Seoul
National
University

Convolutional Neural Network

Taewook Ko

SCONE
Lab.

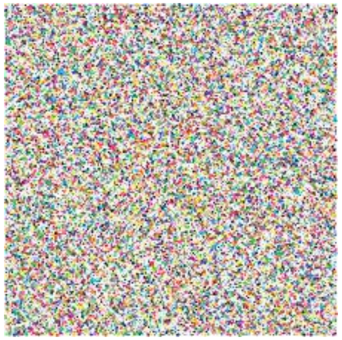
Neural Network

- Network Types
  - Multi-layer perceptron (MLP)
    - Fully-connected neural network (FCNN)
    - Dense Network
  - Convolutional Neural Network (CNN)
  - Recurrent Neural Network (RNN)

Limitation of MLP

- Feature order and shape
  - Some data has there own neighboring feature distribution
    - The context of the feature distribution
  - Images!
    - Images are set of pixels
      - There is a dog image
      - There is a dog image after randomly shuffling the order of pixels
    - The shuffled image is no more a dog

Real Image

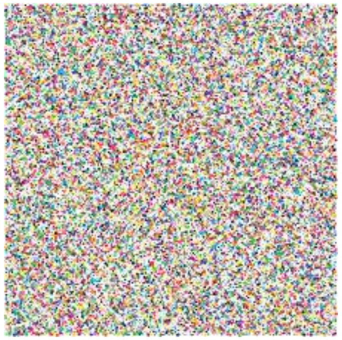


Re-ordering Image

Limitation of MLP

- Feature order and shape
  - Neighboring pixels (features) has there own reason
    - Why they are neighboring together
  - If there is different distribution
    - It is different data
  - A feature order or shape of a data
    - May contain important information

Real Image

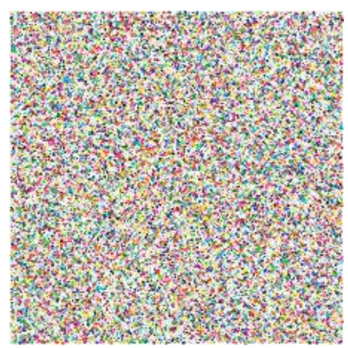


Re-ordering Image

Limitation of MLP

- Feature order and shape
  - MLP always takes vector input
  - Need Flattening when using 2D or higher input
  - The important information can be lost
    - When Flattening or any manipulation on raw data
  - For MLP, it has the same result when using real image and re-ordered image

Real Image

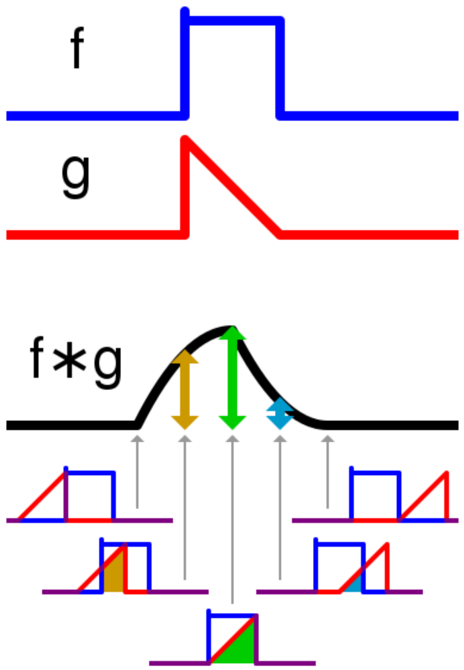


Re-ordering Image

MLP Flatten Input

Convolutional Operation

- Convolution operation in signal processing
  - $f * g(x) = \int f(\tau)g(x-\tau)d\tau$
    $= \sum f(\tau)g(x-\tau)$
  - Mathematically meaning
    - A weighted average of all the inputs **around the 'x'**
    - Not only for the 'x'
      - Consider values around the 'x'
      - Expressed $'\tau'$ in the equation
  - Convolutional is good for considering
    - Neighbor information



Convolution Layer

- Convolution Operation
  - $O[i,j] = x[i,j] * w[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[i,j] \cdot w[i-m, j-n]$
  - When also consider adjacent neighboring information
    - 3x3 filter
    - $\mathrm{O}_{\mathrm{i,j}} = \sum_{\mathrm{m}=-1}^{1} \sum_{\mathrm{n}=-1}^{1} \mathrm{w}_{\mathrm{m,n}} \cdot \mathrm{x}_{\mathrm{i+m+1,i+n+1}}$

2	4	1	3	4
5	1	5	6	1
3	1	2	3	2
4	3	1	2	3
2	2	1	2	4

Data (5x5)

$\times$

$\mathrm{w}_{0,0}$	$\mathrm{w}_{0,1}$	$\mathrm{w}_{0,2}$
$\mathrm{w}_{1,0}$	$\mathrm{w}_{1,1}$	$\mathrm{w}_{1,2}$
$\mathrm{w}_{2,0}$	$\mathrm{w}_{2,1}$	$\mathrm{w}_{2,2}$

3x3 filter

$=$

$\mathrm{O}_{0,0}$	$\mathrm{O}_{0,1}$	$\mathrm{O}_{0,2}$
$\mathrm{O}_{1,0}$	$\mathrm{O}_{1,1}$	$\mathrm{O}_{1,2}$
$\mathrm{O}_{2,0}$	$\mathrm{O}_{2,1}$	$\mathrm{O}_{2,2}$

Output (3x3)

Convolution Layer

- Convolution Operation
  - $O_{0,0}$: Convolute Data[:3,:3] with filter

$$O_{0,0} = (\, 2 \times W_{0,0} + 4 \times W_{0,1} + 1 \times W_{0,2} \\ + 5 \times W_{1,0} + 1 \times W_{1,1} + 5 \times W_{1,2} \\ + 3 \times W_{2,0} + 1 \times W_{2,1} + 2 \times W_{2,2} \,)$$

2	4	1	3	4
5	1	5	6	1
3	1	2	3	2
4	3	1	2	3
2	2	1	2	4

Data (5x5)

×

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$

3x3 filter

=

$O_{0,0}$	$O_{0,1}$	$O_{0,2}$
$O_{1,0}$	$O_{1,1}$	$O_{1,2}$
$O_{2,0}$	$O_{2,1}$	$O_{2,2}$

Output (3x3)

Convolution Layer

- Convolution Operation
  - $O_{0,1}$: Convolute Data[1:4,1:4] with filter

$$O_{0,1} = (\,4 \times W_{0,0} + 1 \times W_{0,1} + 3 \times W_{0,2}$$
$$+\ 1 \times W_{1,0} + 5 \times W_{1,1} + 6 \times W_{1,2}$$
$$+\ 1 \times W_{2,0} + 2 \times W_{2,1} + 3 \times W_{2,2}\,)$$

2	4	1	3	4
5	1	5	6	1
3	1	2	3	2
4	3	1	2	3
2	2	1	2	4

Data (5x5)

×

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$

3x3 filter

=

$O_{0,0}$	$O_{0,1}$	$O_{0,2}$
$O_{1,0}$	$O_{1,1}$	$O_{1,2}$
$O_{2,0}$	$O_{2,1}$	$O_{2,2}$

Output (3x3)

