Outline

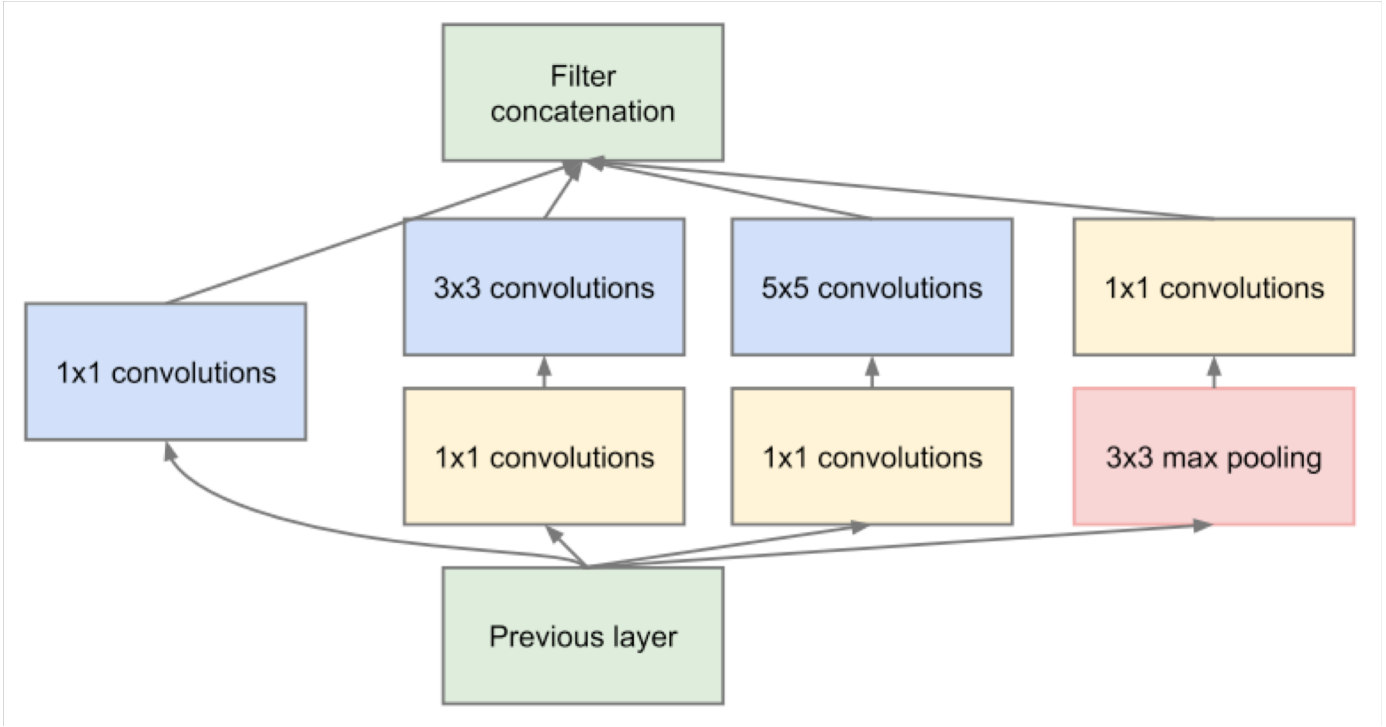
- Architecture related
- Other topics

Outline

- Architecture related
  - *Tensor decomposition*
  - *Channel shuffle*
  - *Inverted bottleneck*
  - *Spatial operations*
  - *Densely connections*
  - *Attention blocks*
  - *Normalization*
  - *Inference efficiency*
- Other topics

Tensor Decomposition

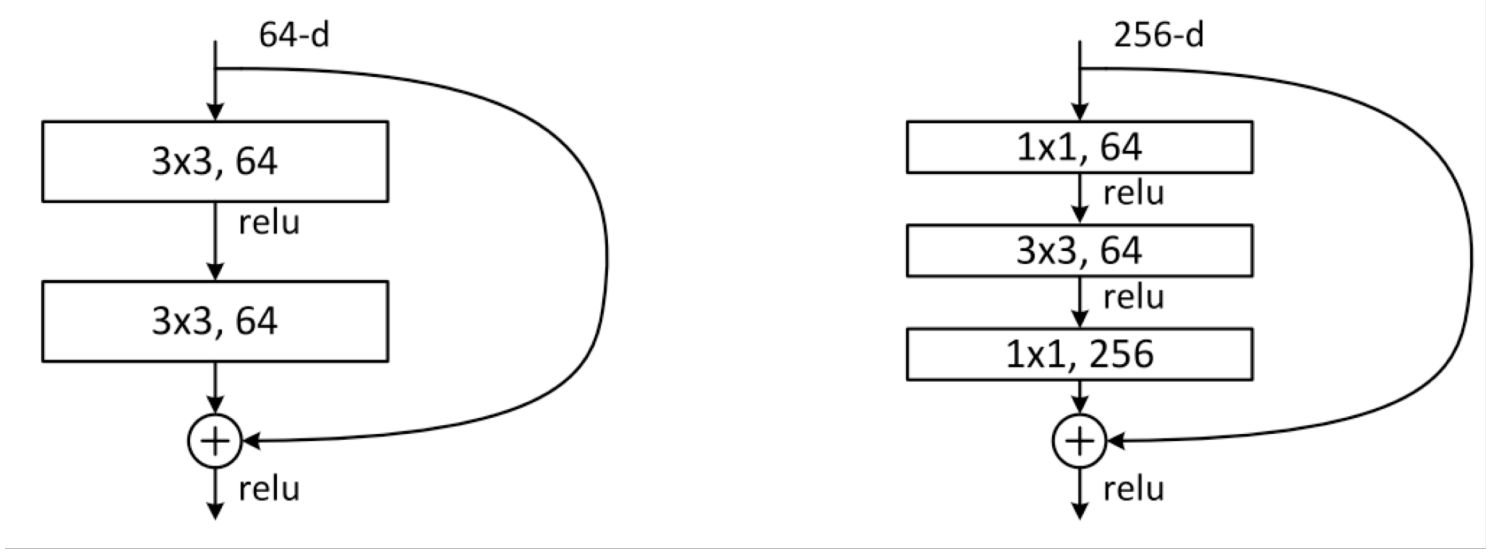
- 1x1 + KxK
  - *GoogleNet*



Szegedy, Christian, et al. "Going deeper with convolutions." Cvpr, 2015.

Tensor Decomposition

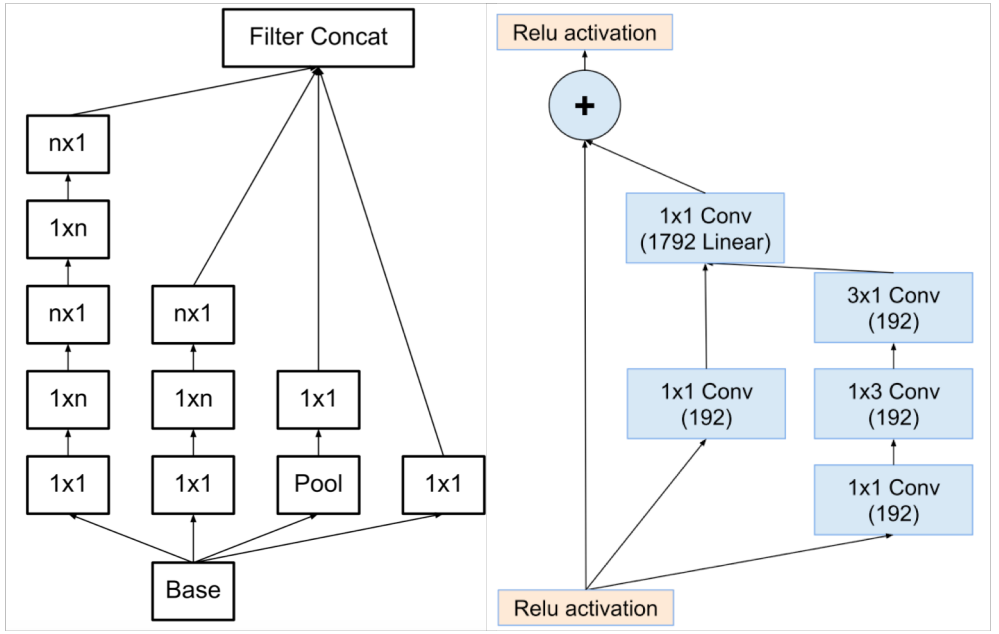
- 1x1 + KxK + 1x1
  - *Bottlenecks*

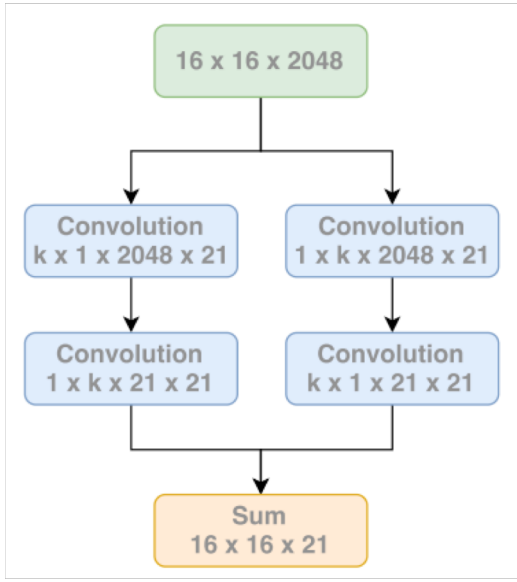


He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition.

Tensor Decomposition

- 1xM + Mx1
  - *Inception v3, v4*
  - *“Large kernel matters”*





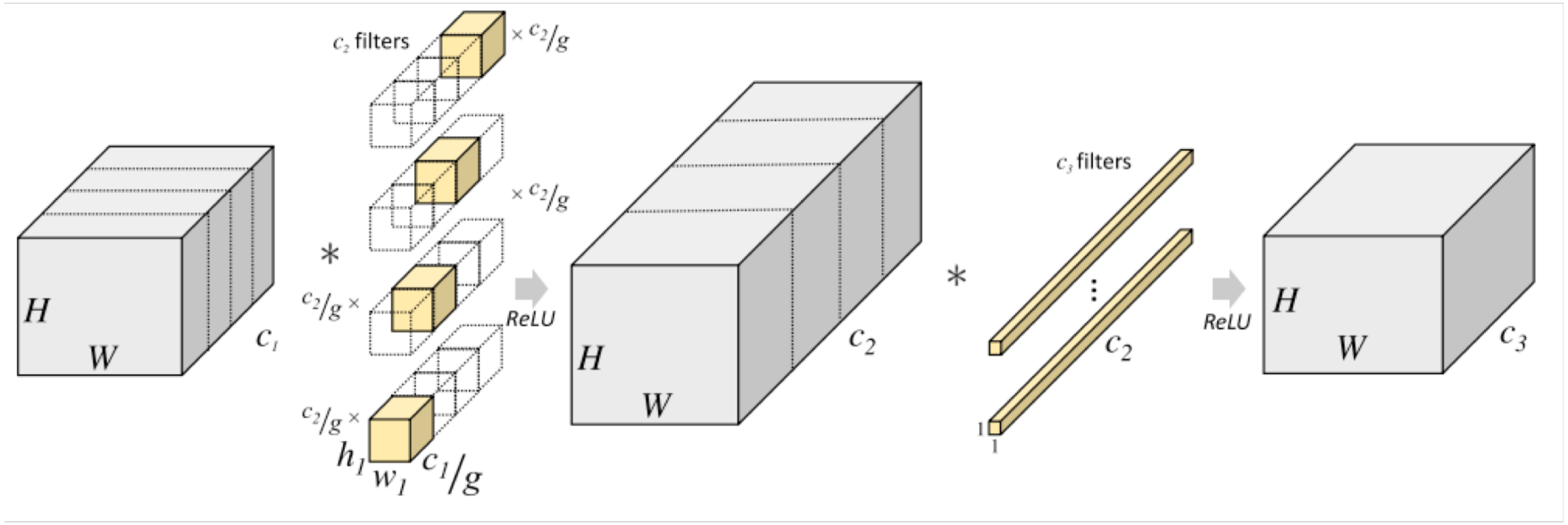
Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *Cvpr*. 2016.

Szegedy, Christian, et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." *AAAI*. Vol. 4. 2017.

Peng, Chao, et al. "Large Kernel Matters--Improve Semantic Segmentation by Global Convolutional Network.” *Cvpr 2017*

Tensor Decomposition

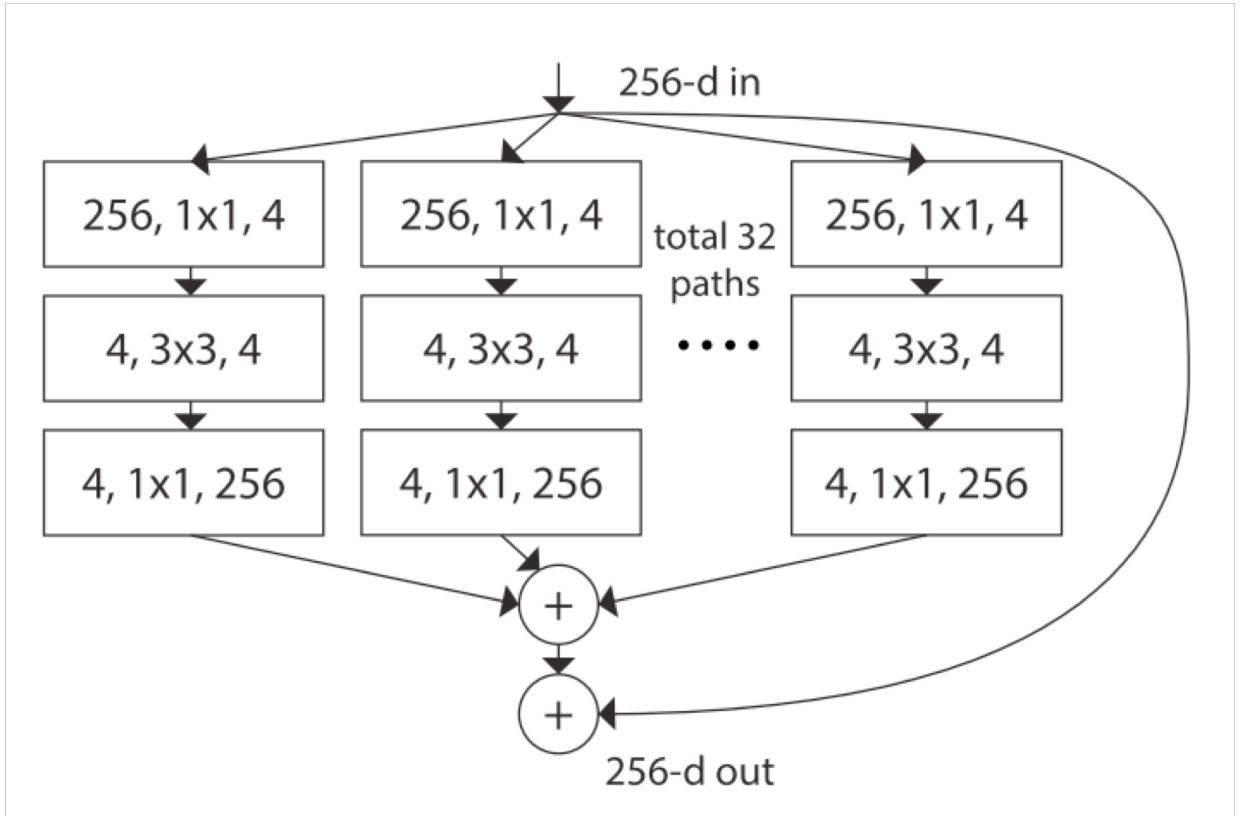
- GConv + 1x1
  - *DeepRoots*



Ioannou, Yani, et al. "Deep roots: Improving CNN efficiency with hierarchical filter groups." (2017).

