LECTURE 11-2

CNN
CASE STUDY

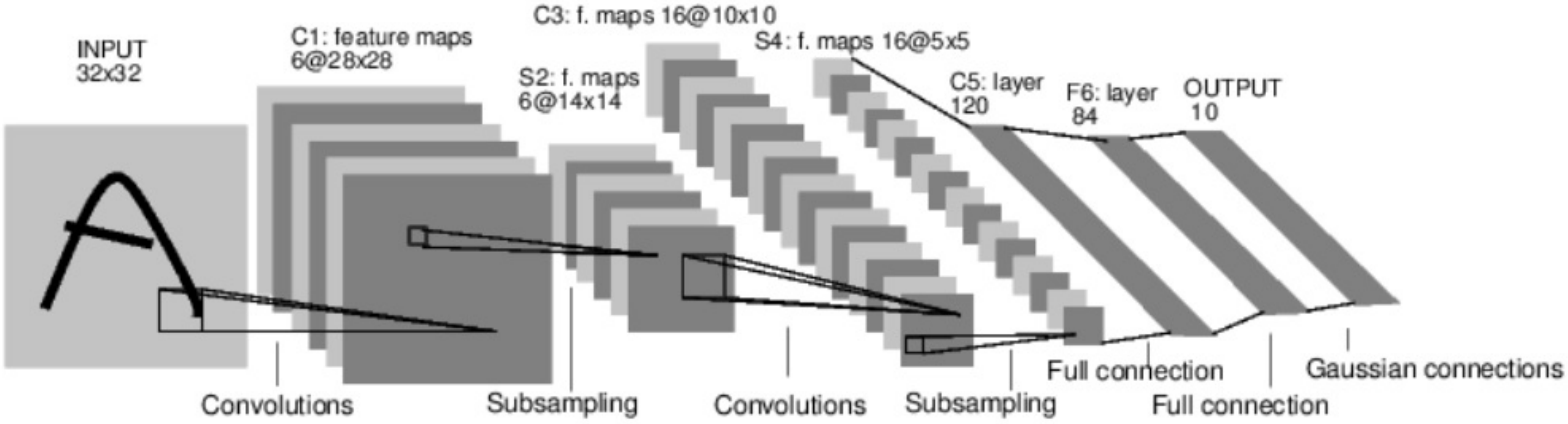
Sung Kim <hunkim+ml@gmail.com>
http://hunkim.github.io/ml

NAVER | Clova

Convolutional Neural Networks

Case Study : LeNet-5

LeCun et al., 1998



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e architecture is [CONV-POOL-CONV-POOL-CONV-FC]

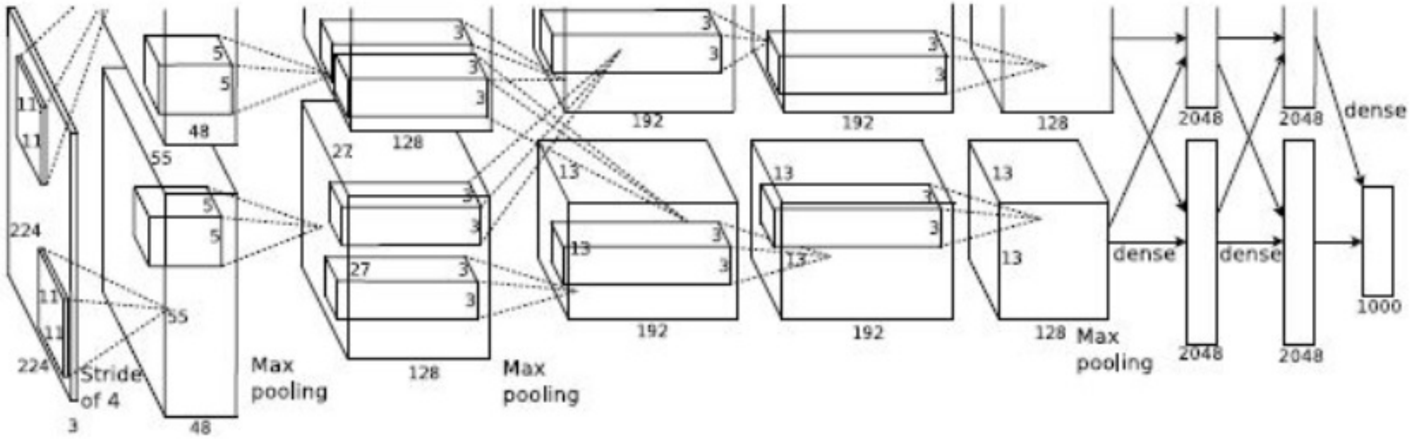
Case Study : AlexNet

Krizhevsky et al. 2012

Input:
227 X 227 X 3 images

First Layer (CONV1):
96 11×11 filters applied at stride 4

-> Output Volume [55×55×96]



