Self-moving Point Representations for Continuous Convolution - Appendix

Sanghyeon Kim[1] Eunbyung Park[1,2*]
[1]Department of Electrical and Computer Engineering, Sungkyunkwan University
[2]Department of Artificial Intelligence, Sungkyunkwan University

1. Experimental details

1.1. Sequential data and image classification

In each convolution filter, SMPConv has 30 weight points for 1D and 16 weight points for 2D. For SMPConv1D, we sample the point locations from zero mean truncated gaussian distribution with $\sigma = 0.1$. Because of causal convolution, we sample in the range $(-1, 0)$ rather than $(-1, 1)$. For SMPConv2D, we sample the point locations from 2D zero mean truncated gaussian with $\Sigma = [[\sigma_1, 0], [0, \sigma_2]]$, where $\sigma_1 = \sigma_2 = 0.05$. We initialize radius as $r \approx \frac{2}{k}d$, where k is kernel size and d is dimension of input (i.e., $d = 1$ for 1D, $d = 2$ for 2D). In 2D, the kernel size means the width of the kernel. The size of the additional small kernel is 5 for 1D and 3×3 for 2D, respectively. Following FlexConv [7], we use batch normalization [4] after convolution and skip connection.

We train our networks using Adam [5] optimizer. We use a cosine annealing learning rate scheduling with warm-up epochs. The learning rate for radius parameters is set to be $0.1\times$ smaller than the regular learning rate. During the training, the radius range is clipped from 0.0001 to 0.1. More details for each data are shown in Tab. 1. For sequential data experiments, we train our model with a single NVIDIA A100 GPU. We use a single RTX3090 GPU for CIFAR10 experiments.

1.2. Image classification on ImageNet-1k

Our large-scale variants of SMPConv networks have the same architecture as RepLKNet [3] except for large kernel convolution, which is replaced by our `SMP`. Like [3], we set the kernel size of each stage to [31, 29, 27, 13] and use additional 5×5 convolution for reparameterization trick. We use $\lfloor \frac{k^2}{4} \rfloor$ weight points for each `SMP` depth-wise version, which shares weight points over channels, where k is the kernel size of corresponding each block. The point locations and radius are initialized in the same way as Sec. 1.1 SMPConv2D with $\sigma_1 = \sigma_2 = 0.2$.

Our models are trained for 300 epochs using AdamW [6] optimizer. We set the batch size of 2048. The initial learning rate is set to 4×10^{-3} with cosine annealing scheduling and 10 warm-up epochs. We use RandAugment [1] in Timm [9](”rand-m9-mstd0.5-inc1”), Label Smoothing [8] coefficient of 0.1, Mixup [11] with $\alpha = 0.8$, Cutmix [10] with $\alpha = 1.0$, Rand Erasing [12] with probability of 25%, Stochastic Depth with drop path rate of 10% for SMPConv-T, and 50% for SMPConv-B, and model EMA(exponential moving average) with a decay factor of 0.9999. For fast depth-wise convolution computation, we use block-wise(inverse) *implicit gemm* algorithm implemented by [3]. We train both SMPConv-T and SMPConv-B with 4 NVIDIA A100 GPUs.

2. Additional results

2.1. Larger kernels

We set the kernel size of each stage to [31, 29, 27, 13] for large-scale variants of SMPConv networks following RepLKNet [3]. Although the current kernel sizes are larger than conventional convolution, we evaluate whether our model is trained without performance degradation even when using larger kernels.

To conduct this experiment, we design a new variant, SMPConv-mobile. For the mobile variant, the number of blocks and the number of channels for each stage is [2, 2, 2, 2] and [64, 128, 256, 320], respectively. Also, we use $\lfloor \frac{k^2}{8} \rfloor$ weight points for each `SMP` and reduce the expansion ratio of feed-forward networks from 4 to 2. We train this variant for 120 epochs and do not use Stochastic Depth. Other training settings are same as Sec. 1.2. We set the kernel size of each stage to [31, 29, 27, 13] for SMPConv-mobile31 and [51, 49, 47, 13] for SMPConv-mobile51.

In ImageNet-1k [2] image classification, SMPConv-mobile31 and SMPConv-mobile51 get **73.5**% and **73.7**% top-1 accuracy, respectively. Thus, using our `SMP`, convolution kernel sizes can be increased without performance degradation, even in large-scale data.

References

[1] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation

*Corresponding authors

	sMNIST	pMNIST	sCIFAR10	CT	SC	SC-raw	CIFAR10
lr	0.0001	0.0001	0.0002	0.0001	0.001	0.001	0.005
epoch	200	200	200	300	300	160	210
warm-up	5	5	5	5	5	10	10
dropout	0	0	0	0	0.2	0.1	0.1
# of batch	64	64	64	64	64	64	64
weight decay	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
kernel size	784	784	1024	182	101	16000	32×32

Table 1. Hyper-parameter details

with a reduced search space. In *CVPRW*, pages 702–703, 2020. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 1

[3] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *CVPR*, pages 11963–11975, 2022. 1

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. PMLR, 2015. 1

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[7] David W Romero, Robert-Jan Bruintjes, Jakub M Tomczak, Erik J Bekkers, Mark Hoogendoorn, and Jan C van Gemert. Flexconv: Continuous kernel convolutions with differentiable kernel sizes. *arXiv preprint arXiv:2110.08059*, 2021. 1

[8] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016. 1

[9] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 1

[10] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6023–6032, 2019. 1

[11] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1

[12] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, volume 34, pages 13001–13008, 2020. 1