Tensor Decomposition

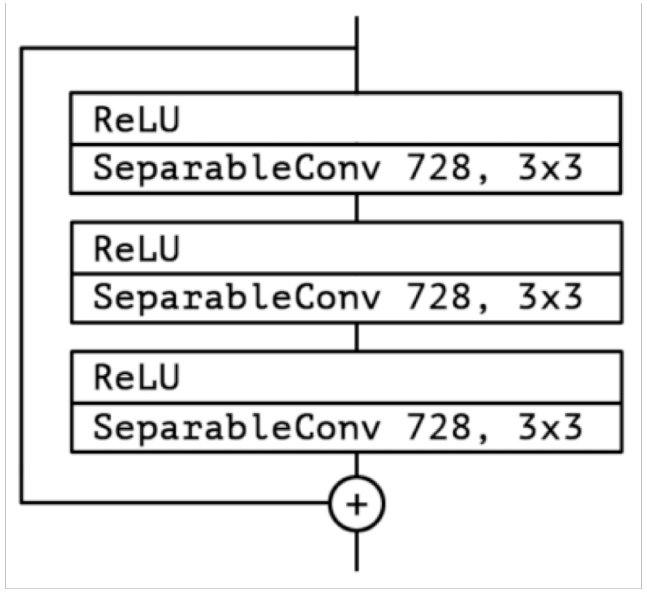
- 1x1 + GConv + 1x1
  - *ResNeXt*
- Applications
  - *For competitions (with SE)*
  - *Liveness*

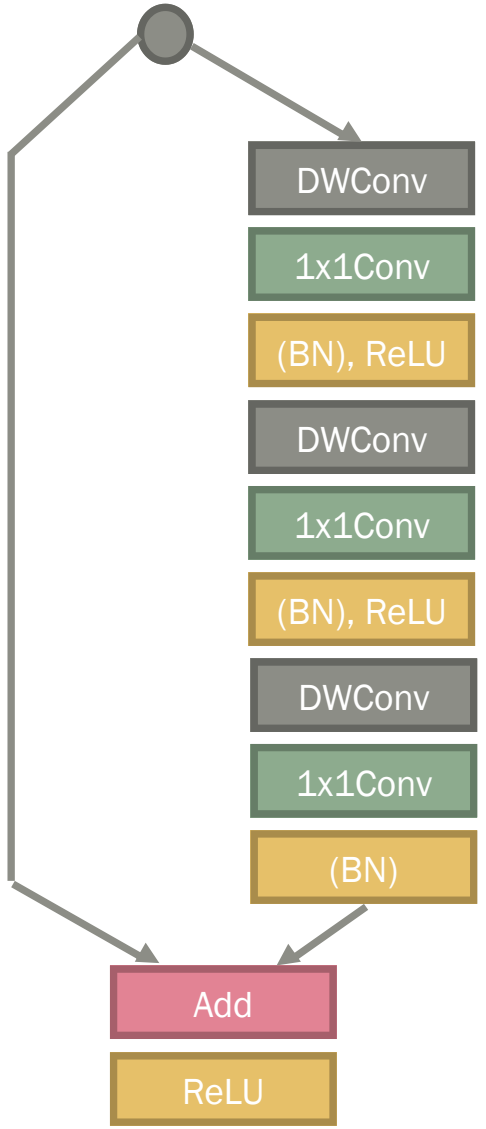


Xie, Saining, et al. "Aggregated residual transformations for deep neural networks." *(CVPR)*

Tensor Decomposition

- DWConv + 1x1
  - *Xception*
  - *Light-head DET*
  - *Waterfall Xception*





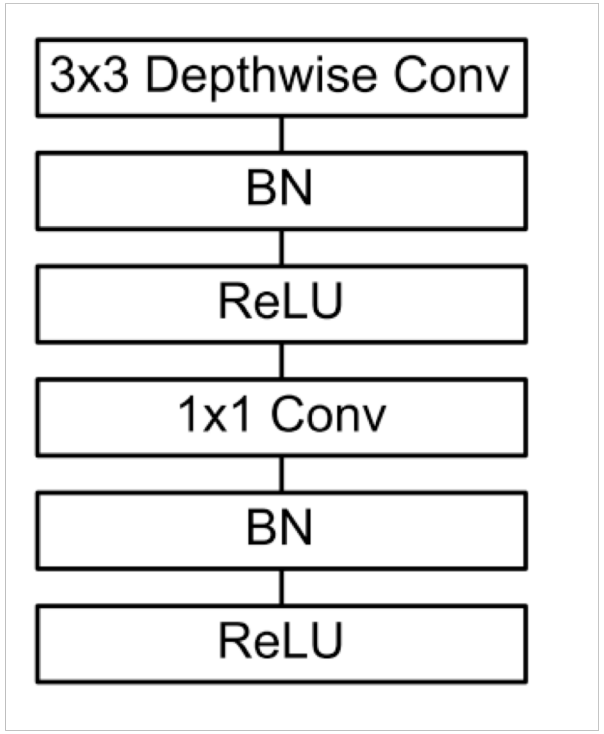
- Applications
  - *Large RF required (e.g. segmentation)*

Chollet, François. "Xception: Deep learning with depthwise separable convolutions."

Li, Zeming, et al. "Light-Head R-CNN: In Defense of Two-Stage Object Detector."

Tensor Decomposition

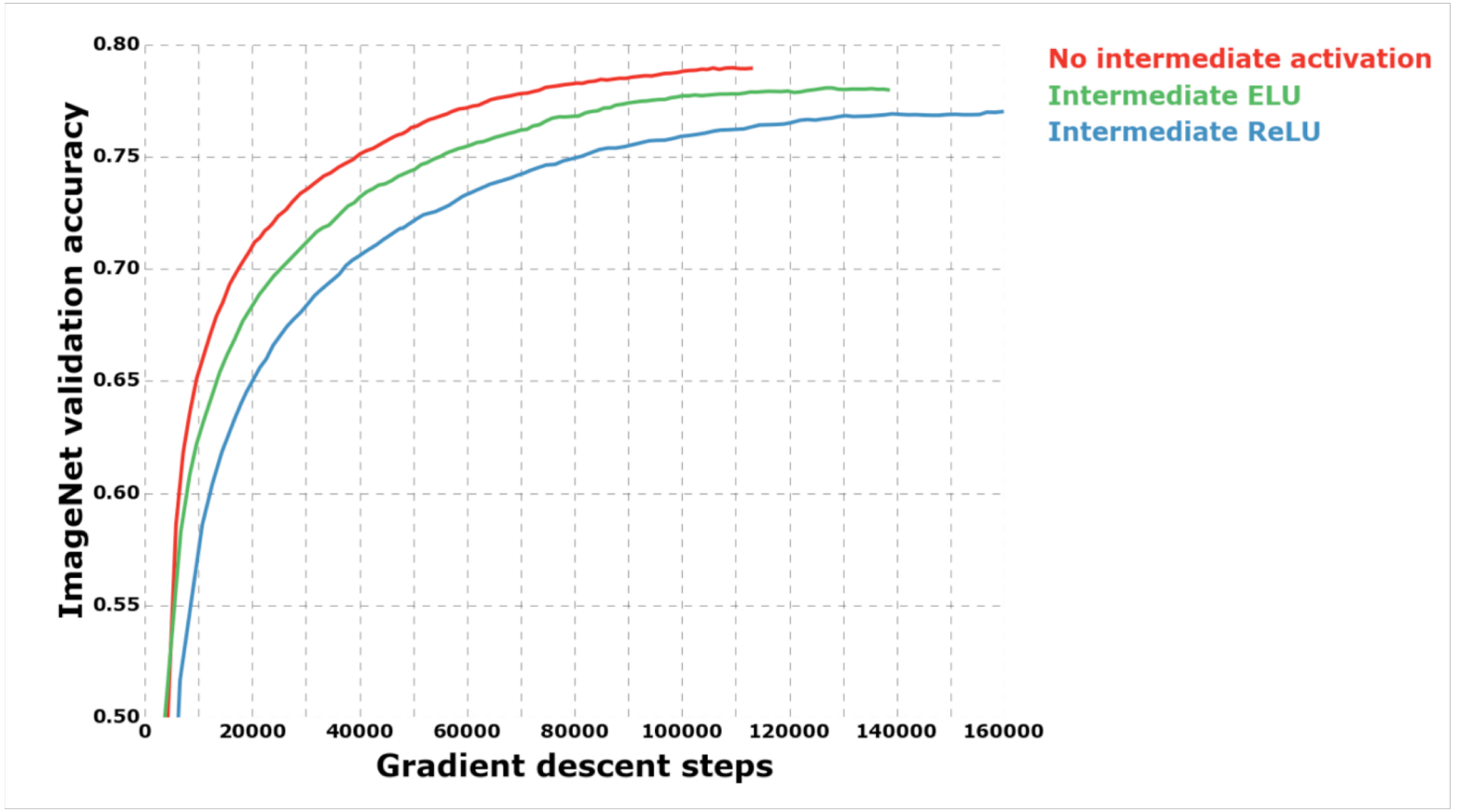
- DWConv + 1x1
  - *MobileNet v1*
- Applications
  - *None, unless none of other components supported*



Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications."

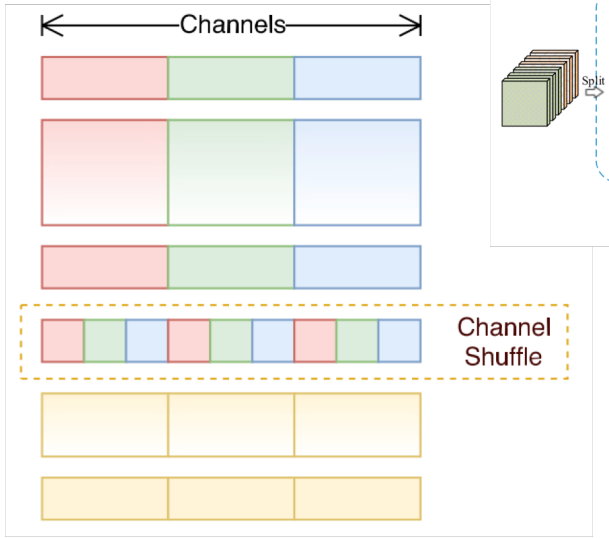
Tensor Decomposition

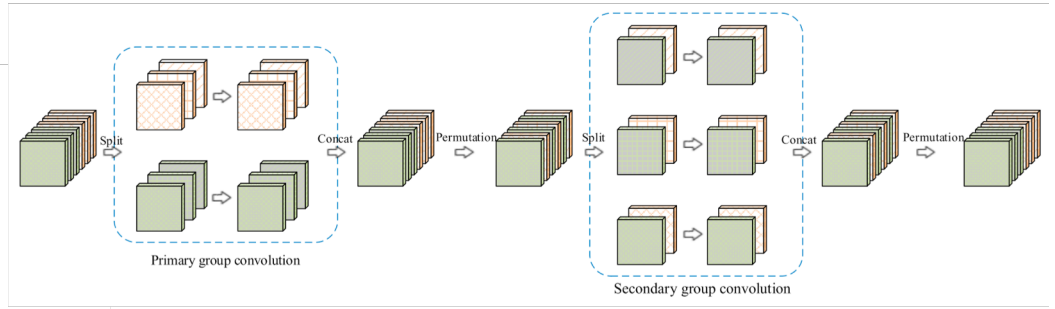
- Some details
  - *With/without BN?*
  - *With/without ReLU?*

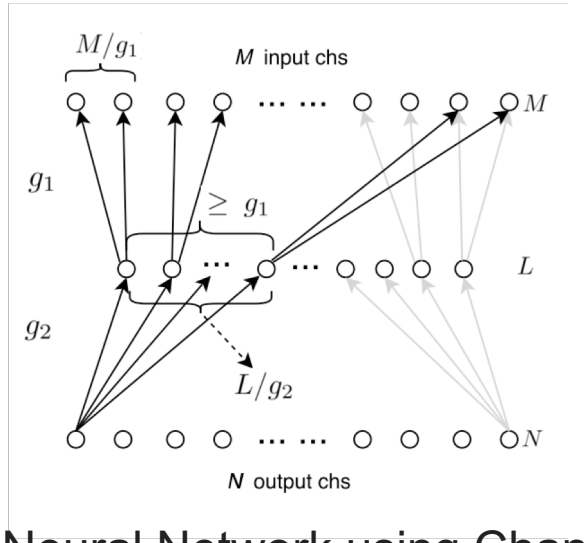


Channel Shuffle

- GConv + Channel Shuffle
  - *ShuffleNet*
  - *IGCNet*
  - *CLCNet*







Zhang, X.*, Zhou, X.*, Lin, M. and Sun, J., 2017. Shufflenet: An extremely efficient convolutional neural network for mobile devices

Zhang, Dong-Qing. "clcNet: Improving the Efficiency of Convolutional Neural Network using Channel Local Convolutions."

