Efficient Machine Learning for Visual and Textual Data

Sachin Mehta

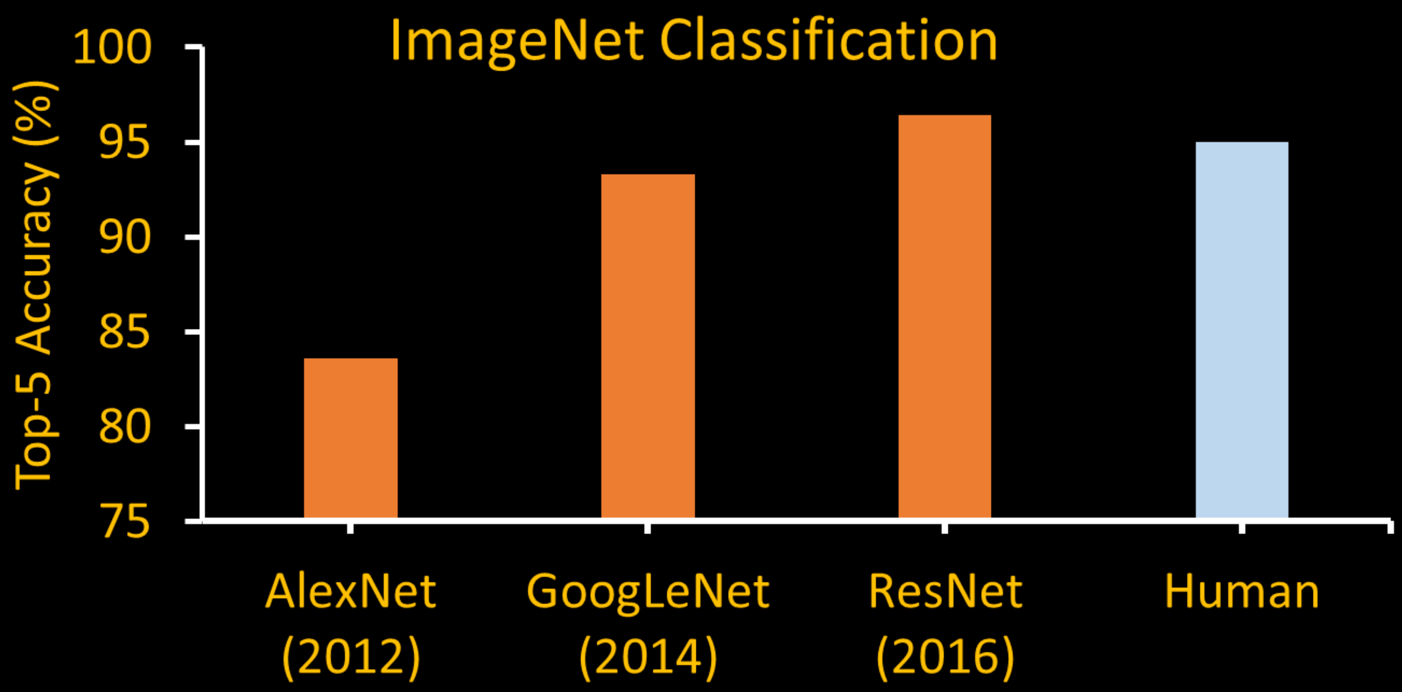
Joint work with Mohammad Rastegari, Linda Shapiro, and Hannaneh Hajishirzi

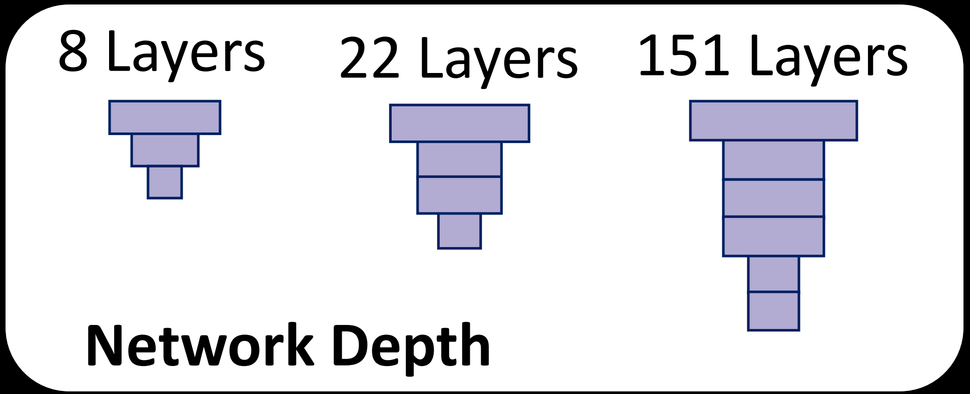
https://h2lab.cs.washington.edu/

https://sacmehta.github.io/

https://github.com/sacmehta/

Introduction






Accuracy **improves**


Speed **reduces**


Energy consumption **increases**

These models cannot be used on **Resource-Constrained Devices**

Resource-constrained devices

Embedded Devices

Mobile phones

Limited energy overhead

Restrictive memory constraints

Limited compute

My Research

Light-weight, Low latency, and SOTA Neural Networks

My Research

Light-weight, Low latency, and SOTA Neural Networks

Different Tasks

ESPNets: ECCV'18, CVPR'19
PRU: EMNLP'18

Object Detection

Semantic Segmentation

Language Modeling

My Research

Light-weight, Low latency, and SOTA Neural Networks

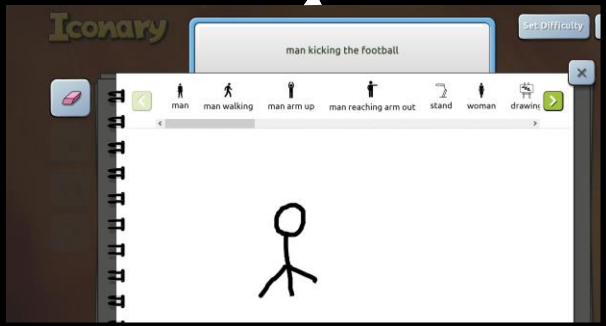
Different Tasks

- Object Detection
- Semantic Segmentation
- Language Modeling
- .....

Different Modalities

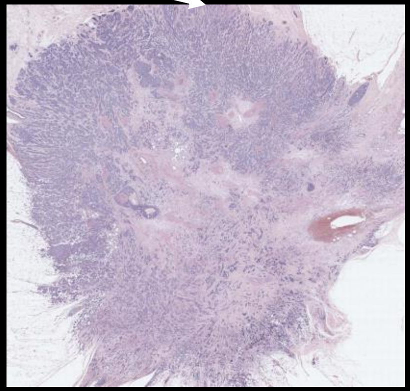
Medical Imaging

- JAMA Network Open'19
- MICCAI'18
- WACV'18

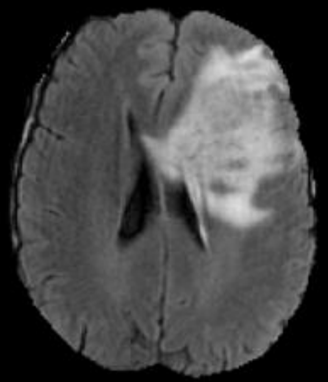


Sketches (Iconary@AI2)

Natural Images



Whole Slide Images



3D MRIs

My Research

Light-weight, Low latency, and SOTA Neural Networks

Different Tasks

- Object Detection
- Semantic Segmentation
- Language Modeling
- .....

Different Modalities

- Natural Images
- Medical Images
- Text
- .....

Different Devices

NVIDIA TX2

Mobile Phone

My Research

Light-weight, Low latency, and SOTA Neural Networks

Different Tasks

- Object Detection
- Semantic Segmentation
- Language Modeling
- .....

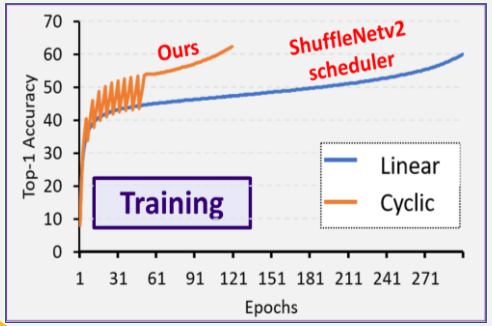
Different Modalities

- Natural Images
- Medical Images
- Text
- .....

Different Devices

- Desktop
- Embedded Devices
- Mobile Devices
- .....

