RefConv: Reparameterized Refocusing Convolution for Powerful ConvNets

Zhicheng Cai, Xiaohan Ding, Qiu Shen, *Member, IEEE*, and Xun Cao, *Member, IEEE*

***Abstract*—We propose reparameterized refocusing convolution (RefConv) as a replacement for regular convolutional layers, which is a plug-and-play module to improve the performance without any inference costs. Specifically, given a pretrained model, RefConv applies a trainable Refocusing Transformation to the basis kernels inherited from the pretrained model to establish connections among the parameters. For example, a depthwise RefConv can relate the parameters of a specific channel of convolution kernel to the parameters of the other kernel, i.e., make them refocus on the other parts of the model they have never attended to, rather than focus on the input features only. From another perspective, RefConv augments the priors of existing model structures by utilizing the representations encoded in the pretrained parameters as the priors and refocusing on them to learn novel representations, thus further enhancing the representational capacity of the pretrained model. The experimental results validated that RefConv can improve multiple convolutional neural network (CNN)-based models by a clear margin on image classification (up to 1.47% higher top-1 accuracy on ImageNet), object detection, semantic segmentation, and adversarial attacks without introducing any extra inference costs or altering the original model structure. Further studies demonstrated that RefConv can strengthen the spatial skeletons of kernels, reduce the redundancy of channels, and smooth the loss landscape, which explains its effectiveness.**

***Index Terms*—Computer vision, convolutional neural network (CNN), deep learning, reparameterization method.**

I. Introduction

CONVOLUTIONAL neural networks (CNNs) have indeed been the dominant tool for a wide range of computer vision tasks. One of the mainstream approaches to improving the performance of CNNs is to elaborately design the model structures, including macromodel architectures [1], [2], [3] and micro-plug-and-play components [4], [5], [6]. The success of CNN can be partly attributed to the locality of operations. For the spatial dimensions, a typical example is the sliding-window mechanism of convolution, which utilizes the local priors of images. For the channel dimension, a depthwise convolution (referred to as *DW conv* for brevity) operates on each input channel with an independent 2-D convolution kernel, significantly reducing the parameters and computations, compared to a regular *dense conv* (which means each output channel attends to every input channel, i.e., the number of groups is 1).

In this article, we propose to further improve the performance of CNNs from another perspective—augmenting the priors of existing structures. For example, a DW conv can be regarded as a concatenation of multiple mutually independent 2-D conv kernels (referred to as *kernel channels*), and the only input to a specific kernel channel is its corresponding channel of the feature map (referred to as *feature channel*), which may limit the model's representational capacity. We seek to add more priors without changing the model's definition or introducing any inference costs (e.g., letting the kernel channel operate with the other feature channels will make the operation no longer a DW conv), so we propose a reparameterization technique to *augment the priors of model structures by making their parameters attend to the parameters of other structures*.

Specifically, we propose a technique named *reparameterized refocusing*, which establishes connections among the parameters of existing structures. Given a pretrained CNN, we replace its conv layers with our proposed *reparameterized refocusing convolution* (RefConv), as shown in Fig. 1. Taking DW conv again as an example, a DW conv of a pretrained CNN will be replaced by a RefConv, which freezes its pretrained conv kernel as the *basis weights* W_b and apply a trainable operation, which is referred to as *Refocusing Transformation* $\mathrm{T}(\cdot)$, to W_b to generate a new DW conv kernel, which is referred to as *transformed weights* W_t. We use *refocusing weights* W_r to denote the trainable parameters additionally introduced by Refocusing Transformation so that $\mathrm{W}_t = \mathrm{T}(\mathrm{W}_b, \mathrm{W}_r)$. We then use W_t, instead of the original parameters, to operate on the input features. In other words, we use a different parameterization of the conv kernel. With a properly designed Refocusing Transformation, we can relate the parameters of a specific kernel channel to the parameters of the other kernel channels, i.e., make them *refocus* on the other parts of the model (rather than the input features only) to learn new representations. As the latter are trained with the other feature channels, they encode the representations condensed from the other feature channels, so that we can *indirectly* establish connections among the feature channels, which cannot be realized directly (by the definition of DW conv). After a training process (referred to as *Refocusing Learning*) of the constructed model (*RefConv model*), we use the trained refocusing weights and the frozen basis weights to generate the final transformed weights, which are saved and used for inference only. Eventually, the resultant model (*reparameterized*

Received 9 March 2024; revised 17 September 2024 and 13 December 2024; accepted 12 March 2025. Date of publication 16 April 2025; date of current version 4 June 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFF0713300 and in part by the National Natural Science Foundation of China under Grant 62231002. *(Corresponding author: Qiu Shen.)*

Zhicheng Cai, Qiu Shen, and Xun Cao are with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China, and also with the Key Laboratory of Optoelectronic Devices and Systems with Extreme Performances of MOE, Nanjing University, Nanjing 210023, China (e-mail: caizc@smail.nju.edu.cn; shenqiu@nju.edu.cn; caoxun@nju.edu.cn).

Xiaohan Ding is with the Tencent AI Laboratory, Shenzhen 518057, China (e-mail: xiaohding@gmail.com).

Digital Object Identifier 10.1109/TNNLS.2025.3552654

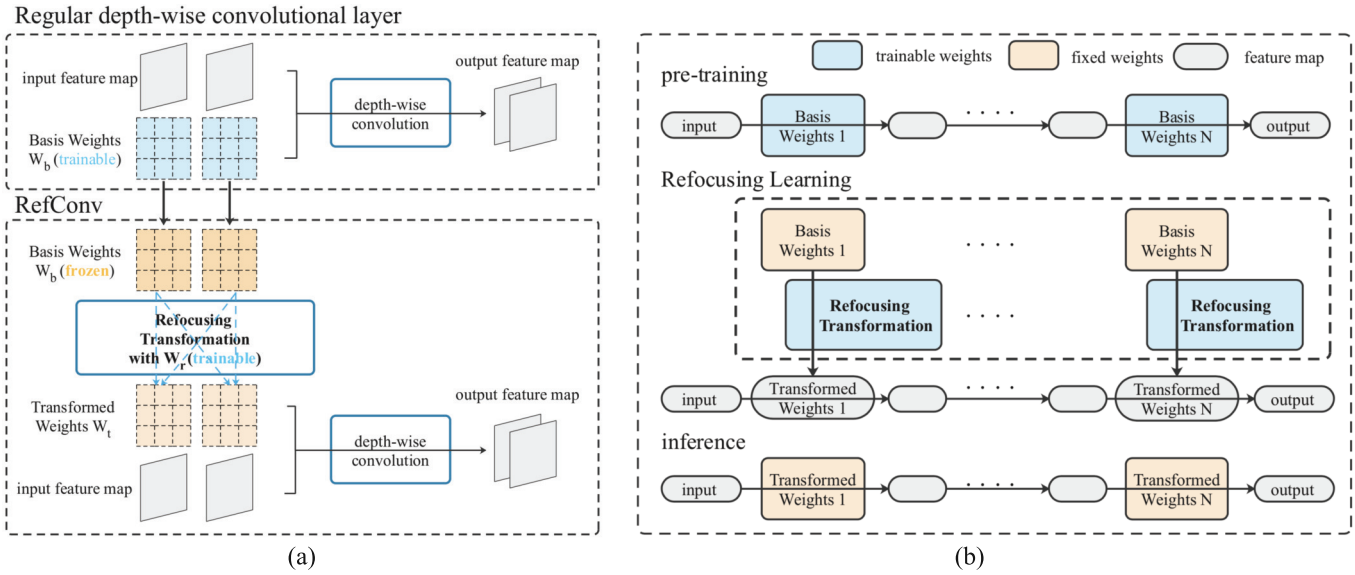


Fig. 1. (a) We showcase a depthwise RefConv with two input channels, whose kernel size is 3×3. We apply a trainable Refocusing Transformation to the basis weights W_b (which are inherited from the pretrained model and frozen) to generate W_t, which then operates on the input features. A specific channel of the original model's conv kernel (i.e., a 3×3 matrix in this case) only attends to a single channel of the input feature map, by the definition of DW conv. In contrast, RefConv can establish connections between each channel of the conv kernel and every other channel of W_b through the Refocusing Transformation. (b) We adopt a two-stage pipeline to improve CNN with RefConv. After a regular pretraining stage (which can be skipped if an off-the-shelf model is available), we construct a RefConv model by replacing the regular conv layers by the corresponding RefConv layers, which are built with the basis weights inherited from the pretrained model. During Refocusing Learning, the basis weights are frozen and the Refocusing Transformations are learnable. Finally, we save the transformed weights W_t only and use them for inference. Consequently, Refocusing Learning avoids changing the original model structure or producing extra inference costs.

RefConv model) will deliver higher performance with identical inference costs to the original model. In addition, since the Refocusing Transformation in RefConv is conducted on the basis weights instead of the batches of training examples, the Refocusing Learning process is computationally efficient and memory-saving.

Except for DW conv, RefConv can easily generalize to other forms such as groupwise and dense conv. As a generic design element, RefConv can be applied to any off-the-shelf CNN models with different structures. Our experimental results show that RefConv can improve the performance of multiple ConvNets on image classification, object detection, semantic segmentation, and adversarial attacks by a clear margin. For example, RefConv improves MobileNetv3 [7] and ShuffleNetv2 [8] by up to 1.47% and 1.26% higher top-1 accuracy on ImageNet. To be emphasized, such performance improvements are realized with no extra inference costs or alterations to the original model structure. We further seek to explain the effectiveness of RefConv and discover that RefConv can enlarge the KL divergence between the pairs of kernel channels, which validates that RefConv can reduce the channel similarity and redundancy [9], [10] through attending to other channels. This enables RefConv to learn more diversified representations and enhance the model's representational capacity. In addition, it is observed that the model with RefConv has a smoother loss landscape, suggesting a better generalization ability [11]. Both phenomena validate that the added priors of RefConv make the local structures of original convolution more informative, gaining a better holistic view of the data. Moreover, it is discovered that RefConv can automatically strengthen the central parts of convolution kernels, which also contributes to the effectiveness [12].

Our contributions are summarized as follows.

1) We propose reparameterized refocusing, which augments the priors to existing structures by establishing connections to the learned kernels. Consequently, the reparameterized kernels can learn more diverse representations, thus further improving the representational capacity of the trained CNNs.
2) We propose RefConv to replace the original conv layers and experimentally validate that RefConv can improve the performance of various backbone models on ImageNet by a clear margin without extra inference costs or altering model structure. Moreover, RefConv can also improve the ConvNets on object detection, semantic segmentation, and adversarial attacks.
3) We explore the effects of RefConv on the kernel weights and model training dynamics. We demonstrate that RefConv can strengthen the kernel spatial skeletons, reduce the channel redundancy, and smooth the loss landscape, which explains the effectiveness.

