ReAF

Reducing approximation of channels by reducing feature reuse within convolution

Zhu, Baozhou; Al-Ars, Zaid; Hofstee, H. Peter

DOI
10.1109/ACCESS.2020.3024252

Publication date
2020

Document Version
Final published version

Published in
IEEE Access

Citation (APA)
Zhu, B., Al-Ars, Z., & Hofstee, H. P. (2020). ReAF: Reducing approximation of channels by reducing feature reuse within convolution. *IEEE Access*, *8*, 169957-169965. Article 9197593. https://doi.org/10.1109/ACCESS.2020.3024252

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 9, 2020, accepted September 8, 2020, date of publication September 15, 2020,
date of current version September 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3024252

REAF: Reducing Approximation of Channels by Reducing Feature Reuse Within Convolution

ZHU BAOZHOU[1,2], ZAID AL-ARS[1], (Member, IEEE), AND H. PETER HOFSTEE[1,3], (Member, IEEE)

[1]Department of Quantum and Computer Engineering, Delft University of Technology, 2628 Delft, The Netherlands
[2]College of Computer, National University of Defense Technology, Changsha 410073, China
[3]IBM, Austin, TX 78758, USA

Corresponding author: Zhu Baozhou (b.zhu-1@tudelft.nl)

This work was supported by the Dutch National e-Infrastructure through SURF Cooperative.

ABSTRACT High-level feature maps of Convolutional Neural Networks are computed by reusing their corresponding low-level feature maps, which brings into full play feature reuse to improve the computational efficiency. This form of feature reuse is referred to as feature reuse *between* convolutional layers. The second type of feature reuse is referred to as feature reuse *within* the convolution, where the channels of the output feature maps of the convolution are computed by reusing the same channels of the input feature maps, which results in an approximation of the channels of the output feature maps. To compute them accurately, we need specialized input feature maps for every channel of the output feature maps. In this paper, we first discuss the approximation problem introduced by full feature reuse *within* the convolution and then propose a new feature reuse scheme called Reducing Approximation of channels by Reducing Feature reuse (REAF). The paper also shows that group convolution is a special case of our REAF scheme and we analyze the advantage of REAF compared to such group convolution. Moreover, we develop the REAF+ scheme and integrate it with group convolution-based models. Compared with baselines, experiments on image classification demonstrate the effectiveness of our REAF and REAF+ schemes. Under the given computational complexity budget, the Top-1 accuracy of REAF-ResNet50 and REAF+-MobileNetV2 on ImageNet will increase by 0.37% and 0.69% respectively. The code and pre-trained models will be publicly available.

INDEX TERMS Convolutional neural networks, group convolution, feature reuse.

I. INTRODUCTION

Convolutional Neural Networks (CNNs) have achieved a series of breakthroughs on non-trivial visual tasks [1]–[5]. The features in a dataset can be learned in an end-to-end manner by CNNs with minimal human effort and can be transferred to diverse visual tasks [6]. Accordingly, researchers are dedicated to designing better networks for learning representations [7]–[10] instead of handcrafted features.

To enrich the representational power of CNNs, recent work investigates various aspects of CNN network architecture [11]–[14]. Constructing deep CNNs by stacking building blocks of the same shape is an effective strategy [11] since higher layers learn more abstract and invariant representations [15], [16]. Inherited from this, networks with skip connections [12], [17] enable the training CNNs with extreme depth. Inspired from the Hebbian principle and multi-scale processing, multi-branch CNNs [18] achieve compelling accuracy if the topology of each branch is carefully designed, and multiple branches are expected to approximate large and dense layers of powerful representational capability. Except for depth, the width of a network is an essential dimension to increase the model capability [19]. Exposing the new dimension of cardinality [20] or deploying the new approaches [8], [21], [22] can enlarge the representational ability of the model. To address the limitation of spatial locality in convolution, the attention mechanism [21], [23] is used to capture a larger feature interaction. Using automated strategy, Neural Architecture Search achieves state-of-the-art accuracy [22] and platform-aware efficiency [24].

When it comes to the efficiency of designing CNNs, feature reuse is key to making it feasible [16], [25]–[27]. More precisely, the features computed by earlier layers will be reused by the latter layers, which is the feature reuse *between* convolutional layers and is popular in both a plain network [28] and a multi-branch network [18]. The pursuit of maximizing

The associate editor coordinating the review of this manuscript and approving it for publication was Shiping Wen.

feature reuse between convolutional layers is an important concept in designing these networks. Deeper CNNs encourage more feature reuse, which resulted in designing VGG-net [28] and ResNet [17]. Deep CNNs with identity mapping have a problem of diminishing feature reuse in [29], which motivates the development of wide ResNet [19], ResNet with stochastic depth [30], and DenseNet [31]. Also, feature reuse plays a critical role *within* the convolution in addition to *between* the convolutional layers. Specifically, all the channels of the output activations of the convolution are computed by reusing all and the same channels of the input activations.