Zhang, T., et al. Interleaved Group Convolutions for Deep Neural Networks

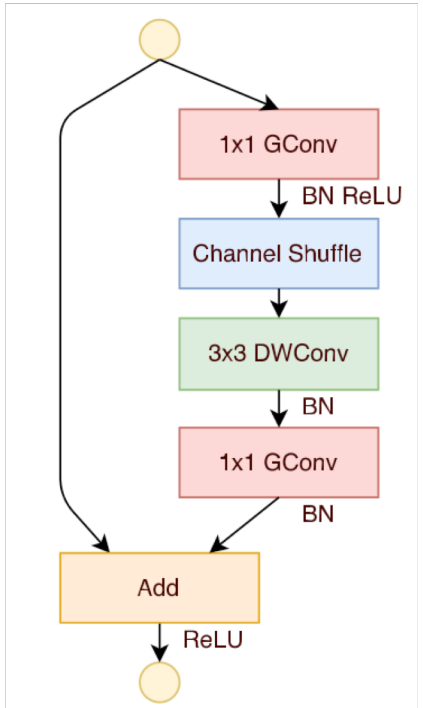
Channel Shuffle

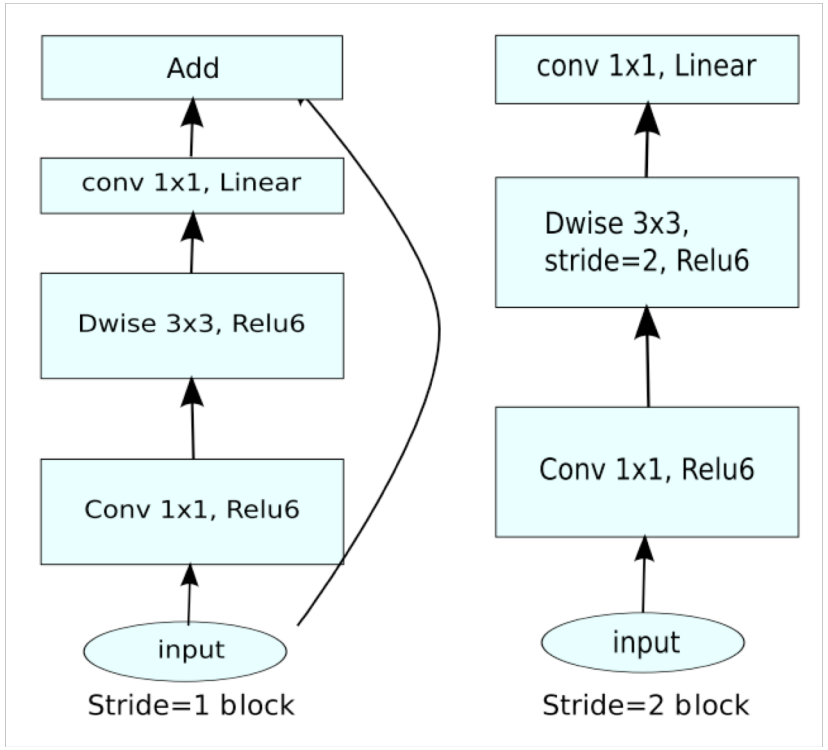
- Network configuration
  - *"fast downsampling"*

Layer	Output size	KSize	Stride	Repeat	Output channels (g groups)				
					$g = 1$	$g = 2$	$g = 3$	$g = 4$	$g = 8$
Image	224×224				3	3	3	3	3
Conv1	112×112	3×3	2	1	24	24	24	24	24
MaxPool	56×56	3×3	2						
Stage2[1]	28×28		2	1	144	200	240	272	384
	28×28		1	3	144	200	240	272	384
Stage3	14×14		2	1	288	400	480	544	768
	14×14		1	7	288	400	480	544	768
Stage4	7×7		2	1	576	800	960	1088	1536
	7×7		1	3	576	800	960	1088	1536
GlobalPool	1×1	7×7							
FC					1000	1000	1000	1000	1000
Complexity[2]					143M	140M	137M	133M	137M

Bottleneck vs. Inverted Bottleneck

- MobileNet v2
  - *Linear bottleneck*
  - *Inverted residual*
  - *ReLU6*





Sandler, Mark, et al. "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation."

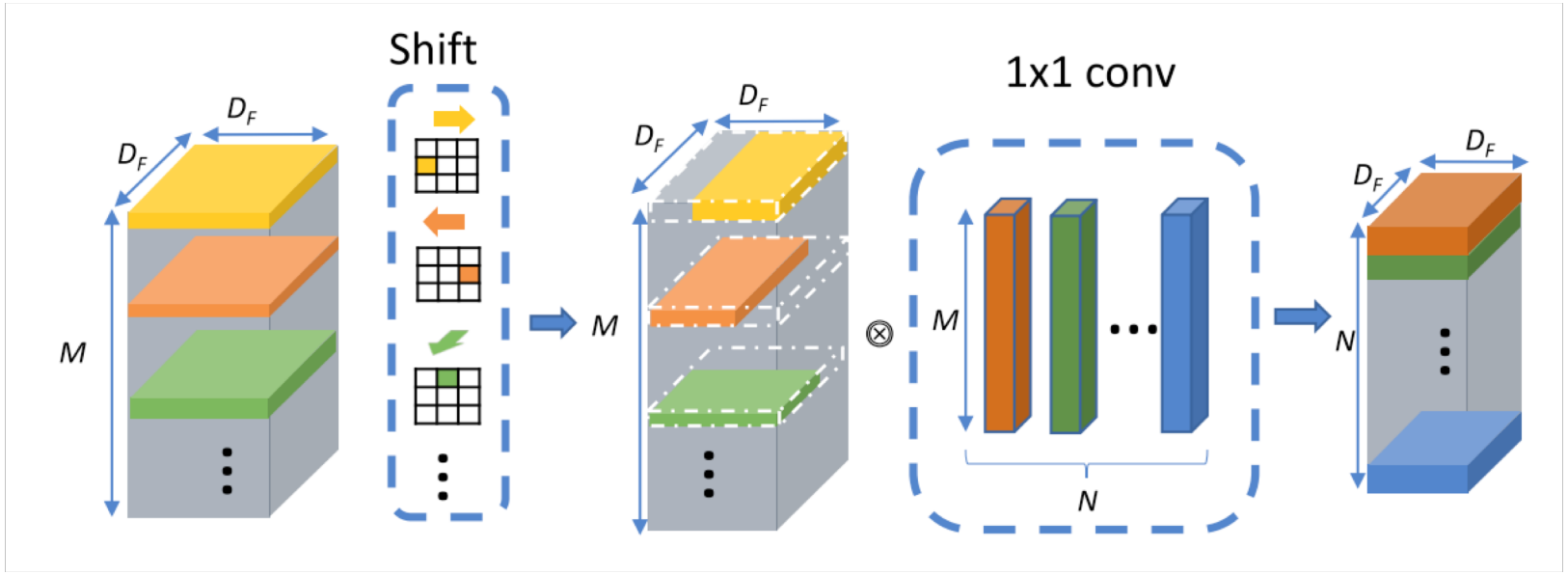
Bottleneck vs. Inverted Bottleneck

- Trick in MobileNet v2

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$28^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times k$	conv2d 1x1	-	k	-	

Spatial Operators

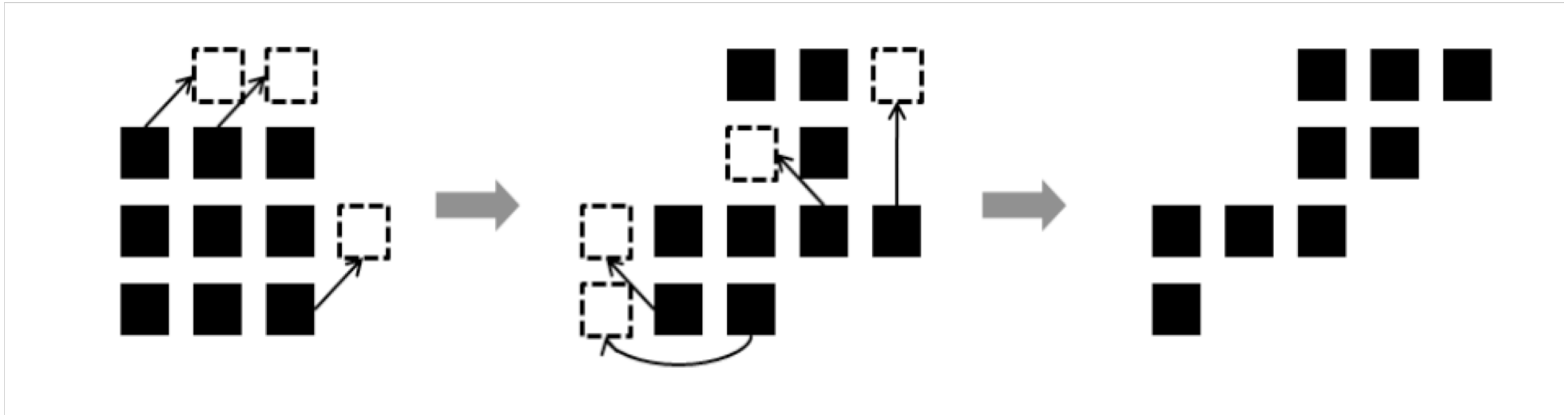
- “Shift” operator



Wu, Bichen, et al. "Shift: A Zero FLOP, Zero Parameter Alternative to Spatial Convolutions."

Spatial Operators

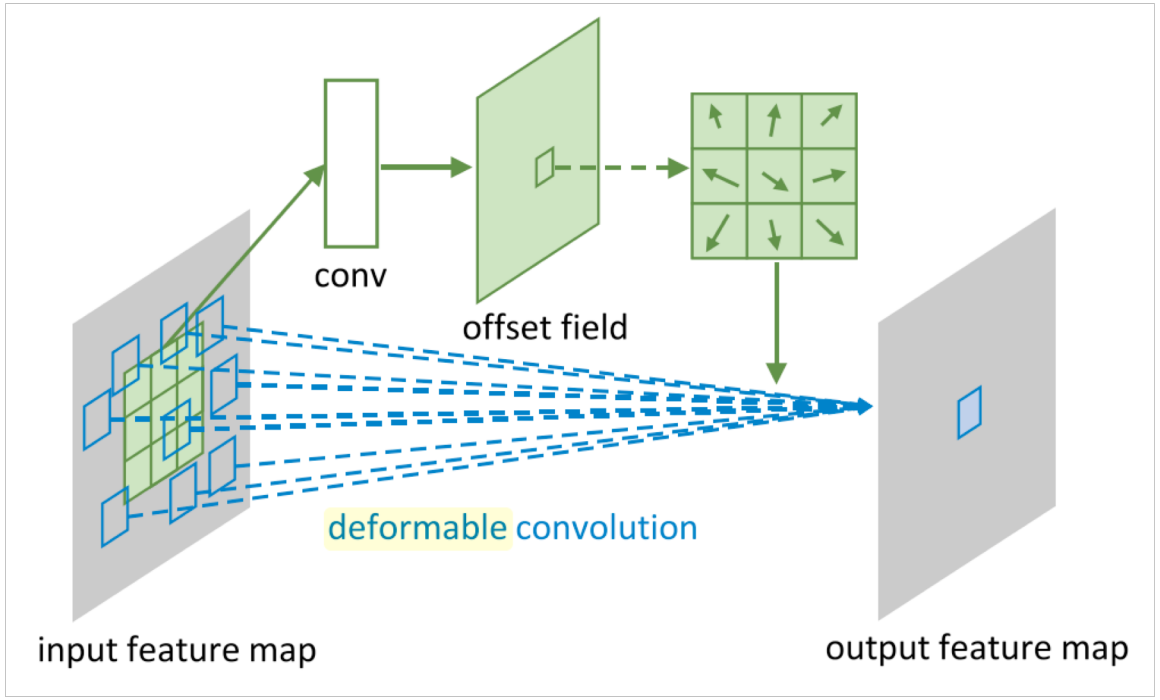
- Irregular convolution



Ma, Jiabin, Wei Wang, and Liang Wang. "Irregular Convolutional Neural Networks."

Spatial Operators

- Deformable Convolution/Pooling



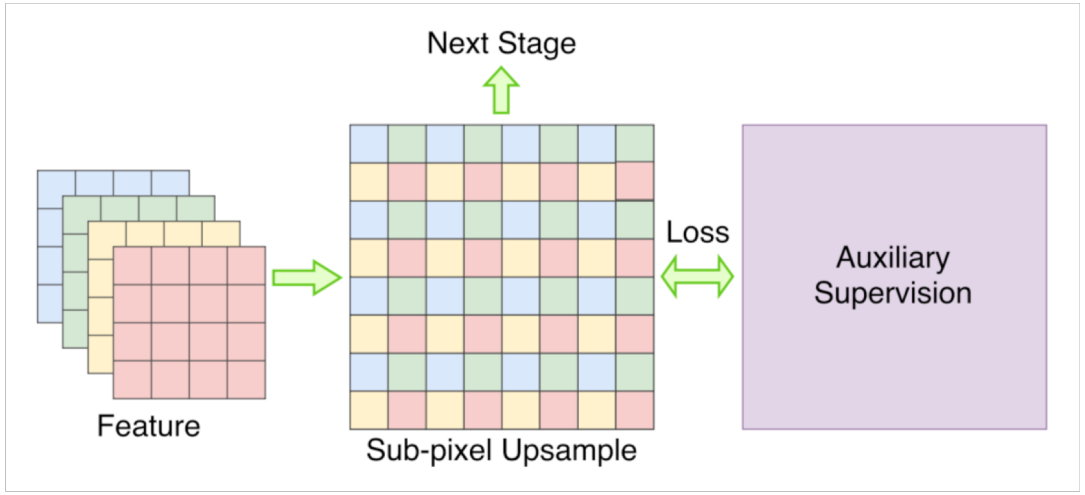
Dai, Jifeng, et al. "Deformable convolutional networks."

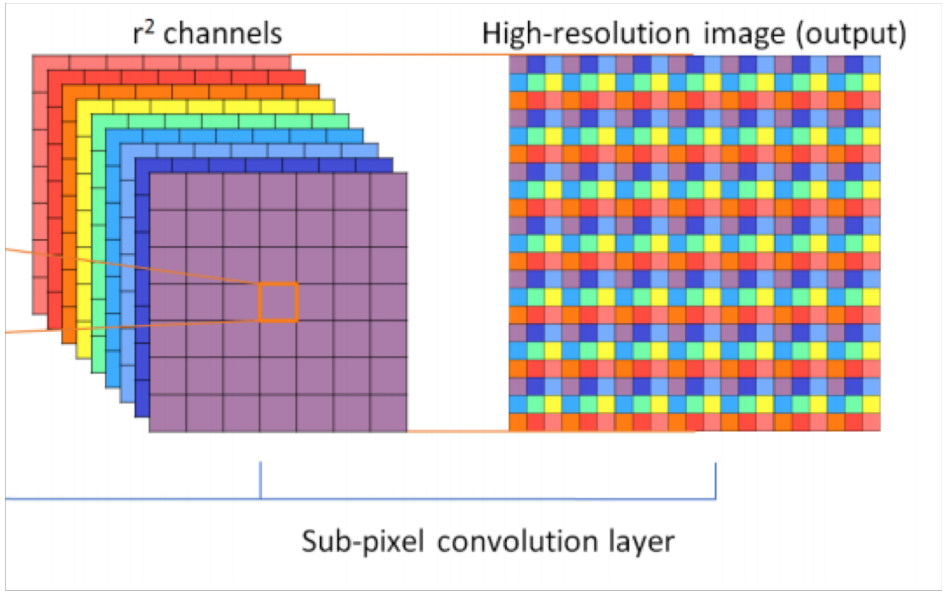
Spatial Operators

- Upsampling
  - *Deconvolution*
  - *Resize*
  - *“Reshape”*
  - *Unpooling*
  - *…*

Spatial Operators

- Upsampling
  - *Deconvolution*
  - *Resize*
  - *"Reshape"*
  - *Unpooling*
  - *…*





Shi, W., et al. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. (2016)