Faster Training



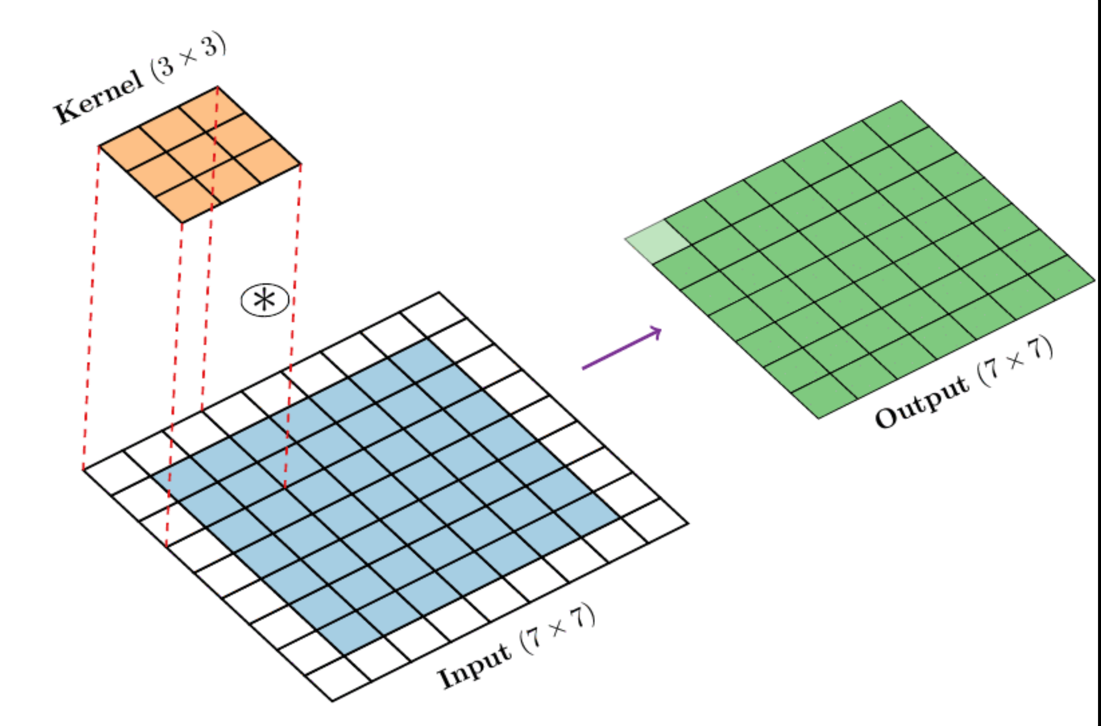
Ours: 1 - 1.5 days

SOTA: 4 - 5 days

Outline

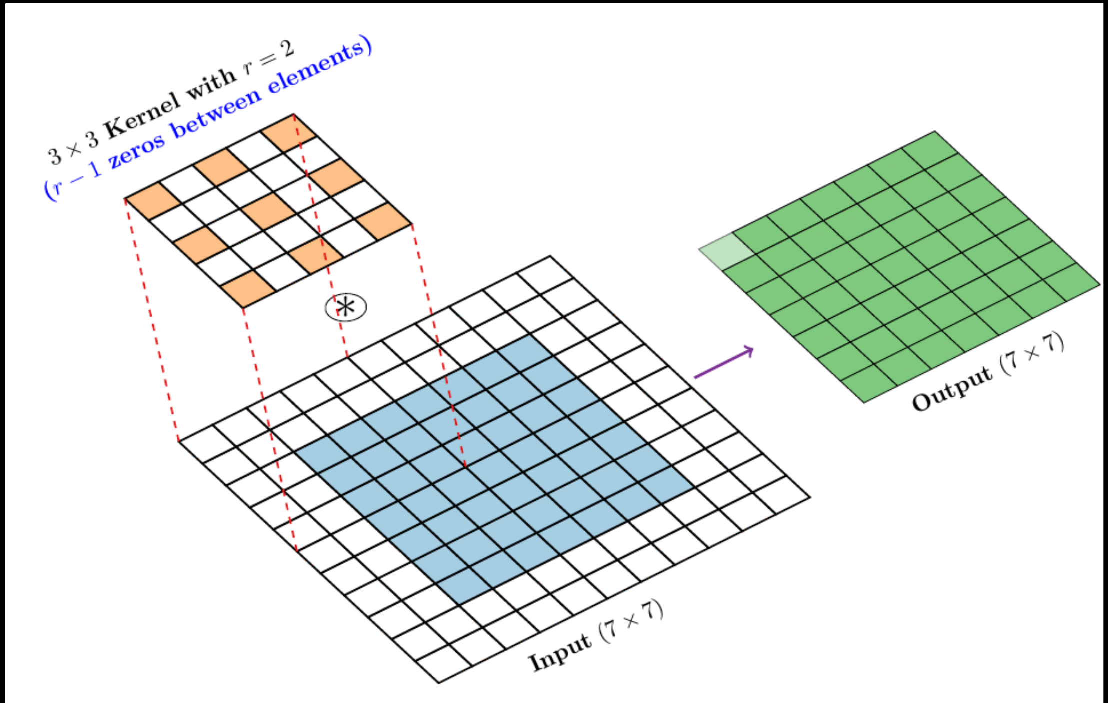
- Brief overview about convolutions
- Image Classification
  - VGG Unit
  - ResNet Unit
  - ESPNet Unit
- Semantic Segmentation
  - Vanilla Encoder-Decoder
  - U-Net Encoder-Decoder
- Results

Convolutions



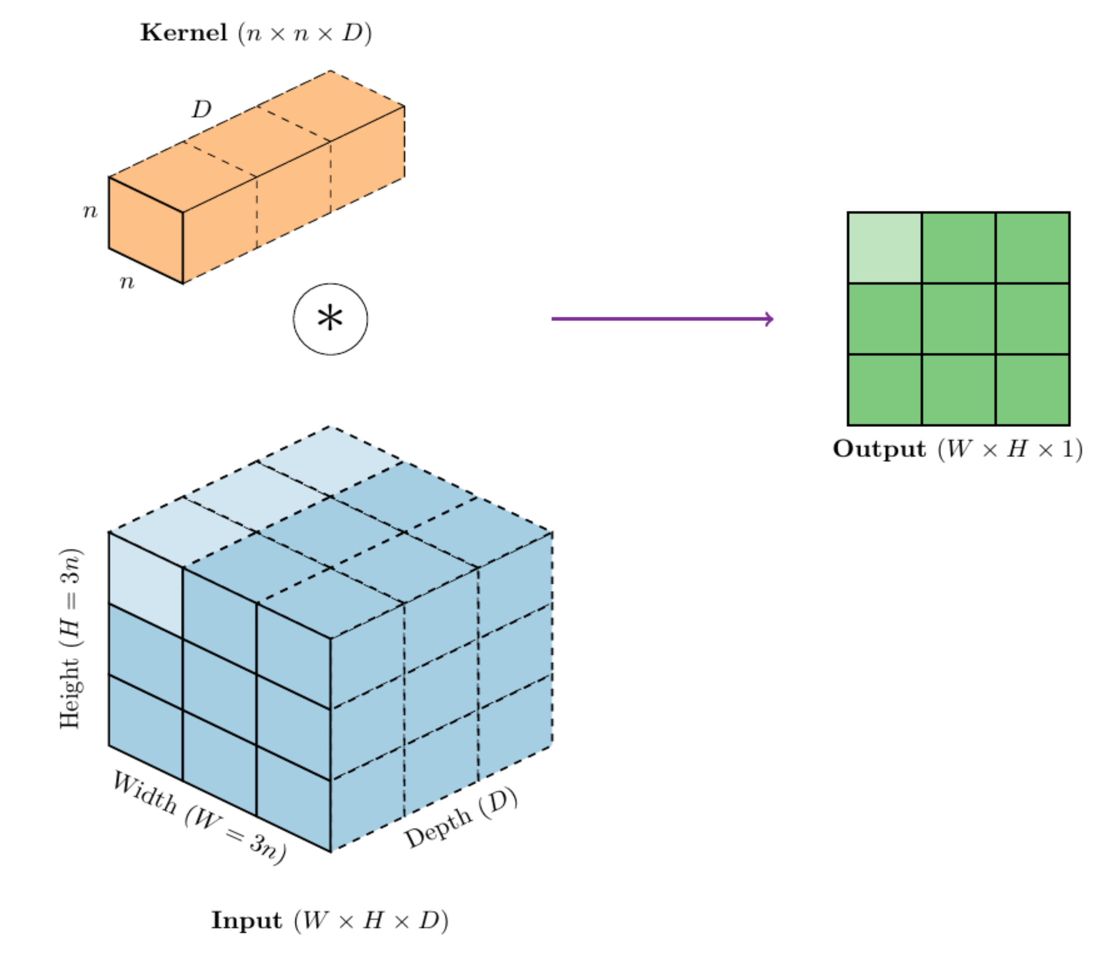
- A widely used operation in computer vision
- Traditionally, kernels are manually designed
  - Sobel filter for edge detection
  - Gaussian filter
- Nowadays, we learn kernels from the data
  - Convolutional neural networks (CNNs)
- A $n \times n$ kernel
  - Has n^2 parameters
  - Performs n^2WH multiplication-addition operations

Convolutions



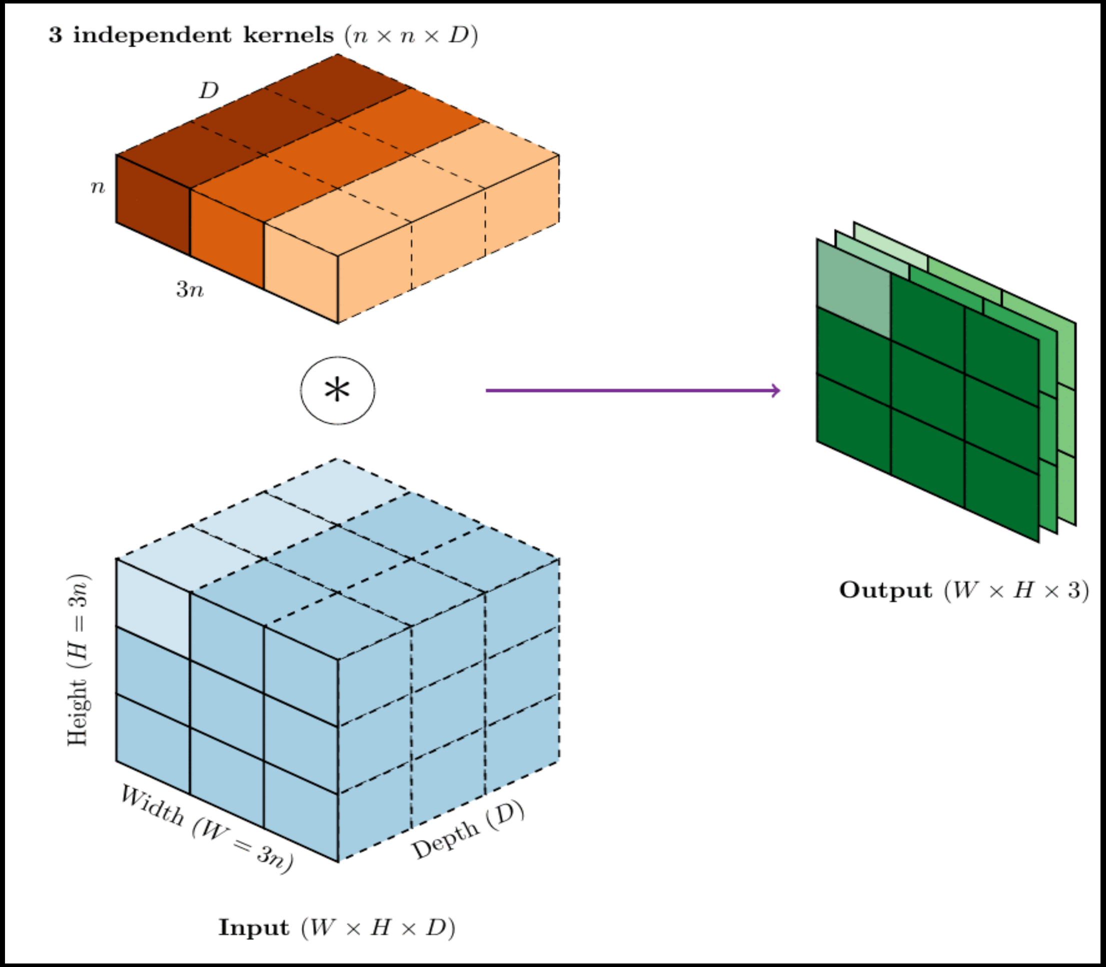
- Higher receptive field
  - Inserts zeros between kernel elements
- A $n \times n$ kernel with a dilation rate of r
  - Has a receptive field of $[(n-1)r+1]^2$
  - Learns n^2 parameters
  - Performs n^2WH multiplication-addition operations