Feature reuse is at the heart of the theoretical advantages behind deep learning and explains the power of distributed representations [16]. However, it is a limitation of the representational capability of networks since the reused features are an approximation of accurate features. Reducing or eliminating feature reuse from a given model will result in comparable or higher model accuracy, which supports the approximation drawback of feature reuse. There is some research investigating eliminating feature reuse *between* convolutional layers. For example, CondenseNet [25] removes such connections between layers to avoid superfluous feature reuse in the network architecture. Besides, the lottery ticket [32], selective allocation of channels [33], pruning [34], [35], and group convolution [13], [36] can all be regarded as methods for reducing feature reuse *within* the convolution as analyzed in our work, which has not been pointed out in their original illustration. We initially pointed out that feature reuse *within* the convolution leads to the problem of approximation of the channels. Thus, by reducing feature reuse we can reduce the approximation *within* the convolution, which makes the calculation of the channels more accurate.

Our main contributions are summarized as follows.

- To our best knowledge, we are the first to point out and analyze the approximation problem introduced by the feature reuse *within* the convolution. To solve the problem, we propose the Reducing Approximation of channels by Reducing Feature reuse (REAF) scheme, which is a moderate version of feature reuse *within* the convolution.
- We compare our REAF scheme with group convolution and show that there are more merged channels in our REAF scheme than those in group convolution even though group convolution is a special case of our REAF scheme.
- We develop our REAF+ scheme with Bn and Relu layers as the parameterized operations and integrate it with group convolution-based models.
- We use extensive experiments to demonstrate the effectiveness of our REAF and REAF+ schemes.

II. RELATED WORK

A. MULTI-BRANCH CONVOLUTIONAL NETWORKS

To ease the difficulty of training deep neural networks, an adaptive gating unit is used in Highway networks [29], which evolves into identity mapping in ResNet [17]. Replacing the identity mapping with more residual blocks, shake-shake networks [37], and multi-residual networks [38] are extended to improve the accuracy and speed. FractalNets [39] and Multilevel ResNets [40] expand the multiple paths in a fractal and recursive way, respectively. The Inception series [18] aggregate the multifarious features of multi-scale with a careful configuration for each branch.

B. CONVOLUTION VARIANTS

To enrich the representational capability, deformable convolution [7], [41] and active convolution [42] augment the spatial sampling locations with additional offsets. Considering the computational complexity, group convolution [43], flattened convolution [44], tiled convolution [45], octave convolution [46], and dilated convolution [1] are developed as variants. Depthwise convolution [47] is an example of group convolution with the number of groups being the same as the number of channels. Based on these convolution variants, compact models are built, including IGCV series [48], MobileNet series [49], ShuffleNet series [50], and Espnet series [51].

III. METHOD

In this section, we analyze the limitation of the convolution, i.e., the approximation of the channels caused by the full feature reuse *within* the convolution. To address this problem, we propose our REAF scheme for convolution, which introduces specialization to compute the channels of the output feature maps. We optimize the REAF scheme and compare it with group convolution since group convolution is a special case of our scheme. Finally, we develop the REAF+ scheme to improve the performance of the group convolution-based models.

A. PROBLEM DEFINITION

The output activations $O \in R^{C_{out}\times H\times W}$ of the convolution are convolved between the input activations $I \in R^{C_{in}\times H\times W}$ and the weights $W \in R^{C_{out}\times C_{in}\times h\times w}$, where the batch size N is omitted. The channel-wise representation of the output activations O is shown as follows, where O_j refers to the j^{th} channel of the output activations O and $j = 0, \ldots, C_{out} - 1$.

$$O = \{O_0, O_1, \ldots, O_{C_{out}-1}\} \tag{1}$$

To study feature reuse for every individual channel, we construct a variable $A \in R^{C_{out}\times C_{in}\times H\times W}$ to compute all the channels of the output activations, where A_j is the input activation to compute the j^{th} channel of the output activation O_j. The feature reuse *within* the convolution is shown in the left of Fig. 1, where we reuse all the channels of the input activations I to calculate every individual channel of the output activations O. Therefore, O_j is convolved by the input activations A_j and the weights W_j as follows and $i = 0, \ldots, C_{in} - 1$.

$$O_j = \sum_{i=0}^{C_{in}-1} A_{ji} \times W_{ji} = \sum_{i=0}^{C_{in}-1} I_i \times W_{ji} \tag{2}$$

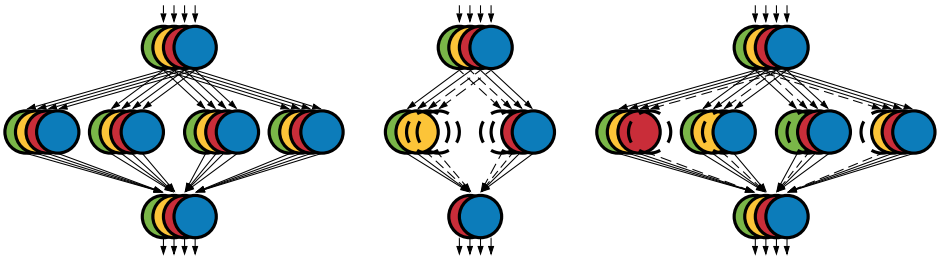

FIGURE 1. Illustration of convolution using different feature reuse schemes. Every circle stands for every channel of the feature maps. Circles on the top, in the middle, and on the bottom refer to the input feature maps, the input feature maps to compute every channel of the output feature maps, and the output feature maps, respectively. Circles in dashed lines indicate the absence of the channels.