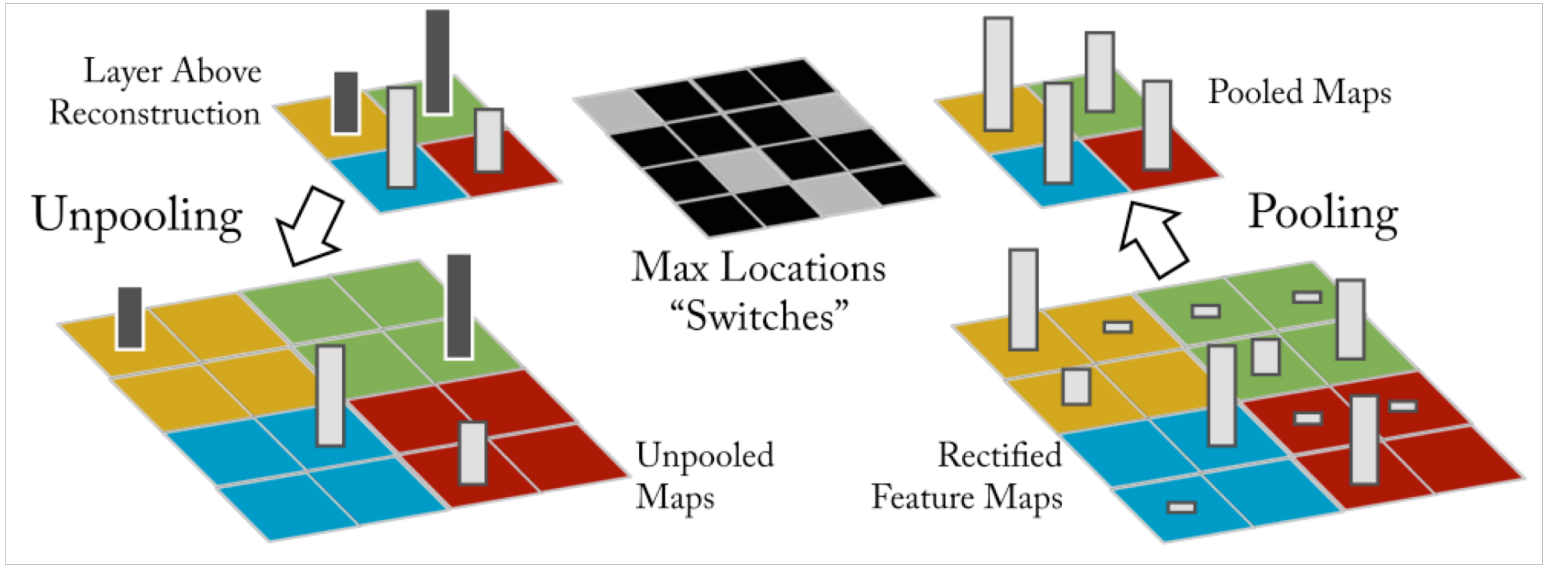
Aitken, A., et al.: Checkerboard 684 artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize. (2017)

Zhenli Zhang, et al. ExFuse: Enhancing Feature Fusion for Semantic Segmentation.

Spatial Operators

- Upsampling
  - *Deconvolution*
  - *Resize*
  - *“Reshape”*
  - *Unpooling*

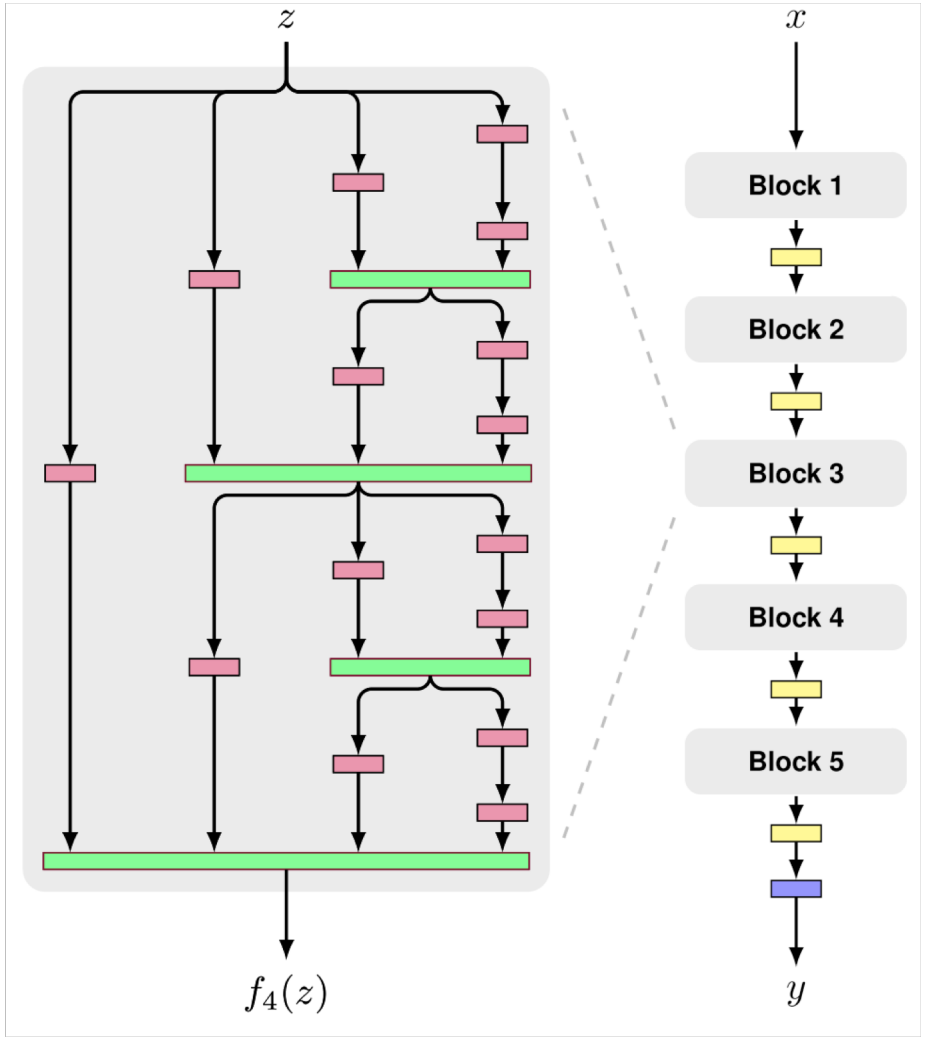
  …



Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." *European conference on computer vision*. Springer, Cham, 2014.

Densely Connections

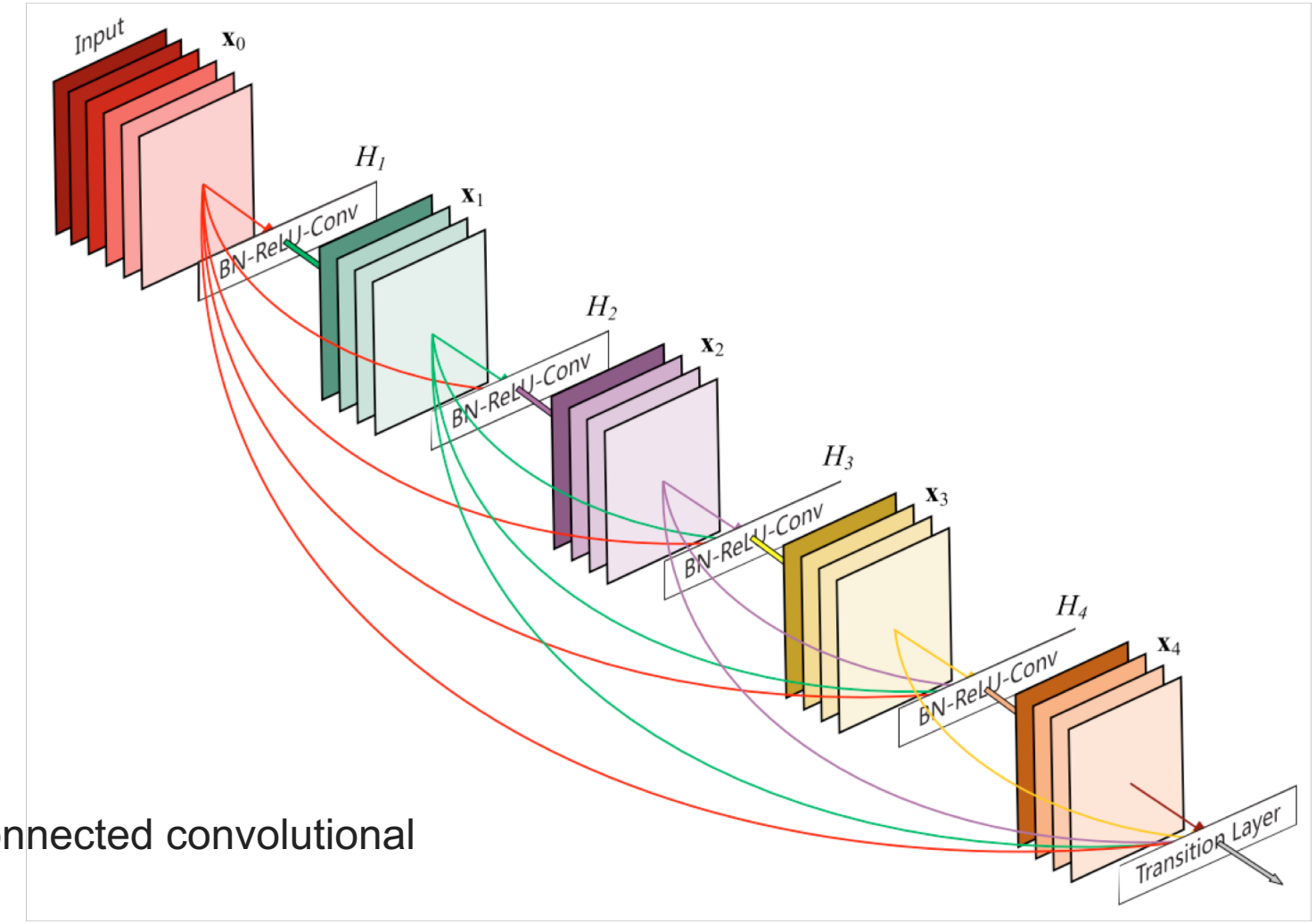
- FractalNet



Larsson, Gustav, Michael Maire, and Gregory Shakhnarovich. "Fractalnet: Ultra-deep neural networks without residuals."

Densely Connections

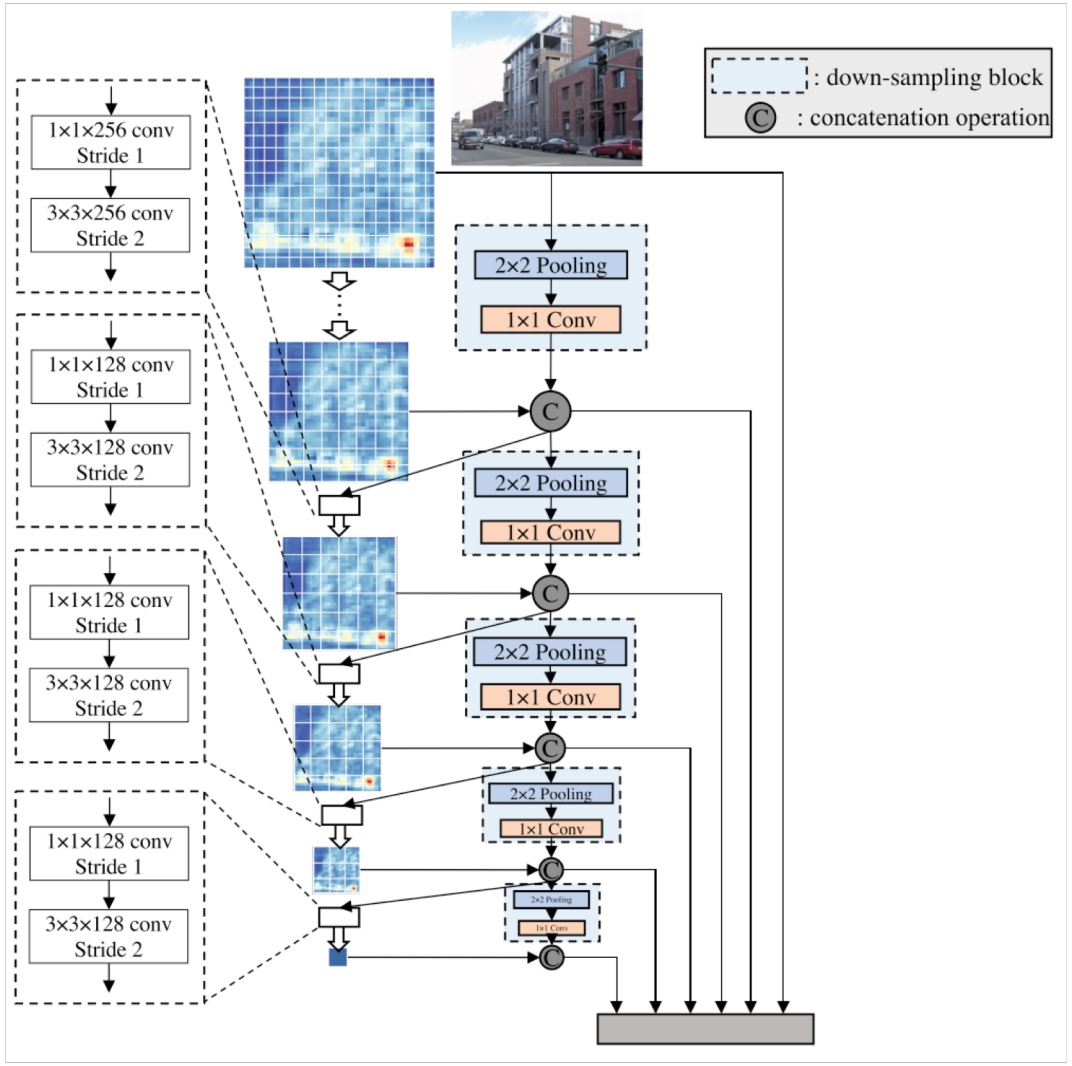
- DenseNet



Huang, Gao, et al. "Densely connected convolutional networks." *CVPR 2017.*

Densely Connections

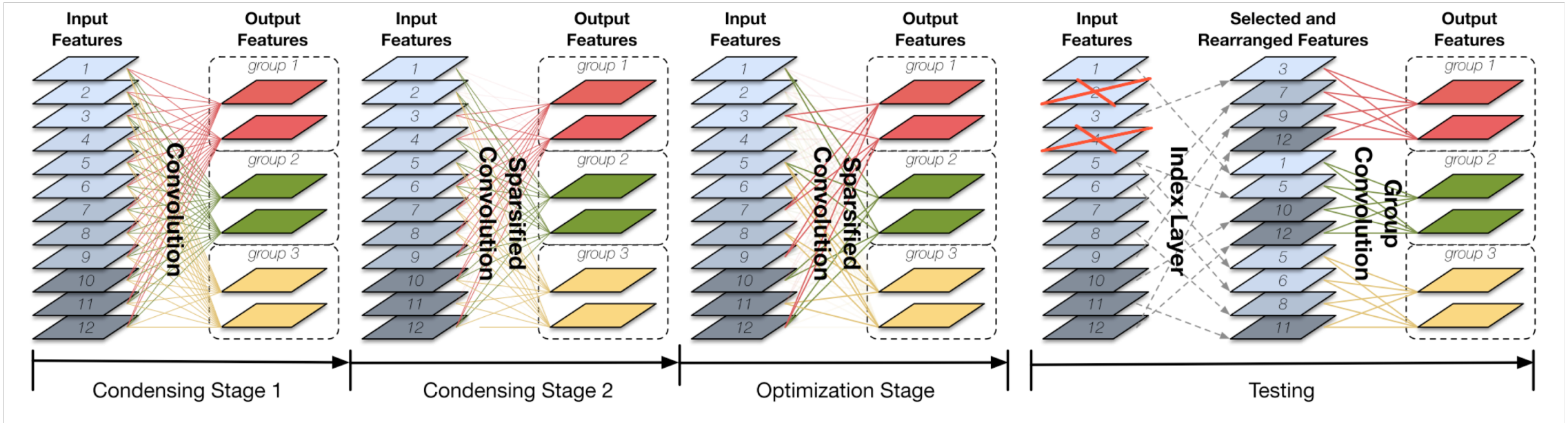
- DenseNet Applications
  - *DSOD*



Shen, Zhiqiang, et al. "Dsod: Learning deeply supervised object detectors from scratch." ICCV 2017