Convolution in 3D



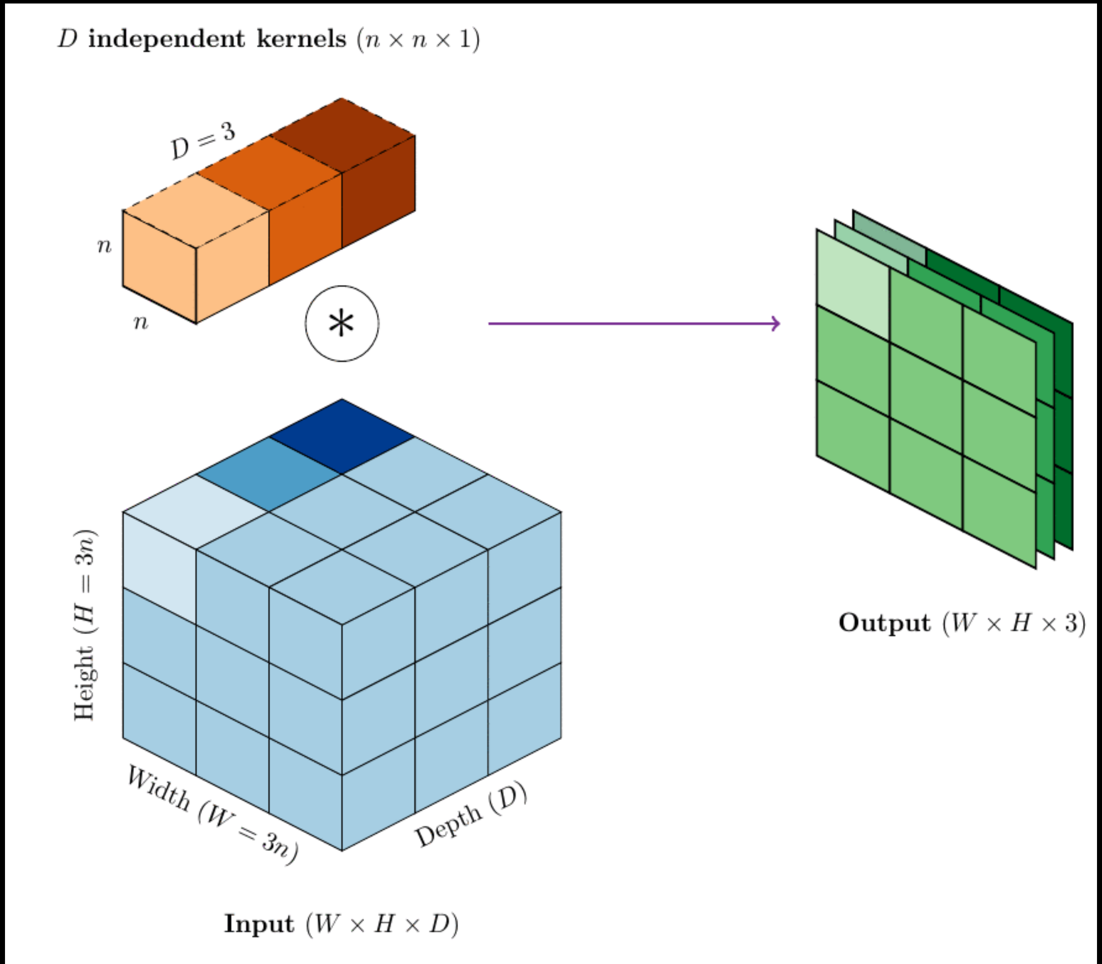
- Both Input and Kernel are 3D
- A $n \times n \times D$ kernel
  - Learns $n^2 D$ parameters
  - Performs $n^2 DWH$ operations to produce an output plane

Convolution in 3D



- Both Input and Kernel are 3D
- A $n \times n \times D$ kernel
  - Learns $n^2 D$ parameters
  - Performs $\mathrm{n}^2 \mathrm{D} WH$ operations to produce an output plane
- Multiple independent kernels are applied to produce high-dimensional output
- K $n \times n \times D$ kernels
  - Learns $n^2 DK$ parameters
  - Performs $n^2 WHDK$ operations

Depth-wise Convolution



- Improves the efficiency of standard convolutions
- Each convolutional filter is applied per spatial plane
- D $n \times n$ kernels
  - Learns n^2D parameters
  - Performs n^2WHD operations

Image Classification

Image Classification

CU	Convolutional Unit
DU	Down-sampling Unit
FC	Fully-connected Or Linear Layer
GAP	Global Avg. Pooling

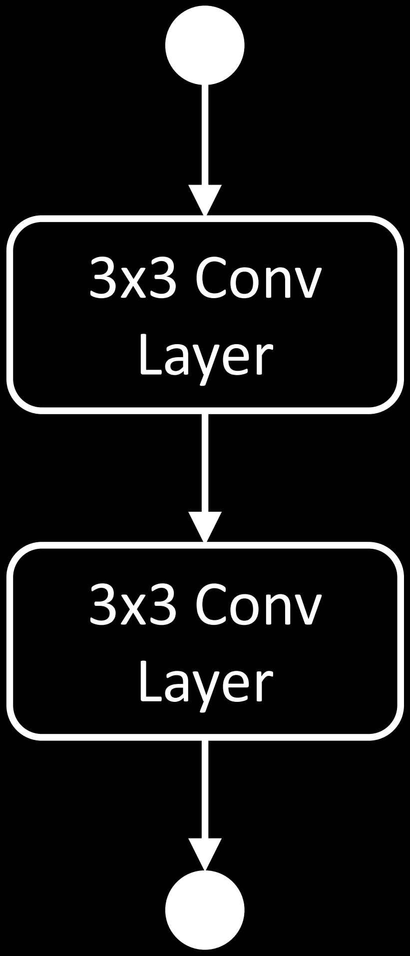
Image Classification

CU	Convolutional Unit	FC	Fully-connected Or Linear Layer
DU	Down-sampling Unit	GAP	Global Avg. Pooling

→ CU → DU → CU → DU → CU → GAP → FC → Cat

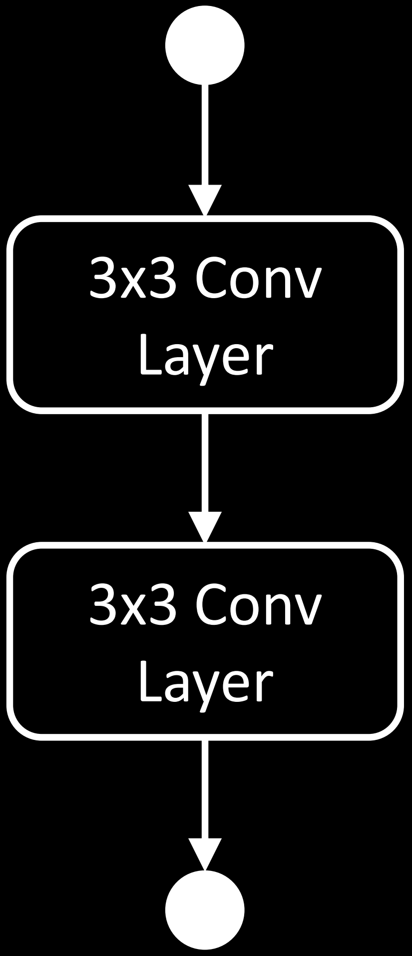
28 x 28 = $[28]^2$	CU	DU	CU	DU	CU	GAP	FC
	$[28]^2$	$[14]^2$	$[14]^2$	$[7]^2$	$[7]^2$	$[1]^2$	

Convolutional Unit (CU) - VGG



VGG: Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *ICLR, 2015*.