Convolution Layer

- Convolution Operation
  - $O_{0,2}$: Convolute Data[2:5,2:5] with filter

$$O_{0,2} = (\,1 \times W_{0,0} + 3 \times W_{0,1} + 4 \times W_{0,2} + 5 \times W_{1,0} + 6 \times W_{1,1} + 1 \times W_{1,2} + 2 \times W_{2,0} + 3 \times W_{2,1} + 2 \times W_{2,2}\,)$$

2	4	1	3	4
5	1	5	6	1
3	1	2	3	2
4	3	1	2	3
2	2	1	2	4

Data (5x5)

×

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$

3x3 filter

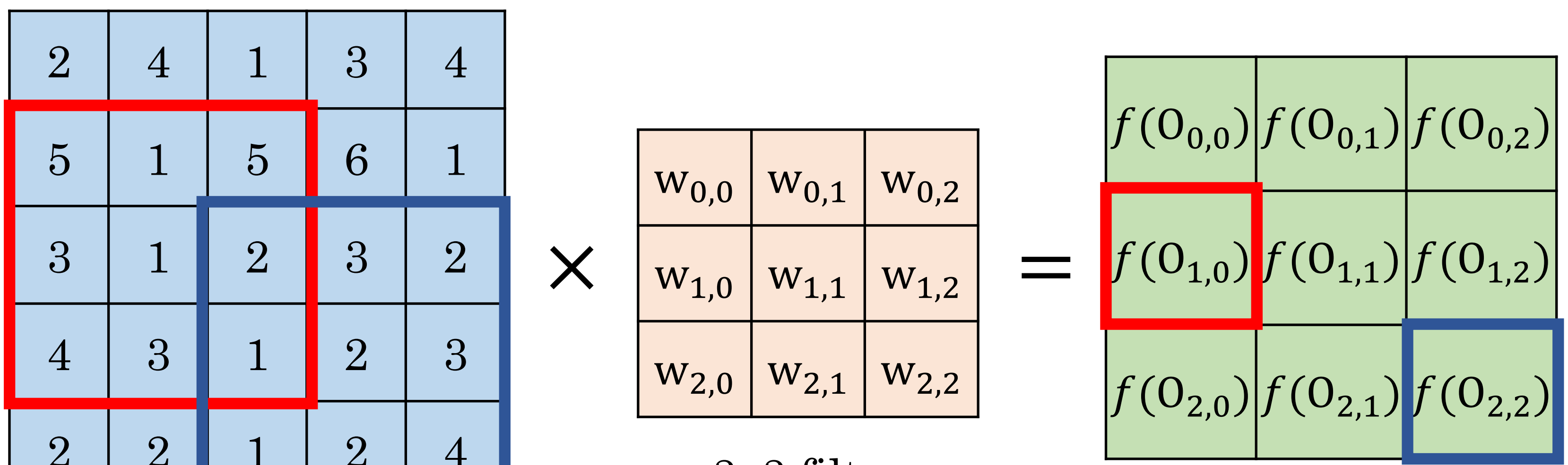
=

$O_{0,0}$	$O_{0,1}$	$O_{0,2}$
$O_{1,0}$	$O_{1,1}$	$O_{1,2}$
$O_{2,0}$	$O_{2,1}$	$O_{2,2}$

Output (3x3)

Convolution Layer

- Convolutional Operation
  - Apply activation on the output
  - The output of convolutional layer is called
    - Activation map



Data (5x5)

3x3 filter

Output (3x3)

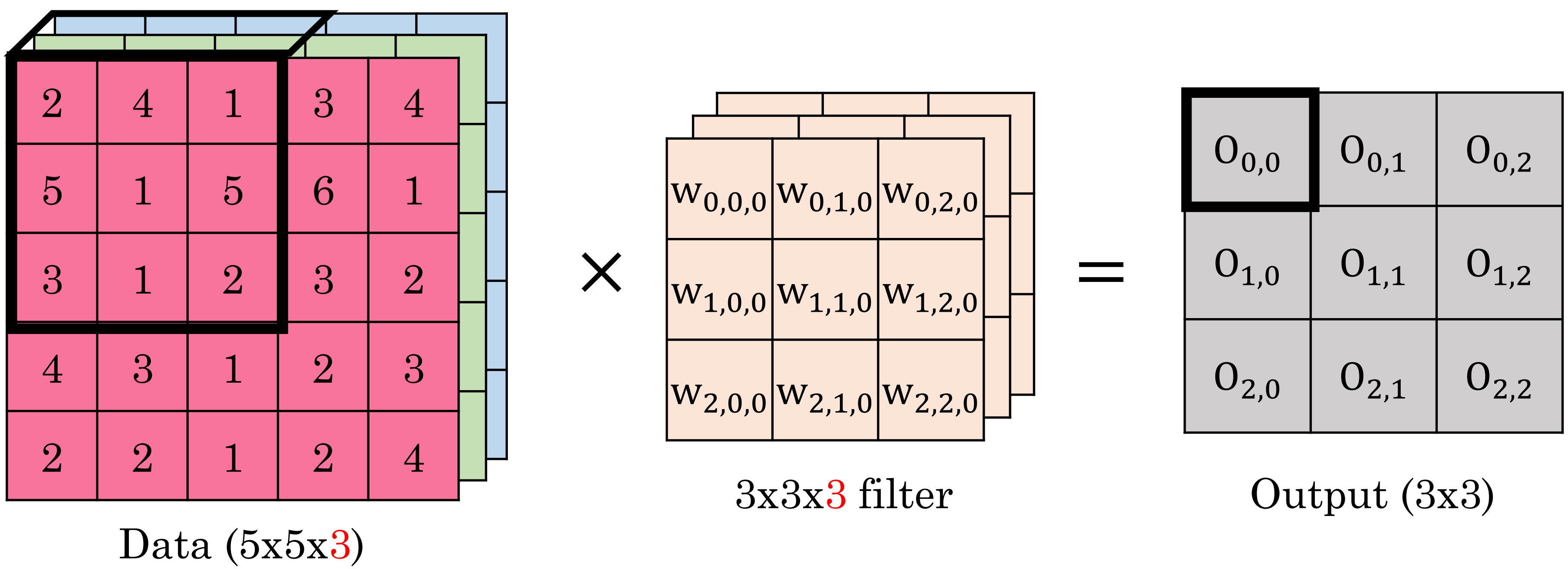
Convolution Layer

- Meaning of convolutional operation
  - Aggregate filter sized features together from raw data
    - 9 features for 3x3 filter
  - Aggregate features with neighboring information
    - Preserves the neighboring context of the raw data
- Meaning of filters
  - Parameters to train
    - 9 parameters for 3x3 filter
  - Weights, how much important a pixel is
  - Extract pattern or information by weighted sum

$$\begin{bmatrix} 2 & 4 & 1 & 3 & 4 \\ 5 & 1 & 5 & 6 & 1 \\ 3 & 1 & 2 & 3 & 2 \\ 4 & 3 & 1 & 2 & 3 \\ 2 & 2 & 1 & 2 & 4 \end{bmatrix} \times \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} \\ w_{1,0} & w_{1,1} & w_{1,2} \\ w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix} = \begin{bmatrix} O_{0,0} & O_{0,1} & O_{0,2} \\ O_{1,0} & O_{1,1} & O_{1,2} \\ O_{2,0} & O_{2,1} & O_{2,2} \end{bmatrix}$$