Q.
What is the total number of parameters in this layer?

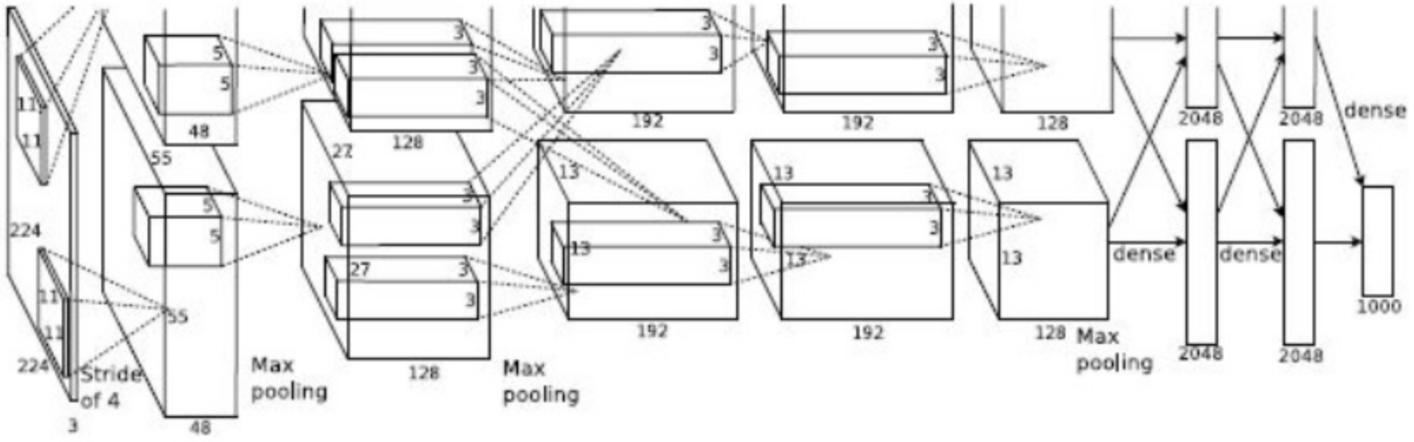
Case Study : AlexNet

Krizhevsky et al. 2012

Input:
227 X 227 X 3 images

First Layer (CONV1):
96 11x11 filters applied at stride 4

->



Q.
What is the output volume size?
Hint: (227-11)/4+1=55

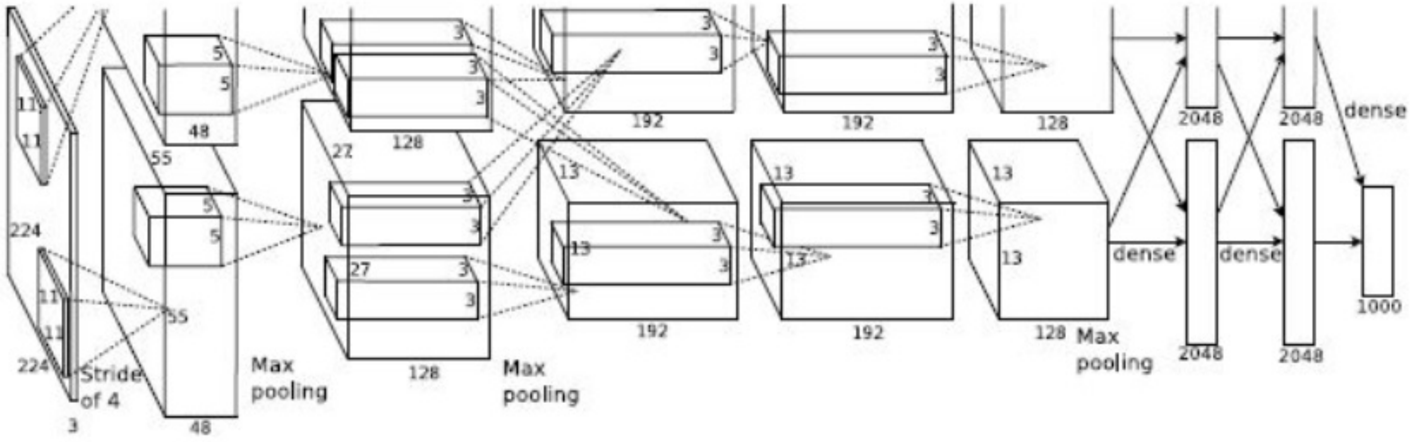
Case Study : AlexNet

Krizhevsky et al. 2012

Input:
227 X 227 X 3 images