The information of the CNN transitions gradually from spatial coding to channel coding by a hierarchy of representations. Regarding the feature reuse *between* the convolutional layers, the learned hierarchical representation makes it reasonable to save computational complexity for CNNs since the high-level features are composed of the low-level features. However, when it comes to feature reuse *within* the convolution, i.e., every channel of the output activations is computed by reusing the same input activations. The reused input activations are approximated tensors for all the channels of the output activations. As every channel of feature maps is considered as a feature detector [52], the input feature maps to compute every channel of the output feature maps of the convolution are expected to be customized and specialized to make the computation more accurate.

B. REDUCING APPROXIMATION OF CHANNELS BY REDUCING FEATURE REUSE

The approximation problem of the channels introduced by the feature reuse *within* the convolution can be expressed as $A_j = A_{j'}$ and $j, j' = 0, \ldots, C_{out} - 1$. To introduce specialized input feature maps for every channel of the output feature maps, a straightforward scheme for the convolution is shown in the middle of Fig. 1, where there is no feature reuse *within* the convolution at all. The input activations are divided into G_I groups, where $C_{in} = G_I \times A_{in}$ and A_{in} is the number of channels of every group to compute every channel of the output activations. We have $G_I = C_{out}$. The input activations to compute the j^{th} channel of the output activations are A_j, where A_j is part of the input activations I as follows. Index refers to the function that indexes the j^{th} group from I.

$$A_j = \text{Index}(I, j, A_{in}) \tag{3}$$

In this way, we do not reuse any channels of the input activations I to compute every channel of the output activations O. O_j is convolved by the input activations A_j and the weights W_j as follows.

$$\begin{aligned} O_j &= \sum_{i=0}^{A_{in}-1} A_{ji} \times W_{ji} \\ &= \sum_{i=0}^{A_{in}-1} \text{Index}(I, j, A_{in}) \times W_{ji} \end{aligned} \tag{4}$$

Considering computational complexity and distributed representations, it is inadvisable to remove feature reuse totally *within* the convolution. To keep feature reuse and introduce customized input activations to compute the channels, we introduce our scheme called REAF, which is a moderate version of feature reuse for convolution as shown in the right part of Fig. 1.

All the channels of the input activations I and the output activations O are divided into G_I and G_O groups respectively, and there is only one channel in every group as shown in the right part of Fig. 1. $C_{out} = G_O \times B_{out}$ and B_{out} is the number of channels of every group of the output activations. To keep feature reuse and introduce specialization, we draw G_M groups from the G_I groups of the input activations to compute every group of the output activations. If all the channels of the input activations I are considered as a set S of G_I elements, every element is a group. The number of the (G_M)-combinations is denoted using elementary combinatorics text as $C(G_I, G_M)$. All the (G_M)-combinations are enumerated as $E_0, \ldots, E_l, \ldots, E_{G_O-1}$. Here $l = \lfloor j/B_{out} \rfloor$ and $l = 0, \ldots, G_O - 1$. Since it is hard to get prior knowledge on how many times the groups of the input activations should be reused, we try to keep the homogeneity of computing channels. Therefore, we have $C(G_I, G_M) = G_O$. Index refers to the function that indexes G_M groups from I based on E_l and concatenate them together.

$$A_j = \text{Index}(I, E_l, A_{in}) \tag{5}$$

To compute different groups of the output activations, the reused G_M groups of the input activations are different from each other. O_j is convolved by the input activations A_j and the weights W_j as follows.

$$\begin{aligned} O_j &= \sum_{i=0}^{A_{in} \times G_M - 1} A_{ji} \times W_{ji} \\ &= \sum_{i=0}^{A_{in} \times G_M - 1} \text{Index}(I, E_l, A_{in}) \times W_{ji} \end{aligned} \tag{6}$$

C. OPTIMIZING THE CONFIGURATIONS OF OUR SCHEME

Given a convolution with a computational complexity budget, the configurations of $G_I - G_M - G_O$ can be optimized since they have an influence on the feature reuse *within* the

TABLE 1. ResNet50 and REAF-ResNet50 with a 4-3-4 template using the reformulation of the REAF scheme. Inside the brackets is the shape of a residual block, and outside the brackets is the number of stacked blocks on a stage. "C = 72" refers to the base width of the mode. "4-3-4" suggests that the REAF scheme with the configurations of $G_I = 4$, $G_M = 3$, and $G_O = 4$ is applied to the convolution. The numbers of parameters and FLOPs are comparable between these two models.

Stage	Output	ResNet50	REAF-ResNet50 (72-72, 4-3-4)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
conv2	56×56	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 72 \\ 3 \times 3, 72, 4\text{-}3\text{-}4 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 144 \\ 3 \times 3, 144, 4\text{-}3\text{-}4 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 288 \\ 3 \times 3, 288, 4\text{-}3\text{-}4 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 576 \\ 3 \times 3, 576, 4\text{-}3\text{-}4 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
Parameters		25.56×10^6	26.11×10^6
FLOPs		3.86×10^9	3.97×10^9

TABLE 2. ResNeXt50 and REAF-ResNet50 with a 4-3-4 template using the reformulation of the REAF scheme. Inside the brackets is the shape of a residual block, and outside the brackets is the number of stacked blocks on a stage. "C = 64" refers to the base width of the mode. "4-3-4" suggests that the REAF scheme with the configurations of $G_I = 4$, $G_M = 3$, and $G_O = 4$ is applied to the convolution. "4" refers to the number of groups $G = 4$ in ResNeXt50. The numbers of parameters and FLOPs are comparable between these two models.