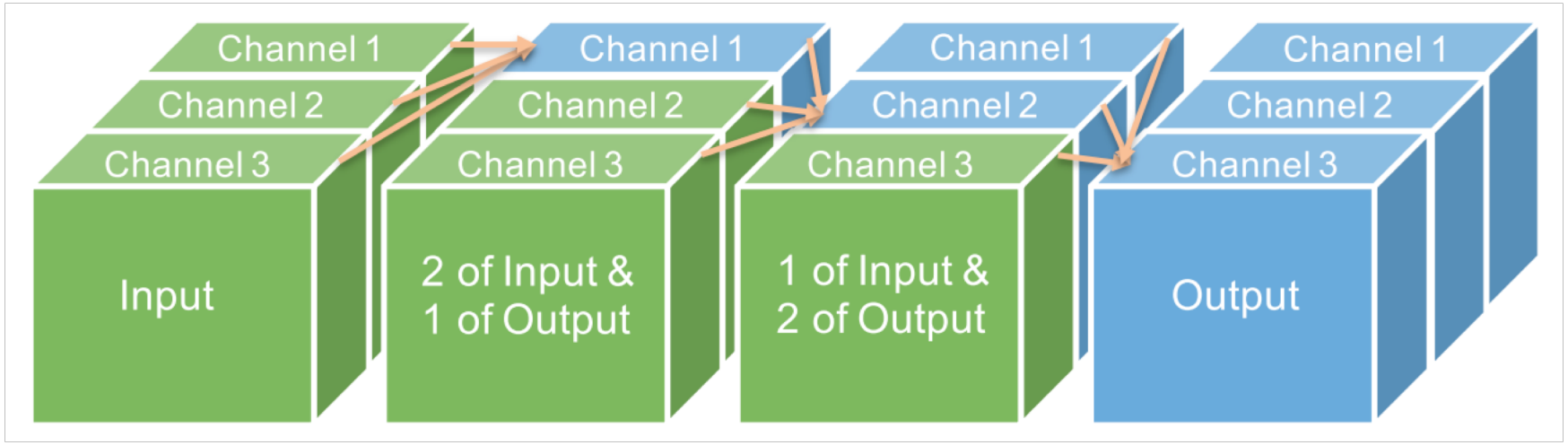
Densely Connections

- DenseNet Optimizations
  - *CondenseNet*



Huang, Gao, et al. "CondenseNet: An Efficient DenseNet using Learned Group Convolutions."

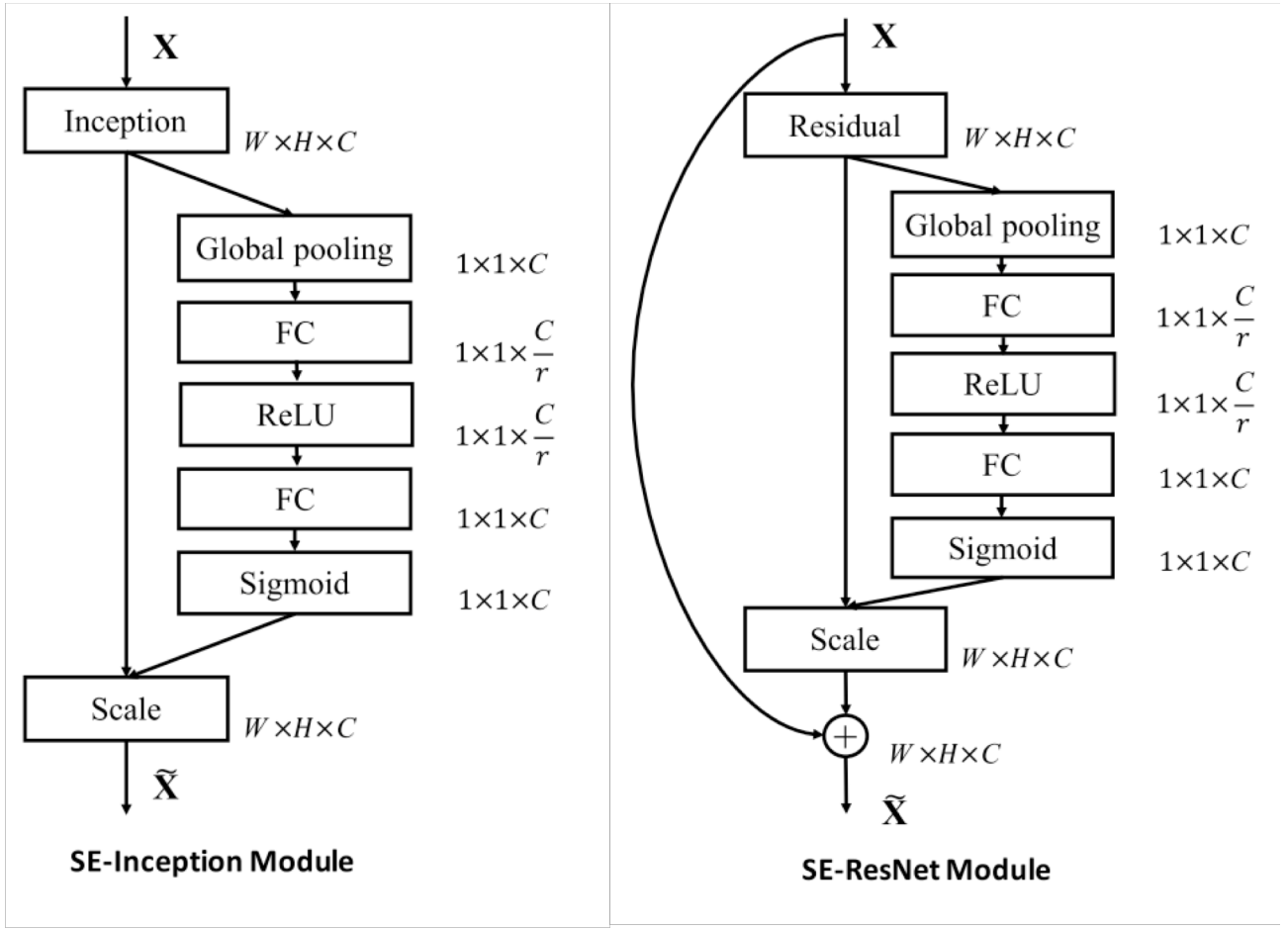
Feature Reuse



Qiao, Siyuan, et al. "Gradually Updated Neural Networks for Large-Scale Image Recognition."

Attention Blocks

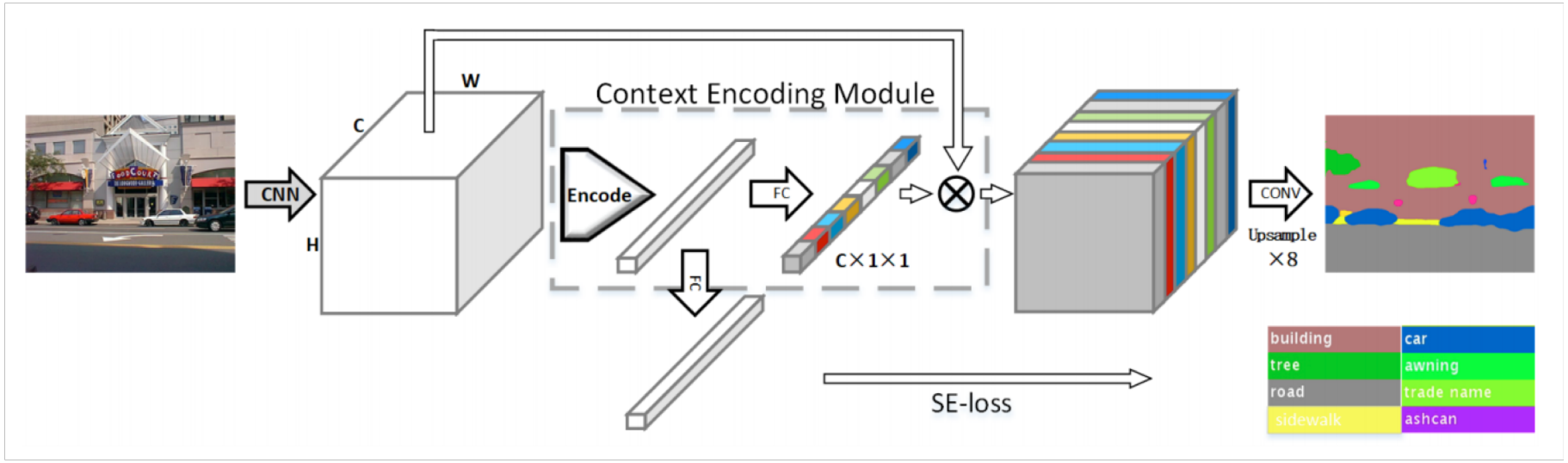
- Channel Attention
  - *SENet*
  - *With/without BN?*
- Applications
  - *Recommended*



Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks."

Attention Blocks

- Applications
  - *EncNet*



Hang Zhang, et al. Context Encoding for Semantic Segmentation

Normalization

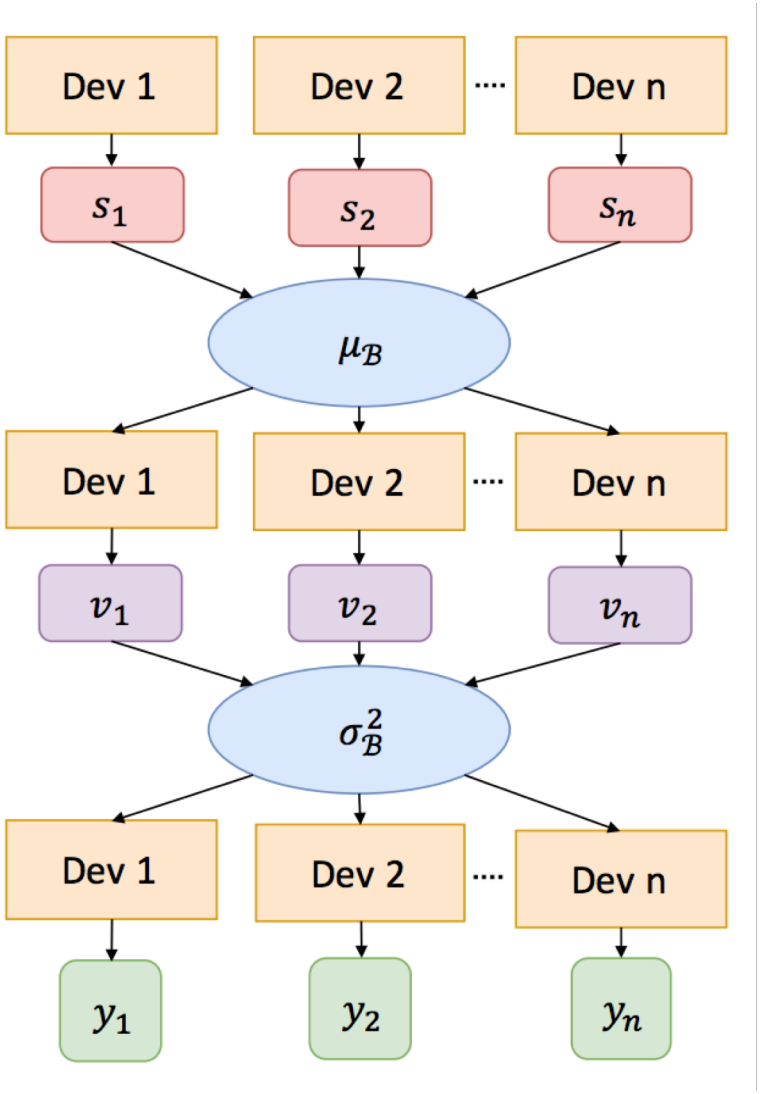
- Batch Normalization/Batch Renormalization
- Multi-GPU Batch Normalization
- Layer Normalization
- Weight Normalization
- Gradient Normalization
- Group Normalization

...

Normalization

- Batch Normalization/Batch Renormalization
- Multi-GPU Batch Normalization
- Layer Normalization
- Weight Normalization
- Gradient Normalization
- Group Normalization

...

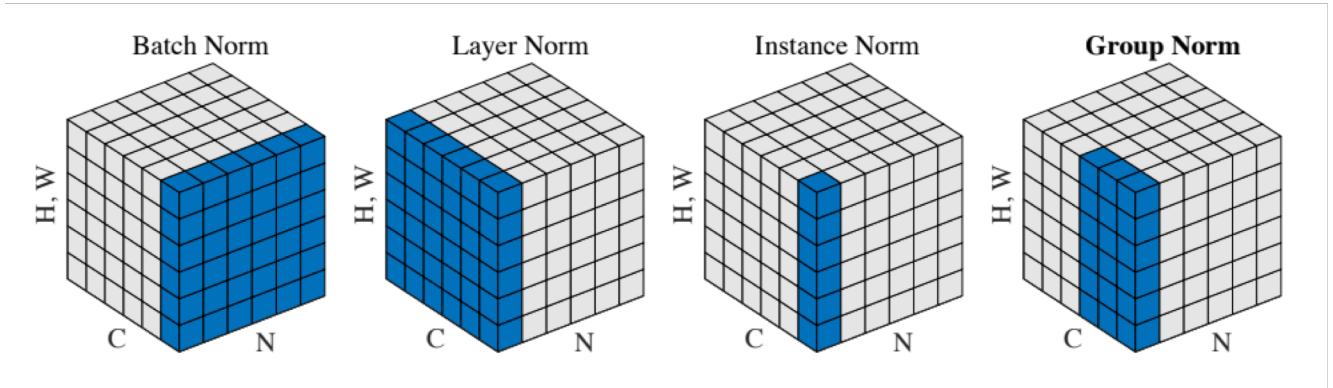


Peng, Chao, et al. "MegDet: A Large Mini-Batch Object Detector."