First Layer (CONV1):
96 11x11 filters applied at stride 4

-> Output Volume [55 x 55 x 96]
Parameters: (11 x 11 x 3) x 96 = 35k



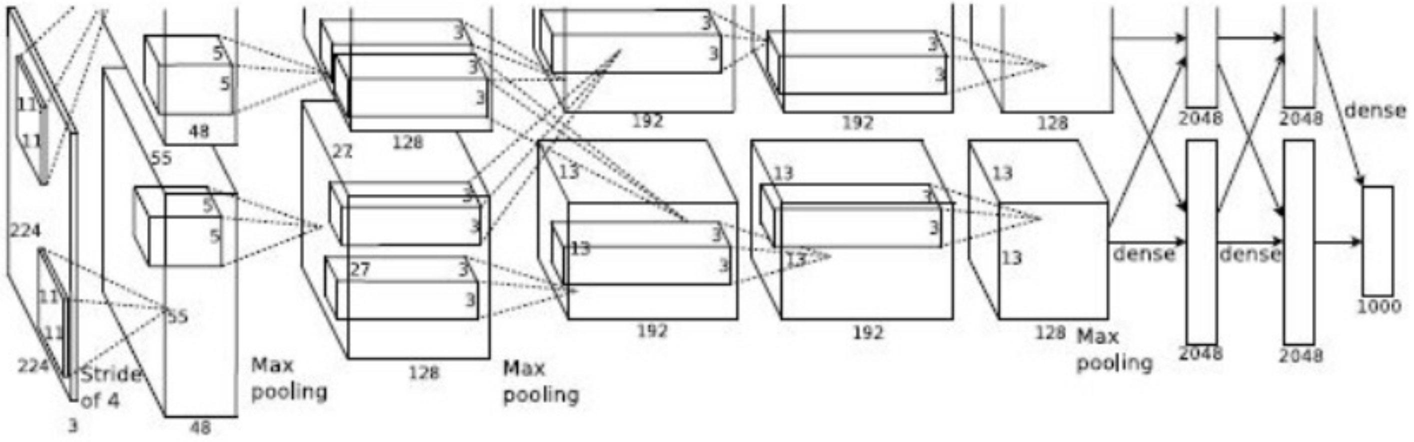
Case Study : AlexNet

Krizhevsky et al. 2012

Input: 227 X 227 X 3 images
After CONV1 : 55 X 55 X 96

Second Layer (POOL1):
3X3 filters applied at stride 2

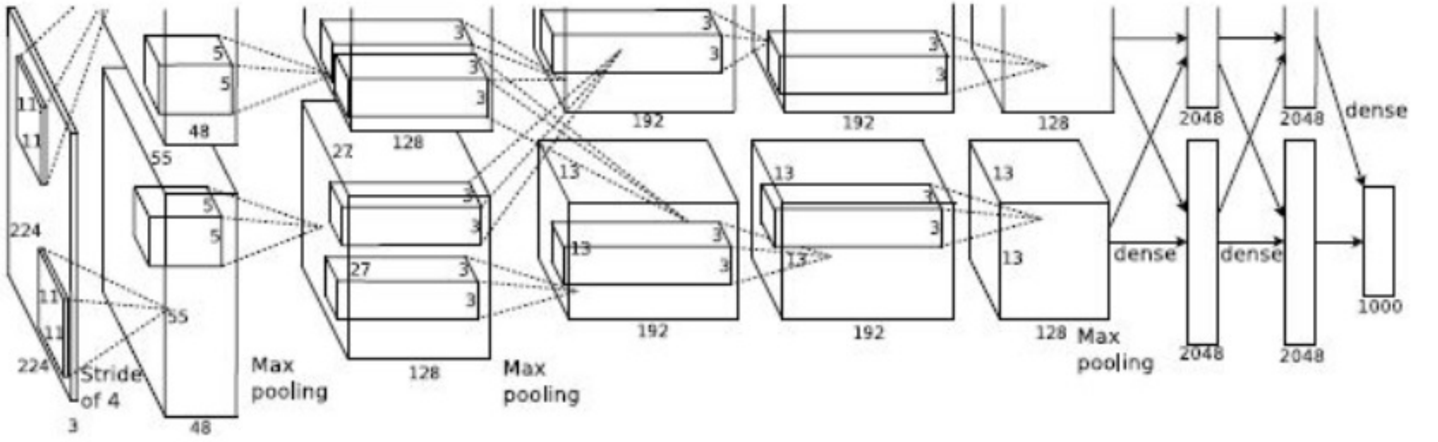
-> Output Volume: 27x27x96
Parameters: 0!