Stage	Output	ResNeXt50	REAF-ResNet50 (64-64, 4-3-4)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
conv2	56×56	$\begin{bmatrix} 1 \times 1, 64, 4 \\ 3 \times 3, 64, 4 \\ 1 \times 1, 256, 4 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 72, 4\text{-}3\text{-}4 \\ 3 \times 3, 72, 4\text{-}3\text{-}4 \\ 1 \times 1, 256, 4\text{-}3\text{-}4 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1 \times 1, 128, 4 \\ 3 \times 3, 128, 4 \\ 1 \times 1, 512, 4 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 144, 4\text{-}3\text{-}4 \\ 3 \times 3, 144, 4\text{-}3\text{-}4 \\ 1 \times 1, 512, 4\text{-}3\text{-}4 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1 \times 1, 256, 4 \\ 3 \times 3, 256, 4 \\ 1 \times 1, 1024, 4 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 288, 4\text{-}3\text{-}4 \\ 3 \times 3, 288, 4\text{-}3\text{-}4 \\ 1 \times 1, 1024, 4\text{-}3\text{-}4 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1 \times 1, 512, 4 \\ 3 \times 3, 512, 4 \\ 1 \times 1, 2048, 4 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 576, 4\text{-}3\text{-}4 \\ 3 \times 3, 576, 4\text{-}3\text{-}4 \\ 1 \times 1, 2048, 4\text{-}3\text{-}4 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
Parameters		20.89×10^6	20.39×10^6
FLOPs		2.98×10^9	2.72×10^9

convolution and the number of the merged channels. When $G_I - G_M$ remains unchanged and G_O increases, the feature reuse within the group reduces since every group of the output activations is computed by reusing different input activations. On the contrary, the feature reuse between the groups increases since the number of different channels between the reused input activations of computing different groups of the output activations decreases. When G_O remains unchanged and $G_I - G_M$ increases, the feature reuse *within* the convolution reduces while the number of the merged channels decreases. In this paper, we focus on studying the extreme case of $G_M = G_I - 1$ since $G_M = 1$ (i.e., group convolution) has been explored in [20]. Given a convolutional neural network, we apply our scheme to its convolutions and optimize the configurations. Taking ResNet50 as an example, and the overall REAF-ResNet50 architecture, i.e., apply our scheme on ResNet50, is listed in as shown in Table 1. We keep the topology and computational complexity of the model

TABLE 3. Experimental results of REAF-ResNet50 with different configurations.

Model	Top-1 err.	Top-5 err.	GFLOPs	#Params (M)
ResNet50 Baseline	22.06%	5.54%	1.22	23.71
REAF-ResNet50 (C=C'=64, 16-15-16)	21.70%	5.52%	1.18	23.01
REAF-ResNet50 (C=C'=64, 8-7-8)	21.57%	5.54%	1.14	22.30
REAF-ResNet50 (C=C'=70, 5-4-5)	20.89%	5.14%	1.22	24.11
REAF-ResNet50 (C=C'=72, 4-3-4)	20.58%	5.11%	1.24	24.32
REAF-ResNet50 (C=C'=72, 3-2-3)	21.80%	5.42%	1.20	23.20
REAF-ResNet50 (C=C'=80, 2-1-2)	22.52%	5.72%	1.22	23.75
REAF-ResNet50 (C=88, C'=56, 8-6-28)	22.78%	5.73%	1.23	23.60
REAF-ResNet50 (C=77, C'=63, 7-5-21)	21.94%	5.65%	1.18	22.76
REAF-ResNet50 (C=72, C'=75, 6-4-15)	22.26%	5.44%	1.23	23.74
REAF-ResNet50 (C=80, C'=70, 5-3-10)	22.14%	5.84%	1.21	23.23
REAF-ResNet50 (C=88, C'=72, 4-2-6)	21.51%	5.69%	1.23	23.40
REAF-ResNet50 (C=C'=90, 3-1-3)	21.50%	5.61%	1.25	23.66
REAF-ResNet50 (C=96, C'=56, 8-5-56)	20.82%	5.29%	1.23	23.21
REAF-ResNet50 (C=84, C'=70, 7-4-35)	20.91%	5.24%	1.23	23.50
REAF-ResNet50 (C=78, C'=80, 6-3-20)	20.77%	5.36%	1.21	23.22
REAF-ResNet50 (C=90, C'=80, 5-2-10)	20.80%	5.27%	1.23	23.36
REAF-ResNet50 (C=C'=96, 4-1-4)	21.30%	5.39%	1.25	23.44
REAF-ResNet50 (C=88, C'=70, 8-4-70)	20.56%	4.96%	1.21	23.00
REAF-ResNet50 (C=98, C'=70, 7-3-35)	21.35%	5.28%	1.24	23.29
REAF-ResNet50 (C=C'=90, 6-2-15)	20.58%	5.11%	1.25	23.56
REAF-ResNet50 (C=C'=100, 5-1-5)	21.58%	5.96%	1.24	23.19
REAF-ResNet50 (C=64, C'=112, 8-3-56)	20.71%	5.33%	1.21	23.59
REAF-ResNet50 (C=98, C'=84, 7-2-21)	21.38%	5.41%	1.21	22.76
REAF-ResNet50 (C=C'=102, 6-1-6)	**20.28%**	**5.06%**	1.21	22.74
REAF-ResNet50 (C=90, C'=84, 9-3-84)	21.28%	5.23%	1.20	22.68
REAF-ResNet50 (C=104, C'=84, 8-2-28)	21.11%	5.06%	1.21	22.71
REAF-ResNet50 (C=C'=105, 7-1-7)	21.08%	5.27%	1.22	22.75
REAF-ResNet50 (C=90, C'=108, 9-2-36)	20.59%	5.23%	1.24	23.58
REAF-ResNet50 (C=C'=112, 8-1-8)	20.46%	5.18%	1.28	23.75