Convolutional Unit (CU) - VGG



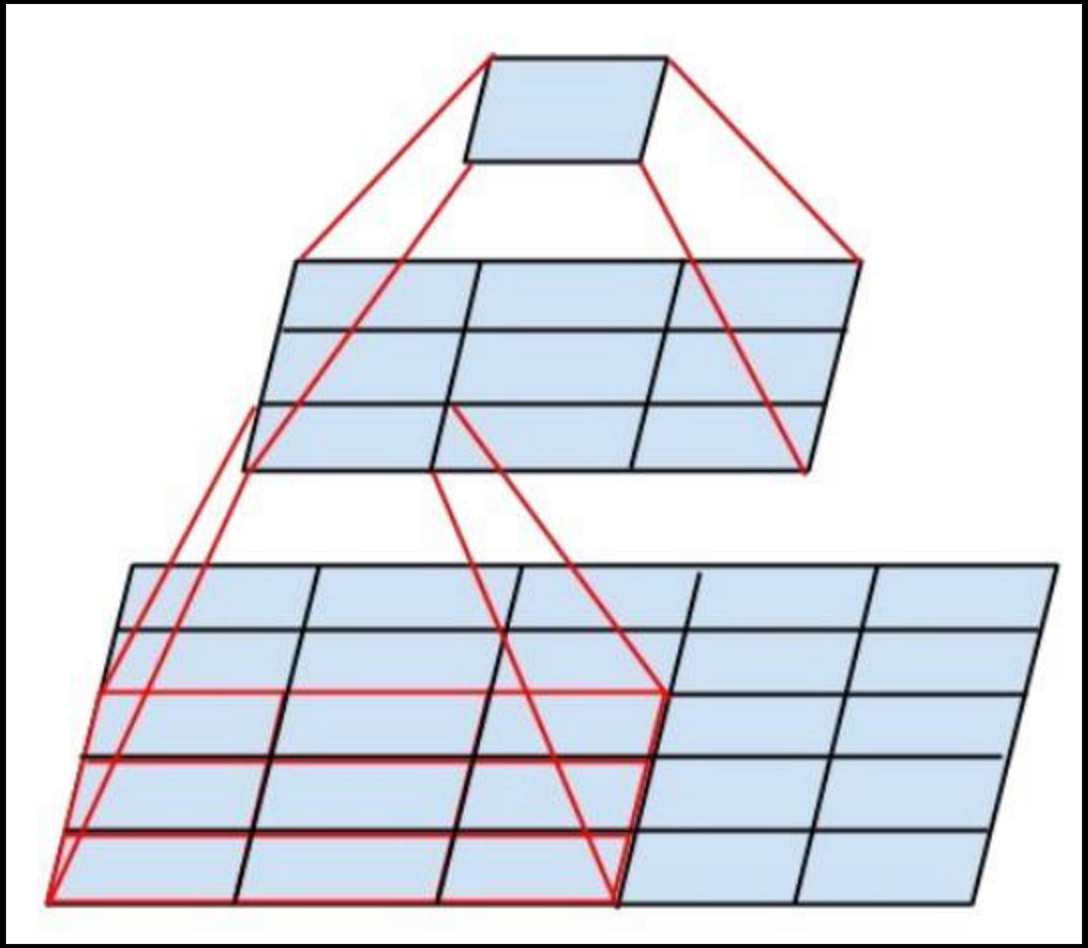
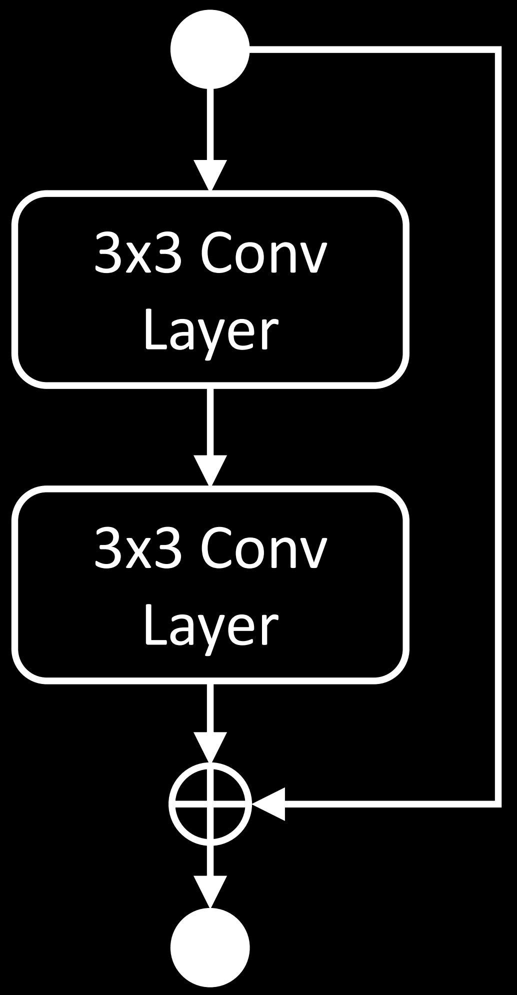


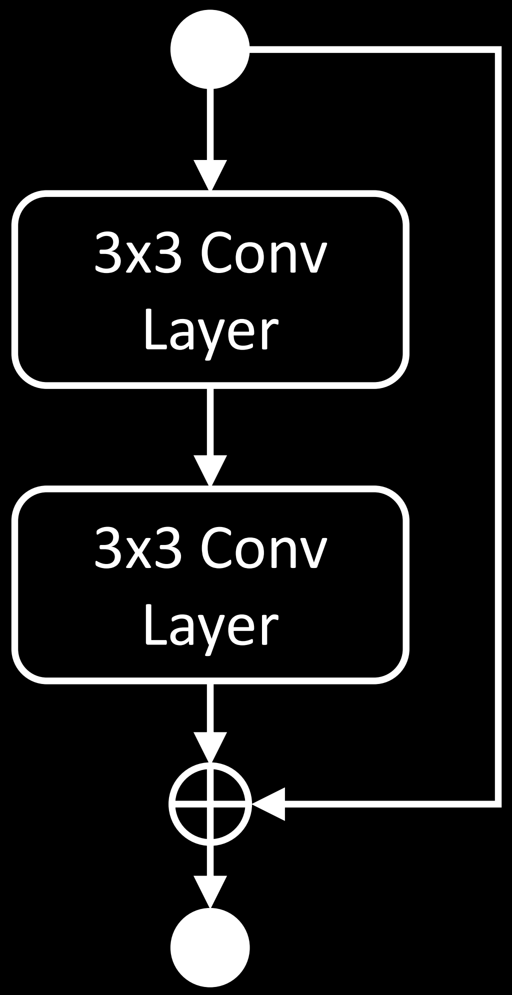
Image Source (Inception): Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *CVPR*. 2016.

Convolutional Unit (CU) - ResNet



ResNet: He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.

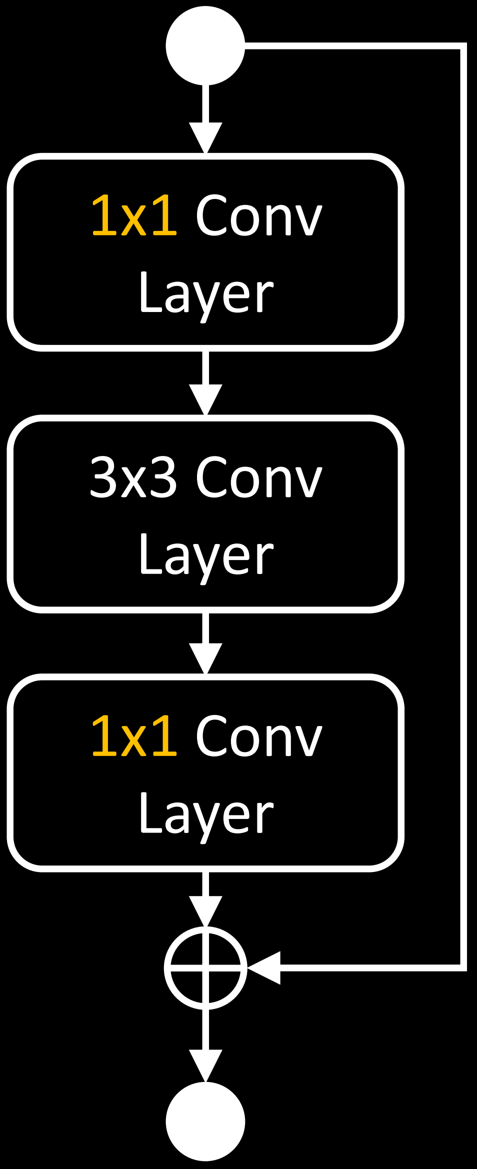
Convolutional Unit (CU) - ResNet



- Element-wise addition of input and output
- Often referred as Residual Connection
- Improves gradient flow and accuracy
- Computationally expensive
  - Hard to train very deep networks (101-151 layers)

ResNet: He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.

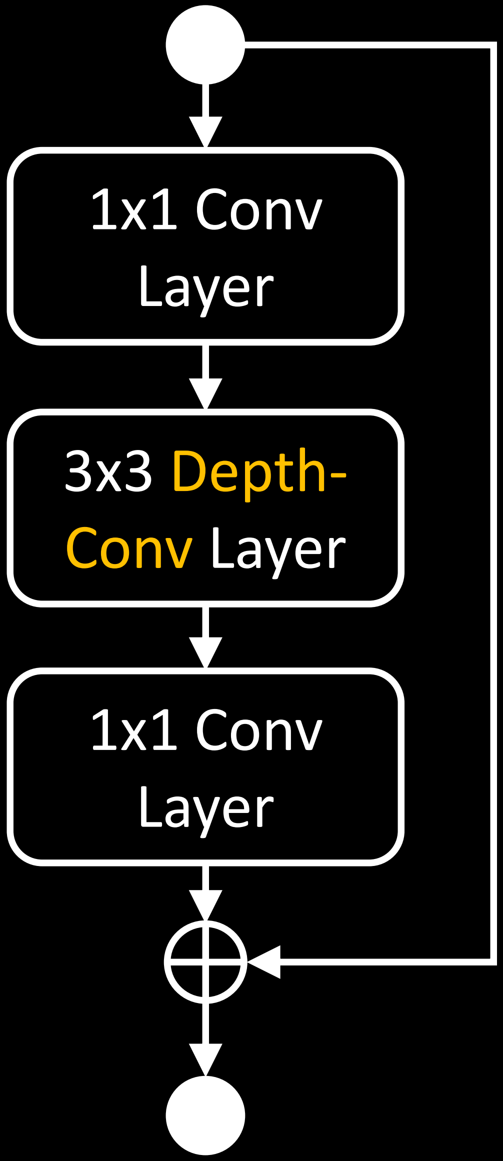
Convolutional Unit (CU) - ResNet



- Bottleneck unit
- Exercise?
  - Validate this unit is efficient

ResNet: He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.