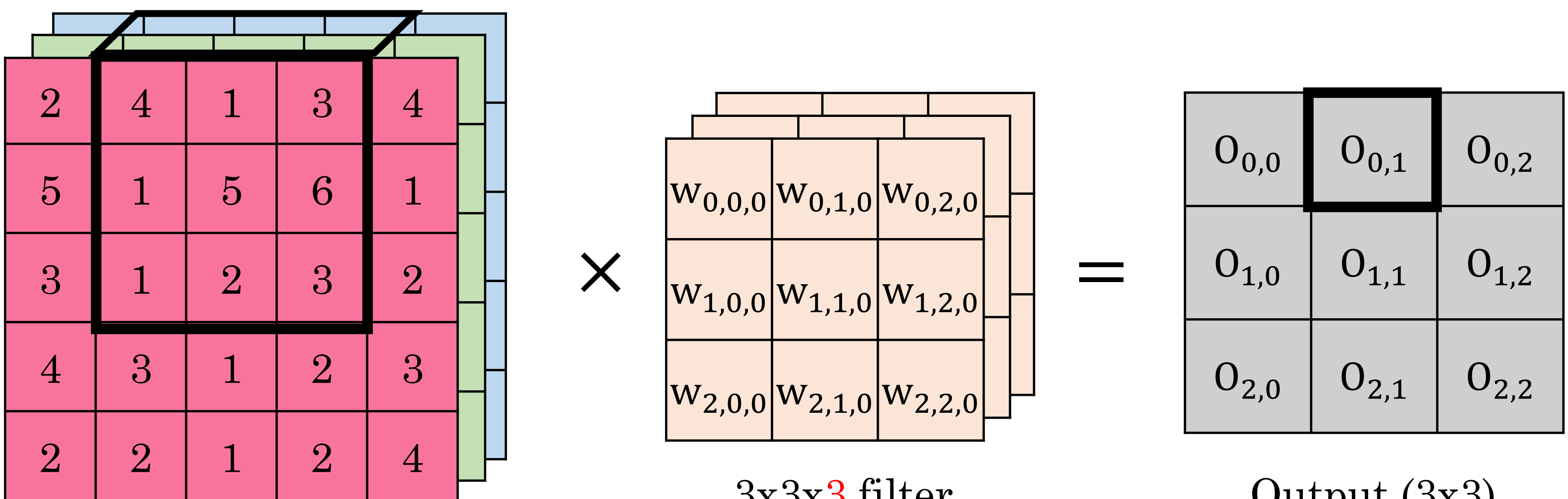
Convolution Layer

- Convolutional Operation
  - For 3 dimensional data
    - Ex. (R,G,B) of a image data
    - Filter is expanded to the same size of input channel
    - Element-wise multiplication for (i, j, k), make one scalar output



Convolution Layer

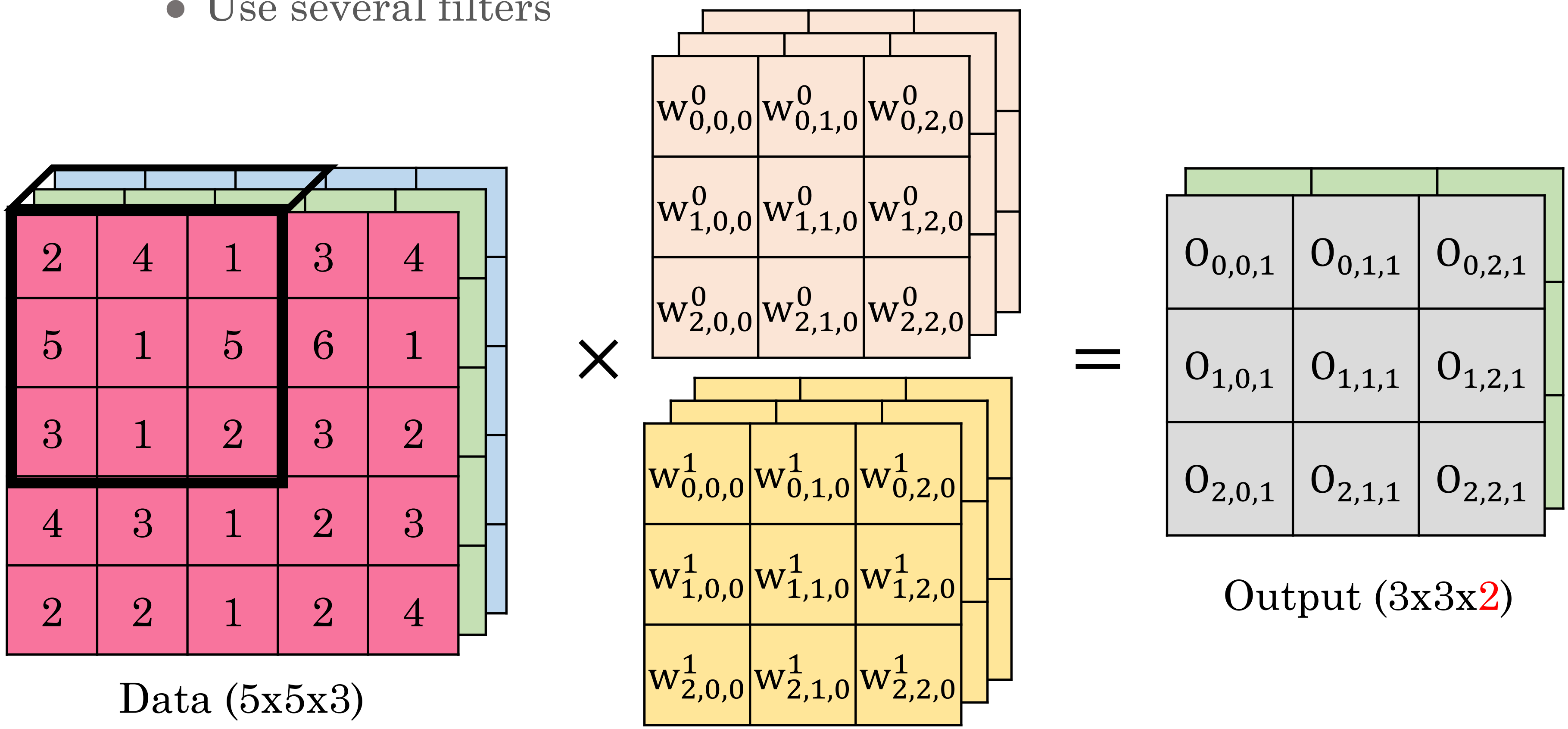
- Convolutional Operation
  - For 3 dimensional data
    - Ex. (R,G,B) of a image data
    - Filter is expanded to the same size of input channel
    - Element-wise multiplication for (i, j, k), make one scalar output



Data (5x5x3) 3x3x3 filter Output (3x3)

Convolution Layer

- Convolutional Operation
  - For 3 dimensional data
    - To extract multiple patterns
    - Use several filters



Data (5x5x3)

Output (3x3x2)

Two 3x3x3 filters

Convolution Layer

- Convolutional filter
  - Filter size
    - Receptive field are varying depends on the filter size
      - Aggregating window size
    - (1x1), (3x3), (5x5) are commonly-used
  - Filter numbers
    - Normally many filters are in advantages
    - Each filters are trained to extract different patters
    - For example
      - Filter 1, extracts horizontal lines
      - Filter 2, extracts vertical lines
      - Filter 3, extracts circular lines

CNN

- Padding
  - Maintain original data size
  - Reduce edge side data loss

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

Data (5x5) with zero padding

×

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$

3x3 filter

=

$O_{0,0}$	$O_{0,1}$	$O_{0,2}$	$O_{0,3}$	$O_{0,4}$
$O_{1,0}$	$O_{1,1}$	$O_{1,2}$	$O_{1,3}$	$O_{1,4}$
$O_{2,0}$	$O_{2,1}$	$O_{2,2}$	$O_{2,3}$	$O_{2,4}$
$O_{3,0}$	$O_{3,1}$	$O_{3,2}$	$O_{2,3}$	$O_{2,4}$
$O_{4,0}$	$O_{4,1}$	$O_{4,2}$	$O_{2,3}$	$O_{2,4}$

Output (5x5)

CNN

SCONE Lab.