Normalization

- Batch Normalization/Batch Renormalization
- Multi-GPU Batch Normalization
- Layer Normalization
- Weight Normalization
- Gradient Normalization
- Group Normalization

...



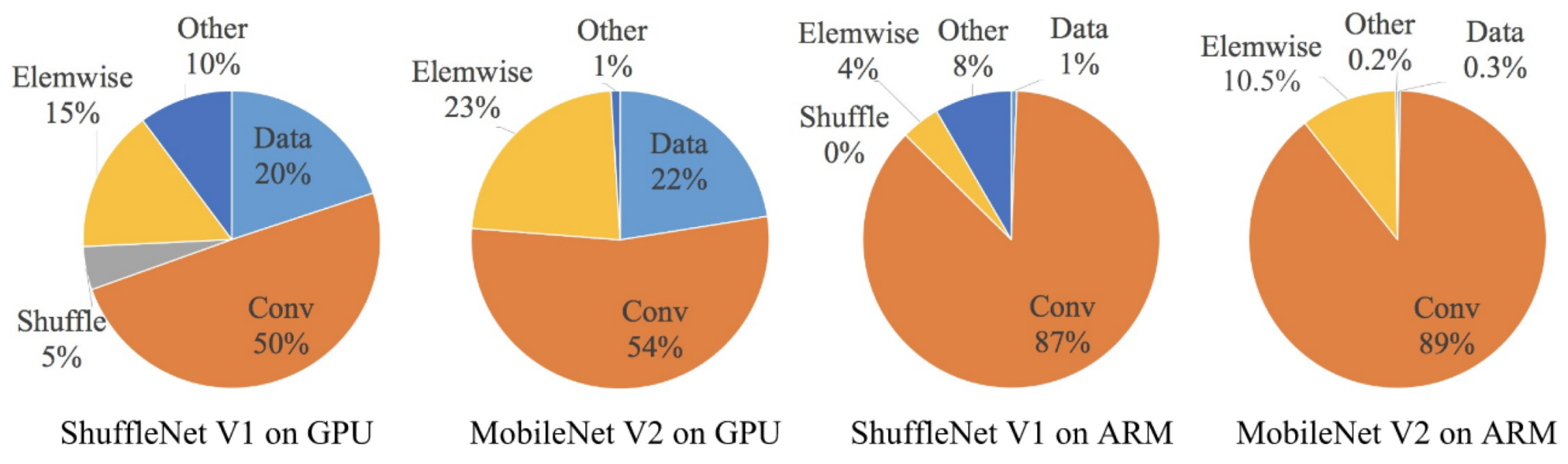
Yuxin Wu, et al. Group normalization

Normalization

- Best practice
  - *Large batch: BN*
  - *Small batch: Multi-gpu BN > GN/BRN > BN/LN*
  - *Weight/meta network: LN/WN*
  - *GAN/Multi-task: Gradient normalization*

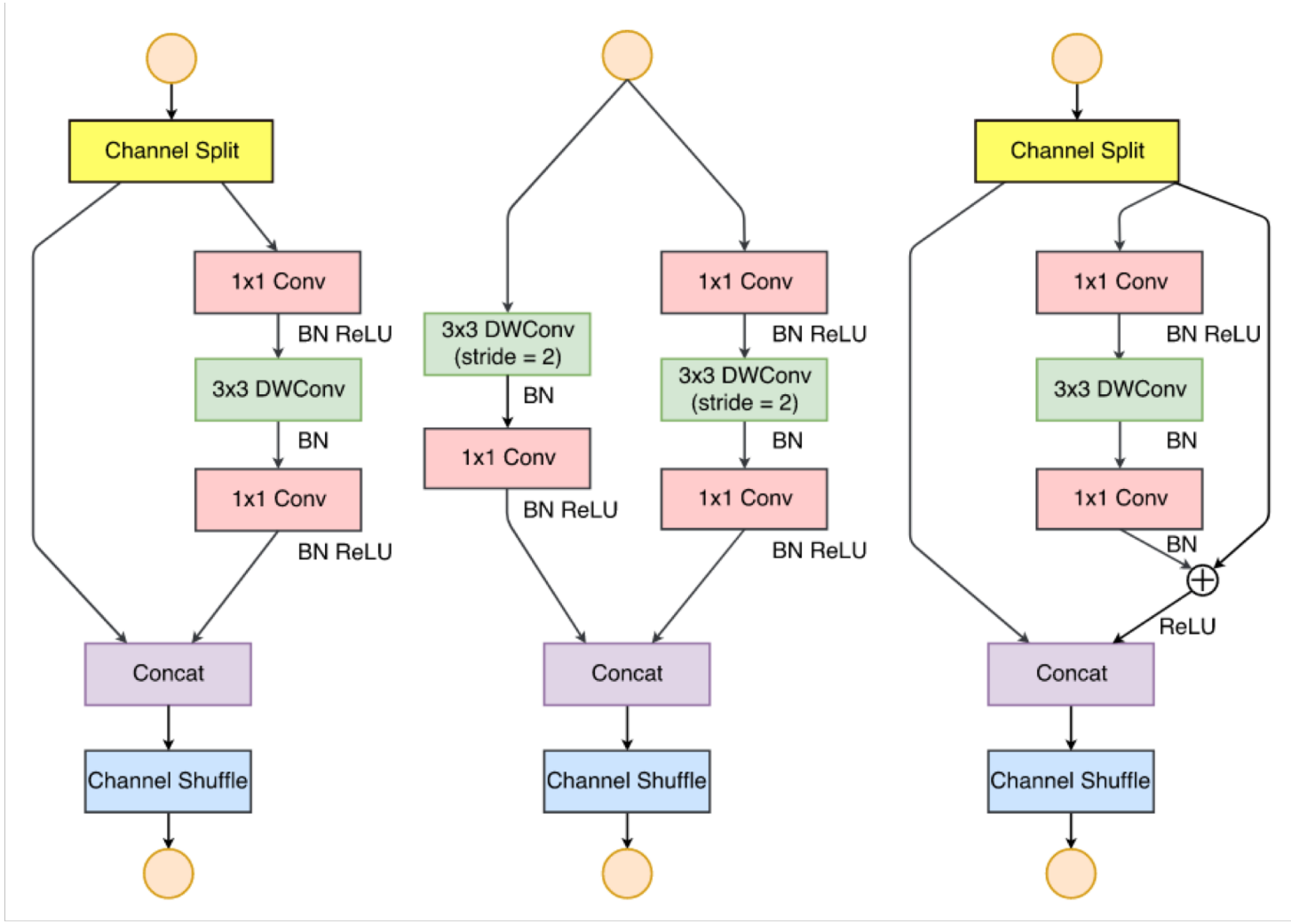
Analysis: Inference Efficiency

- Actual running time
  - *Convolutions and elementwise operators*



ShuffleNet v2

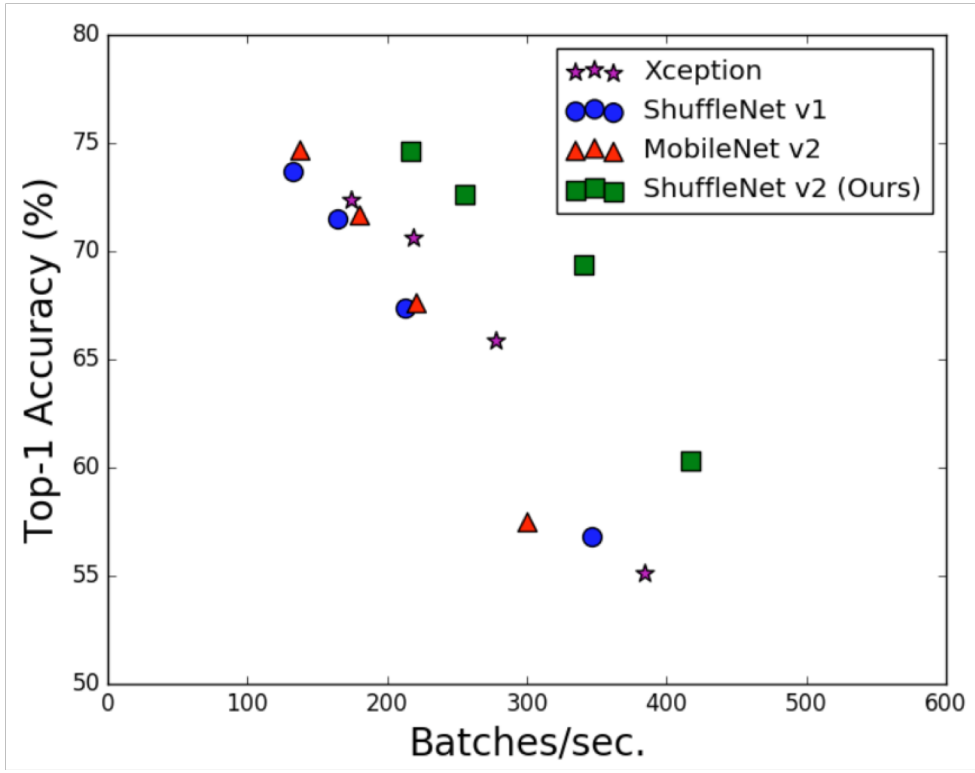
- Inference-efficient structrue

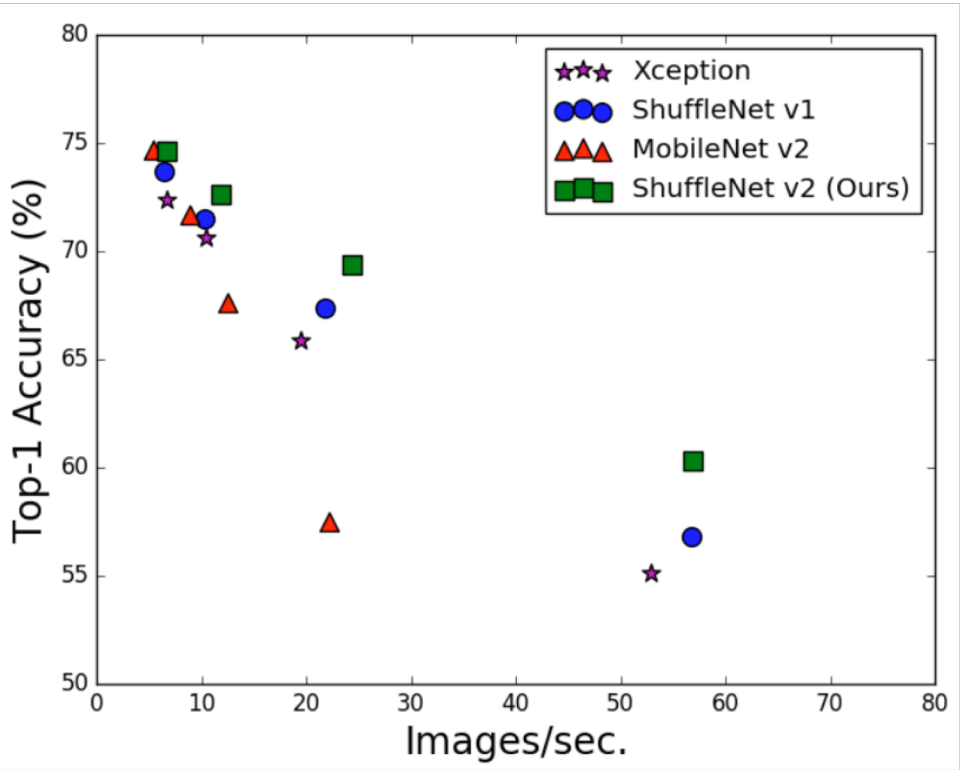


Ninging Ma, Xiangyu Zhang, Jian Sun. Light-weight CNN Architecture Design for Fast Inference

ShuffleNet v2

- Better accuracy/actual speed trade-offs





ShuffleNet v2

- Known issues
  - *Limited RF*
- Applications
  - *Classification-oriented tasks*
  - *GPU efficiency*
  - *Competition (not sure)*

Model	mmAP (%)		GPU Speed (Batches/sec.)	
FLOPs	300M	500M	300M	500M
Xception	31.3	32.9	25.3	20.8
ShuffleNet v1	29.9	32.9	19.0	15.0
MobileNet v2	30.0	30.6	23.5	18.0
ShuffleNet v2 (ours)	31.8	33.3	**27.3**	**21.8**
ShuffleNet v2* (ours)	**33.0**	**34.8**	21.3	16.5

Conclusion: Model Design

- Task specific
  - *Cls or Loc*
  - *Representation or Receptive Field*
- Shortcut matters!
- Feature reuse
- Attention if needed
- Inference efficiency

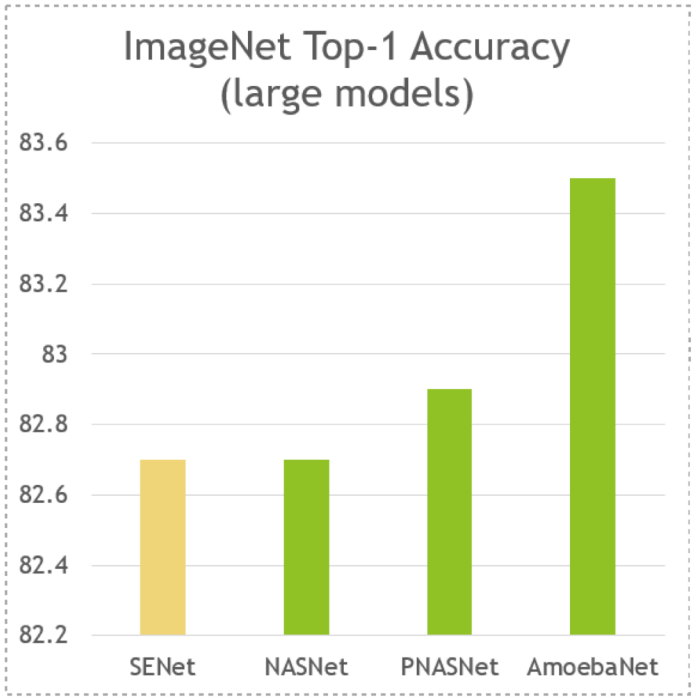
Outline

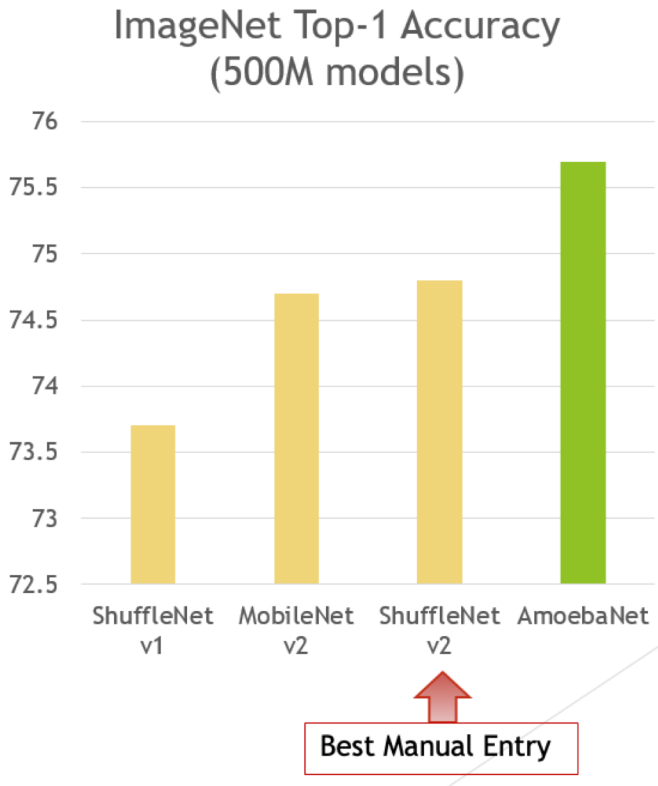
- Architecture related
- Other topics
  - *Model search*
  - *Large batch training*

