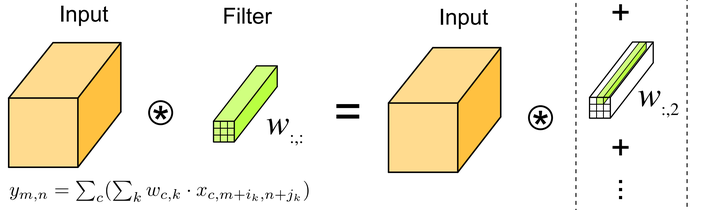
Constructing Fast Network through Deconstruction of Convolution

Yunho Jeon and Junmo Kim
jyh2986@kaist.ac.kr, junmo.kim@kaist.ac.kr
School of Electrical Engineering, KAIST, South Korea

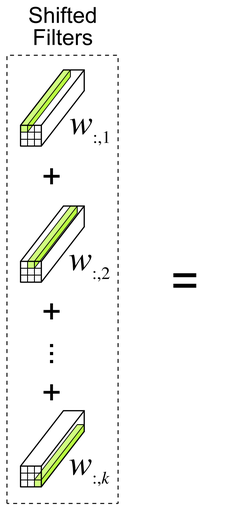
NeurIPS 2018
Neural Information Processing Systems

Deconstruction of Convolution

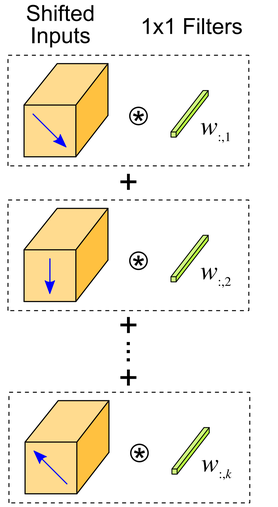
- Spatial Convolution
  - ➔ Shift function + 1x1 convolution



$y_{m,n} = \sum_c (\sum_k w_{c,k} \cdot x_{c,m+i_k,n+j_k})$

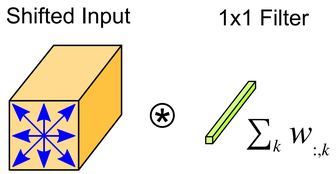


$y_{m,n} = \sum_k (\sum_c w_{c,k} \cdot x_{c,m+i_k,n+j_k})$



$y_{m,n} = \sum_k (\sum_c w_{c,k} \cdot S_k(x_{c,m,n}))$

Reduce Shifted Input
$S_k \rightarrow S$



$y_{m,n} = \sum_c (\sum_k w_{c,k} \cdot S(x_{c,m,n}))$
$= \sum_c S(x_{c,m,n}) \cdot (\sum_k w_{c,k})$

If sharing shifted inputs,

- Reduction in FLOPs & memory access
- But, limited if shifting to one direction
- ➢ **How to shift input values?**

How to make a fast network?

- Reduce computational complexity (FLOPs)
  - Use Depthwise or Grouped convolution
  - Network pruning
  - → **But, Lower FLOPs ≠ Faster Speed**
- *Reduce memory access*
  - Reduce spatial convolutions
- *Maximize utilization of accessed memory*
  - Use 1x1 convolutions

Active Shift Layer (ASL)

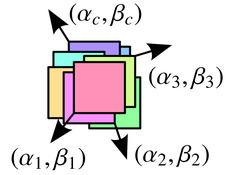
- Use depthwise shift
- **Introduce new shift parameters** for each channel

$\theta_s = \{(\alpha_c, \beta_c) | 1 \le c \le C\}$

$$S_C^{\theta_s}(\boldsymbol{X}) = \begin{bmatrix} \boldsymbol{X}_{1,:}^{(\alpha_1,\beta_1)} \\ \boldsymbol{X}_{2,:}^{(\alpha_2,\beta_2)} \\ \cdots \\ \boldsymbol{X}_{C,:}^{(\alpha_C,\beta_C)} \end{bmatrix}$$

- Expand to *non-integer shift* using interpolation

$$\begin{aligned} \tilde{x}_{c,m+\alpha_c,n+\beta_c} =& Z_c^1 \cdot (1-\Delta\alpha_c) \cdot (1-\Delta\beta_c) + Z_c^3 \cdot \Delta\alpha_c \cdot (1-\Delta\beta_c) \\ &+ Z_c^2 \cdot (1-\Delta\alpha_c) \cdot \Delta\beta_c + Z_c^4 \cdot \Delta\alpha_c \cdot \Delta\beta_c, \\ &\Delta\alpha_c = \alpha_c - \lfloor\alpha_c\rfloor, \Delta\beta_c = \beta_c - \lfloor\beta_c\rfloor, \end{aligned}$$

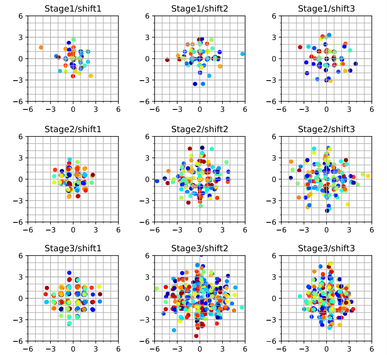


- Shift values are **Differentiable & Learnable**

Experimental Results (CIFAR)

- vs ShiftNet [CVPR2018]
  - Grouped integer shift
    - Heuristically assigned
  - [Conv1-Shift-Conv1] block

Param(M)	ShiftNet[23] C10	ShiftNet[23] C100	ASNet(ours) C10	ASNet(ours) C100
0.035	86.66	55.62	89.14	63.43
0.1	90.08	62.32	91.62	68.83
0.19	90.59	68.64	92.54	70.68
0.28	91.69	69.82	92.93	71.83
0.28	-	-	93.52	73.07
1.2	93.17	72.56	93.73	73.46
0.99	-	-	94.53	76.73



Experimental Results (ImageNet)

Network	Top-1	Top-5	Param(M)	FLOPs(M)	Inference Time[a] CPU(ms)	Inference Time[a] GPU(ms)
MobileNetV1[8]	70.6	-	4.2	569	-	-
ShiftNet-A[23]	70.1	89.7	4.1	1.4G	74.1	10.04
MobileNetV2[18]	71.8	**91**	3.47	300	54.7	7.07
AS-ResNet-w68(ours)	**72.2**	90.7	3.42	729	**47.9**	**6.73**
ShuffleNet-×1.5[26]	71.3	-	3.4	292	-	-
MobileNetV2-×0.75	69.8	**89.6**	2.61	209	40.4	6.23
AS-ResNet-w50(ours)	**69.9**	89.3	1.96	404	32.1	6.14
MobileNetV2-×0.5	65.4	86.4	1.95	97	**26.8**	**5.73**
MobileNetV1-×0.5	63.7	-	1.3	149	-	-
SqueezeNet[10]	57.5	80.3	1.2	-	-	-
ShiftNet-B	61.2	83.6	1.1	371	31.8	7.88
AS-ResNet-w32(ours)	**64.1**	**85.4**	0.9	171	**18.7**	**5.37**
ShiftNet-C	58.8	82	0.78	-	-	-

[a]Measured by Caffe [12] using an Intel i7-5930K CPU with a single thread and GTX Titan X (Maxwell). Inference time for MobileNet and ShiftNet (including FLOPs) are measured by using their network description.

Code is available at https://github.com/jyh2986/Active-Shift