unchanged for a fair comparison. We adopt the REAF scheme for the middle convolution of the bottleneck and adjust its width.

As shown in Table 3, the experimental results of REAF-ResNet50 on CIFAR-100 classification are presented. C and C' refer to the base number of the input and output channels of the middle convolution of the bottleneck and $C = C'$ in default. When applying the REAF scheme to the middle convolution, the accuracy will improve with a wide range of configurations. When $G_I - G_M = 1$, REAF-ResNet50 with $G_I = 4$ achieves the best Top-1 accuracy among all the variants and 1.48% better than the baseline. With the increase and decrease of the number of the groups from $G_I = 4$, the accuracy will degrade, which suggests that $G_I = 4$ refers to an optimized configuration of the REAF scheme for the ResNet50.

D. ADVANTAGE OVER GROUP CONVOLUTION

Group convolution [20] has been widely adopted in the design of CNNs since it exposes a new dimension, i.e., the size of the set of transformations. Meanwhile, group convolution benefits from reducing feature reuse *within* the convolution as explained in our work, which has not been pointed out by the original reference [20]. Group convolution is an example of our REAF scheme when $G_M = 1$. The number of merged channels in our proposed scheme is $G_M - 1$ times larger than that in group convolution. Also note that the number of merged channels in the REAF scheme with the optimized configuration is much more than that in group convolution with the optimized number of groups G. Based on the experiments in [20], a larger number of cardinality indicates a more effective choice for a given model and the number of merged channels is 4 in group convolution with the optimized groups $G = 32$. For example, in our proposed scheme with the optimized configuration 4-3-4 and the number of merged channels is 54.

E. REAF+ SCHEME

We propose the REAF+ scheme, which adopts parameterized or parameter-free operations to enable $A_j \neq A_{j'}$. Applying the REAF+ scheme, the performance of the models, including the group convolution-based models or REAF scheme-based models, will improve further. When the Bn and Relu layers are included, O_j is convolved by the output activations O' of the last convolution and the weights W_j as follows.

$$\begin{aligned} O_j &= \text{Conv}(A_j, W_j) = \text{Conv}(I, W_j) \\ &= \text{Conv}(\text{Relu}(\text{Bn}(O')), W_j) \end{aligned} \tag{7}$$

Taking the parameterized operations of Bn and Relu layers as an example, the REAF+ scheme can be expressed as follows. There are g_O pairings of Bn and Relu layers introduced when the output activations are divided into g_O groups.

$$O_j = \text{Conv}(A_j, W_j) = \text{Conv}(\text{Relu}_l(\text{Bn}_l(O')), W_j) \tag{8}$$

TABLE 4. Comparisons of REAF-ResNet and ResNet on CIFAR-100 classification.

Model	Top-1 err.	Top-5 err.	GFLOPs	#Params (M)
ResNet18 Baseline	22.21%	6.12%	0.56	11.22
REAF-ResNet18 (C=72, 4-3-4)	**21.86%**	**5.82%**	0.56	11.17
ResNet34 Baseline	21.30%	5.32%	1.16	21.33
REAF-ResNet34 (C=72, 4-3-4)	**20.86%**	**5.23%**	1.13	18.10
ResNet101 Baseline	21.20%	5.40%	2.44	42.70
REAF-ResNet101 (C=72, 4-3-4)	**20.44%**	**4.89%**	2.48	43.81

TABLE 5. Comparisons of REAF-ResNet and ResNet on ImageNet classification.

Model	Top-1 err.	Top-5 err.	GFLOPs	#Params (M)
ResNet18 Baseline	29.56%	10.36%	1.81	11.69
REAF-ResNet18 (C=72, 4-3-4)	**29.29%**	**10.28%**	1.95	11.74
ResNet50 Baseline	23.65%	6.89%	3.86	25.56
REAF-ResNet50 (C=88, C'=70, 8-4-70)	**23.18%**	**6.90%**	3.74	24.85
REAF-ResNet50 (C=72, 4-3-4)	23.28%	6.74%	3.97	26.11
WideResNet50 Baseline (widen=2.0)	22.09%	6.08%	11.44	68.88
REAF-WideResNet50 (C=136, 4-3-4)	**21.89%**	**6.10%**	10.50	63.11
ResNet101 Baseline	22.09%	6.18%	7.57	44.55
REAF-ResNet101 (C=72, 4-3-4)	**21.84%**	**5.98%**	7.79	45.83

TABLE 6. Comparisons between our REAF scheme and group convolution on classification.