- Pooling
  - Max pooling
  - Average pooling
  - Min pooling
    - Reduce the size of activation map / Reduce model size
    - Drop less important feature

13	10	1	24
7	8	5	4
21	11	5	9
3	19	34	7

Activation map

13	24
21	34

Max pooling

10	9
14	14

Avg pooling

7	1
3	5

Min pooling

CNN

- CNN notations
  - Filter
    - Parameter matrix or a tensor
  - Activation map
    - Output features of convolutional operation
  - Channel
    - Dimension of data or activation map
    - For color image data = (R,G,B) three channel
  - Padding
  - Pooling
  - Stride
    - Moving step size of a filter

CNN

- Stride
  - Moving step size of a filter

Stride = 1

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

Stride = 2

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

CNN

- Activation map size
  - W: image width
  - F: filter width
  - P: padding size
  - S: stide size
    - $\frac{W-F+2\times P}{S}+1$

$$\frac{5-3+2\times 0}{1}+1=3$$

$$\begin{bmatrix} 2 & 4 & 1 & 3 & 4 \\ 5 & 1 & 5 & 6 & 1 \\ 3 & 1 & 2 & 3 & 2 \\ 4 & 3 & 1 & 2 & 3 \\ 2 & 2 & 1 & 2 & 4 \end{bmatrix} \times \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} \\ w_{1,0} & w_{1,1} & w_{1,2} \\ w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix} = \begin{bmatrix} O_{0,0} & O_{0,1} & O_{0,2} \\ O_{1,0} & O_{1,1} & O_{1,2} \\ O_{2,0} & O_{2,1} & O_{2,2} \end{bmatrix}$$

Data (5x5) | 3x3 filter | Output (3x3)

CNN

- Activation map size
  - W: image width
  - F: filter width
  - P: padding size
  - S: stide size
    - $\frac{W-F+2\times P}{S}+1$

Padding =1
Stride = 2

$$\frac{5-3+2\times 1}{2}+1=3$$

0	0	0	0	0	0	0
0	2	4	1	3	4	0
0	5	1	5	6	1	0
0	3	1	2	3	2	0
0	4	3	1	2	3	0
0	2	2	1	2	4	0
0	0	0	0	0	0	0

Data (5x5)

×

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$

3x3 filter

=

$O_{0,0}$	$O_{0,1}$	$O_{0,2}$
$O_{1,0}$	$O_{1,1}$	$O_{1,2}$
$O_{2,0}$	$O_{2,1}$	$O_{2,2}$

Output (3x3)

Compare to MLP

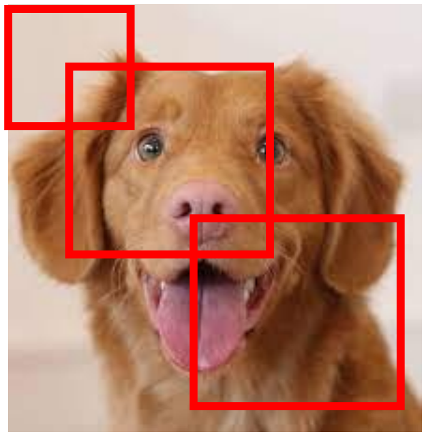
- MLP requires input shape of 1D vector
  - When dealing image data, need flattening
  - MNIST data : $(28 \times 28 \times 3)$ image to $(1 \times 2{,}352)$ vector
- MLP requires to many parameters
  - $W^l \in \mathbb{R}^{d_1 \times d_2}$
    - Needs huge number of parameters with large image
    - For $(256 \times 256 \times 3)$ image
      - The first layer parameters are size of $(196{,}608 \times d_2)$
  - Not depend on the input size for CNN
    - Parameter size = filter size

- CNN can reduce parameter size a lot

Compare to MLP

- Neighboring Feature aggregation
  - Some neighboring features have their own pattern
  - CNN works better if data has their neighboring context
    - Image
    - Sentence (word order is important)
      - I am a boy and you are a girl
      - A I you and are girl boy a am

Image



CNN

MLP

Limitations

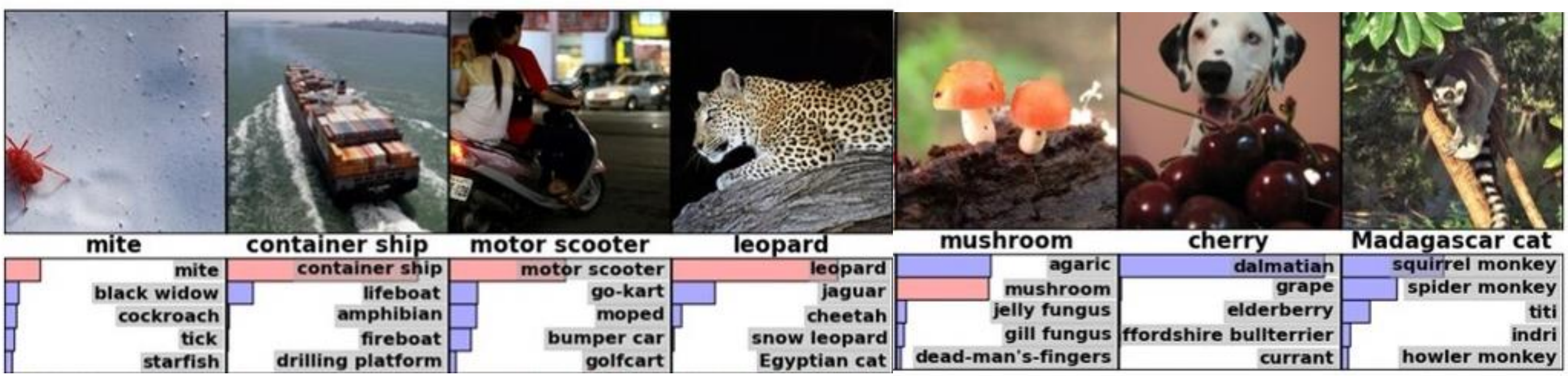
- Restricted receptive field
  - Convolutional operation is an advantage of CNN
    - Aggregate neighbor features or data together
  - But the receptive field is fixed by the size of filter
  - Out of receptive field data are not considered together
    - Example, For sentence data

    "KENTECH is an energy-specialized college that focuses on energy science and technology located in Naju."
    - If the receptive field is small
      - Cannot aggregate 'KENTECH' and 'Naju' together
    - Reason why 'Attention mechanism' proposed

- Nevertheless, CNN has made unprecedent advances in performance in computer vision.