What is Model Search

- A powerful tool for DL model research, towards:
  - *Better*
  - *Faster*
  - *Cheaper*
- Automatically derive new architectures or settings
- Usually supported by clusters of computing devices

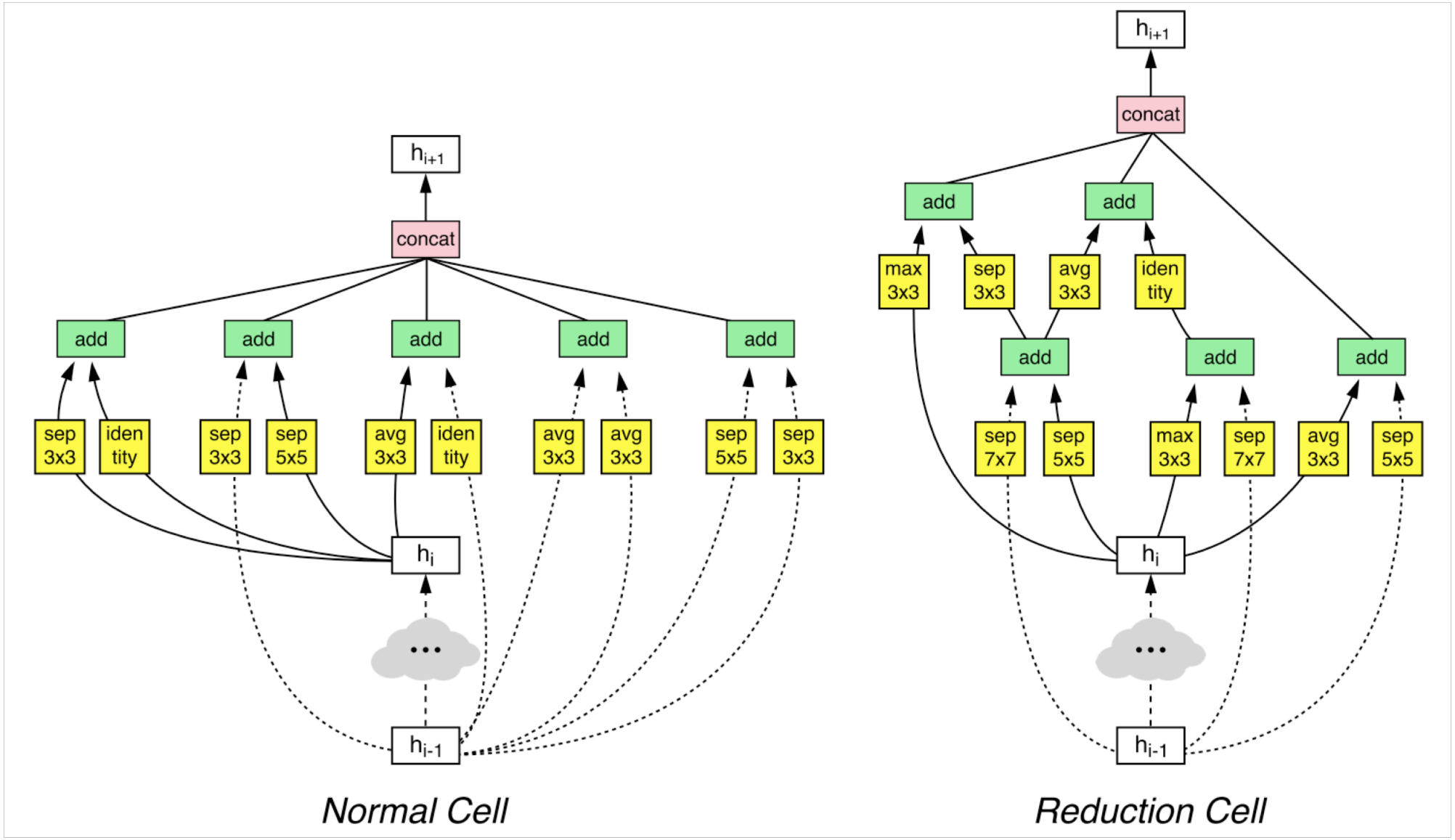
Power of Model Search





Research Community

- Google (Quoc V. Le, et al)
  - NASNet (2017)
  - PNASNet (2017)
  - SWISH (2017)
  - AmoebaNet (2018)
  - NetAdapt (2018)
  - Auto-augmentation (2018)
  - …
- Sensetime
  - PolyNet (2017)
  - BlockQNN (2017)


h_{i+1}
concat
add
add
add
add
add
sep 3x3
iden tity
sep 3x3
sep 5x5
avg 3x3
iden tity
avg 3x3
avg 3x3
sep 5x5
sep 3x3
h_i
h_{i-1}
Normal Cell
h_{i+1}
concat
add
add
max 3x3
sep 3x3
avg 3x3
iden tity
add
add
add
sep 7x7
sep 5x5
max 3x3
sep 7x7
avg 3x3
sep 5x5
h_i
h_{i-1}
Reduction Cell

Neural Architecture Search with Reinforcement Learning

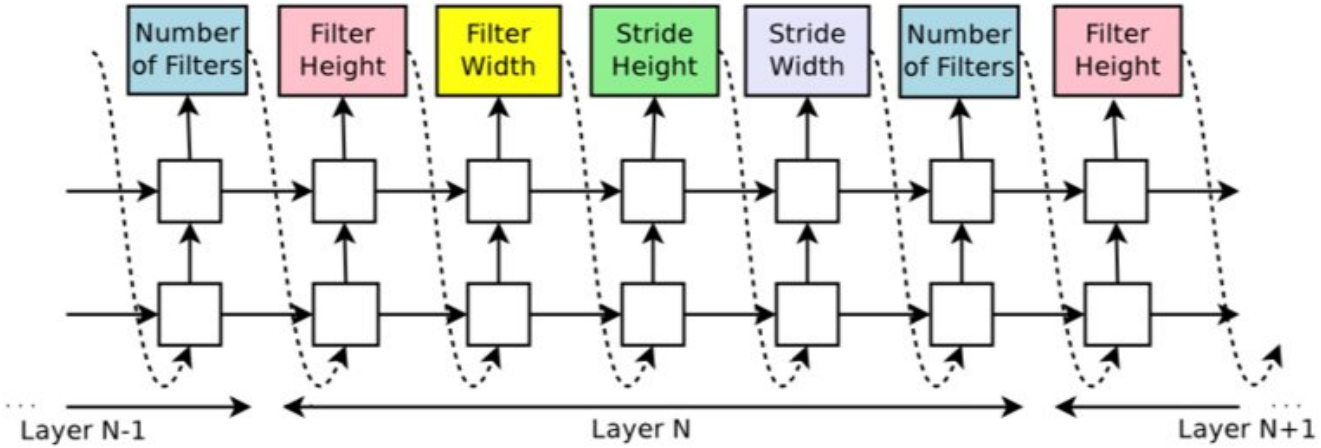


Figure 2: How our controller recurrent neural network samples a simple convolutional network. It predicts filter height, filter width, stride height, stride width, and number of filters for one layer and repeats. Every prediction is carried out by a softmax classifier and then fed into the next time step as input.

Neural Architecture Search with Reinforcement Learning

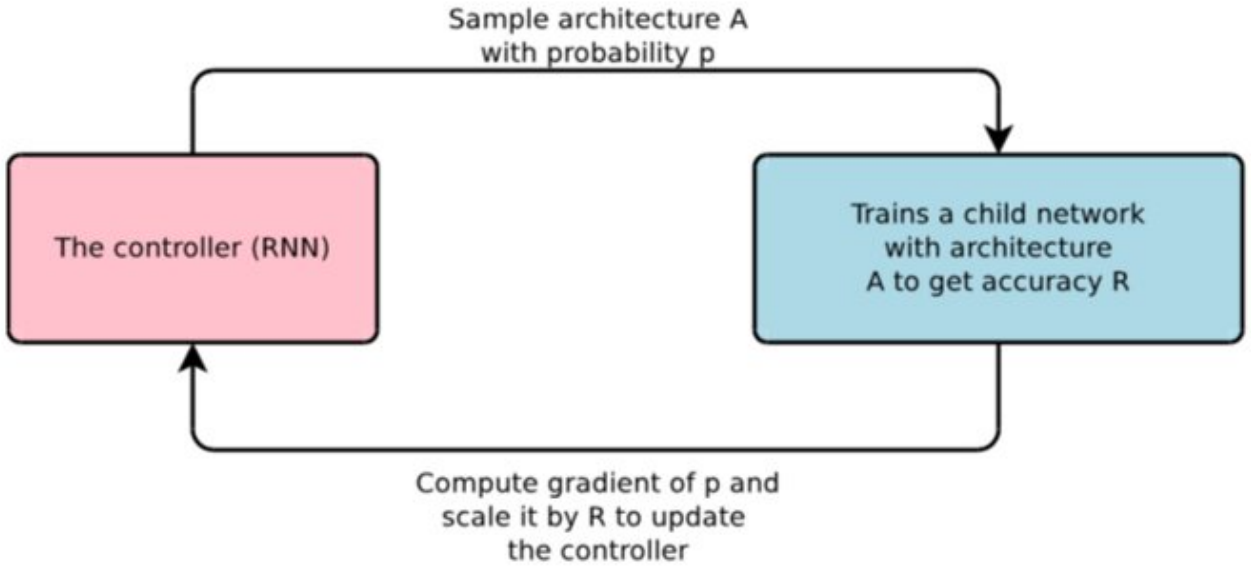


Figure 1: An overview of Neural Architecture Search.

NAS

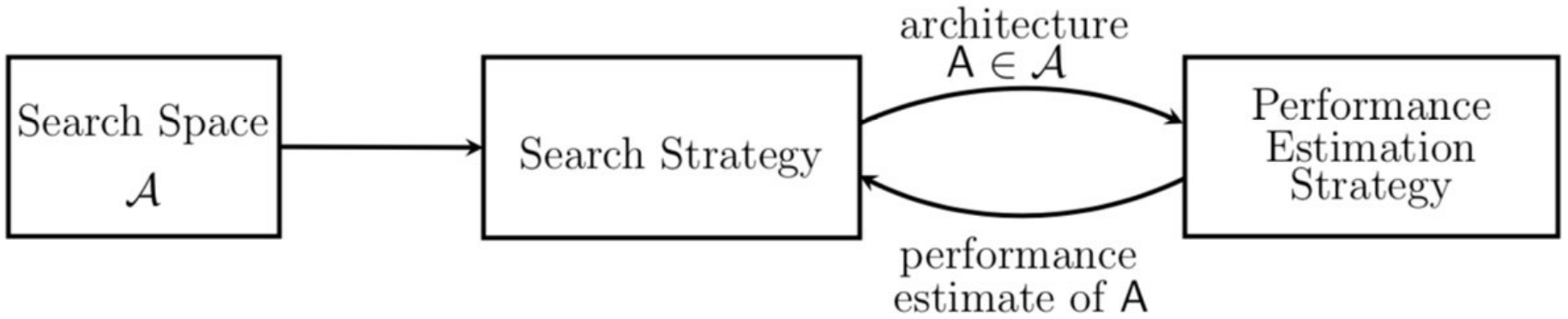
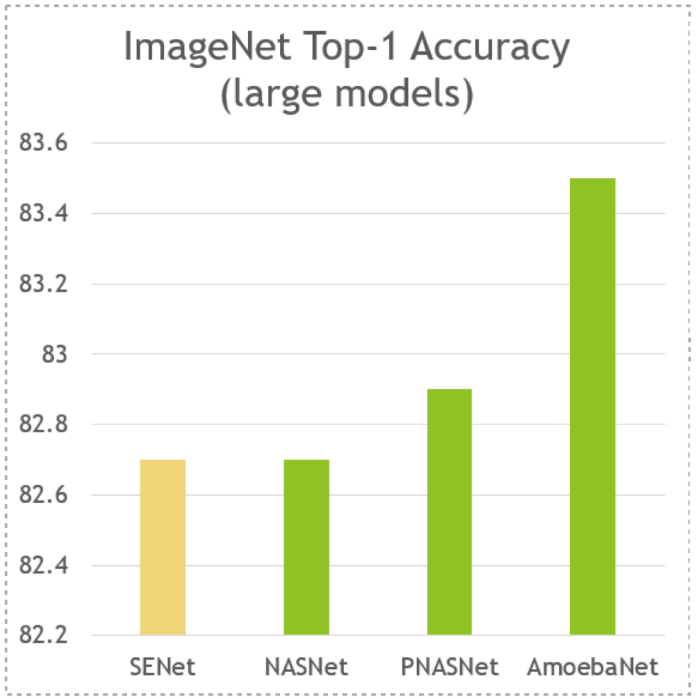


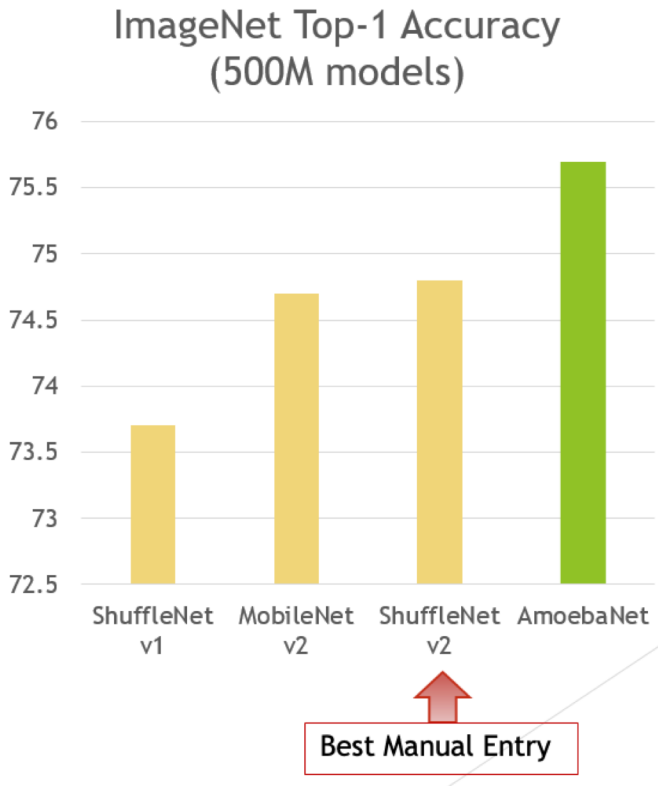
Figure 1: Abstract illustration of Neural Architecture Search methods. A search strategy selects an architecture A from a predefined search space $\mathcal{A}$. The architecture is passed to a performance estimation strategy, which returns the estimated performance of A to the search strategy.