Model	Top-1 err.	Top-5 err.	GFLOPs	#Params (M)
ResNeXt50 on CIFAR-100	22.89%	6.19%	1.02	19.04
REAF-ResNet50 on CIFAR-100	**22.15%**	**5.54%**	1.00	18.73
ResNeXt101 on CIFAR-100	21.70%	5.62%	1.96	33.60
REAF-ResNet101 on CIFAR-100	**20.59%**	**5.21%**	1.92	33.01
ResNeXt50 on ImageNet	25.04%	7.80%	2.98	20.89
REAF-ResNet50 on ImageNet	**24.30%**	**7.32%**	2.72	20.39

IV. EXPERIMENTAL RESULTS

We trained and evaluated our REAF-Net and REAF+-Net modes, i.e., applying our REAF and REAF+ schemes on baseline models, on CIFAR-100 and ImageNet ILSVRC2012 classification dataset [53].

A. EXPERIMENTS ON CIFAR-100 CLASSIFICATION

We apply our scheme on ResNet models and conduct experiments on low-resolution imagery CIFAR-100 datasets. We adopt crop translation and flipping data augmentation. We train 200 epochs in total and the learning rate decays at the steps of 60, 120, and 160 with a factor of 0.2. As the experimental results presented in Table 4, the Top-1 accuracy of our REAF-ResNet outperforms that of ResNet baseline with various depths using comparable computational complexity. Especially, our REAF-ResNet101 achieves a 1.76% top-1 accuracy improvement compared with the ResNet101 baseline.

B. EXPERIMENTS ON ImageNet CLASSIFICATION

To show the performance of our proposed scheme on high-resolution images and large datasets, we experiment with ResNet and REAF-ResNet on ImageNet classification. We train 100 epochs in total with a batch size of 256. The learning rate starts at 0.1 and decays every 30 epochs with a factor of 0.1. The weight decay is $1e-4$ and the momentum is 0.9. The image is resized for scale augmentation. A 224×224 crop is randomly sampled from an image or its horizontal flip, with per-pixel normalization. As summarized in Table 5, the Top-1 accuracy of our REAF-ResNet increases by 0.27%, 0.37%, and 0.25% for the depth of 18, 50 and 101 layers respectively compared to the ResNet baseline.

C. COMPARISONS WITH GROUP CONVOLUTION

This subsection reports on experiments on CIFAR-100 and ImageNet classification datasets to show the advantage of our proposed scheme over group convolution. We build ResNeXt and REAF-ResNet according to a given computational complexity budget, and their architectures can be found as shown in Table 2. The network architecture of ResNeXt50 and REAF-ResNet50 in the section of comparisons with group convolution are listed. The base width of ResNeXt and REAF-ResNet is $C = 64$. In ResNeXt, all the convolutions in the building bottlenecks of ResNet are replaced with the group convolutions $G = 4$. Specifically, the three convolutions in a bottleneck are replaced with

TABLE 7. Results of group convolution-based and our REAF+ scheme-based models on classification.

Model	Top-1 err.	Top-5 err.	GFLOPs	#Params (M)
MobileNetV2 on CIFAR-100	31.53%	9.34%	67.59×10^{-3}	2.37
REAF+-MobileNetV2 on CIFAR-100	**30.98%**	**8.46%**	67.90×10^{-3}	2.38
ResNeXt50 on CIFAR-100	20.13%	4.99%	1.36	23.18
REAF+-ResNeXt50 on CIFAR-100	**19.70%**	**4.76%**	1.36	23.19
MobileNetV2 on ImageNet	35.02%	13.62%	3.14	3.50
REAF+-MobileNetV2 on ImageNet	**34.33%**	**13.46%**	3.17	3.52

group convolutions. Similarly, all the convolutions in the building bottlenecks of REAF-ResNet adopt our proposed scheme with a configuration of $4 - 3 - 4$. In this way, the difference between group convolution and our proposed scheme, introduced by the number of merged channels, can be observed clearly. On the CIFAR-100 dataset, the Top-1 accuracy of the REAF-ResNet50 model is 0.74% better than that of ResNeXt50, and the gap is 1.11% for 101 layers as in Table 6. Compared to ResNeXt50, the Top-1 accuracy of REAF-ResNet50 increases by 0.74% on the ImageNet dataset.

D. EXPERIMENTS OF REAF+ SCHEME

As shown in Table 7, we apply the REAF+ scheme to a group convolution-based model and conduct experiments on the classification to show accuracy improvement. Applying the REAF+ scheme with $g_O = 2$ to the third convolution (i.e., the second pointwise convolution) of the bottleneck, the Top-1 accuracy of MobileNetV2 and ResNeXt50 on CIFAR-100 will improve by 0.55% and 0.43% respectively. The Top-1 accuracy of MobileNetV2 on ImageNet will increase by 0.69%.

E. EFFECTS ON LEARNED REPRESENTATION

In this section, we compare the class selectivity index for the features in the ResNet baseline and REAF-ResNet to interpret the effect of our proposed scheme on learned representation. The class selectivity index is computed for every channel of the feature maps, as $selectivity = (u_{max} - u_{-max})/(u_{max} + u_{-max})$. u_{max} represents the highest class-conditional mean activity and u_{-max} represents the mean of class-conditional mean activity across all other classes over given data distribution. Selectivity provides the degree to which features are being shared across classes, which is a central property of distributed representations and measure the extent of feature reuse. Using the validation dataset of the ImageNet, we calculate the selectivity for the features of "layer1", "layer2", "layer3", "layer4", and "avgpool", which are the output of the stage2, stage3, stage4, stage5, and AvgPool2d, respectively.