II. Related Work

A. Structure Designs for Better Performance

The designs of CNN structures for better performance include specific macroarchitecture and generic microcomponents. Representatives of classic macroarchitectures include VGGNet [13] and ResNet [1], [2], [7], [14], [15], [16]. Recently, more modern and powerful CNN models have been

raised, such as ConvNeXt [3], RepLKNet [17], and FasterNet [18], [19], [20]. Microcomponents, such as SE block [4] and CBAM block [5], [6], [21], [22], [23], [24], [25], [26], are usually architecture-agnostic [12], which can be incorporated into various models and bring generic benefits. However, all of these model designs change the predefined model structure. In contrast, RefConv focuses on the parameters of convolution kernels and intends to augment the priors of existing structures. As RefConv does not change the model structure, it is complementary to the advancements in the designs of architectures or components.

B. Structural Reparameterization

Structural reparameterization [12], [17], [27], [28], [29], [30] is a representative reparameterization methodology to parameterize a structure with the parameters transformed from another structure. Typically, it adds extra branches to the model in training to enhance the representational capacity and improve the performance, then equivalently simplifies the training structure into the same as the original model for inference. For example, ACNet [12] constructs two extra vertical and horizontal convolution branches in training and converts them into the original branch in inference. RepVGG [29] constructs identity mappings parallel to the 3×3 convolution during training and converts the shortcuts into the 3×3 branch. Due to the effectiveness of structural reparameterization, it is recently attached great importance and utilized in a wide range of reals, such as lightweight model design [31], [32], [33], object detection [34], [35], [36], and super-resolution [37], [38], [39], [40], [41], [42].

Similar to the aforementioned designs in the structure space, structural reparameterization builds extra branches to process the feature maps, which incurs considerable extra computational and memory costs during training. In contrast, the extra transformations in RefConv are applied to the basis weights only, which is computationally efficient and memory saving, compared to structural reparameterization.

C. Weight Reparameterization Methods

As a representative weight reparameterization method, DiracNet [43] encodes the convolution kernel as the linear combination of the normalized kernel and the identity tensor. ExpandNet [44] and depthwise over-Parameterized convolution (DO-Conv) [45] expand a linear layer to multiple linear layers in training and equivalently multiply these layers together according to linear algebra. Weight normalization includes standard normalization [46], centered normalization [47], and orthogonal normalization [48], which aims to normalize the weights in order to accelerate and stabilize training instead of enhancing the model representational capacity. Specifically, these weight reparameterization methods are independent of the data.

Dynamic convolution [49], [50], [51], such as CondConv [52] and DyConv [53], can be viewed as data-dependent weight reparameterization, as it uses specifically designed over-parameterized hypernetworks [54], [55], [56], [57], which take data as the input and generate specific model weights for the certain data. For further extension, dynamic filter network [58] and kernel prediction networks [59], [60] introduce two filter generation frameworks, which uses a deep neural network to generate sample-adaptive filters conditioned on the input. However, due to the dependence on the input data, such additional hypernetworks cannot be removed in inference, thus introducing significant extra parameters and computational costs in both training and inference.

Refocusing Learning derives new weights with some meta weights (instead of the data), and then utilizes the new weights for computations, so it can be categorized as a data-independent weight reparameterization method, which enhances the model representational capacity without introducing additional parameters and computational costs in inference.

Model compression, represented by pruning [61], [62], [63], [64], [65] and quantization [66], [67], [68], [69], [70], can be regarded as another approach to optimize the models through manipulating the model parameterization. DepGraph [63] explicitly models the dependence between layers and comprehensively group coupled parameters for structural pruning. SAP [65] adaptively drops redundant weights according to the measure of sparsity. QCE [67] provides a more accurate estimation of gradients by leveraging the Taylor expansion to compensate for the quantization error. SBS [69] uses a single-path bit-sharing scheme to perform pruning and quantization jointly for automatic loss-aware model compression.

While weight reparameterization like RefConv primarily enhances the model representational capacity, model compression directly targets the reduction of model size and computational load, and maintains the accuracy as much as possible. Both approaches are essential for optimizing deep learning models for deployment in resource-constrained environments.

III. Reparameterized Refocusing Convolution

In this section, we first elaborate on the design of a RefConv as a replacement for a regular depthwise conv, and then describe how to generalize it to groupwise or dense cases.

A. Depthwise RefConv

Denote the number of input channels by C_{in}, output channels by C_{out}, and groups by g. A depthwise convolution is configured by $C = C_{\text{in}} = C_{\text{out}} = g$. Assume the kernel size is K, so that the basis weights and transformed weights can be formulated as $\mathrm{W}_b, \mathrm{W}_t \in \mathbb{R}^{C\times1\times K\times K}$. Note that we desire not to change the model's inference structure, so that W_t should be of the same shape as W_b.

We desire a proper design of the Refocusing Transformation T, which transforms the frozen W_b into W_t. For a specific channel of W_t, such a function T is expected to establish connections between it and every single channel of W_b. In this article, we propose to use a dense convolution as T, so that the Refocusing Transformation is parameterized by the kernel tensor of such a dense convolution $\mathrm{W}_r \in \mathbb{R}^{C\times C\times k\times k}$. We use $k = 3$ by default so that it should have padding $= 1$ to ensure W_t has the same shape as W_b. Intuitively, such an operation can be seen as *scanning the basis weights*

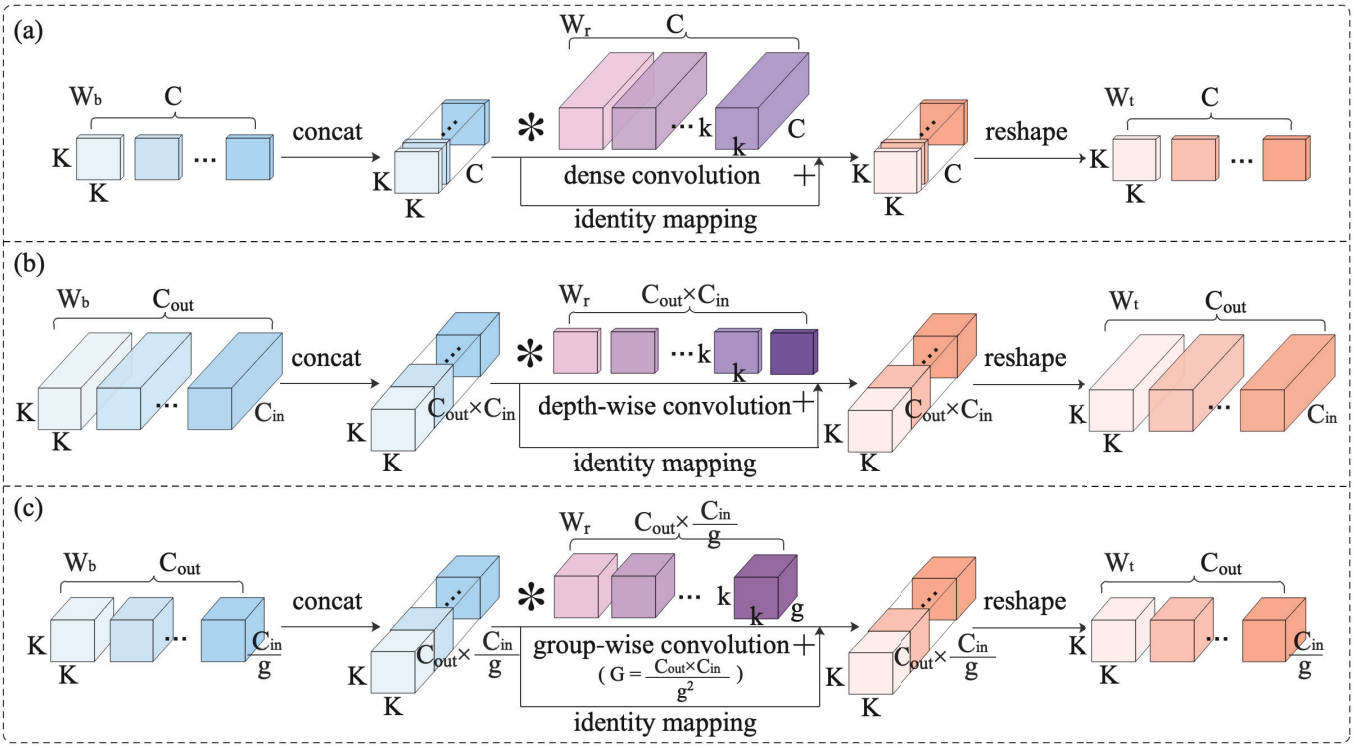


Fig. 2. Detailed forms of Refocusing Transformation. (a) Depthwise case ($h = C$). (b) Dense case ($g = 1$). (c) General groupwise case.

with a 3×3 *sliding window parameterized by* W_r *to extract representations to construct the desired kernel, just like we scan the feature maps with a regular conv kernel to extract patterns*. As such a convolution is dense, the interchannel connections are established, so that each channel of W_t relates to all the channels of W_b, as shown in Fig. 1(a). Just as the metaphor indicates, W_b can be regarded as the input "feature map" to the Refocusing Transformation, which may be designed more carefully, borrowing novel ideas from the model structure design literature. For example, we may employ nonlinearity or more advanced operations, which may perform better. In this article, we use a single convolution because it is simple, intuitive, and effective enough. Better implementations of Refocusing Transformations are scheduled as our future work.

Moreover, as inspired by residual learning [1], we desire the Refocusing Transformation to learn the increments over the basis weights, rather than the original mapping, just like we use the residual blocks to learn the increments over the base feature maps in ResNets. Therefore, we add a similar "identity mapping," so that

$$\mathrm{W}_t = \mathrm{T}(\mathrm{W}_b, \mathrm{W}_r) = \mathrm{W}_b * \mathrm{W}_r + \mathrm{W}_b \tag{1}$$

where $*$ denotes the convolution operator. Fig. 2(a) exhibits the detailed transformation of the depthwise-case RefConv.