ILSVRC

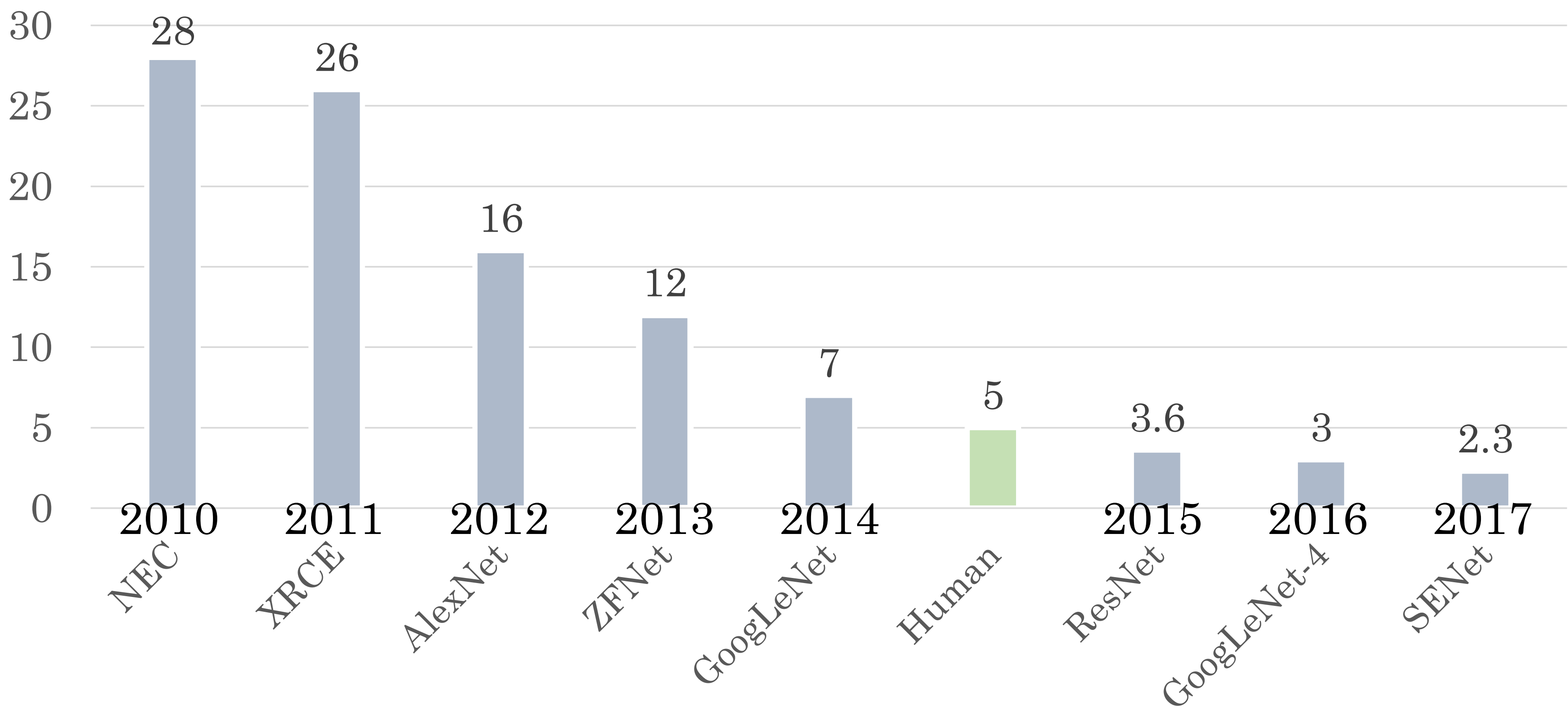
- ImageNet Large Scale Visual Recognition Challenge
  - Image classification task
    - Classify given images out of 1,000 categories
  - Train/Valid/Test = 1M/50k/150k
  - 2010 to 2017
  - During the ImageNet challenges
    - Hundreds of papers published
    - Huge breakthrough in computer vision and AI
    - Higher score than human



[1] https://image-net.org/challenges/LSVRC/

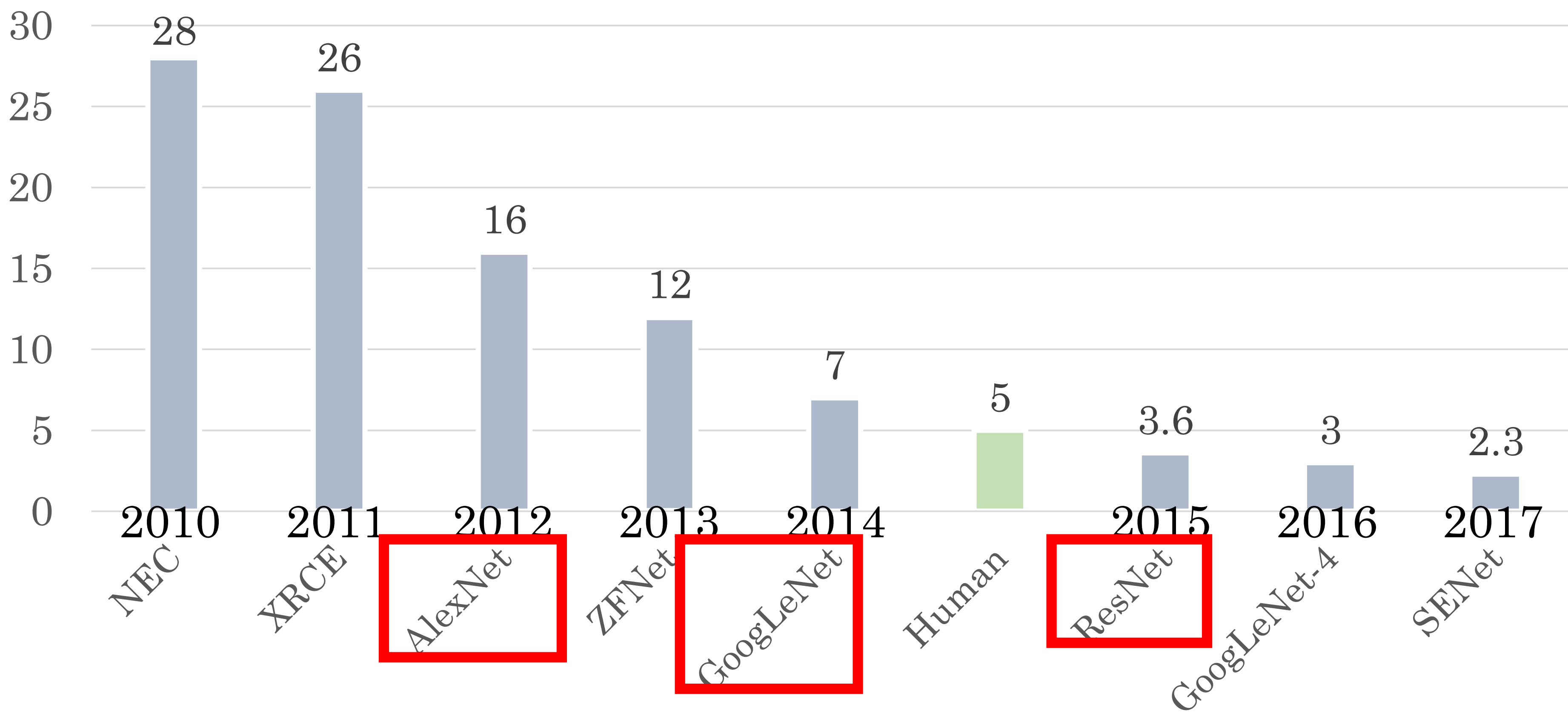
ILSVRC

- ImageNet Large Scale Visual Recognition Challenge
  - Accuracy loss for the winner models



