Parameter Efficient Dynamic Convolution via Tensor Decomposition

Zejiang Hou
zejiangh@princeton.edu
Sun-Yuan Kung
kung@princeton.edu

Department of Electrical Engineering
Princeton University
Princeton, NJ, USA

Appendix

This appendix is organized as follows. In Appendix A, we give the visualization analysis of PEDConv to further evidence its effectiveness. In Appendix B, we discuss the inference run-time of the proposed PEDConv compared to static convolutional baseline and other dynamic convolution methods. In Appendix C, we investigate the performance of PEDConv at different model sizes. In Appendix D, we perform a series of ablation studies on the generator module design. In Appendix E, we compare the multiplicative type reparameterization in PEDConv versus a variant of additive type reparameterization. In section F, we provide the mathematical derivations for Equation 4 and 6 in the main paper.

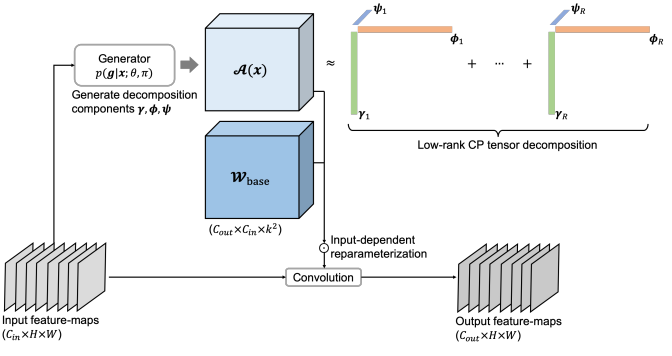
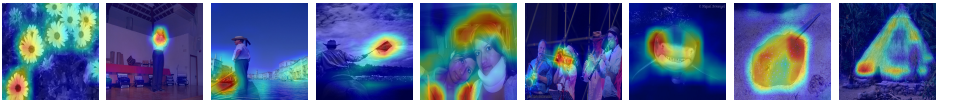


Figure 1: Illustration of the proposed parameter-efficient dynamic convolution (PEDConv).

A Visualization

To further evidence the effectiveness of PEDConv, we adopt Grad-CAM [4] as the tool to visualize the attention-map produced by ResNet50 with PEDConv for classification on ImageNet. Grad-CAM generates the heatmap representation that helps understand as to what region of the input image influence most to the model's prediction. The results are shown in Figure 2. It can be clearly seen that the attention-maps produced by ResNet50 with PEDConv can more precisely locate the target objects without expanding to the backgroud areas or non-target objects compared to baseline ResNet50. These visualization may help explain why PEDConv achieves significant accuracy gain over the baseline models.

PEDConv



Baseline

Figure 2: Grad-CAM visualization of PEDConv applied to ResNet50 compared to baseline static convolution on images randomly sampled from ImageNet. It is obvious that ResNet50 with PEDConv captures richer context information and more precisely locates objects of interest, which may explain the accuracy improvement of PEDConv over the baseline.

B Inference run-time analysis

In Table 1, we report the CPU inference run-time of PEDConv applied to diverse CNN architectures. The run-time is measured by averaging the feed-forward inference time of 5000 images with batch size 1 on the PyTorch platform. Firstly, compared with baseline static convolution, PEDConv consumes only $4 \sim 7\%$ more inference run-time for the considered four CNN architectures on ImageNet, while achieving significant Top-1 accuracy improvement. This result suggests that PDConv can mostly maintain the inference efficiency. Moreover, compared with previous dynamic convolution [2], our method demonstrates less inference run-time and better accuracy. This result further evidences the advantage brought by our parameter-efficient tensor-decomposition based reparameterization, compared to the parameter-inefficient multi-weight aggregation schema in [2].

Model	Method	FLOPs	Run-time	Top-1 (%)
MobileNetV1	Baseline	569M	167ms	71.9
	PEDConv	579M	177ms	75.9
MobileNetV2	Baseline	300M	222ms	72.0
	DYConv [2]	313M	248ms	75.2
	PEDConv	307M	240ms	75.5
ResNet18	Baseline	1.81G	112ms	70.4
	DYConv [2]	1.85G	125ms	72.7
	PEDConv	1.83G	120ms	74.2
ResNet50	Baseline	4.1G	276ms	76.2
	PEDConv	4.2G	288ms	80.5

Table 1: Realistic inference run-time of the proposed PEDConv applied to diverse CNN architectures on ImageNet. PEDConv can mostly maintain the inference efficiency with small penalty in run-time while significantly improving Top-1 accuracy over baseline. Moreover, PEDConv demonstrates better accuracy-efficiency trade-off compared to prior art.