B. General RefConv

For a general dense or groupwise RefConv, where the basis weights and transformed weights are denoted by $\mathrm{W}_b, \mathrm{W}_t \in \mathbb{R}^{C_{\text{out}} \times (C_{\text{in}}/g) \times K \times K}$, we generalize the Refocusing Transformation designed for the depthwise case. Just like the depthwise case transforms the C_{out} basis kernel channels into C_{out} transformed kernel channels, in the general case, we transform the C_{out} basis kernel slices (each with (C_{in}/g) channels) into C_{out} transformed kernel slices. We still use convolution as the Refocusing Transformation T, whose input and output feature maps have the same shape of $\mathbb{R}^{(C_{\text{out}} \times (C_{\text{in}}/g)) \times K \times K}$, so that it should be configured with output channels = input channels = $(C_{\text{out}} \times (C_{\text{in}}/g))$, namely, $\mathrm{W}_r \in \mathbb{R}^{(C_{\text{out}} \times (C_{\text{in}}/g)) \times (C_{\text{out}} \times (C_{\text{in}}/g)) \times k \times k}$. However, such W_r may be too large if W_b is dense, in which case $g = 1$ and W_r has $C_{\text{out}}^2 C_{\text{in}}^2 k^2$ parameters (which is approximately $C_{\text{in}} C_{\text{out}}$ times larger than the $C_{\text{out}} C_{\text{in}} K^2$ parameters of W_b, and C_{in} and C_{out} are very large), making it difficult for the model to train stably. To reduce the parameters of such a Refocusing Transformation T and stable the process of refocusing learning, we make it groupwise, introducing its number of groups G as a hyperparameter, so that $\mathrm{W}_r \in \mathbb{R}^{(C_{\text{out}} \times (C_{\text{in}}/g)) \times (C_{\text{out}} \times (C_{\text{in}}/gG)) \times k \times k}$ and the number of parameters becomes $(C_{\text{out}}^2 C_{\text{in}}^2 k^2 / g^2 G)$.

We propose a formula to automatically determine G according to the input basis conv as

$$G = \frac{C_{\text{out}} C_{\text{in}}}{g^2}. \tag{2}$$

Thus, the shape of W_r becomes $\mathrm{W}_r \in \mathbb{R}^{(C_{\text{out}} \times (C_{\text{in}}/g)) \times g \times k \times k}$. Such a design makes the Refocusing Transformation complementary to the original convolution: a larger g means a sparser original convolution, which needs more cross-channel connections established by Refocusing Transformation; according to 2, a larger g results in a smaller G, which exactly meets such a demand. The detailed computation of the general groupwise case of RefConv, including the identity mapping and necessary reshaping operations, is depicted in Fig. 2(c).

For example, if W_b is dense, as shown in Fig. 2(b), we have $G = C_{\text{out}}C_{\text{in}}$ and $\mathrm{W}_r \in \mathbb{R}^{(C_{\text{out}}C_{\text{in}})\times 1\times k\times k}$, which is a depthwise convolution with $C_{\text{out}}C_{\text{in}}$ kernel channels, so it only aggregates the learned representations across the spatial dimensions but performs no cross-channel recombinations (which are not desired since W_b can operate across the feature channels by itself). On the contrary, if W_b is a depthwise kernel, as shown in Fig. 2(a), we will have $G = 1$ and $\mathrm{W}_r \in \mathbb{R}^{C_{\text{out}}\times C_{\text{in}}\times K\times K}$, which is exactly the dense convolution kernel discussed in Section III-A, which fully establishes the required cross-channel connections.

1) Comparison of Parameters: To compare the amount of parameters, W_b has $C_{\text{out}}C_{\text{in}}K^2/g$ parameters, while W_r has $C_{\text{out}}C_{\text{in}}k^2$ parameters, which is approximately g times of W_b, thus strengthening the capacity of conv kernels to capture more representations.

2) Comparison of FLOPs: Nevertheless, we would like to note that RefConv adds only minor extra computations during training. Assume the feature map is of $B \times C_{\text{in}} \times H \times W$, the FLOPs of the original convolution will be $(\text{BHW}C_{\text{in}}C_{\text{out}}K^2/g)$, while the FLOPs of Refocusing Transformation is only $(K^2C_{\text{out}}^2C_{\text{in}}^2k^2/g^2G) = K^2k^2C_{\text{in}}C_{\text{out}}$, which is irrelevant to the batch size B. Consequently, the ratio of FLOPs between RefConv and the original conv layer is (k^2g/BHW). For example, assume $B = 256$, $H = W = 28$, $C_{\text{in}} = C_{\text{out}} = g = 512$, and $K = k = 3$ (common case of a DW layer in a regular CNN trained on ImageNet), the FLOPs is 925M for the DW conv while only 21M for the Refocusing Transformation (only 2.3% of the original DW conv, and the ratio will further decrease with the increase of the batch size and input size). Thus, the computational cost of applying RefConv is negligible, validating the practicality of RefConv.

C. Refocusing Learning

Refocusing Learning begins with a given pretrained CNN, which can be obtained through a regular pretraining stage if we have no available off-the-shelf model. Note that the Refocusing Learning is conducted on the same dataset as the pretrained models, intending to further improve the performance of the pretrained models. We construct a RefConv model by replacing the regular conv layers with the corresponding RefConv layers. We do not replace the 1×1 conv layers because they are dense in the channel and encode no spatial patterns; hence, there is no need to establish cross-channel connections or extract spatial representations from it. The RefConv layers are built with the W_b inherited from the pretrained model and W_r initialized with Xavier random initialization. Moreover, W_r can be initialized as zeros to make the initial model equivalent to the pretrained model (since $\mathrm{W}_t = \mathrm{W}_b$ for every RefConv), which is tested in Section IV-F.

During Refocusing Learning, a RefConv layer computes the transformed weights $\mathrm{W}_t = \mathrm{T}(\mathrm{W}_b, \mathrm{W}_r)$, where W_b is fixed and W_r is learnable, and uses W_t to operate on the input features. Therefore, the gradients will backpropagate through $\mathrm{W}_t - \mathrm{W}_r$, so that W_r will be updated by the optimizer just like the routines of training a regularly parameterized model.

After Refocusing Learning, we compute the final transformed weights with W_b and the trained W_r. We save the final transformed weights only and use them as the parameters of the original CNN for inference. In this way, the inference-time model will have exactly the same structure as the original.

IV. Experiments

A. Performance Evaluation on ImageNet

1) Dataset and Models: We first conduct abundant experiments to validate the effectiveness of RefConv in enhancing the representational capacity and improving the performance of CNNs on ImageNet [71], which is one of the most widely used but challenging real-world benchmark datasets for computer vision. ImageNet comprises 1.28M images for training and 50k images for validation from 1000 classes. We experiment with multiple representative CNN architectures, covering different types of convolution layers (namely, DW conv, groupwise conv, and dense conv). The tested CNNs include MobileNetv1,v2,v3 [7], [15], [16], MobileNeXt [72], HBONet [73], EfficientNet [74], ShuffleNetv1,v2 [8], [75], ResNet [1], DenseNet [2], FasterNet [18], and ConvNeXt [3]. In addition, since transformer-based models are widely used in computer vision realm and to fully explore the potential of RefConv, we also test RefConv on various vision transformers (ViTs) hybridized with convolutions, namely, CeiT [76], VisFormer [77], CMT [78], iFormer [79], and ConvFormer [80]; we use RefConv to replace the original convolution operations in these convolutional-hybrid ViTs.

2) Configurations: For training the baseline models, we adopt an SGD optimizer with a momentum of 0.9, a batch size of 256, and a weight decay of 4×10^{-5}, as the common practice [17]. We use a learning rate schedule with a five-epoch warmup, an initial value of 0.1, and cosine annealing for 100 epochs. The data augmentation uses random cropping and horizontal flipping. The input resolution is 224×224. For Refocusing Learning, we initialize the weights of Refocusing Transformations with Xavier random initialization [81] and freeze the basis weights inherited from the corresponding pretrained models. Refocusing Learning uses the same optimization strategy as the baselines. Both baseline training and Refocusing Learning are conducted on the ImageNet dataset. Besides, we make no difference to the final model architectures.

3) Performance Improvements: Table I shows the experimental results. It can be observed that RefConv can significantly boost the performance of various baseline models with a clear margin. For example, RefConv improves the top-1 accuracy of MobileNetv3-S (depthwise conv), ResNet-18 (dense conv), ShuffleNetv2 (groupwise conv), and FasterNet-S (depthwise and dense conv) by 1.47%, 0.94%, 1.26%, and 1.15%, respectively, indicating the comprehensive effectiveness of RefConv for different types of convolution operations and model structures. Moreover, RefConv can also significantly enhance the accuracy of various convolutional-hybrid ViTs. Specifically, RefConv improves the top-1 accuracy of CMT and ConvFormer up to 1.25% and 1.07%, respectively. To analyze the reason, ViTs tend to capture long-range dependencies and global information, which is less effective at representing local features, while RefConv can enhance the

TABLE I

RESULTS OF REFCONV MODELS AND THE NORMALLY TRAINED BASELINES ON IMAGENET. WE ALSO REPORT THE NUMBER OF TRAINING-TIME PARAMETERS (T. PARAMS), FLOPS, AND MEMORY COSTS OF BASELINES AND REFCONV MODELS. TO EMPHASIZE THAT THE INFERENCE-TIME PARAMETERS (I. PARAMS) AND FLOPS OF THE FINAL REPARAMETERIZED REFCONV MODEL ARE IDENTICAL TO THOSE OF THE CORRESPONDING BASELINE

Index	Top-1 Accuracy		Params (M)		FLOPs (G)		Memory (G)	
Model	Baseline	RefConv	Baseline	RefConv	Baseline	RefConv	Baseline	RefConv
MobileNetv1	72.18%	72.96% **(+0.82%)**	3.22	28.29	150.76	150.96	19.83	20.21
MobileNetv2	71.68%	72.35% **(+0.67%)**	3.56	44.11	90.37	90.72	24.21	24.98
MobileNetv3-S	61.95%	63.42% **(+1.47%)**	2.94	11.15	17.12	17.20	14.49	14.68
MobileNetv3-L	71.73%	72.91% **(+1.18%)**	5.48	34.06	61.15	61.41	24.33	24.85
MobileNeXt	71.57%	72.81% **(+1.24%)**	3.31	109.35	79.37	80.92	30.29	32.21
HBONet	71.61%	72.59% **(+0.98%)**	4.56	44.49	83.71	84.10	25.26	25.66
EfficientNet-B0	75.78%	76.74% **(+0.96%)**	4.98	72.67	103.53	104.20	31.02	31.78
ShuffleNetv1	63.17%	64.30% **(+1.13%)**	1.81	4.56	35.52	35.55	14.61	14.82
ShuffleNetv2	67.66%	68.92% **(+1.26%)**	2.28	5.94	39.65	39.69	13.17	13.30
ResNet-18	70.69%	71.63% **(+0.94%)**	11.72	22.74	472.08	472.20	15.52	15.76
ResNet-50	76.16%	76.96% **(+0.80%)**	25.61	36.97	1063.35	1063.55	32.14	32.54
ResNet-101	77.14%	77.72% **(+0.68%)**	44.63	66.01	2018.72	2018.96	42.97	43.92
DenseNet-169	76.17%	76.90% **(+0.73%)**	14.18	17.24	884.81	884.95	49.10	49.96
FasterNet-S	78.76%	79.91% **(+1.15%)**	28.44	34.65	1091.28	1091.85	24.24	24.62
ConvNeXt-T	80.82%	81.68% **(+0.96%)**	28.59	57.37	1139.10	1139.97	38.53	38.62
CeiT	75.18%	76.14% **(+0.96%)**	6.05	69.76	297.23	297.80	21.19	22.66
VisFormer	76.02%	76.55% **(+0.53%)**	10.19	14.27	308.98	309.00	27.31	27.53
CMT	76.37%	77.62% **(+1.25%)**	11.29	95.29	344.48	345.21	30.45	30.97
iFormer	79.40%	80.11% **(+0.71%)**	19.33	42.18	1157.69	1157.90	51.26	51.66
ConvFormer	81.92%	82.99% **(+1.07%)**	26.75	93.92	1008.89	1012.46	47.50	47.92

local structures of the original convolutions in these ViTs, thus making the ViTs more locally representational and providing a better holistic view of the data. These experimental results further verify the generic effectiveness of RefConv.