Case Study : AlexNet

Krizhevsky et al. 2012

Input: 227 X 227 X 3 images
After CONV1: 55x55x96
After POOL1: 27x27x96
...

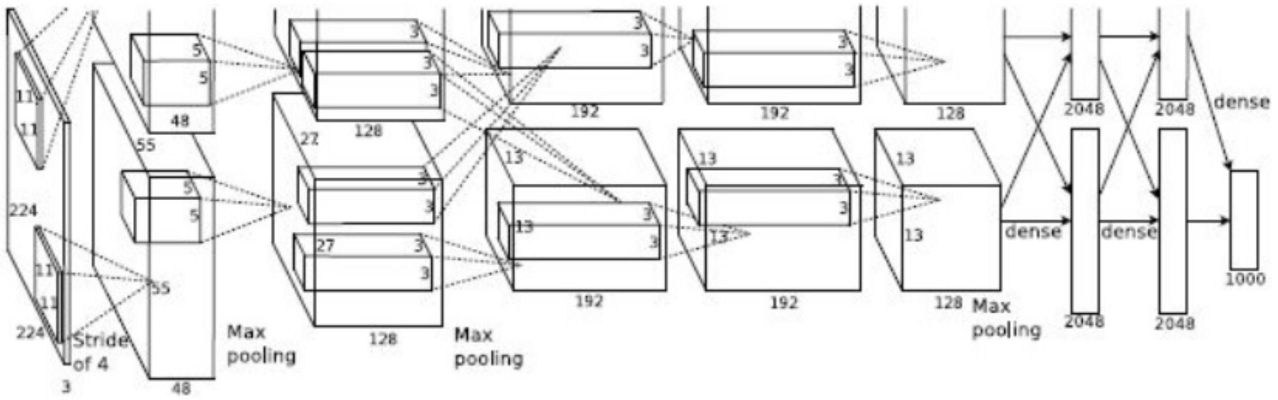


Case Study : AlexNet

Krizhevsky et al. 2012

Full (simplified) AlexNet architecture:

[227 x 227 x 3]	INPUT
[27 x 27 x 96]	CONV1: 96 11 x 11 filters at stride 4, pad 0
[27 x 27 x 96]	MAX POOL1: 3x3 filters at stride 2
[27 x 27 x 96]	NORM1: Normalization layer
[27 x 27 x 256]	CONV2 : 256 5x5 filters at stride 1, pad 2
[13 x 13 x 256]	MAX POOL2: 3x3 filters at stride 2
[13 x 13 x 256]	NORM2: Normalization layer
[13 x 13 x 384]	CONV3: 384 3x3 filters at stride 1, pad 1
[13 x 13 x 384]	CONV4: 384 3x3 filters at stride 1, pad 1
[13 x 13 x 256]	CONV5 : 256 3x3 filters at stride 1, pad 1
[6 x 6 x 256]	MAX POOL3: 3x3 filters at stride 2
[4096]	FC6: 4096 neurons
[4096]	FC7: 4096 neurons
[1000]	FC8: 1000 neurons (class scores)