C PEDConv at different model sizes

We investigate the performance of PEDConv applied to different model sizes. On ImageNet dataset, we carry out experiments on [0.25x, 0.5x, 0.75x, 1x] of MobileNetV1 by using PEDConv to replace the standard convolutions. The results are shown in Figure 3. As observed, smaller models with PEDConv shows larger accuracy improvements than larger models. We suspect this is because smaller models have too few channels in each layer to extract useful features. By replacing standard convolution with PEDConv, small models may improve their representation ability, leading to better accuracy.

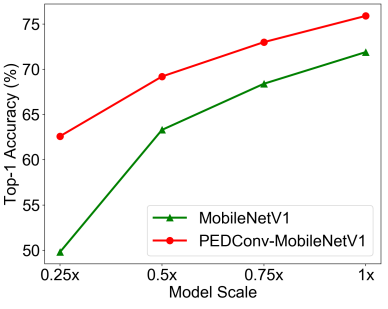


Figure 3: Results of applying PEDConv to different model sizes of MobileNetV1 on ImageNet.

D Ablation study on generator module design

We investigate different design choices of the generator module for generating the decomposition components (cf. Equation 7 in the main paper). By default, the generator takes as input the global context extracted by applying global average pooling (GAP) to the layer's input feature-maps corresponding a certain input image. Then, a bottleneck two-layer MLP is used to generate the decomposition components, where the bottleneck size is set to $C_{in}/16$. In the first part of Table 2, we compare GAP (1^{st}-order pooling) versus bilinear pooling [3] (2^{nd}-order pooling) to examine whether second-order statistic of the feature-maps can better guide the input-dependent weight reparameterization for dynamic convolution. The result shows that GAP yields higher Top-1 accuracy than bilinear pooling, suggesting that 1^{st}-order global context is already discriminative and informative, while being more efficient than 2^{nd}-order one. In the second part of Table 2, we vary the bottleneck size of the MLP, where a wider bottleneck size can further improve the accuracy of PEDConv, but at cost of more model parameters. In the third part of Table 2, we consider a partial sharing scheme, where each residual block shares the same generator module. The result indicates that using separate generator module for each individual layer gives the best accuracy.

E Multiplicative vs. Additive reparameterization

By default, PEDConv adopts multiplicative type reparameterization, i.e., $\mathbf{W}(\mathbf{x}) = \mathbf{W}_{\text{base}} \odot (\gamma(\mathbf{x}) \otimes \phi(\mathbf{x}) \otimes \psi(\mathbf{x}))$ (cf. Equation 2 in the main paper). For comparison, we experiment with a variant of PEDConv employing additive type reparameterization, i.e., $\mathbf{W}(\mathbf{x}) = \mathbf{W}_{\text{base}} + (\gamma(\mathbf{x}) \otimes \phi(\mathbf{x}) \otimes \psi(\mathbf{x}))$. The results are shown in Table 3, where we apply PEDConv to ResNet18 on ImageNet. As shown, our multiplicative type tensor decomposition based weight reparameterization leads to better Top-1 accuracy.

Generator ablations		Params.	FLOPs	Top-1 (%)
Global context	GAP (1^{st}-order)	11.9M	1.83G	74.2
	Bilinear Pooling (2^{nd}-order)	11.9M	2.04G	73.6
Bottleneck size	$C_{in}/16$	11.9M	1.83G	74.2
	$C_{in}/4$	12.6M	1.84G	74.9
Partial sharing	X	11.9M	1.83G	74.2
	✓	11.8M	1.83G	73.4

Table 2: Ablation study on the generator module design. Results are obtained by applying PEDConv to ResNet18 on ImageNet.