4) Number of Parameters: Table I also exhibits the total number of parameters in training. The baseline models have the same training parameters as the inference stage, while the RefConv models have an extra amount of parameters *during training only*. However, as we only utilize the transformed weights for inference, *the inference parameter number of reparameterized RefConv model is identical to the baseline*, introducing completely no extra inference costs. Moreover, these extra parameters are conducted on the basis weights instead of the feature maps; thus, the extra FLOPs and memory cost introduced in training are negligible as stated in Section III-B.

5) Training-Time FLOPs and Memory Costs: To measure the extra training cost brought by the extra computations of RefConv during training, we present in Table I the total *training-time* FLOPs and memory costs of baselines and RefConv models, which are tested on four RTX 3090 GPUs with a total batch size of 256 and full precision (fp32). As exhibited, the additional FLOPs and memory that RefConv introduces is *negligible* compared to the baseline, complying with the discussion in Section III-B that the computational cost of Refocusing Transformation is minor since it is conducted on the kernels instead of the feature maps. It is worth noting that only the training-time RefConv needs minor extra computations to generate W_t, and the reparameterized RefConv model will be structurally identical to the baseline after converting the weights (as there will be no Refocusing Transformation during inference at all), introducing completely no additional memory or computational cost in inference.

B. Performance Evaluation on Other Datasets

We also test the effectiveness of RefConv on CIFAR-10, CIFAR-100, and Tiny-ImageNet-200. We resize the images to 224×224, and the training strategy is in accordance with the experiments on ImageNet. It is emphasized that both baseline training and Refocusing Learning are conducted on the same dataset. A set of representative CNN architectures are tested, including Cifar-quick [82], SqueezeNet [83], VGGNet [13], ResNet [1], ShuffleNetv1,v2 [8], [75], ResNeXt [84], RegNet [85], ResNeSt [14], FasterNet [18], MobileNetv1,v2,v3 [7], [15], [16], MobileNeXt [72], HBONet [73], and EfficientNetv1,v2 [74], [86]. The results are shown in Table II. As can be observed, the performance of all models is consistently improved by a clear margin. For example, RefConv significantly enhances the performance of Cifar-quick (with dense conv) by 6.11%, 5.85%, and 3.34% on CIFAR-10, CIFAR-100, and Tiny-ImageNet-200, respectively. Besides, RefConv improves the performance of ShuffleNetv1 (with groupwise conv) by 1.34%, 1.27%, and 1.76% on CIFAR-10, CIFAR-100, and Tiny-ImageNet-200, respectively. As for the DW Conv, RefConv improves the top-1 accuracy of MobileNetv3-S by 1.32% 2.56%, and 2.05% on CIFAR-10, CIFAR-100, and

TABLE II
RESULTS OF REFCONV AND THE NORMALLY TRAINED BASELINES ON CIFAR-10, CIFAR-100, AND TINY-IMAGENET-200

Dataset	CIFAR-10		CIFAR-100		Tiny-ImageNet-200	
Model	Baseline	RefConv	Baseline	RefConv	Baseline	RefConv
Cifar-quick	83.02%	89.13% **(+6.11%)**	55.21%	61.06% **(+5.85%)**	56.90%	60.24% **(+3.34%)**
SqueezeNet	85.83%	86.64% **(+0.81%)**	61.14%	61.94% **(+0.80%)**	51.60%	53.04% **(+1.44%)**
VGGNet-16	92.50%	93.13% **(+0.63%)**	73.75%	74.88% **(+1.13%)**	61.88%	63.52% **(+1.64%)**
ResNet-18	93.10%	94.15% **(+1.05%)**	73.65%	74.51% **(+0.86%)**	61.12%	62.44% **(+1.32%)**
ResNet-34	93.95%	94.97% **(+1.02%)**	74.82%	95.74% **(+0.92%)**	65.21%	66.45% **(+1.24%)**
ShuffleNetv1	91.35%	92.69% **(+1.34%)**	68.51%	69.78% **(+1.27%)**	56.86%	58.62% **(+1.76%)**
ShuffleNetv2	92.31%	93.56% **(+1.25%)**	70.08%	71.52% **(+1.44%)**	60.38%	61.90% **(+1.52%)**
ResNeXt-18	93.00%	93.68% **(+0.68%)**	72.02%	72.84% **(+0.82%)**	62.02%	62.84% **(+0.82%)**
RegNetX_200MF	91.11%	91.93% **(+0.82%)**	68.08%	69.73% **(+1.65%)**	59.84%	61.23% **(+1.39%)**
ResNeSt-50	93.52%	95.22% **(+1.70%)**	69.18%	70.82% **(+1.64%)**	65.18%	65.82% **(+0.64%)**
FasterNet-T0	92.94%	94.08% **(+1.14%)**	68.02%	69.24% **(+1.22%)**	58.32%	60.14% **(+1.82%)**
MobileNetv1	92.45%	93.01% **(+0.56%)**	72.43%	73.41% **(+0.98%)**	62.52%	63.89% **(+1.37%)**
MobileNetv2	92.58%	93.71% **(+1.13%)**	73.04%	74.35% **(+1.31%)**	61.94%	63.96% **(+2.02%)**
MobileNetv3-S	90.91%	92.23% **(+1.32%)**	68.20%	70.76% **(+2.56%)**	60.66%	62.71% **(+2.05%)**
MobileNetv3-L	92.95%	93.89% **(+0.94%)**	73.89%	75.27% **(+1.38%)**	62.21%	64.28% **(+2.07%)**
MobileNeXt	93.01%	94.45%**(+1.44%)**	67.42%	69.07%**(+1.65%)**	58.12%	60.44%**(+2.32%)**
HBONet	92.32%	93.59%**(+1.27%)**	72.39%	73.91%**(+1.52%)**	64.20%	66.44%**(+2.24%)**
EfficientNetv1-B0	94.21%	95.38%**(+1.17%)**	75.68%	76.90%**(+1.22%)**	66.62%	68.57%**(+1.95%)**
EfficientNetv2-S	93.85%	95.13%**(+1.28%)**	75.42%	76.76%**(+1.34%)**	67.18%	69.32%**(+2.15%)**

TABLE III
COMPARISON WITH OTHER REPARAMETERIZATION METHODS ON IMAGENET

Model	ResNet-18			MobileNetv2		
Method	Top1-Accuracy	FLOPs (G)	Memory (G)	Top1-Accuracy	FLOPs (G)	Memory (G)
Baseline	70.69% (+0.00%)	472.08	15.52	71.68% (+0.00%)	90.37	24.21
ACB	71.47% (+0.78%)	761.25	18.76	71.99% (+0.31%)	98.62	32.02
RepVGGB	71.21% (+0.52%)	522.85	17.02	72.11% (+0.43%)	94.39	28.38
DBB	71.25% (+0.56%)	1097.40	26.18	72.25% (+0.57%)	125.61	43.52
RepLKNetB	71.15% (+0.46%)	905.84	20.38	71.95% (+0.27%)	102.75	35.93
RepViTB	71.01% (+0.32%)	522.67	19.88	72.11% (+0.43%)	95.18	27.77
RepMixer	70.93% (+0.24%)	473.45	17.34	72.05% (+0.37%)	92.24	25.20
WN	70.81% (+0.12%)	472.10	15.53	71.76% (+0.08%)	90.37	24.21
CWN	70.85% (+0.16%)	472.10	15.53	71.78% (+0.10%)	90.37	24.21
OWN	70.83% (+0.14%)	472.10	15.53	71.75% (+0.07%)	90.37	24.21
DO-Conv	71.13% (+0.44%)	472.15	15.64	71.84% (+0.16%)	90.55	24.59
RefConv	**71.63% (+0.94%)**	472.20	15.76	**72.35% (+0.67%)**	90.72	24.98

Tiny-ImageNet-200, respectively. In summary, the results validate that RefConv can enhance the representational capacity of various models with different types of convolution layers, including dense conv, groupwise conv, and DW conv.

C. Comparison With Other Reparameterizations

We compare RefConv with other data-independent reparameterization methods on ImageNet, i.e., structural reparameterization (SR, including asymmetric convolution blocks (ACBs) [12], RepVGGB [29], DBB [28], RepLKNetB [17], RepViT Block [87], and RepMixer [88]) and weight reparameterization (WR, including WN [46], CWN [47], OWN [48], and DO-Conv [45]). For RepVGGB, we parallelly construct extra 1×1 convolution and identity mapping branches. For RepLKNetB, we parallelly construct an extra 3×3 convolution branch. The baseline models are ResNet-18 (with dense conv) and MobileNetv2 (with DW conv). Note that all the inference models of these methods are the same as the baselines. As shown in Table III, WR brings negligible improvement for that WR is intended to accelerate and stabilize the train-

TABLE IV
PASCAL VOC DETECTION AND CITYSCAPES SEGMENTATION

Pascal VOC (mAP)					Cityscapes (mIOU)				
Method	Baseline	ACB	DO-Conv	**RefConv**	Method	Baseline	ACB	DO-Conv	**RefConv**
YOLOv7	73.25%	73.60%	73.35%	**74.02%**	PSPNet	70.78%	71.02%	70.86%	**71.45%**
↑	+0.00%	+0.35%	+0.10%	**+0.77%**	↑	+0.00%	+0.24%	+0.08%	**+0.67%**
YOLOv8	73.71%	74.14%	73.84%	**74.46%**	DeepLabv3	72.31%	72.60%	72.44%	**72.93%**
↑	+0.00%	+0.43%	+0.13%	**+0.75%**	↑	+0.00%	+0.29%	+0.13%	**+0.62%**

ing. While SR improves the performance more significantly than WR, it introduces tremendous extra training costs since the extra branches are conducted on the feature maps. Furthermore, RefConv brings the highest improvements with little extra computational and memory costs during training, demonstrating the superiority over other data-independent reparameterization methods.