Model Search

- Automatic model structure generation
- Agents
  - *Greedy*
  - *RL*
  - *Evolutionary/Genetic*
  - *Model Pruning*
- Usually very costly

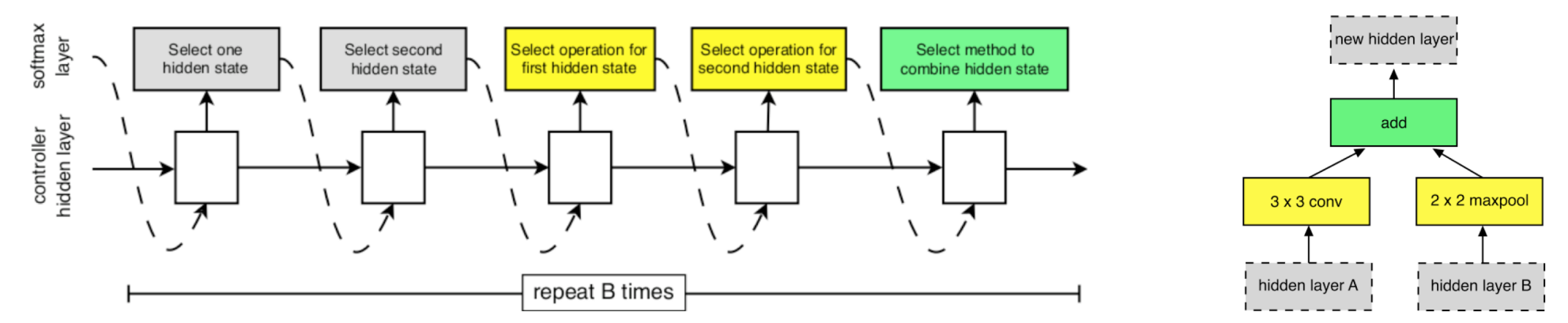
Power of Model Search





Model Search

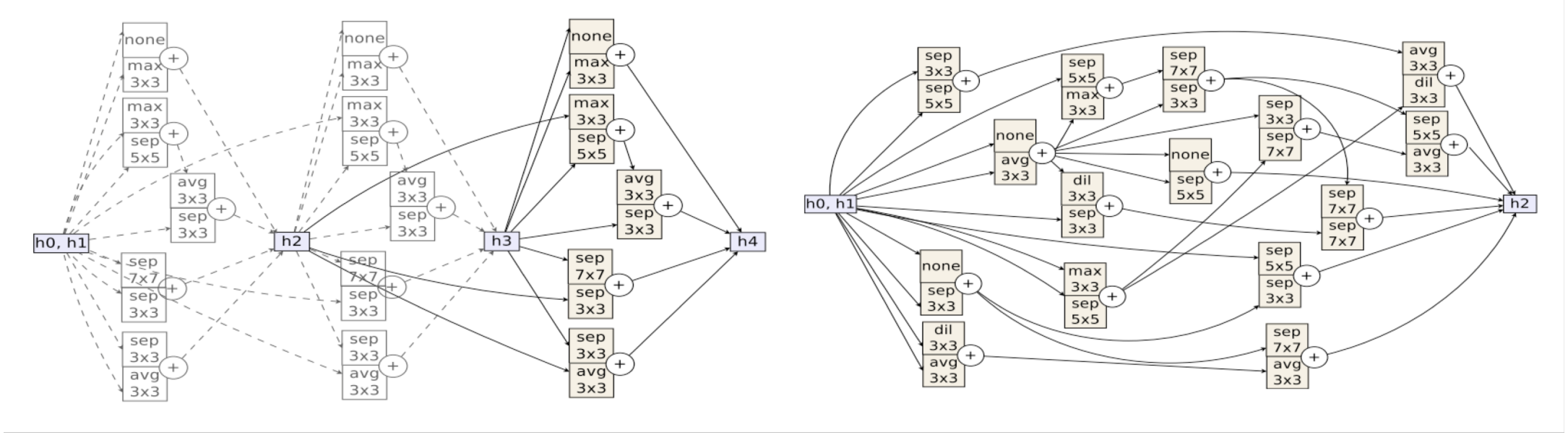
- RL
  - *Search on Cifar-10, test on ImageNet (transferable)*
  - *Generator: RNN*
  - *Reward: score*



Zoph, Barret, et al. "Learning transferable architectures for scalable image recognition."

Model Search

- Evolution
  - *AmoebaNet*



Real, Esteban, et al. "Regularized Evolution for Image Classifier Architecture Search."

DARTS: Differentiable Architecture Search

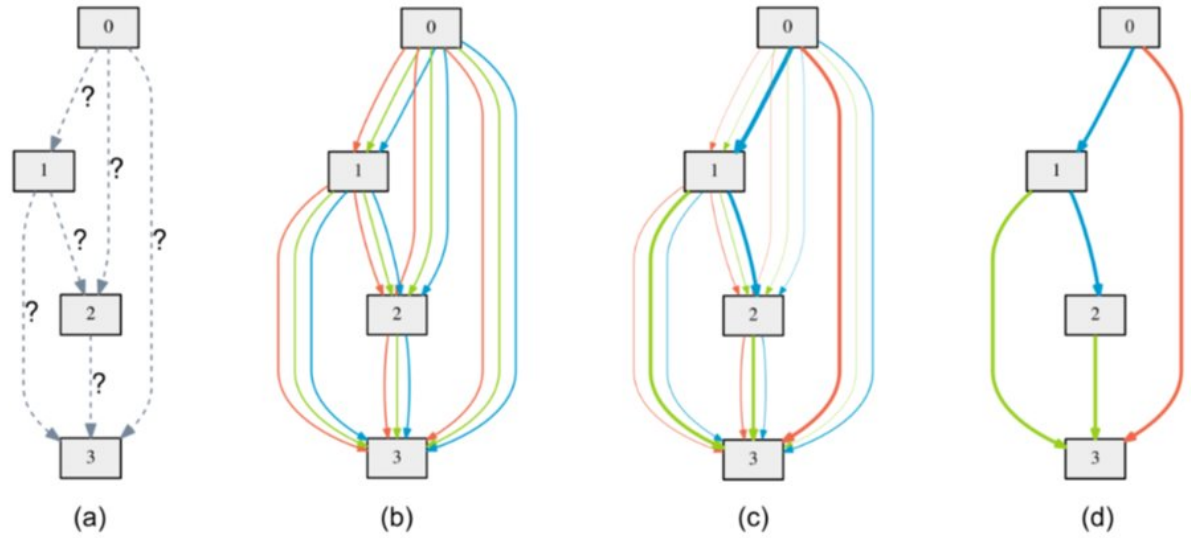


Figure 1: An overview of DARTS: (a) Operations on the edges are initially unknown. (b) Continuous relaxation of the search space by placing a mixture of candidate operations on each edge. (c) Joint optimization of the mixing probabilities and the network weights by solving a bilevel optimization problem. (d) Inducing the final architecture from the learned mixing probabilities.

Large Batch Training

- Distributed training
  - *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*
  - *Extremely Large Minibatch SGD: Training ResNet-50 on ImageNet in 15 Minutes*
  - *ImageNet Training in Minutes*

Large Batch Training

- Equivalence rules
  - *Linear scaling rule (lr and wc)*

> ***Linear Scaling Rule:*** *When the minibatch size is multiplied by k, multiply the learning rate by k.*

  - *Batch normalization vs. batch size*
  - *Warm up*
  - *Momentum correction*

Goyal, Priya, et al. "Accurate, large minibatch SGD: training imagenet in 1 hour."

Large Batch Training

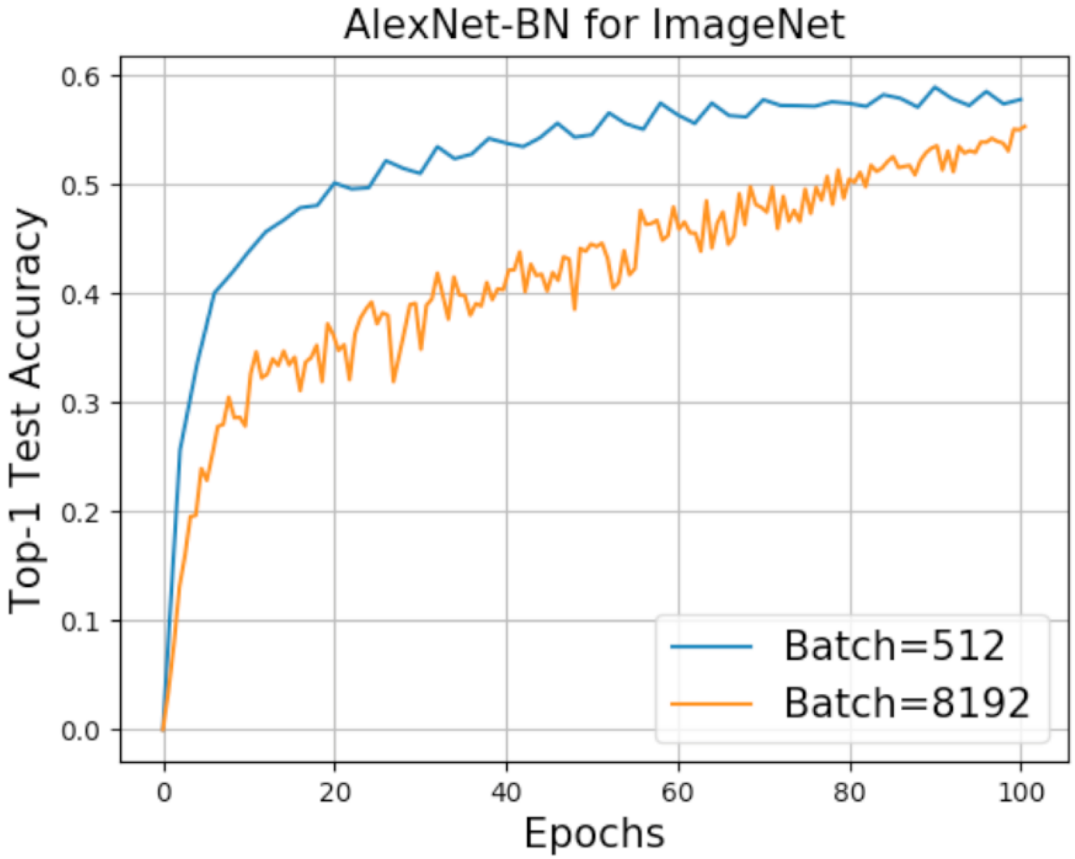
- Layer-wise Adaptive Rate Scaling (LARS)

Algorithm 1 SGD with LARS. Example with weight decay, momentum and polynomial LR decay.

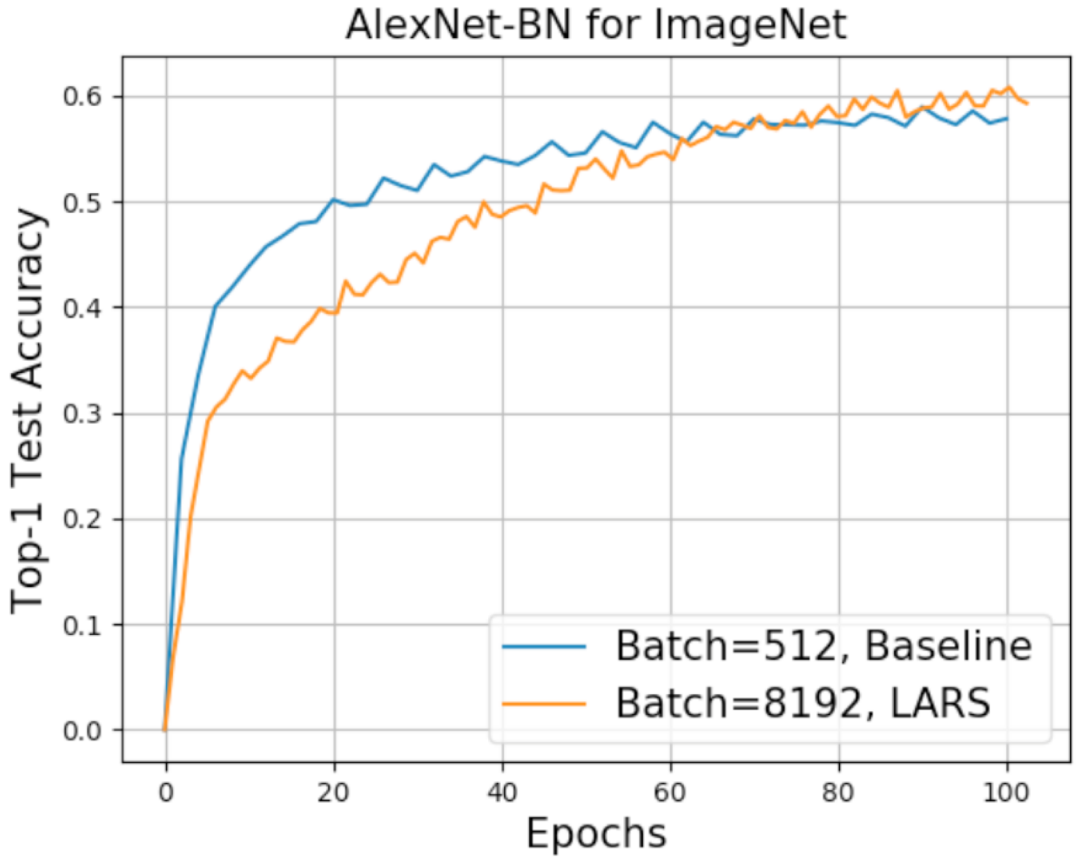
Parameters: base LR γ_0, momentum m, weight decay β, LARS coefficient η, number of steps T
Init: $t = 0, v = 0$. Init weight w_0^l for each layer l
while $t < T$ for each layer l **do**
$g_t^l \leftarrow \nabla L(w_t^l)$ (obtain a stochastic gradient for the current mini-batch)
$\gamma_t \leftarrow \gamma_0 * \left(1 - \frac{t}{T}\right)^2$ (compute the global learning rate)
$\lambda^l \leftarrow \frac{||w_t^l||}{||g_t^l|| + \beta||w_t^l||}$ (compute the local LR λ^l)
$v_{t+1}^l \leftarrow m v_t^l + \gamma_{t+1} * \lambda^l * (g_t^l + \beta w_t^l)$ (update the momentum)
$w_{t+1}^l \leftarrow w_t^l - v_{t+1}^l$ (update the weights)
end while

Yang You, et al. LARGE BATCH TRAINING OF CONVOLUTIONAL NETWORKS

Large Batch Training



(a) Training without LARS



(b) Training with LARS

Large Batch Training

- Trend
  - *Higher performance*
  - *Flexibility*
  - *Multi-task*