One trend is that the selectivity for the features of deep layers is more than that of earlier layers, which conforms to the observations in [23], [54] and can be attributed to the results of feature reuse *between* convolutional layers. Another trend identified in our work is that our REAF scheme

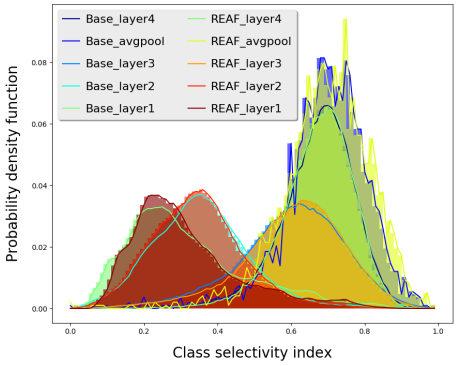

FIGURE 2. ResNet50 + REAF-ResNet50.

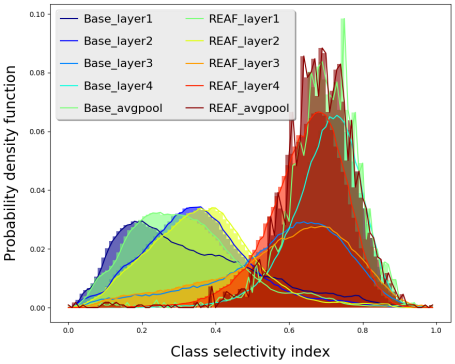

FIGURE 3. ResNet101 + REAF-ResNet101.

increases the selectivity for the features, which verifies its reduction of feature reuse *within* the convolution. The distribution of the selectivity for ResNet and REAF-ResNet appears to be closely matched, so we only analyze the layer, at which the distribution separates most. Figures 2 and 3 present the selectivity comparison for the features in ResNet and REAF-ResNet for 50 and 101 layers. The most distinct distribution of the class selectivity appears at "layer1" in Figures 2 and 3 and the selectivity for the features of the "layer1" in REAF-ResNet are more than that in ResNet. We compare the selectivity for the features in ResNet50 and ResNeXt50 (i.e., REAF-ResNet50 with the configuration of $G_M = 1$) in Figure 4, respectively. Since ResNeXt is an example of our REAF scheme, its learned representation is expected to increase the selectivity. The largest mismatch of the selectivity distribution for the features between ResNet50 and ResNeXt50 falls to "layer4", where ResNeXt50 exhibits more selectivity that ResNet50.

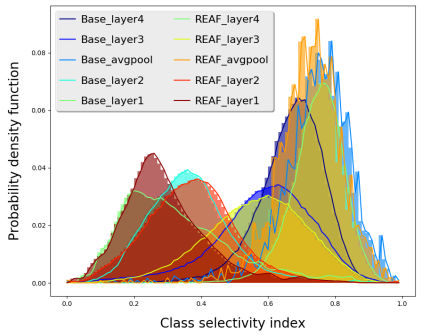

FIGURE 4. ResNet50 + ResNeXt50.

V. CONCLUSION

In this paper, we analyze the approximation problem introduced by the full use of feature reuse *within* the convolution, where the channels of the output activations are computed by reusing the same input activations. To make computing channels more accurate, we propose the REAF scheme, which is a moderate feature reuse version for convolution. Moreover, the configuration of the REAF scheme is optimized for a given CNN. Besides, we clarify the advantage of keeping more merged channels over group convolution. Also, we develop the REAF+ scheme and integrate it with the group convolution-based models. Last but not least, we present sufficient experiments on the classification to support our analysis concerning the REAF and REAF+ schemes.

REFERENCES

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[2] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei, "Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 82–92.

[3] H. Seong, J. Hyun, and E. Kim, "FOSNet: An end-to-end trainable deep neural network for scene recognition," *IEEE Access*, vol. 8, pp. 82066–82077, 2020.

[4] T. Vu, H. Jang, T. X. Pham, and C. Yoo, "Cascade RPN: Delving into high-quality region proposal network with adaptive convolution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1430–1440.

[5] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Feb. 2020.

[6] S. Kornblith, J. Shlens, and Q. V. Le, "Do better ImageNet models transfer better?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2661–2671.

[7] A. Dalca, M. Rakic, J. Guttag, and M. Sabuncu, "Learning conditional deformable templates with convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 804–816.

[8] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 510–519.

[9] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," 2019, *arXiv:1911.05722*. [Online]. Available: http://arxiv.org/abs/1911.05722

[10] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 4, 2020, doi: 10.1109/TPAMI.2020.2992393.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 630–645.

[13] X. Wang, M. Kan, S. Shan, and X. Chen, "Fully learnable group convolution for acceleration of deep neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9049–9058.

[14] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," 2019, *arXiv:1905.11946*. [Online]. Available: http://arxiv.org/abs/1905.11946

[15] A. Romero, N. Ballas, S. Ebrahimi Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*. [Online]. Available: http://arxiv.org/abs/1412.6550

[16] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[18] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. 21st AAAI Conf. Artif. Intell.*, 2017, pp. 1–7.