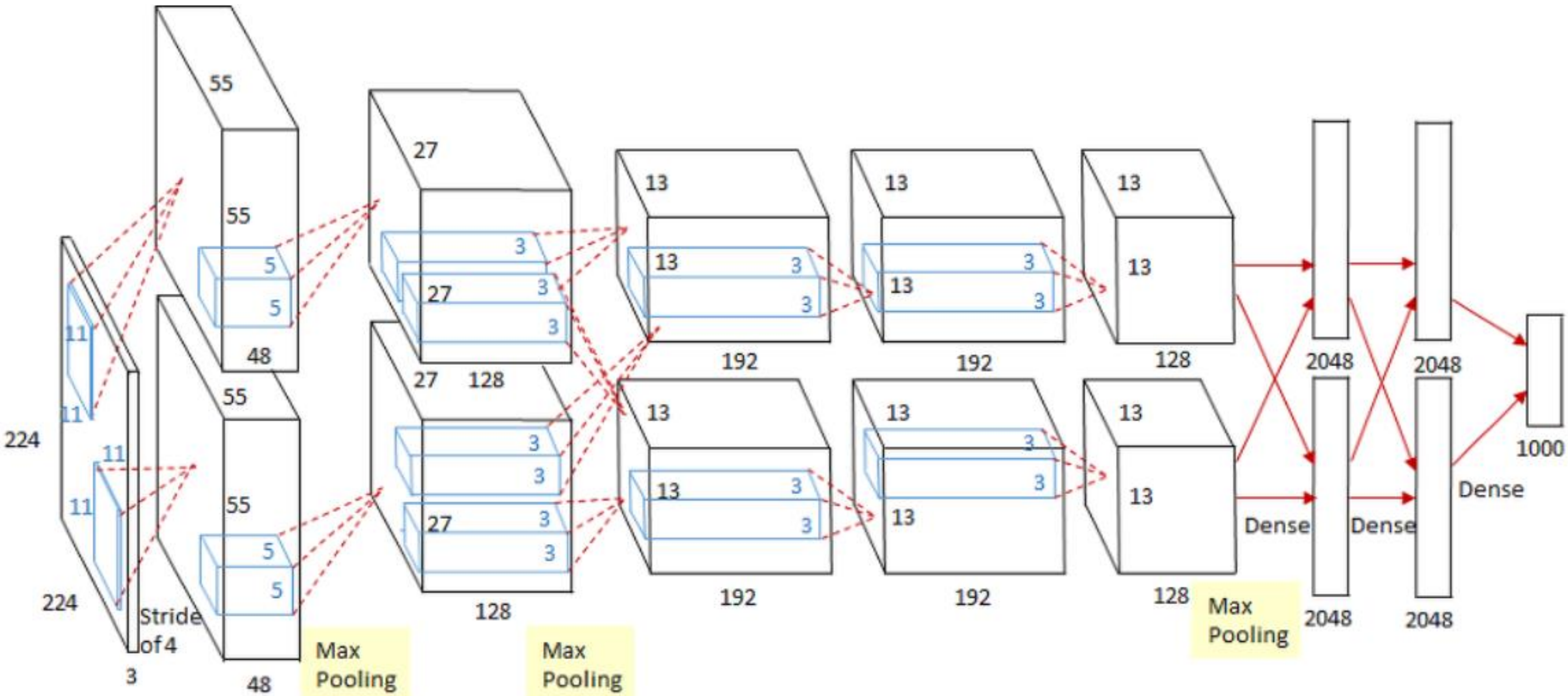
ILSVRC

- ImageNet Large Scale Visual Recognition Challenge
  - Accuracy loss for the winner models



ILSVRC

- AlexNet
  - Input: (224x224x3)
  - Output: (1x1000)
  - 5 convolution (CNN) and 2 fully connected (MLP)



[2] K, Alex, G. Hinton. NIPS 2012

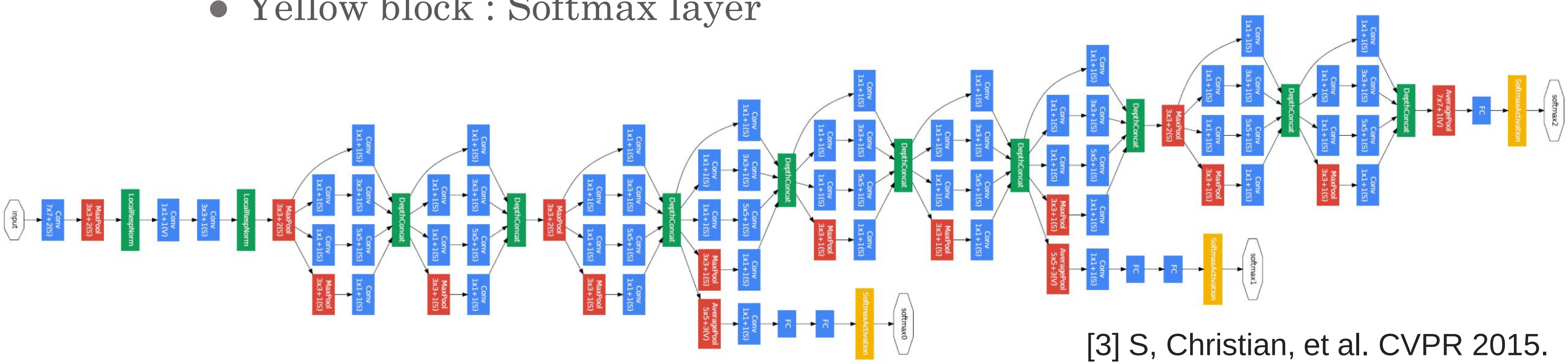
ILSVRC

- AlexNet

	Filter	Stride	Padding	Input	Output
Conv1	11x11x3 (96)	4	0	224x224x3	55x55x96
Maxpool1	3x3	2	-	55x55x96	27x27x96
Norm1	-	-	-	27x27x96	27x27x96
Conv2	5x5x96 (256)	1	2	27x27x96	27x27x256
Maxpool2	3x3	2	-	27x27x256	13x13x256
Norm2	-	-	-	13x13x256	13x13x256
Conv3	3x3x256 (348)	1	1	13x13x256	13x13x384
Conv4	3x3x384 (348)	1	1	13x13x384	13x13x384
Conv5	3x3x384 (256)	1	1	13x13x384	13x13x256
Maxpool3	3x3	2	-	13x13x256	6x6x256
FC1	-	-	-	6x6x256	4096
FC2	-	-	-	4096	1000