Case Study : AlexNet

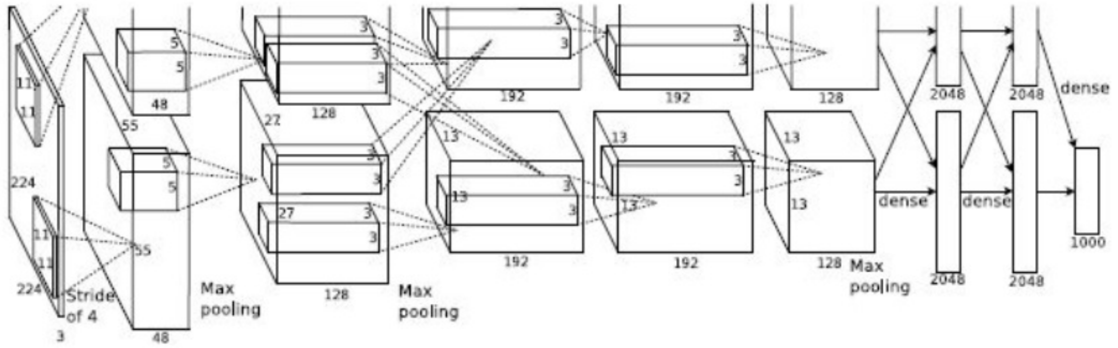
Krizhevsky et al. 2012

Full (simplified) AlexNet architecture:

[227 x 227 x 3]	INPUT
[27 x 27 x 96]	CONV1: 96 11 x 11 filters at stride 4, pad 0
[27 x 27 x 96]	MAX POOL1: 3x3 filters at stride 2
[27 x 27 x 96]	NORM1: Normalization layer
[27 x 27 x 256]	CONV2 : 256 5x5 filters at stride 1, pad 2
[13 x 13 x 256]	MAX POOL2: 3x3 filters at stride 2
[13 x 13 x 256]	NORM2: Normalization layer
[13 x 13 x 384]	CONV3: 384 3x3 filters at stride 1, pad 1
[13 x 13 x 384]	CONV4: 384 3x3 filters at stride 1, pad 1
[13 x 13 x 256]	CONV5 : 256 3x3 filters at stride 1, pad 1
[6 x 6 x 256]	MAX POOL3: 3x3 filters at stride 2
[4096]	FC6: 4096 neurons
[4096]	FC7: 4096 neurons
[1000]	FC8: 1000 neurons (class scores)

Details / Retrospectives:

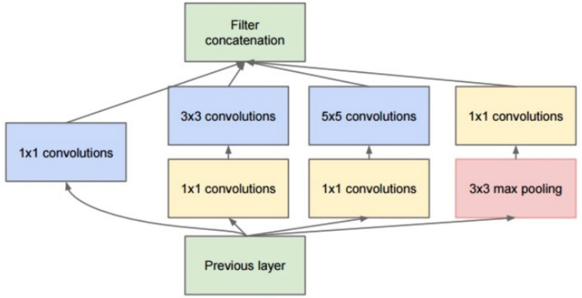
- First use of ReLU
- Used Norm layers (not common anymore)
- Heavy data augmentation
- Dropout 0.5
- Batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%



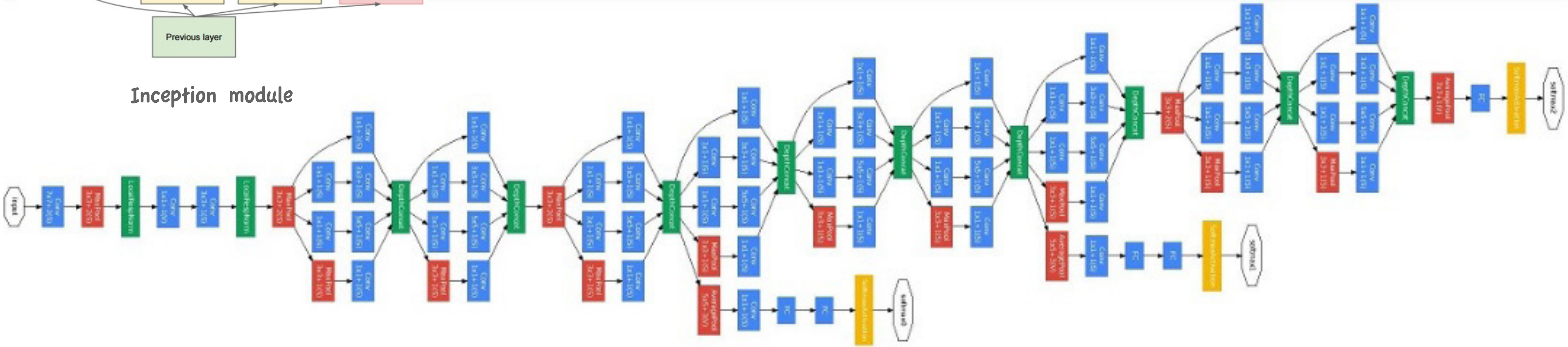
Case Study : GoogLeNet

Szegedy et al., 2014

ILSVRC 2014 winner (6.7% top 5 error)