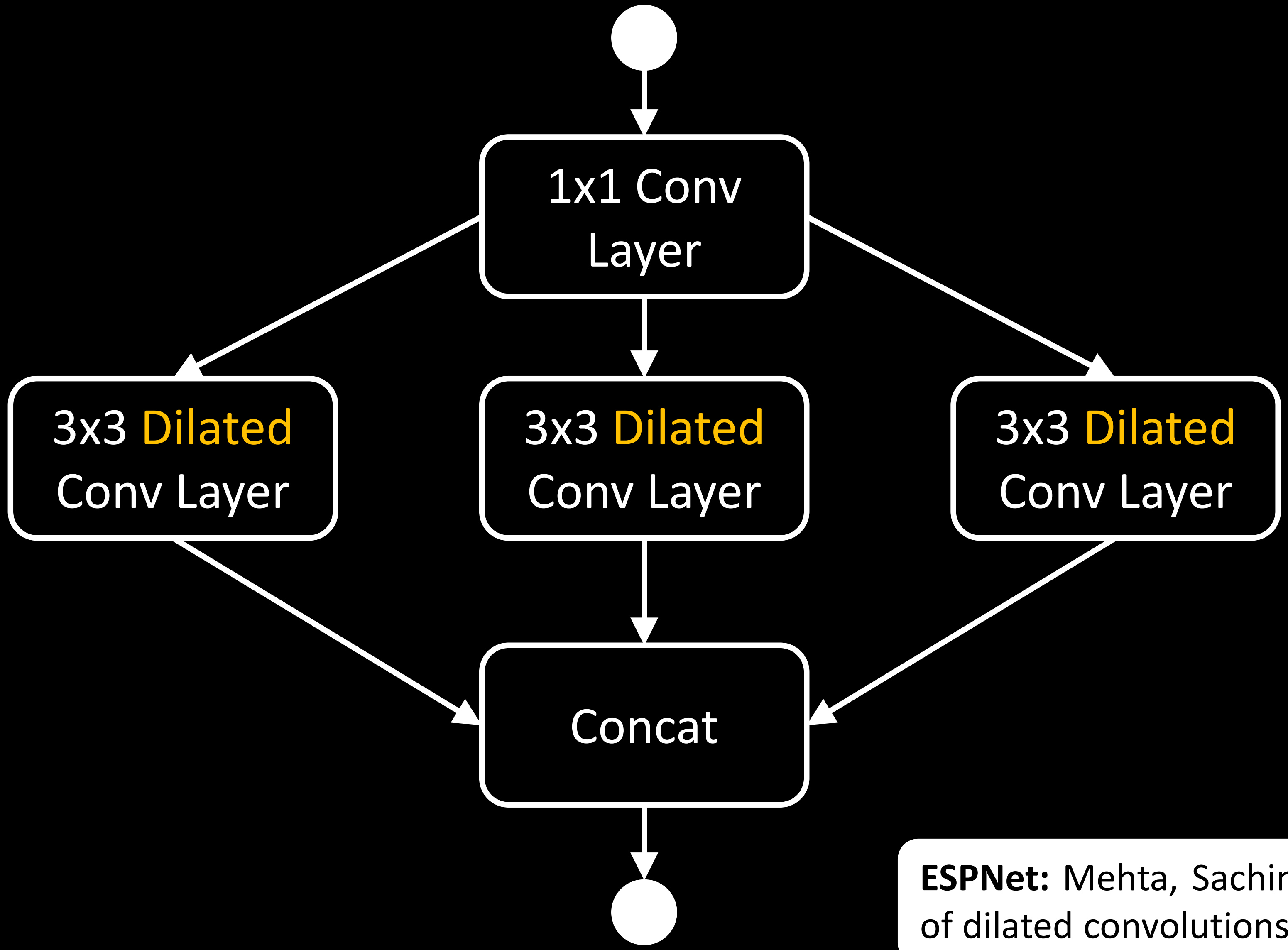
Convolutional Unit (CU) - ResNet



- Bottleneck unit with Depth-wise convs
  - MobileNetv2
  - ShuffleNetv2

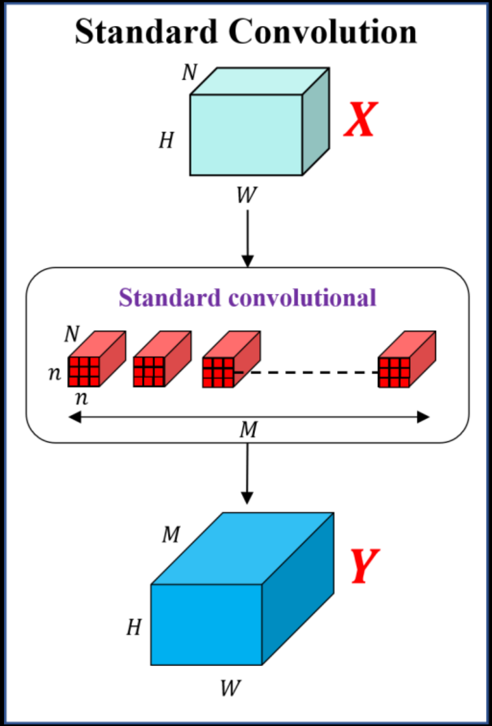
- **MobileNetv2:** Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." CVPR, 2018.
- **ShuffleNetv2:** Ma, Ningning, et al. "Shufflenet v2: Practical guidelines for efficient cnn architecture design." ECCV, 2018.

Convolutional Unit - ESPNet

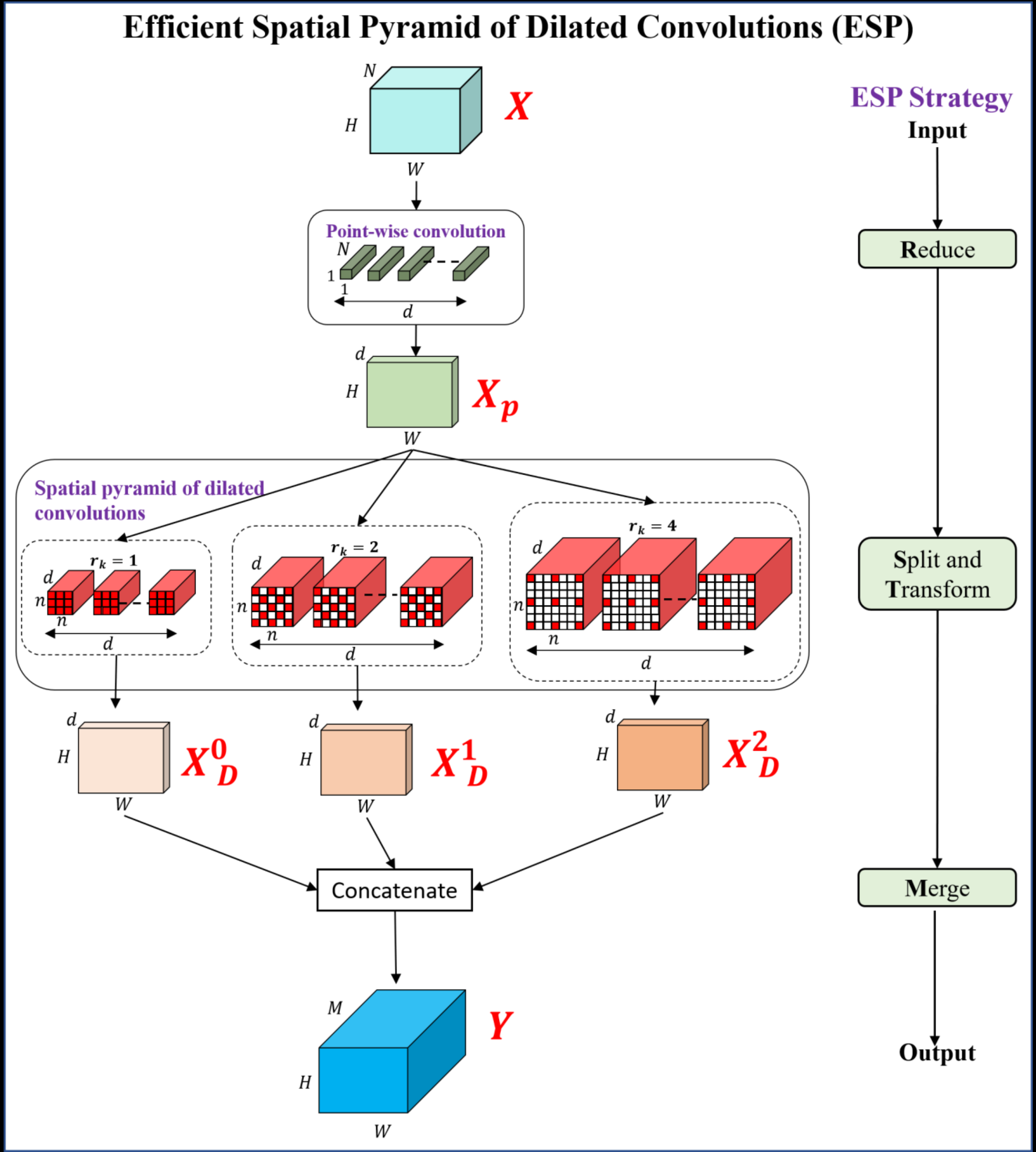


ESPNet: Mehta, Sachin, et al. "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation." ECCV, 2018.