ILSVRC

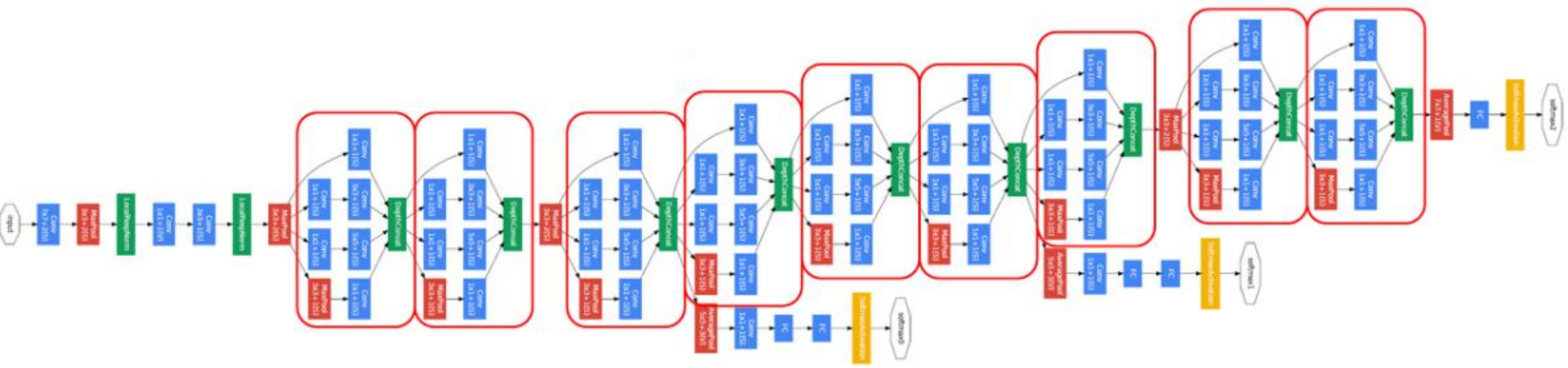
- GoogLeNet
  - a.k.a. Inception Net
  - Inception ← The movie name!
    - It is an SF movie we go deeper and deeper into a dream
    - The authors want to make deeper CNN layers
  - 22 Layers
    - Blue block : Convolution layer with filter
    - Red block : Pooling layer
    - Green block : Concatenation
    - Yellow block : Softmax layer



[3] S, Christian, et al. CVPR 2015.

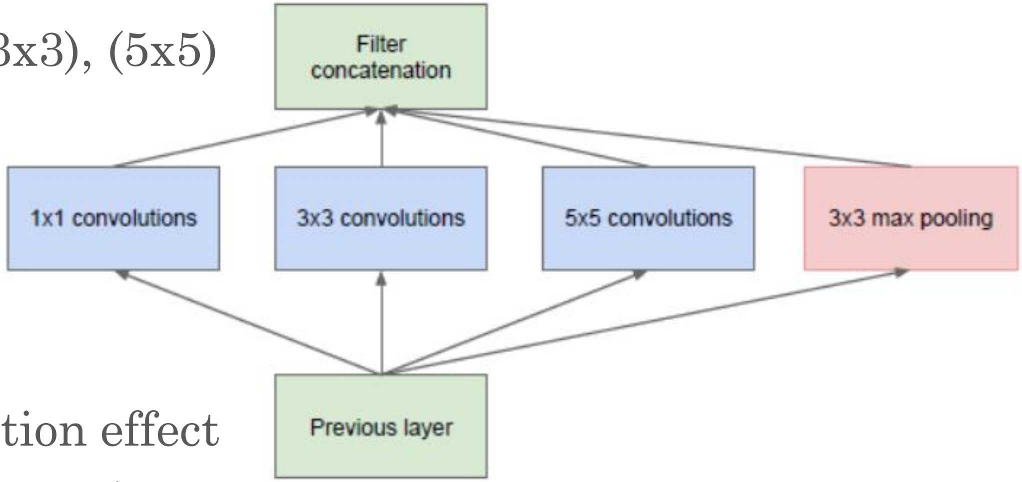
ILSVRC

- GoogLeNet
  - Inception module
    - 8 inception modules in GoogLeNet
    - Apply (1x1), (3x3), (5x5) filters and max pooling
    - Then concatenate the activation maps



ILSVRC

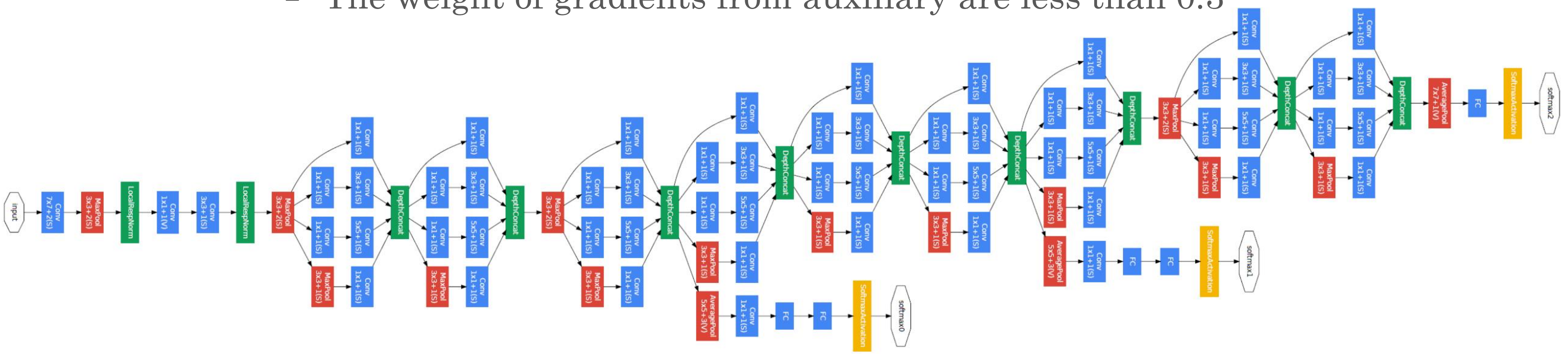
- GoogLeNet
  - Inception module overview
    - The concatenated output has
    - All information of (1x1), (3x3), (5x5)



  - What is the (1x1) filter for?
    - (1x1) filter has no convolution effect
    - But apply convolutional operation,
      - We can adjust activation map channel
    - If input data is size of (100x100x30)
      - We apply 10 (1x1x30) filters
      - Then the output shape is (100x100x10)
      - Still had channel-wise convolution effect

ILSVRC

- GoogLeNet
  - x12 less parameters than AlexNet
    - Use less number of filters
  - There are three outputs for the model
    - Called auxiliary classification
  - To prevent gradient vanishing issue
    - Make intermediate prediction and calculate loss
    - Add gradient
    - Our main goal is the last prediction performance
      - The weight of gradients from auxiliary are less than 0.3