D. Object Detection and Semantic Segmentation

We further transfer the ImageNet-trained MobileNetv2 to Pascal VOC detection task with YOLOv7 [35] and YOLOv8 [89], following the configuration in [35], and Cityscapes segmentation with PSPNet [90] and DeepLabv3+ [91], following the configuration in [17]. We also compare RefConv with structural reparameterization (ACB) and weight reparameterization (DO-Conv). Table IV shows that RefConv can enhance the performance of the baselines by a clear margin on object detection and semantic segmentation tasks, validating the transfer capability of RefConv. Moreover, RefConv brings larger improvements than ACB and DO-Conv do, further substantiating the superiority of RefConv.

E. Adversarial Attacks and Noisy Data

We evaluate the robustness of our models against four distinct transparent attacks. We begin with the fast gradient sign method (FGSM) [92], a widely used attack that provides a baseline evaluation of model vulnerability. We then move to more powerful iterative methods, including the projected gradient descent (PGD) [93], a stronger iterative version of FGSM, the Carlini and Wagner (C&W) [94] attack, which optimizes perturbations to minimize the adversarial impact while remaining undetectable, and the momentum iterative method (MIM) [95], which enhances gradient-based attacks by introducing momentum to accelerate convergence and improve attack success. This progressive evaluation allows us to thoroughly examine the models' resilience against increasingly sophisticated adversarial threats. Moreover, we test a range of perturbation strengths ϵ for each method. For the latter three iterative methods, we set the iterations as 30 epochs and the step size as 0.01 according to the common practice. In addition, we also test out models against two types of noisy data as suggested, namely, data with Gaussian noise and Pepper-Salt noise. Specifically, we test Gaussian noise $\sim \mathcal{N}(0, \sigma^2)$ with a range of σ for different perturbation strengths, and we test Pepper-Salt noise with different probabilities p. All the experiments are conducted on the CIFAR-10 dataset, and MobileNetv2 is adopted as the baseline model. Table V shows the experimental results. As can be observed, RefConv can significantly improve the accuracy of the baseline against various adversarial attacks and noisy data. To be specific, RefConv enhances the accuracy of the baseline up to 4.18%, 4.48%, 5.02%, and 4.44% under the attack of FGSM, PGD, C&W, and MIM, respectively. The experimental results validate that RefConv can also significantly enhance the model's robustness against various adversarial attacks and noisy data.

F. Ablation Study

1) Refocusing Learning Outperforms Simply Retraining the Baseline Model: To show the superiority of Refocusing Learning over the most naive approach, which is simply training the model for more epochs, we train the already pretrained baseline models for the second time on ImageNet with the same training configurations (namely, the total training epochs is doubled). Table VI shows that retraining the models another time can barely improve the performance, which is expected as a specific kernel parameter during retraining still cannot attend to the other parameters at the other channels (for the case of DW conv in MobileNets) or spatial locations (for the case of regular conv in ResNet-18), thus failing to learn new representations.

2) Refocusing Learning Outperforms Simply Fine-Tuning the Baselines: We also fine-tune the already pretrained baseline models with a smaller learning rate of 10^{-4} for another 100 epochs on ImageNet as a common practice (the total training epochs are also doubled). Table VI shows fine-tuning that still brings negligible benefits, compared to RefConv. Once again, simply fine-tuning the converged models fails to learn any new representations. Thus, we reckon that RefConv is not simply fine-tuning or continue training the original convolution layer for another epoch. It uses learnable transformations to establish the connections between convolution kernels, learning new representations through combining the learned representations by the basis kernels, thus strengthening the model representational capacity. *Given a trained model, RefConv is the only usable and effective reparameterization method for further performance improvement.*

3) Pretrained Basis Weights Are Important Prior Knowledge: W_b is the learned weights of the baseline models and fixed during Refocusing Learning. For validation, we randomly initialize the W_b and freeze it in Refocusing Learning (column R W_b in Table VI). Doing so brings only minor

TABLE V

EXPERIMENTS OF THE MODELS' ROBUSTNESS AGAINST FOUR ADVERSARIAL ATTACKS AND TWO TYPES OF NOISY DATA. THE EXPERIMENTS ARE CONDUCTED ON THE CIFAR-10 DATASET, AND MOBILENETV2 IS ADOPTED AS THE BASELINE MODEL

FGSM [92]	clean	$\epsilon = 0.01$	$\epsilon = 0.02$	$\epsilon = 0.05$	$\epsilon = 0.10$	$\epsilon = 0.15$	$\epsilon = 0.20$	$\epsilon = 0.50$
Baseline	92.58%	45.98%	44.00%	40.64%	36.11%	30.81%	25.77%	19.61%
RefConv	93.71%	50.16%	47.91%	44.24%	39.23%	33.01%	27.29%	20.64%
↑	+1.13%	+4.18%	+3.91%	+3.60%	+3.12%	+2.20%	+1.52%	+1.03%
PGD [93]	clean	$\epsilon = 0.001$	$\epsilon = 0.002$	$\epsilon = 0.005$	$\epsilon = 0.010$	$\epsilon = 0.015$	$\epsilon = 0.020$	$\epsilon = 0.050$
Baseline	92.58%	46.74%	45.18%	42.44%	36.70%	26.46%	12.95%	4.18%
RefConv	93.71%	51.22%	49.73%	46.92%	41.52%	30.75%	15.94%	4.93%
↑	+1.13%	+4.48%	+4.55%	+4.48%	+4.82%	+4.29%	+2.99%	+0.75%
C&W [94]	clean	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 4$	$\epsilon = 5$	$\epsilon = 6$	$\epsilon = 7$
Baseline	92.58%	55.90%	53.22%	51.33%	49.95%	49.15%	48.64%	48.37%
RefConv	93.71%	60.92%	57.99%	55.82%	54.28%	53.47%	52.97%	52.72%
↑	+1.13%	+5.02%	+4.77%	+4.49%	+4.33%	+4.32%	+4.33%	+4.35%
MIM [95]	clean	$\epsilon = 0.001$	$\epsilon = 0.003$	$\epsilon = 0.005$	$\epsilon = 0.007$	$\epsilon = 0.010$	$\epsilon = 0.030$	$\epsilon = 0.050$
Baseline	92.58%	46.39%	43.53%	38.83%	35.05%	30.15%	10.02%	4.38%
RefConv	93.71%	50.83%	47.09%	43.58%	39.93%	34.43%	12.57%	5.71%
↑	+1.13%	+4.44%	+3.56%	+4.75%	+4.88%	+4.28%	+2.55%	+1.33%
Gaussian	clean	$\sigma = 0.01$	$\sigma = 0.05$	$\sigma = 0.10$	$\sigma = 0.15$	$\sigma = 0.20$	$\sigma = 0.30$	$\sigma = 0.50$
Baseline	92.58%	48.09%	47.51%	46.08%	43.57%	41.41%	36.35%	26.73%
RefConv	93.71%	52.66%	51.59%	49.77%	47.02%	43.61%	38.10%	27.74%
↑	+1.13%	+4.57%	+4.08%	+3.69%	+3.45%	+2.20%	+1.75%	+1.01%
Pepper-Salt	clean	$p = 0.01$	$p = 0.02$	$p = 0.05$	$p = 0.07$	$p = 0.10$	$p = 0.12$	$p = 0.15$
Baseline	92.58%	85.28%	76.35%	42.12%	27.82%	17.90%	15.05%	13.32%
RefConv	93.71%	87.68%	79.92%	45.48%	30.12%	19.82%	16.55%	14.32%
↑	+1.13%	+2.40%	+3.57%	+3.36%	+2.30%	+1.92%	+1.50%	+1.00%

improvements to MobileNets and even results in significant degradation of ResNet-18, which is expected as the pretrained basis weights can be regarded as prior knowledge brought into the RefConv models, which provides a good basis for learning new representations. The phenomenon that ResNet-18 is degraded much worse can be explained that its Refocusing Transformation is a DW conv (as discussed in Section III-B) that operates on the randomly initialized basis weights. Since the basis weights contain no priors at all, it is expected that the DW conv extracts no useful representations with only spatial aggregations. In contrast, the Refocusing Transformation of a DW RefConv in MobileNets is a dense conv, which has a large parameter space to learn the representations all by itself, even if it begins with no priors.

Then, we attempt to make the pretrained W_b trainable so that both W_b and W_r are updated in Refocusing Learning. Column T W_b in Table VI shows no improvements over the standard RefConv, suggesting that it is favorable to maintain the prior knowledge of the Refocusing Transformation. Last, we make W_b both randomly initialized and trainable. Column R & T W_b in Table VI shows performance better than the baseline while lower than the standard RefConv. In summary, we conclude that the pretrained basis weights W_b are prior knowledge important to the learning process. In addition, it is noteworthy that through jointly updating W_b and W_r, *RefConv can still significantly enhance the model performance without pretrained weights*.

4) Different Initializations for Refocusing Weights: The weights of CNNs are usually randomly initialized when training the models from scratch. However, for RefConv, W_r can be initialized as zeros so that the initial value of W_t will be identical to W_b [by (1)], making the initial RefConv model equivalent to the pretrained model. Column *ZI* W_r in Table VI shows the results of such zero initialization, demonstrating improvements over the baselines, which are slightly worse than the randomly initialized RefConv.

5) Validation of the Identity Mapping in RefConv: We also discover that the identity mapping in RefConv is critical. Column *w/o shortcut* in Table VI shows the results of RefConv without the shortcut, which are observably better than the baselines but worse than the standard RefConv.

