariance matrix of convolutional kernels, The gradient-based covariance matrix offers a more comprehensive view of the dynamic behavior of filters. We define the gradients from Guided Backpropagation as G and compute its gradient correlation matrix $corr(G)$ as:

$$(\text{diag}(K_{GG}))^{-\frac{1}{2}} K_{GG} (\text{diag}(K_{GG}))^{-\frac{1}{2}} \tag{7}$$

where $K_{GG} = \frac{1}{M}\left((G - \mathbb{E}[G])(G - \mathbb{E}[G])^T\right)$, M is the number of channels. Fig. 1 of the off-diagonal elements of $corr(G)$ shows a left-shifted distribution for the Householder-regularized model, confirming its effectiveness in enhancing filter diversity and reducing redundancy.

4. Dynamic Routing in Capsule Network

Algorithm 1 Dynamic Routing

procedure ROUTING($\hat{u}_{j|i}$, r, l)
  for all capsule i in layer l and capsule j in layer $(l+1)$ **do** $b_{ij} \leftarrow 0$
  for T iterations **do**
    for all capsule i in layer l **do** $c_i \leftarrow \text{softmax}(b_i)$
    for all capsule j in layer $(l+1)$ **do** $s_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$
    for all capsule j in layer $(l+1)$ **do** $v_j \leftarrow \text{squash}(s_j)$
    for all capsule i in layer l and capsule j in layer $(l+1)$ **do** $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$
  return v_j

Algorithm 1 describes the dynamic routing algorithm. This algorithm allows lower-level capsule output vectors to be allocated to higher-level capsules based on their similarity, thereby achieving an adaptive feature combination. However, as evident from $\sum_i c_{ij} \hat{u}_{j|i}$, each higher-level capsule is a weighted sum of lower-level capsules. The higher-level capsules are fully connected with the lower level. Furthermore, the routing algorithm fundamentally serves as an unsupervised clustering process for capsules, requiring r iterations to converge the coupling coefficients c. It's crucial to strike a balance in choosing r: an inadequate number of iterations may hinder convergence of c, impairing routing efficacy, while an excessive count increases computational demands.

In Conclusion, it is crucial to introduce a straightforward, iterative-free routing algorithm.