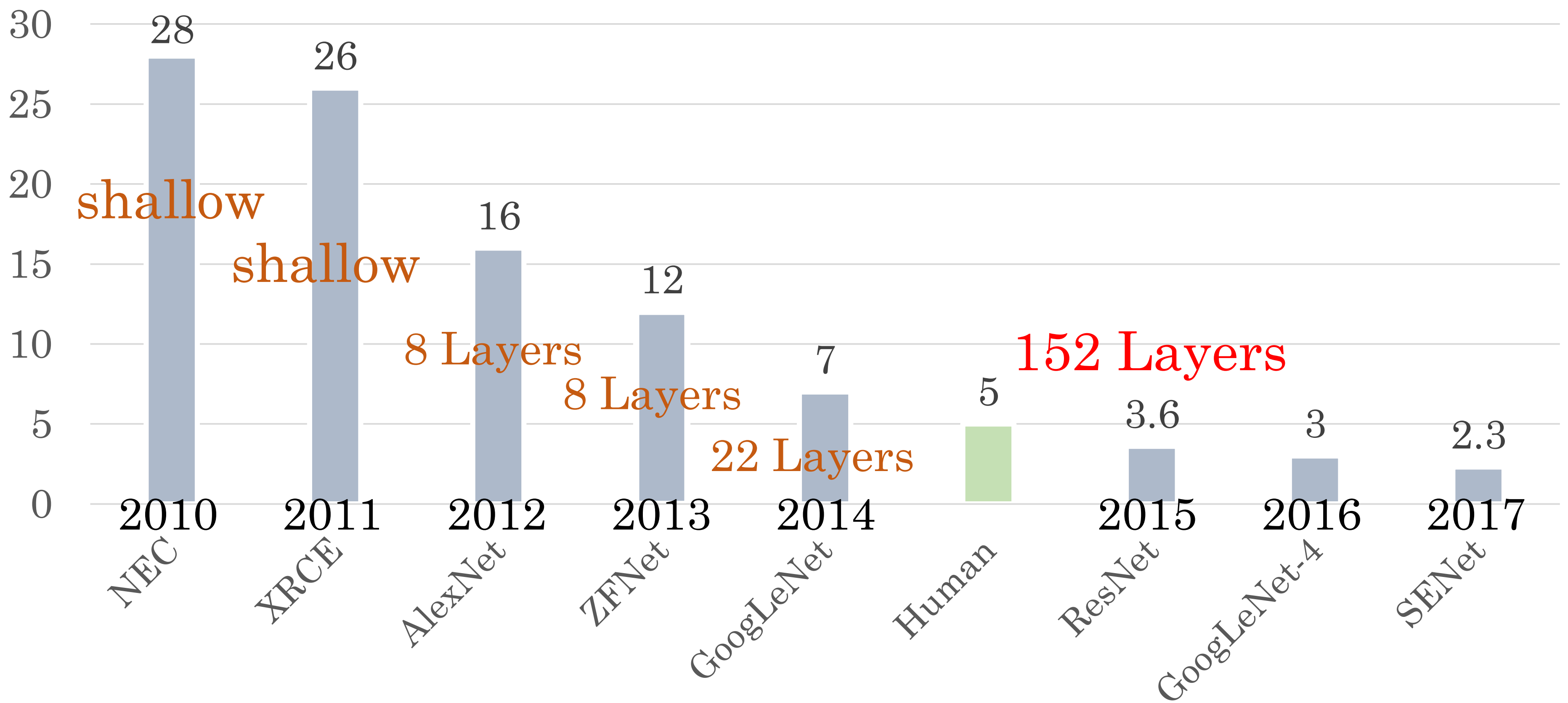
ILSVRC

- GoogLeNet

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

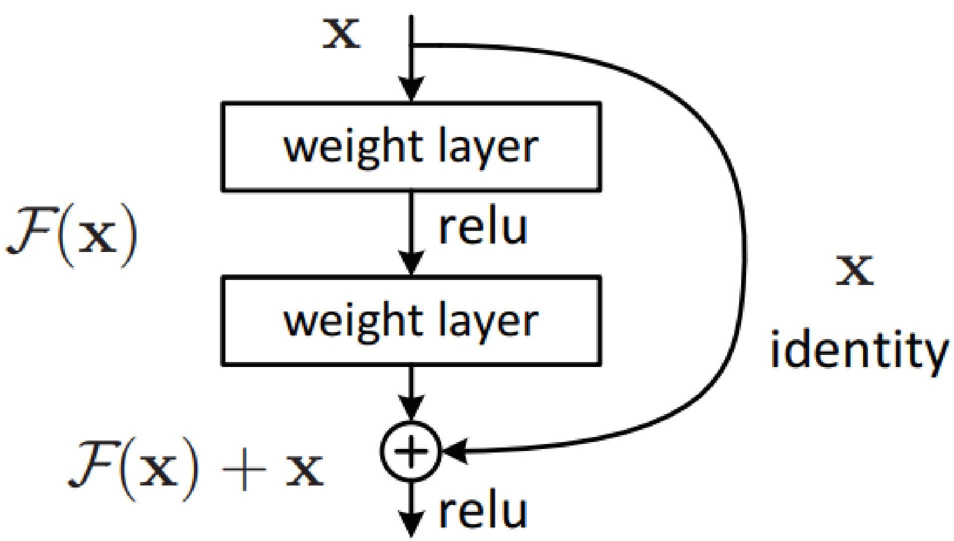
ILSVRC

- ResNet
  - Residual connection
  - Reduce gradient vanishing issue and make deeper layer



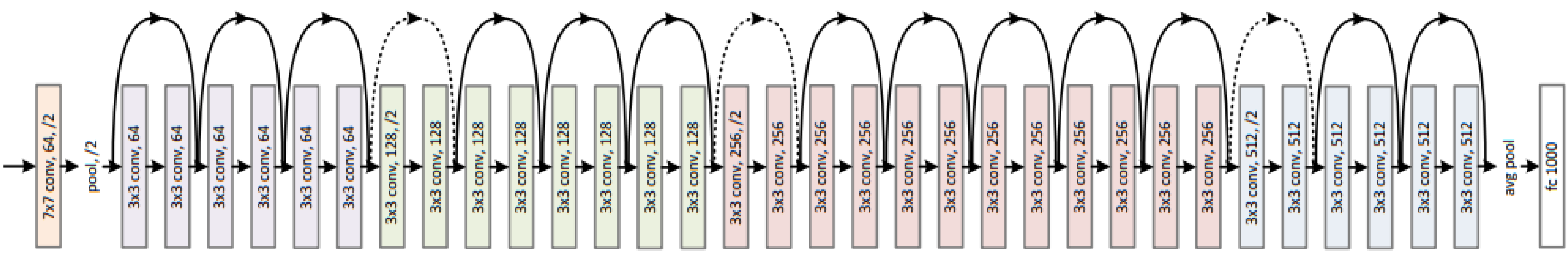
[4] He, Kaiming, et al. CVPR 2016.

- ResNet
  - Residual block
    - Add identity value to layer output
  - Prevent gradient vanishing
    - $(F(x) + x)' = F'(x) + 1$
    - Gradients maintain at least 1



ILSVRC

- ResNet
  - Model overview of 34 layered ResNet
  - Boxes indicates convolution layer
  - Residual connections for every two layers
    - When the addition is possible (same size)
    - Dashed lines are size mis-match
      - Apply (1x1) filter to adjust feature map size



ILSVRC

- ResNet

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				

Reference

[1] https://image-net.org/challenges/LSVRC/

[2] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012).

[3] Szegedy, Christian, et al. "Going deeper with convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.[4] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958.

[4] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.