6) RefConv Connects the Independent Kernels: To validate that a DW RefConv makes each independent kernel channel of W_t attend to the other channels of W_b, we calculate the degree of connection between the ith channel W_t and the jth channel in W_b for all the (i, j) pairs, which forms a correlation matrix. Naturally, as such interchannel connections are established through the filter $W_r^{(i,j)}$, which is a $k \times k$ matrix corresponding to the jth input channel and ith output channel of the W_r, we use the magnitude (i.e., the sum of the absolute values) of

TABLE VI
RESULTS OF RETRAINING AND FINE-TUNING THE BASELINES, AND DIFFERENT CONFIGURATIONS OF THE BASIS WEIGHTS

Model	Baseline	RefConv	Retrain	Finetune	$R\ W_b$	$T\ W_b$	$R\&T\ W_b$	$ZI\ W_r$	w/o shortcut
ResNet-18	70.69%	**71.63%**	70.85%	70.75%	53.04%	71.54%	71.24%	71.52%	71.01%
↑	+0.00%	**+0.94%**	+0.16%	+0.06%	-17.65%	+0.85%	+0.55%	+0.83%	+0.32%
MobileNetv1	72.18%	**72.96%**	72.29%	72.23%	72.43%	72.80%	71.73%	72.89%	72.39%
↑	+0.00%	**+0.78%**	+0.11%	+0.05%	+0.25%	+0.62%	+0.55%	+0.71%	+0.21%
MobileNetv2	71.68%	**72.35%**	71.83%	71.79%	71.90%	72.11%	72.15%	72.25%	71.89%
↑	+0.00%	**+0.67%**	+0.15%	+0.11%	+0.22%	+0.43%	+0.47%	+0.57%	+0.21%
MobileNetv3-S	61.95%	**63.42%**	62.16%	62.11%	62.42%	63.11%	63.04%	63.39%	62.95%
↑	+0.00%	**+1.47%**	+0.21%	+0.16%	+0.47%	+1.16%	+1.09%	+1.44%	+1.00%
MobileNetv3-L	71.73%	**72.91%**	71.95%	71.83%	72.07%	72.67%	72.60%	72.72%	72.27%
↑	+0.00%	**+1.18%**	+0.12%	+0.10%	+0.34%	+0.94%	+0.87%	+0.99%	+0.54%

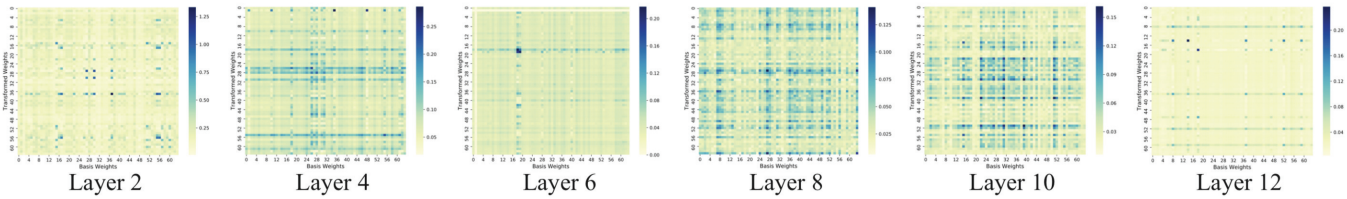


Fig. 3. Connection degree matrix of the first 64 channels of W_t and W_b in different layers. The backbone model is MobileNetv1 trained on ImageNet. Darker colors represent larger values and closer connections. As observed, RefConv establishes connections across different kernel channels, which are previously unlinked.

$W_r^{(i,j)}$ as the numerical metric for the degree of the connection, as a common practice [12], [96], [97], [98], [99]. Briefly, a larger magnitude value indicates a stronger connection. We use the first 64 channels of W_t and W_b and calculate the connection degree between each pair of channels to obtain the 64×64 connection degree matrix. As visualized in Fig. 3, the ith channel in W_t attends to not only the corresponding ith channel in W_b but also multiple other channels in W_b with different magnitude, suggesting that a DW RefConv can attend to all the channels to learn diverse combinations of existing representations.

7) Explore Other Complex Forms of the Refocusing Transformation: As stated above, the Refocusing Transformation can be formed with more complex network structures. This section conducts experiments to explore how different complex forms impact the performance of RefConv. To be specific, we test six more complex Refocusing Transformation as follows.

1) *RefConv-2B1L:* RefConv has one convolution layer with two parallel convolution branches, which can be expressed as

$$W_t = T\left(W_b, W_r^1, W_r^2\right) = W_b * W_r^1 + W_b * W_r^2 + W_b. \quad (3)$$

2) *RefConv-3B1L:* RefConv has one convolution layer with three parallel convolution branches, which can be expressed as

$$\begin{aligned} W_t &= T\left(W_b, W_r^1, W_r^2, W_r^3\right) \\ &= W_b * W_r^1 + W_b * W_r^2 + W_b * W_r^3 + W_b. \end{aligned} \quad (4)$$

3) *RefConv-1B2L:* RefConv has two convolution layers with a single convolution branch, which can be expressed as

$$W_t = T\left(W_b, W_r^1, W_r^2\right) = W_b * W_r^1 * W_r^2 + W_b. \quad (5)$$

4) *RefConv-1B2L+Act:* RefConv has two convolution layers with single convolution branch, and there exits a nonlinear activation function GELU ϕ after the first convolution layer, which can be expressed as

$$W_t = T\left(W_b, W_r^1, W_r^2\right) = \phi\left(W_b * W_r^1\right) * W_r^2 + W_b. \quad (6)$$

5) *RefConv-1B1L+SE:* RefConv has one convolution layer with single convolution branch, and we add one squeeze-and-excitation attention module [4] ψ after the convolution layer, which can be expressed as

$$W_t = T\left(W_b, W_r\right) = \psi\left(W_b * W_r\right) + W_b. \quad (7)$$

6) *RefConv-1B2L+SE:* RefConv has two convolution layers with single convolution branch, and we add one SE attention module ψ after the first convolution layer, which can be expressed as

$$W_t = T\left(W_b, W_r^1, W_r^2\right) = \psi\left(W_b * W_r^1\right) * W_r^2 + W_b. \quad (8)$$

Denote that the original Refocusing Transformation can be regarded as one convolution layer with a single branch (denoted as *RefConv-1B1L* in Table VII). The experiments are conducted on the ImageNet dataset, and the baseline model is MobileNetv2. Table VII exhibits the experimental results. As can be observed, *RefConv-3B1L* with three convolution

TABLE VII

RESULTS OF REFCONV WITH DIFFERENT COMPLEX STRUCTURES ON IMAGENET. THE BASELINE MODEL IS MOBILENETV2. WE ALSO REPORT THE NUMBER OF T. PARAMS, I. PARAMS, FLOPS, AND MEMORY COSTS OF THESE REFCONV MODELS

Model	Top-1 Accuracy	T. Params (M)	I. Params (M)	FLOPs (G)	Memory (G)
Baseline	71.68% (+0.00%)	3.56	3.56	90.37	24.21
RefConv-1B1L	72.35% (+0.67%)	44.11	3.56	90.72	24.98
RefConv-2B1L	72.29% (+0.61%)	84.66	3.56	91.07	25.75
RefConv-3B1L	72.50% (+0.82%)	125.21	3.56	91.14	26.52
RefConv-1B2L	72.31% (+0.63%)	84.66	3.56	91.07	25.65
RefConv-1B2L+Act	72.27% (+0.59%)	84.66	3.56	91.07	25.65
RefConv-1B1L+SE	72.46% (+0.78%)	46.36	3.56	90.79	25.10
RefConv-1B2L+SE	**72.68% (+1.00%)**	86.91	3.56	91.16	25.77

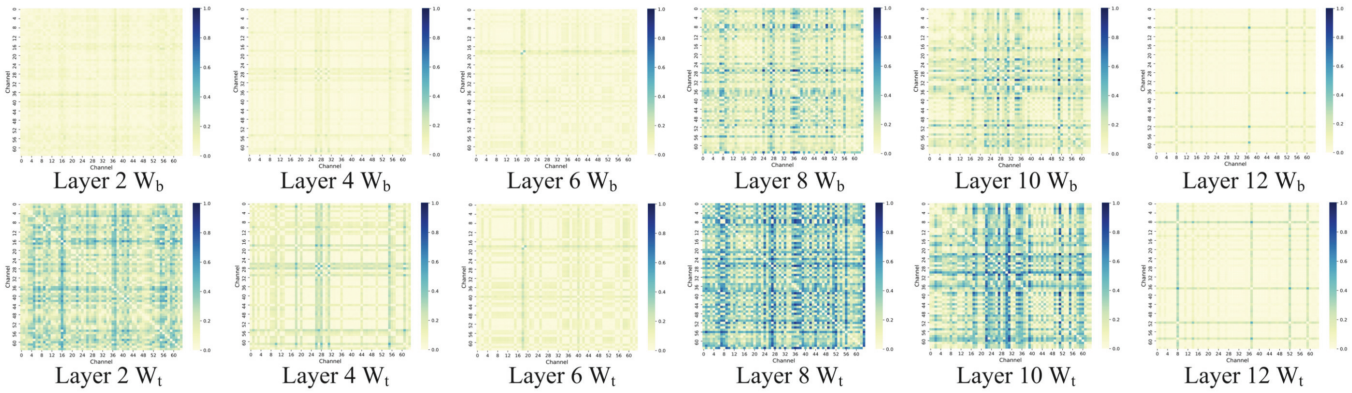


Fig. 4. Similarity matrix of the first 64 channels of W_b and W_t in different layers. The backbone model is MobileNetv1 trained on ImageNet. To improve the readability, the original value of KL divergence is added with 1 and then taken 10-base logarithm. A point with a darker color represents a larger value, and thus a lower similarity.

branches slightly improve the accuracy by 0.15% compared to the original *RefConv-1B1L*, while having almost two times more training parameters. Introducing the nonlinearity of *RefConv-1B2L+Act* slightly depresses the accuracy by 0.08%. Besides, adding one more branch by *RefConv-2B1L* and one more layer by *RefConv-1B2L*, both slightly depress the accuracy by 0.06% and 0.04%, respectively. The experiments validate that the simple convolution operation of the original RefConv has been capable of sufficiently re-establishing the representations of different channels. In addition, we find that the SE attention module can effectively enhance the accuracy of the original RefConv, *RefConv-1B1L+SE* surpasses *RefConv-1B1L* by 0.11%, and *RefConv-1B2L+SE* surpasses *RefConv-1B1L* by 0.33%. This can be attributed to the fact that SE allows the network to automatically learn the dependencies between channels and apply weighting to the features of each channel, which significantly improves the ability of RefConv to establish connections across different channels.

G. RefConv Reduces the Channel Redundancy

To explore the difference between the basis weights W_b and the transformed weights W_t, we compare the channel redundancy of W_b and W_t. As a common practice, we utilize the Kullback–Leibler (KL) divergence to measure the similarity between different pairs of channels [9], [10], so that a larger KL divergence indicates lower similarity and thus a lower degree of channel redundancy. Specifically, we sample a DW RefConv layer from the trained MobileNetv1 and apply softmax to every $K \times K$ kernel channel, and then sample the first 64 channels to calculate the KL divergence between every pair of channels. In this way, we obtain a 64×64 similarity matrix for the sampled layer. Fig. 4 shows the similarity matrices of multiple layers. As can be observed, there exists high redundancy among channels in W_b as the KL divergence is low between most of the channels. In contrast, the KL divergence between channels of W_t is significantly higher, which means the kernel channels become significantly different from the others. Based on such observations, we conclude that RefConv can reduce redundancy consistently and effectively. We explain such phenomena that RefConv can explicitly make every channel able to attend to the other channels of the pretrained kernel, which refocus on the learned representations encoded in the pretrained kernel channels to learn diverse novel representations. Consequently, the channel redundancy is reduced and the representation diversity is enhanced, which results in a higher representational capacity.