Method	Params.	FLOPs	Top-1 (%)
Multiplicative reparameterization	11.9M	1.83G	74.2
Additive reparameterization	11.9M	1.83G	73.0

Table 3: Comparison of multiplicative reparameterization in PEDConv versus a variant of additive reparameterization. Results are obtained by applying PEDConv to ResNet18 on ImageNet.

F Derivations for Equation 4 and 6 in the main paper

Derivation for Eq.4

$$\begin{aligned}
\log p_\theta(\mathbf{y}|\mathbf{x}) &= D_{\mathrm{KL}}(q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{g}|\mathbf{x},\mathbf{y})) + \mathbb{E}_{q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})}[-\log q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y}) + \log p_\theta(\mathbf{y},\mathbf{g}|\mathbf{x})] \\
&\geq \mathbb{E}_{q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})}[-\log q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y}) + \log p_\theta(\mathbf{y},\mathbf{g}|\mathbf{x})] \\
&= \mathbb{E}_{q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})}[-\log q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y}) + \log p_\theta(\mathbf{g}|\mathbf{x})] + \mathbb{E}_{q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{g})] \\
&= -D_{\mathrm{KL}}(q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y}\|p_\theta(\mathbf{g}|\mathbf{x})) + \mathbb{E}_{q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{g})] \\
&\overset{(a)}{=} \mathbb{E}_{p_\theta(\mathbf{g}|\mathbf{x})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{g})] \qquad (1)
\end{aligned}$$

where in step (a), we follow [5] to make the approximate posterior $q_\phi(\mathbf{g}|\mathbf{x},\mathbf{y})$ share the same form with the conditional prior $p_\theta(\mathbf{g}|\mathbf{x})$, such that the KL divergence between them becomes zero. By sharing the same form, the training and testing pipeline would become consistent.

Derivation for Eq.6 Based on the chain rule of mutual information,

$$I((\mathbf{x},\mathbf{y});\mathbf{g}) = I(\mathbf{x};\mathbf{g}) + I(\mathbf{y};\mathbf{g}|\mathbf{x})$$

For the first term, since the true posterior distributions are intractable, we use $p_\theta(\mathbf{x}|\mathbf{g})$ to approximate the posteriors by following the Variational Information Maximization [1]:

$$\begin{aligned}
I(\mathbf{x};\mathbf{g}) &= H(\mathbf{x}) - H(\mathbf{x}|\mathbf{g}) \\
&= H(\mathbf{x}) + \mathbb{E}_{\mathbf{g}\sim p(\mathbf{g})}[\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\mathbf{g})}[\log p(\mathbf{x}|\mathbf{g})]] \\
&= H(\mathbf{x}) + \mathbb{E}_{\mathbf{g}\sim p(\mathbf{g})}[D_{\mathrm{KL}}(p(\mathbf{x}|\mathbf{g})\|p_\theta(\mathbf{x}|\mathbf{g})) + \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\mathbf{g})}[\log p_\theta(\mathbf{x}|\mathbf{g})]] \\
&\geq H(\mathbf{x}) + \mathbb{E}_{\mathbf{g}\sim p(\mathbf{g})}[\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\mathbf{g})}[\log p_\theta(\mathbf{x}|\mathbf{g})]] \\
&= H(\mathbf{x}) + \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\mathbb{E}_{\mathbf{g}\sim p(\mathbf{g}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{g})]]
\end{aligned}$$

where $H(\cdot)$ denotes the entropy of a random variable. $H(\mathbf{x})$ is a constant term for a given input $\mathbf{x}$. Similarly, for the second term, it holds that

$$I(\mathbf{y};\mathbf{g}|\mathbf{x}) \geq H(\mathbf{y}|\mathbf{x}) + \mathbb{E}_{\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})}[\mathbb{E}_{\mathbf{g}\sim p(\mathbf{g}|\mathbf{y},\mathbf{x})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{g})]]$$

By combining the lower bounds of both first and second term, we can obtain the lower bound for the MI objective as Equation 6 in the main paper.

References

[1] David Barber Felix Agakov. The im algorithm: a variational approach to information maximization. *Advances in neural information processing systems*, 16:201, 2004.

[2] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11030–11039, 2020.

[3] Tsung-Yu Lin and Subhransu Maji. Improved bilinear pooling with cnns. *arXiv preprint arXiv:1707.06772*, 2017.

[4] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[5] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.