ESPNet Unit

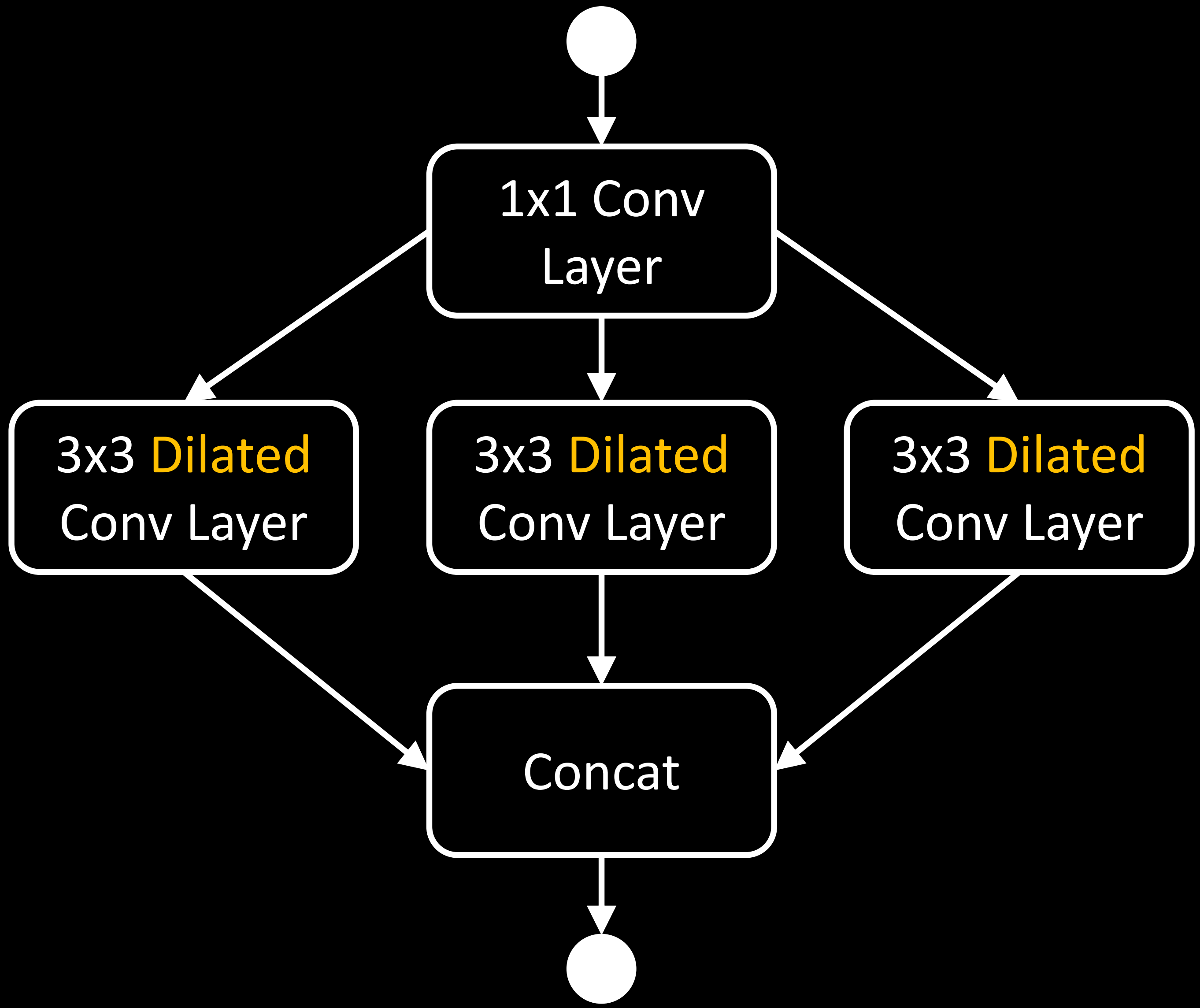


Standard Convolution
N
H
W
X
Standard convolutional
N
n
n
M
M
H
W
Y

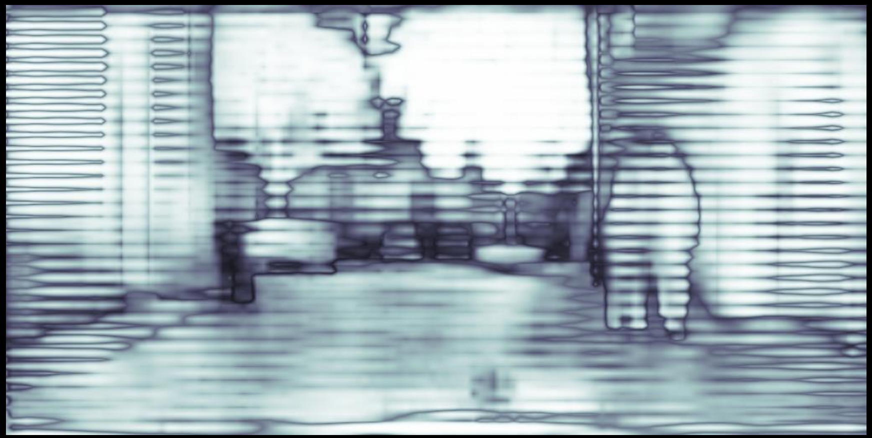


Efficient Spatial Pyramid of Dilated Convolutions (ESP)
N
H
W
X
Point-wise convolution
N
1
1
d
d
H
W
X_p
Spatial pyramid of dilated convolutions
$r_k = 1$
$r_k = 2$
$r_k = 4$
d
n
n
d
X_D^0
X_D^1
X_D^2
Concatenate
M
H
W
Y
ESP Strategy
Input
Reduce
Split and Transform
Merge
Output