H. RefConv Smooths the Loss Landscape

To explore how Refocusing Learning influences the training dynamics, we visualize the loss landscapes of the baseline and the RefConv counterpart with the filterwise normalization visualization [11]. We use MobileNetv1 and MobileNetv2

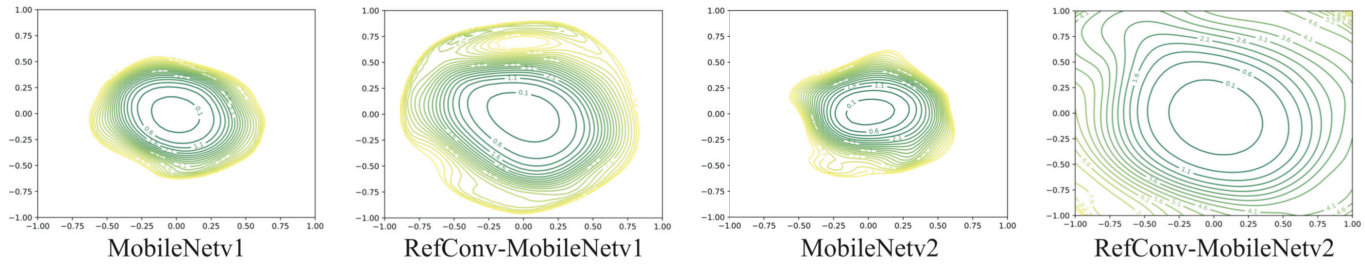


Fig. 5. Loss landscape contour maps of MobileNetv1 and MobileNetv2 with/without RefConv trained on CIFAR-10. As can be observed, the models with RefConv have wider and sparser contours compared to the original models, indicating a flatter loss landscape resulted by RefConv. Thus, RefConv possesses better training properties and generalization ability.

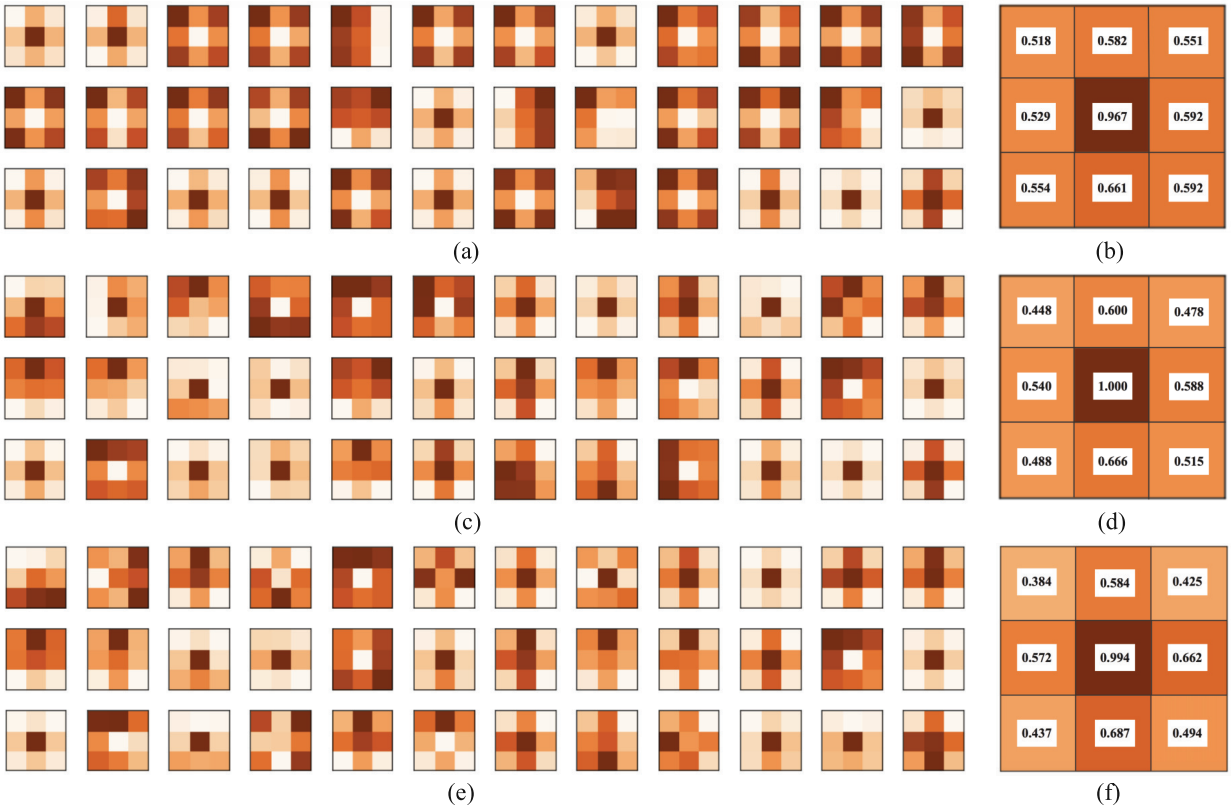


Fig. 6. Visualizations of the magnitude (i.e., the absolute value of weights) of individual kernel channels and the average magnitude matrices (which are averaged across channels). The weights are sampled from the last RefConv layer of the RefConv-MobileNetv2 trained on ImageNet. For the visualization, we normalize each matrix by the maximal value of its entries to facilitate the comparison and improve the readability. A darker color indicates a larger magnitude. As can be observed, the magnitude of W_t exhibits enhanced skeleton patterns compared to W_b, which contributes to the performance improvement. (a) W_b. (b) Avg Mag. of W_b. (c) W_t. (d) Avg Mag. of W_t. (e) $\Delta W = Wt - Wb$. (f) Avg Mag. of ΔW.

trained on CIFAR-10 as the backbone models. Fig. 5 shows that the loss landscapes of RefConv have wider and sparser contours compared to that of the baselines, which indicates that the loss curvature of RefConv is much flatter [11], suggesting a better generalization ability. This phenomenon demonstrates that Refocusing Learning possesses better training properties, which partly explains the performance improvements.

I. RefConv Strengthens the Kernel Skeletons

To explore the spatial differences between the basis kernels and transformed kernels, we visualize the W_b, W_t, and the increments over the basis weights $\Delta W = W_b - W_t$ of the last convolution layer of RefConv-MobileNetv2 trained on ImageNet, as exhibited in the left column of Fig. 6. We find that most of the ΔW exhibits stronger skeleton patterns [12], indicating that the major difference lies in the center rows and columns of the kernels. Consequently, it can be obviously observed that W_t exhibits stronger skeleton patterns than W_b, especially in the central point. Furthermore, we calculate and visualize the average kernel magnitude matrices [12], [96], [97], [99], [100] of these three weights, as exhibits in the right column of Fig. 6. Once again, the magnitude of ΔW shows strong skeleton patterns and small impact factors in corners, suggesting that the skeleton patterns of W_t are strengthened and the corners are weakened, compared to W_b. Moreover, it is

noteworthy that for W_t, the central point has a value of 1.000, which means that location has a dominant importance consistently in every 3×3 layers. To conclude, RefConv can establish spatial connections through establishing connections among the channels in different spatial positions, by which means RefConv can automatically learn the increments complement to the original kernels and strengthen the kernel skeleton patterns. According to [12], enhancing the skeletons results in performance improvement, which explains the effectiveness of RefConv from another perspective.

V. Conclusion

This article proposes reparameterized RefConv, which is the first reparameterization method that augments the priors of existing model structures by establishing extra connections among kernel parameters. As a plug-and-play module to replace the regular convolutional layers, RefConv can significantly improve the performance of various CNNs on multiple tasks without altering the original model structures or introducing extra costs in inference. Moreover, we explain the effectiveness of RefConv by showing its capability of reducing channel redundancy and smoothing the loss landscape, which may inspire further theoretical research on training dynamics. In our future work, we will explore more effective designs of Refocusing Transformations, e.g., by introducing nonlinearity and more advanced operations.

Besides, RefConv is especially designed for convolution, conducting Refocusing Transformation on the learned weights, and refocusing on the representations encoded in these weights to learn new representations. While self-attention projects the input to query, key, and value with three fully connected layers, then queries the attention scores calculated by key and value to obtain the final output. The core of feature extraction in self-attention lies in the calculated attention scores rather than the weights of the projection layers. Consequently, there are no effective weights to be refocused and connected, as the projection layers extract no representations and the attention scores are not weights. Thus, how to design an approach to refocusing on the attention scores or fusing self-attention into the Refocusing Transformation is another future direction.

References

[1] 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.

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

[3] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 11976–11986.

[4] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[5] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 3–19.

[6] 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.

[7] A. Howard et al., "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[8] 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)*, 2018, pp. 116–131.

[9] Y. Zhou, Y. Zhang, Y.-F. Wang, and Q. Tian, "Accelerate CNN via recursive Bayesian pruning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3305–3314.

[10] X. Wang and S. X. Yu, "Tied block convolution: Leaner and better CNNs with shared thinner filters," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, vol. 35, May 2021, pp. 10227–10235.

[11] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, Jan. 2017, pp. 1–11.

[12] X. Ding, Y. Guo, G. Ding, and J. Han, "ACNet: Strengthening the kernel skeletons for powerful CNN via asymmetric convolution blocks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1911–1920.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[14] H. Zhang et al., "ResNeSt: Split-attention networks," 2020, *arXiv:2004.08955*.

[15] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[17] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31×31: Revisiting large kernel design in CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 11963–11975.

[18] J. Chen et al., "Run, don't walk: Chasing higher FLOPS for faster neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2023, pp. 12021–12031.

[19] Y. Rao, W. Zhao, Y. Tang, J. Zhou, S.-N. Lim, and J. Lu, "HorNet: Efficient high-order spatial interactions with recursive gated convolutions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, Jan. 2022, pp. 10353–10366.

[20] S. Liu et al., "More ConvNets in the 2020s: Scaling up kernels beyond 51×51 using sparsity," 2022, *arXiv:2207.03620*.

[21] Y. Chen et al., "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.

[22] R. Zhang, "Making convolutional networks shift-invariant again," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 97, 2019, pp. 7324–7334.

[23] A. Kirchmeyer and J. Deng, "Convolutional networks with oriented 1D kernels," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 6199–6209.

[24] X. Lin, Z. Yan, X. Deng, C. Zheng, and L. Yu, "ConvFormer: Plug-and-play CNN-style transformers for improving medical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, Jan. 2023, pp. 642–651.

[25] 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.*, vol. 31, Jan. 2018, pp. 1–11.

[26] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient channel attention for deep convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11534–11542.

[27] X. Ding et al., "ResRep: Lossless CNN pruning via decoupling remembering and forgetting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 4490–4500.

[28] X. Ding, X. Zhang, J. Han, and G. Ding, "Diverse branch block: Building a convolution as an inception-like unit," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10886–10895.

[29] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "RepVGG: Making VGG-style ConvNets great again," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13733–13742.

[30] X. Ding, H. Chen, X. Zhang, J. Han, and G. Ding, "RepMLPNet: Hierarchical vision MLP with re-parameterized locality," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 578–587.