[19] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*. [Online]. Available: http://arxiv.org/abs/1605.07146

[20] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1492–1500.

[21] I. Bello, B. Zoph, Q. Le, A. Vaswani, and J. Shlens, "Attention augmented convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3286–3295.

[22] S. Xie, A. Kirillov, R. Girshick, and K. He, "Exploring randomly wired neural networks for image recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1284–1293.

[23] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9401–9411.

[24] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2820–2828.

[25] G. Huang, S. Liu, L. V. D. Maaten, and K. Q. Weinberger, "CondenseNet: An efficient DenseNet using learned group convolutions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2752–2761.

[26] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2472–2481.

[27] A. S. Winoto, M. Kristianus, and C. Premachandra, "Small and slim deep convolutional neural network for mobile device," *IEEE Access*, vol. 8, pp. 125210–125222, 2020.

[28] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," 2019, *arXiv:1911.11907*. [Online]. Available: http://arxiv.org/abs/1911.11907

[29] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*. [Online]. Available: http://arxiv.org/abs/1505.00387

[30] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 646–661.

[31] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[32] J. Frankle, G. K. Dziugaite, D. M. Roy, and M. Carbin, "Stabilizing the lottery ticket hypothesis," 2019, *arXiv:1903.01611*. [Online]. Available: http://arxiv.org/abs/1903.01611

[33] J. Jeong and J. Shin, "Training CNNs with selective allocation of channels," 2019, *arXiv:1905.04509*. [Online]. Available: http://arxiv.org/abs/1905.04509

[34] X. Ding, X. Zhou, Y. Guo, J. Han, and J. Liu, "Global sparse momentum sgd for pruning very deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 6379–6391.

[35] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng, and J. Sun, "MetaPruning: Meta learning for automatic neural network channel pruning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3296–3305.

[36] Z. Zhang, J. Li, W. Shao, Z. Peng, R. Zhang, X. Wang, and P. Luo, "Differentiable learning-to-group channels via groupable convolutional neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3542–3551.
[37] X. Gastaldi, "Shake-shake regularization," 2017, *arXiv:1705.07485*. [Online]. Available: http://arxiv.org/abs/1705.07485
[38] M. Abdi and S. Nahavandi, "Multi-residual networks: Improving the speed and accuracy of residual networks," 2016, *arXiv:1609.05672*. [Online]. Available: http://arxiv.org/abs/1609.05672
[39] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," 2016, *arXiv:1605.07648*. [Online]. Available: http://arxiv.org/abs/1605.07648
[40] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1303–1314, Jun. 2018.
[41] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.
[42] Y. Jeon and J. Kim, "Active convolution: Learning the shape of convolution for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4201–4209.
[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[44] J. Jin, A. Dundar, and E. Culurciello, "Flattened convolutional neural networks for feedforward acceleration," 2014, *arXiv:1412.5474*. [Online]. Available: http://arxiv.org/abs/1412.5474
[45] J. Ngiam, Z. Chen, D. Chia, P. W. Koh, Q. V. Le, and A. Y. Ng, "Tiled convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1279–1287.
[46] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, Y. Shuicheng, and J. Feng, "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3435–3444.
[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861
[48] K. Sun, M. Li, D. Liu, and J. Wang, "IGCV3: Interleaved low-rank group convolutions for efficient deep neural networks," 2018, *arXiv:1806.00178*. [Online]. Available: http://arxiv.org/abs/1806.00178
[49] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.
[50] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 116–131.
[51] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNetv2: A lightweight, power efficient, and general purpose convolutional neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9190–9200.
[52] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 818–833.
[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[54] A. S. Morcos, D. G. T. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization," 2018, *arXiv:1803.06959*. [Online]. Available: http://arxiv.org/abs/1803.06959

ZHU BAOZHOU was born in 1992. He received the B.S. degree in mechanical engineering and automation from the South China University of Technology (SCUT) in 2015 and the M.S. degree in microelectronics and solid-state electronics from the National University of Defense Technology (NUDT), China, in 2017. He is currently pursuing the Ph.D. degree in deep learning with the Department of Quantum and Computer Engineering, Delft University of Technology, Delft, The Netherlands. His research interest includes the efficient deployment of deep neural networks.

ZAID AL-ARS (Member, IEEE) is an Associate Professor with the Delft University of Technology (TU Delft), where he leads the Accelerated Big Data Systems Group. He is the co-founder of a couple of big data companies specialized in high-performance analytics solutions and AI. His work focuses on developing computing infrastructures for efficient processing of big data analytics applications. He serves on the advisory board of a number of high-tech start-ups. He published more than 100 peer-reviewed publications and holds two patents.

H. PETER HOFSTEE (Member, IEEE) is a Distinguished Research Staff Member of IBM and a part-time Professor with the Delft University of Technology (TU Delft), best known for his contributions to heterogeneous computing as a chief architect of the synergistic processor elements in the cell broadband engine used in PlayStation3 and the first supercomputer to reach sustained petaflop operation. He currently focuses on optimizing the system roadmap for big data, analytics, and cloud, including the use of accelerated computation. His recent contributions include coherently attached reconfigurable acceleration on POWER7, paving the way for the new coherent attach processor interface on POWER8 through POWER10. He holds more than 100 issued patents.