Hierarchical Feature Fusion in ESPNet

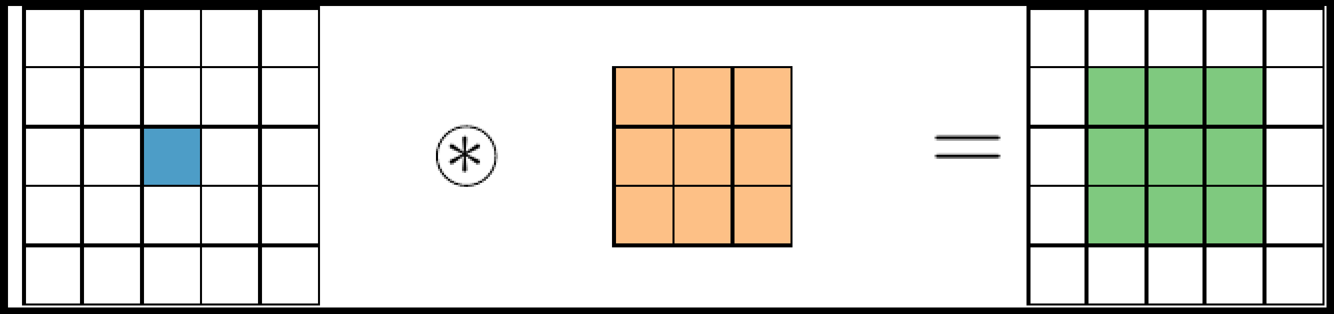


Input

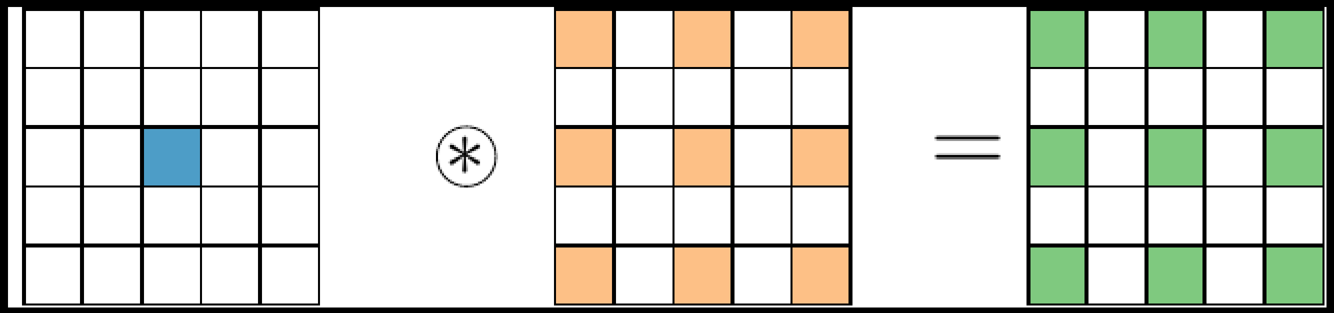


Output of ESP module

Gridding Artifact in Dilated Convolutions

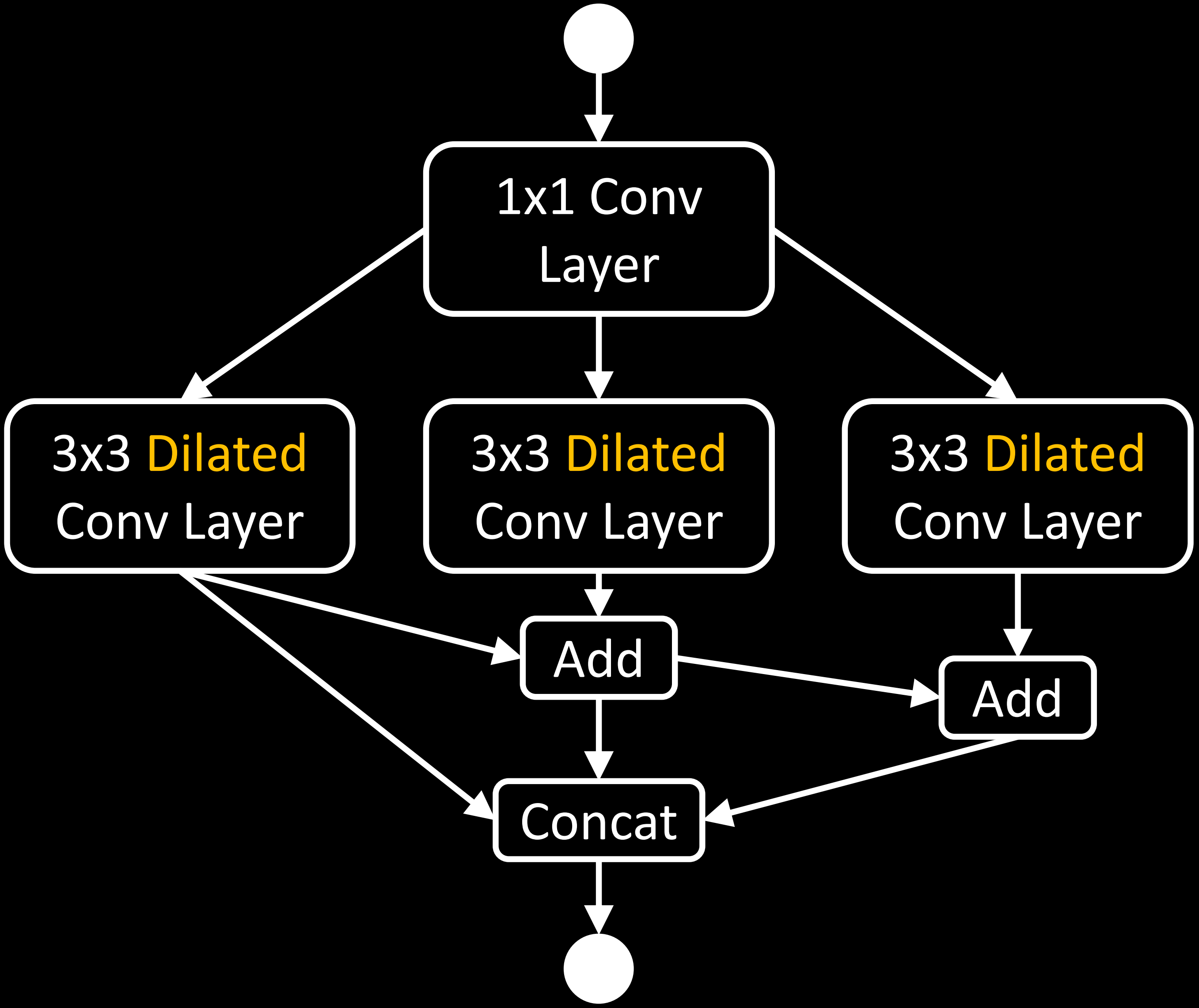


Standard convolution

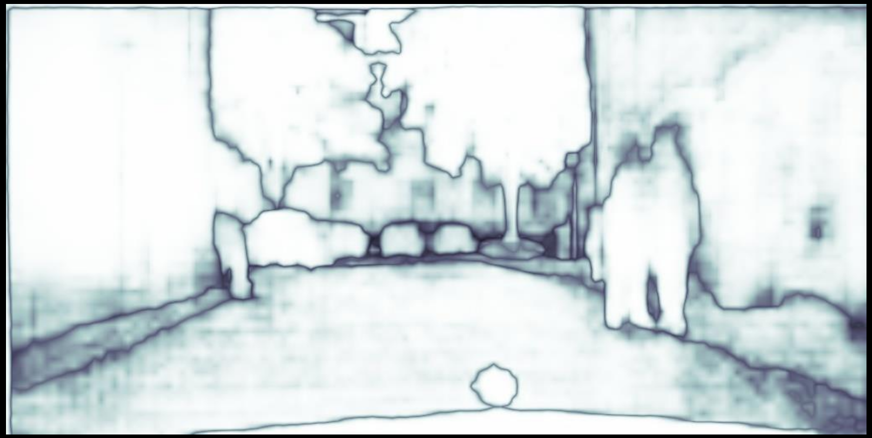


Dilated convolution

Hierarchical Feature Fusion in ESPNet

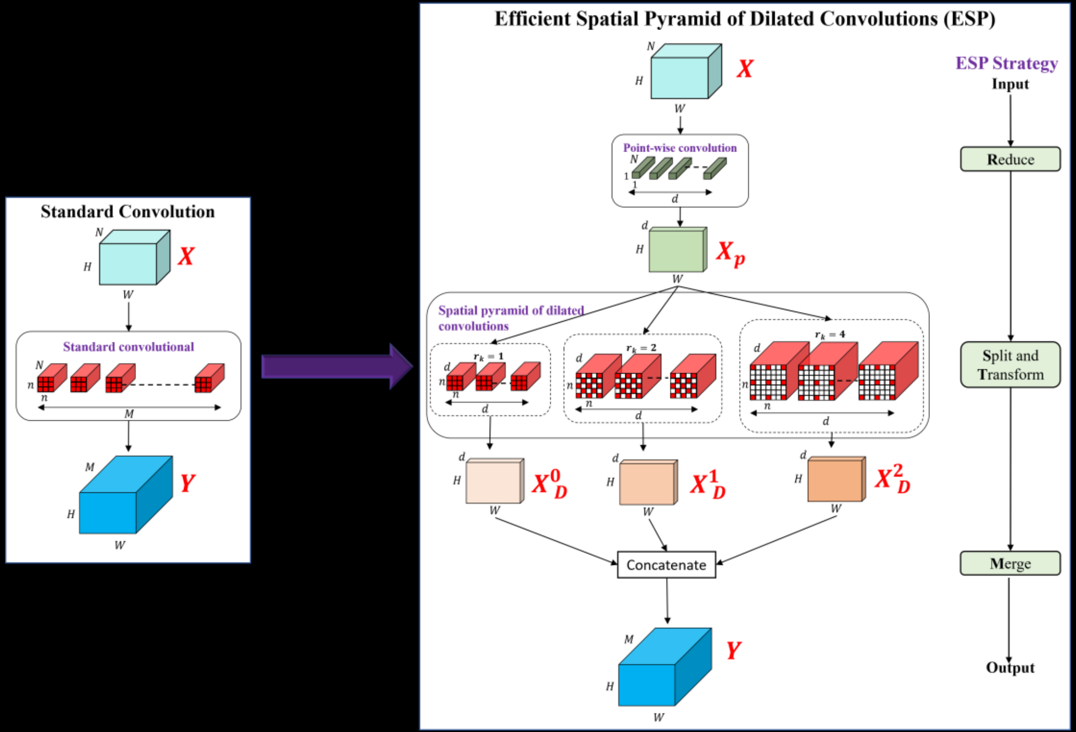


Input



Output of ESP module

Convolutional Unit – ESPNetv2



- ESPNetv2
  - 3x3 dilated convolutions are replaced by 3x3 depth-wise dilated convolutions

- **ESPNet:** Mehta, Sachin, et al. “ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation." ECCV, 2018.
- **ESPNetv2:** Mehta, Sachin, et al. "Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network." CVPR, 2019.

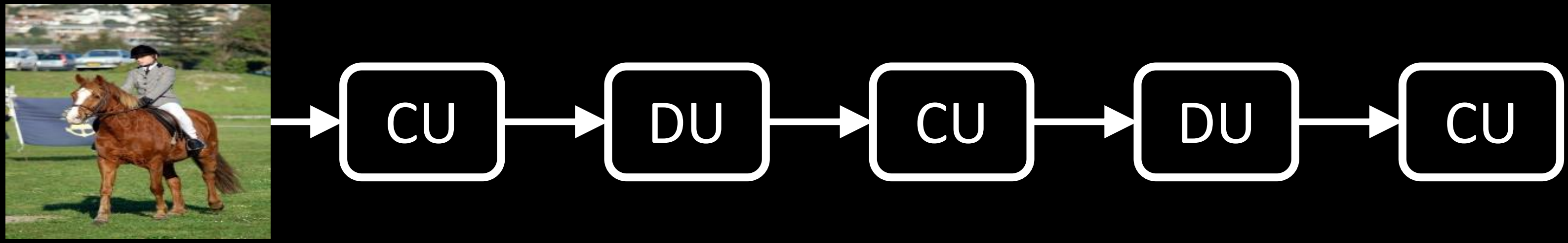
Semantic Segmentation

Image Classification

CU → DU → CU → DU → CU → GAP → FC → Cat

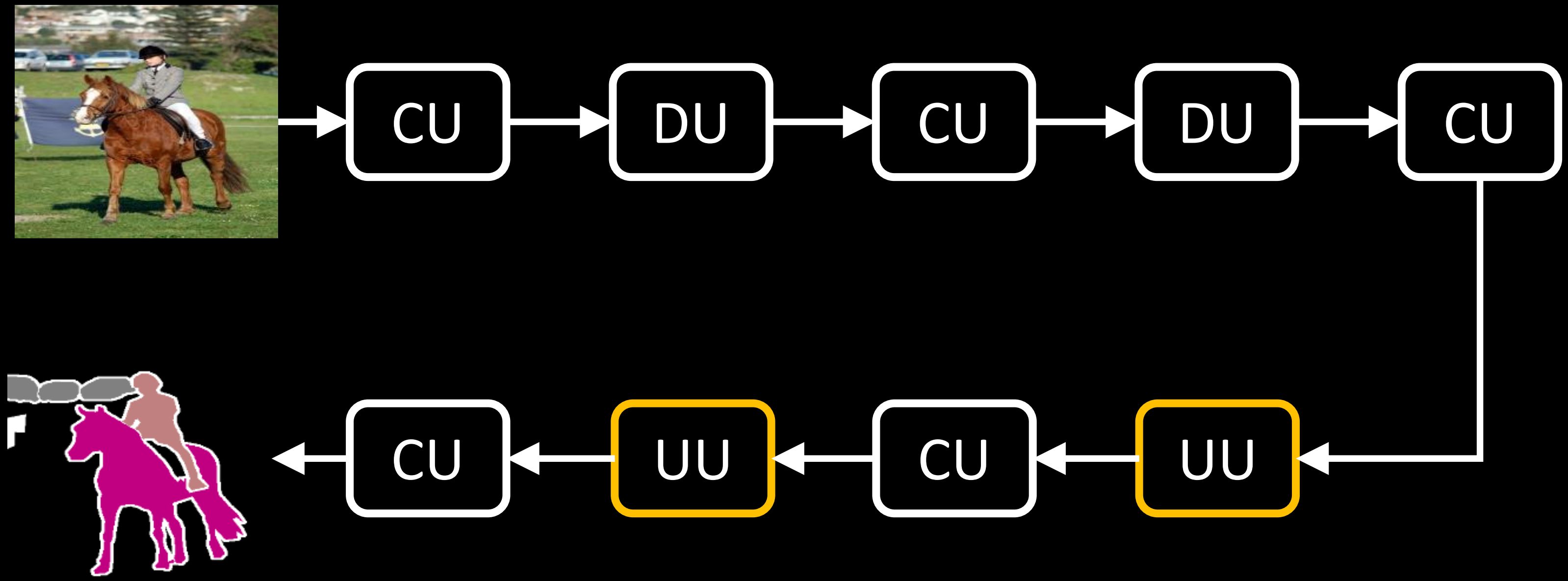
CU	Convolutional Unit
FC	Fully-connected Or Linear Layer
DU	Down-sampling Unit
GAP	Global Avg. Pooling