[31] P. Kumar Anasosalu Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan, "MobileOne: An improved one millisecond mobile backbone," 2022, *arXiv:2206.04040*.

[32] C. Chen, Z. Guo, H. Zeng, P. Xiong, and J. Dong, "RepGhost: A hardware-efficient ghost module via re-parameterization," 2022, *arXiv:2211.06088*.

[33] Z. Cai and Q. Shen, "FalconNet: Factorization for the light-weight ConvNets," 2023, *arXiv:2306.06365*.

[34] C. Li et al., "YOLOv6: A single-stage object detection framework for industrial applications," 2022, *arXiv:2209.02976*.

[35] C.-Y. Wang, A. Bochkovskiy, and H.-Y.-M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 7464–7475.

[36] S. Xu et al., "PP-YOLOE: An evolved version of YOLO," 2022, *arXiv:2203.16250*.

[37] K. Bhardwaj et al., "Collapsible linear blocks for super-efficient super resolution," in *Proc. Mach. Learn. Syst.*, vol. 4, Jan. 2021, pp. 529–547.

[38] Y. Li et al., "NTIRE 2023 challenge on efficient super-resolution: Methods and results," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2023, pp. 1922–1960.

[39] L. Wang, D. Li, L. Tian, and Y. Shan, "Efficient image super-resolution with collapsible linear blocks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 816–822.

[40] S. P. Sharan, A. K. Krishna, A. Siddharth Rao, and V. P. Gopi, "RepArnet: Re-parameterized encoders and attentive feature arsenals for fast video denoising," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 633–639.

[41] L. Guo, J. He, P. Lin, S.-Y. Huang, and J. Wang, "TRScore: A 3D RepVGG-based scoring method for ranking protein docking models," *Bioinformatics*, vol. 38, no. 9, pp. 2444–2451, Apr. 2022.

[42] X. Ding et al., "UniRepLKNet: A universal perception large-kernel ConvNet for audio, video, point cloud, time-series and image recognition," 2023, *arXiv:2311.15599*.

[43] S. Zagoruyko and N. Komodakis, "DiracNets: Training very deep neural networks without skip-connections," 2017, *arXiv:1706.00388*.

[44] D. Marnerides, T. Bashford-Rogers, J. Hatchett, and K. Debattista, "ExpandNet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 37–49, May 2018.

[45] J. Cao et al., "DO-Conv: Depthwise over-parameterized convolutional layer," *IEEE Trans. Image Process.*, vol. 31, pp. 3726–3736, 2022.

[46] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, Dec. 2016, pp. 901–909.

[47] L. Huang, X. Liu, Y. Liu, B. Lang, and D. Tao, "Centered weight normalization in accelerating training of deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2822–2830.

[48] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li, "Orthogonal weight normalization: Solution to optimization over multiple dependent Stiefel manifolds in deep neural networks," in *Proc. AAAI Conf. Artif. Intell.*, Jan. 2017, pp. 1–8.

[49] Y. Zhang, J. Zhang, Q. Wang, and Z. Zhong, "DyNet: Dynamic convolution for accelerating convolutional neural networks," 2020, *arXiv:2004.10694*.

[50] C. Li, A. Zhou, and A. Yao, "Omni-dimensional dynamic convolution," 2022, *arXiv:2209.07947*.

[51] C. Li and A. Yao, "KernelWarehouse: Towards parameter-efficient dynamic convolution," 2023, *arXiv:2308.08361*.

[52] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, "CondConv: Conditionally parameterized convolutions for efficient inference," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, Jan. 2019, pp. 1–12.

[53] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic convolution: Attention over convolution kernels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11030–11039.

[54] N. Ma, X. Zhang, J. Huang, and J. Sun, "WeightNet: Revisiting the design space of weight networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 776–792.

[55] D. Ha, A. Dai, and Q. V. Le, "HyperNetworks," 2016, *arXiv:1609.09106*.

[56] N. Quader, M. M. I. Bhuiyan, J. Lu, P. Dai, and W. Li, "Weight excitation: Built-in attention mechanisms in convolutional neural networks," in *Proc. 16th Eur. Conf. Comput. Vis. (ECCV)*, Glasgow, U.K. Cham, Switzerland: Springer, Jan. 2020, pp. 87–103.

[57] X. Lin, L. Ma, W. Liu, and S.-F. Chang, "Context-gated convolution," in *Proc. Eur. Conf. Comput. Vis.*, Glasgow, U.K. Cham, Switzerland: Springer, Aug. 2020, pp. 701–718.

[58] B. D. Brabandere, X. Jia, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, Jan. 2016, pp. 1–9.

[59] S. Bako et al., "Kernel-predicting convolutional networks for denoising Monte Carlo renderings," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–14, Aug. 2017.

[60] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, "Burst denoising with kernel prediction networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2502–2510.

[61] V. T. Pham, Y. Zniyed, and T. P. Nguyen, "Enhanced network compression through tensor decompositions and pruning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 3, pp. 4358–4370, Mar. 2025.

[62] T. Hao, X. Ding, J. Han, Y. Guo, and G. Ding, "Manipulating identical filter redundancy for efficient pruning on deep and complicated CNN," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 11, pp. 16831–16844, Nov. 2024.

[63] G. Fang, X. Ma, and M. Song, "Depgraph: Towards any structural pruning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2023, pp. 16091–16101.

[64] J. Liu et al., "Discrimination-aware network pruning for deep model compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 8, pp. 4035–4051, Aug. 2022.

[65] E. Diao, G. Wang, J. Zhan, Y. Yang, J. Ding, and V. Tarokh, "Pruning deep neural networks from a sparsity perspective," in *Proc. ICLR*, Jan. 2023, pp. 1–12.

[66] W. Fei, W. Dai, C. Li, J. Zou, and H. Xiong, "General bitwidth assignment for efficient deep convolutional neural network quantization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5253–5267, Oct. 2022.

[67] H. Peng, J. Wu, Z. Zhang, S. Chen, and H.-T. Zhang, "Deep network quantization via error compensation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 4960–4970, Sep. 2021.

[68] J. Cheng, J. Wu, C. Leng, Y. Wang, and Q. Hu, "Quantized CNN: A unified approach to accelerate and compress convolutional networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4730–4743, Oct. 2018.

[69] J. Liu, B. Zhuang, P. Chen, C. Shen, J. Cai, and M. Tan, "Single-path bit sharing for automatic loss-aware model compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 10, pp. 12459–12473, Oct. 2023.

[70] S. Xu, S. Zhang, J. Liu, B. Zhuang, Y. Wang, and M. Tan, "Generative data free model quantization with knowledge matching for classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 12, pp. 7296–7309, Dec. 2023.

[71] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[72] D. Zhou, Q. Hou, Y. Chen, J. Feng, and S. Yan, "Rethinking bottleneck structure for efficient mobile network design," in *Proc. Eur. Conf. Comput. Vis.*, Glasgow, U.K. Cham, Switzerland: Springer, Aug. 2020, pp. 680–697.

[73] D. Li, A. Zhou, and A. Yao, "HBONet: Harmonious bottleneck on two orthogonal dimensions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3315–3324.

[74] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[75] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[76] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, and W. Wu, "Incorporating convolution designs into visual transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 579–588.

[77] Z. Chen, L. Xie, J. Niu, X. Liu, L. Wei, and Q. Tian, "Visformer: The vision-friendly transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 589–598.

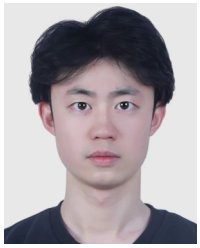
[78] J. Guo et al., "CMT: Convolutional neural networks meet vision transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12175–12185.

[79] C. Si, W. Yu, P. Zhou, Y. Zhou, X. Wang, and S. Yan, "Inception transformer," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 23495–23509.

[80] W. Yu et al., "MetaFormer baselines for vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 2, pp. 896–912, Feb. 2024.

[81] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[82] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.
[83] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50× fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*.
[84] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 1492–1500.
[85] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10428–10436.
[86] M. Tan and Q. V. Le, "EfficientNetv2: Smaller models and faster training," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10096–10106.
[87] A. Wang, H. Chen, Z. Lin, J. Han, and G. Ding, "Rep ViT: Revisiting mobile CNN from ViT perspective," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2024, pp. 15909–15920.
[88] P. K. Anasosalu Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan, "FastViT: A fast hybrid vision transformer using structural reparameterization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 5762–5772.
[89] R. Varghese and M. Sambath, "YOLOv8: A novel object detection algorithm with enhanced performance and robustness," in *Proc. Int. Conf. Adv. Data Eng. Intell. Comput. Syst. (ADICS)*, Apr. 2024, pp. 1–6.
[90] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.
[91] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.
[92] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.
[93] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv:1706.06083*.
[94] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
[95] Y. Dong et al., "Boosting adversarial attacks with momentum," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9185–9193.
[96] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–11.
[97] X. Ding, G. Ding, J. Han, and S. Tang, "Auto-balanced filter pruning for efficient convolutional neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, Apr. 2018, pp. 1–8.
[98] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. Peter Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*.
[99] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient DNNs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, Jan. 2016, pp. 1–9.
[100] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.



Zhicheng Cai received the B.S. degree from Nanjing University, Nanjing, China, in 2022, where he is currently pursuing the master's degree with the School of Electronic Science and Technology.

His research interests include deep learning and computer vision.

Xiaohan Ding received the Ph.D. degree from Tsinghua University, Beijing, China, in 2022.

He proposed a methodology named structural reparameterization and several important applications. He has published pioneering research works on large-kernel convolutional neural networks and model compression. His representative publications include RepVGG, RepLKNet, UniRepLKNet, C-SGD, and ResRep.

Qiu Shen (Member, IEEE) received the B.S. degree in electrical engineering and information science and the Ph.D. degree in signal and information processing from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively.

From 2009 to 2016, she was with the Huawei 2012 Laboratory, Shenzhen, China, and Nanjing University of Aeronautics and Astronautics, Nanjing, China. She is currently a Faculty Member with the Electronic Science and Engineering School, Nanjing University, Nanjing. Her current research focuses on the next-generation video coding, collaborative video compression and analysis, and vision representation.

Xun Cao (Member, IEEE) received the B.S. degree from Nanjing University, Nanjing, China, in 2006, and the Ph.D. degree from the Department of Automation, Tsinghua University, Beijing, China, in 2012.

He held visiting positions at Philips Research, Aachen, Germany, in 2008, and Microsoft Research Asia, Beijing, from 2009 to 2010. He was a Visiting Scholar with The University of Texas at Austin, Austin, TX, USA, from 2010 to 2011. He is currently a Professor with the School of Electronic Science and Engineering, Nanjing University. His research interests include computational photography, image-based modeling and rendering, and virtual reality (VR)/augmented reality (AR) systems.