Encoder-Decoder



CU	Convolutional Unit
FC	Fully-connected Or Linear Layer
DU	Down-sampling Unit
GAP	Global Avg. Pooling

Encoder-Decoder



UU — Up-sampling Unit

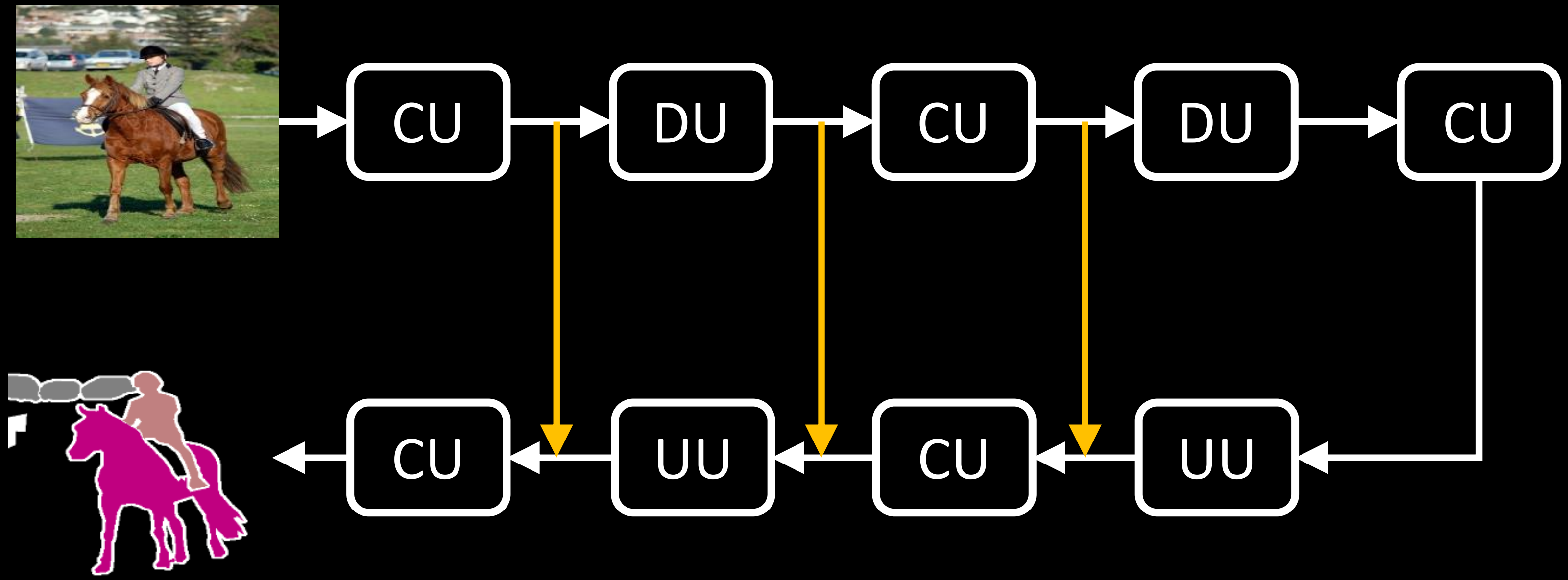
CU — Convolutional Unit

FC — Fully-connected Or Linear Layer

DU — Down-sampling Unit

GAP — Global Avg. Pooling

Encoder-Decoder with Skip-Connections (U-Net)



UU Up-sampling Unit

CU Convolutional Unit

FC Fully-connected Or Linear Layer

DU Down-sampling Unit

GAP Global Avg. Pooling

Papers for Semantic Segmentation

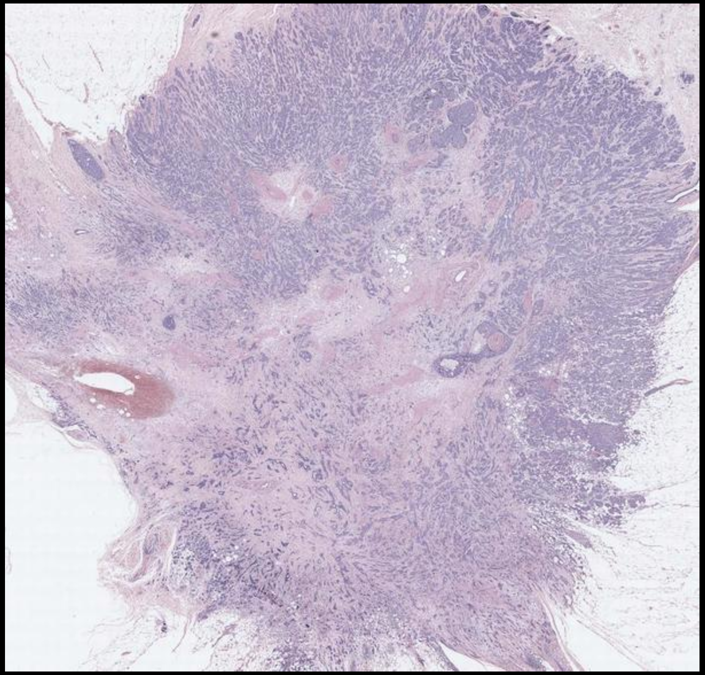
- FCN: Long et al. "Fully convolutional networks for semantic segmentation." CVPR, 2015.
- SegNet: Badrinarayanan et al. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", PAMI, 2017
- DeepLab:
  - Chen, Liang-Chieh, et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs." *PAMI, 2017*.
  - Chen, Liang-Chieh, et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation." ECCV, 2018
- UNet: Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *MICCAI,* 2015.

Results

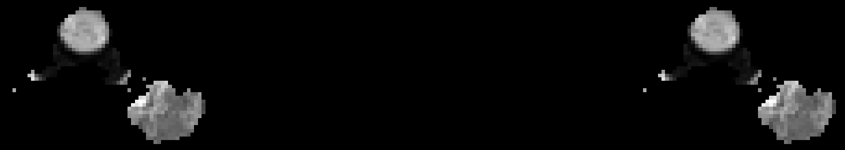
Semantic Segmentation

Semantic Segmentation

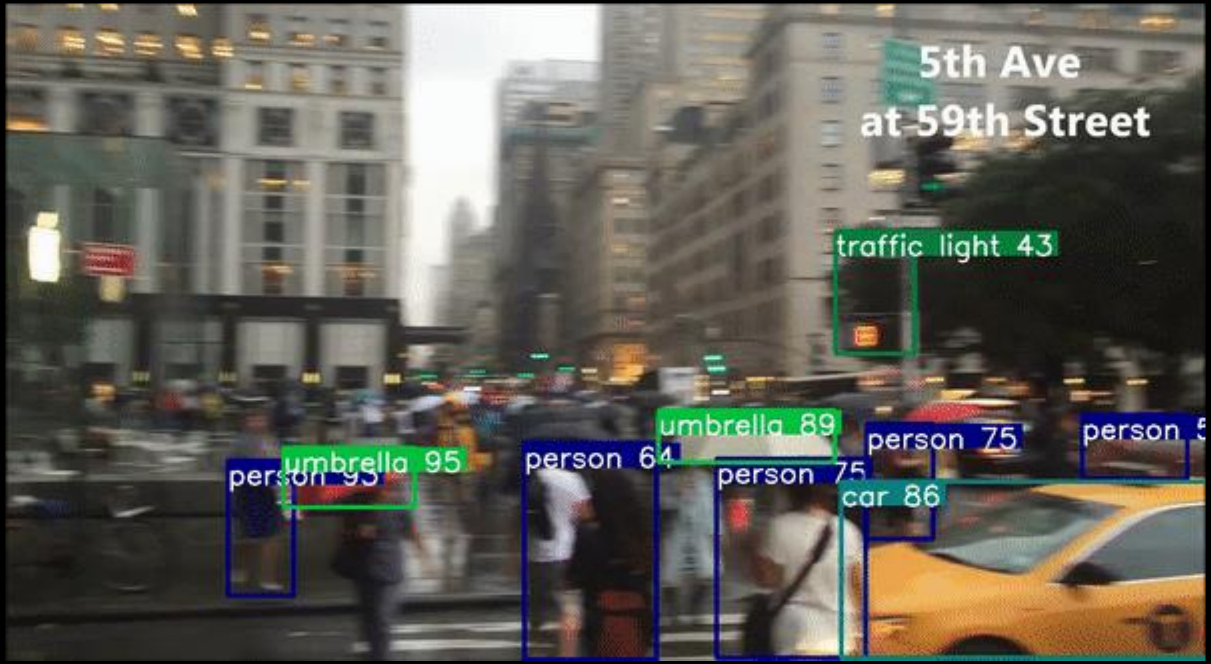
WSIs



3D MRIs



Object Detection



Thanks!!

https://sacmehta.github.io/

https://github.com/sacmehta/