Inception module



Case Study : ResNet

He et al. 2015

ILSVRC 2015 winner (3.6% top 5 error)

MSRA @ ILSVRC & COCO 2015 Competitoions

- 1st places in all 5 main tracks
  - ImageNet Classification : "Ultra-deep" (quote Yann) 152-layer nets
  - ImageNet Detection : 16% better than 2nd
  - ImageNet Localization : 27% better tnan 2nd
  - COCO Detection : 11% better than 2nd
  - COCO Segmentation : 12% better than 2nd

*improvements are relative numbers

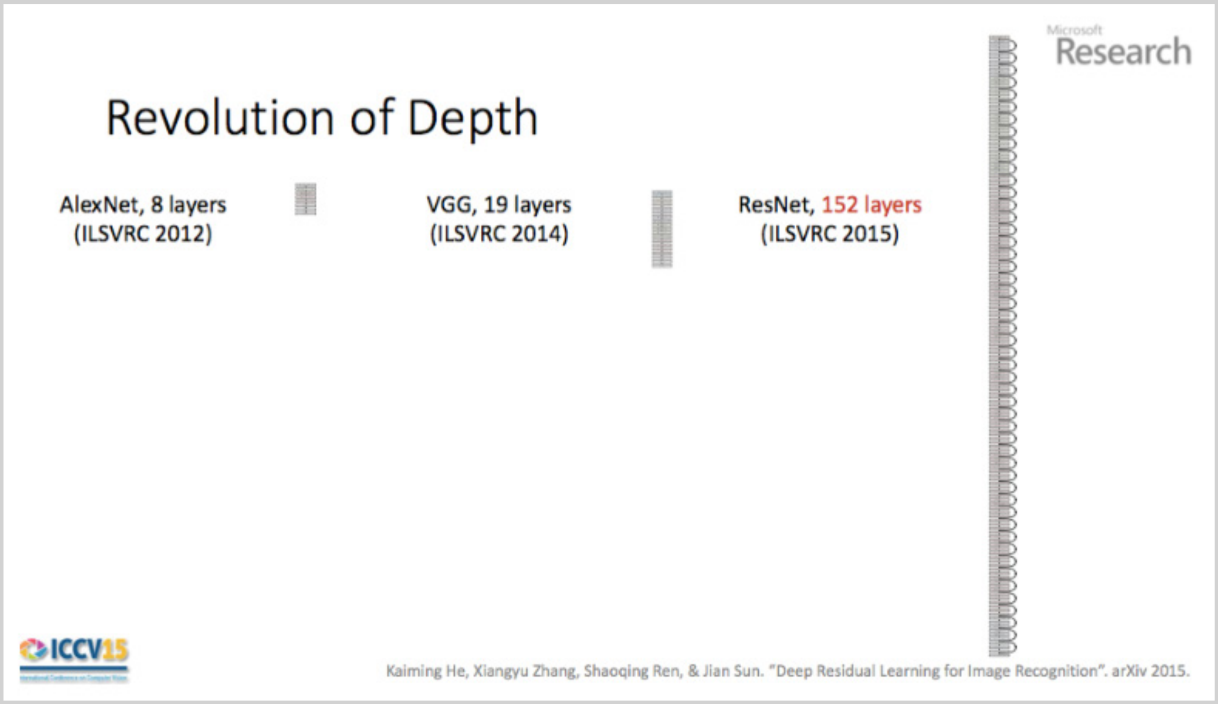
Kaiming, Xiangyu Zhang, Shaoqing Ren, & Jian Sun, "Deep Residual Learning for Image Recognition". arXiv 2015.

Slide from Kaiming He's recent presentation http://www.youtube.com/warch?v=1PGLj-uKT1w

Case Study : ResNet

He et al. 2015

ILSVRC 2015 winner (3.6% top 5 error)

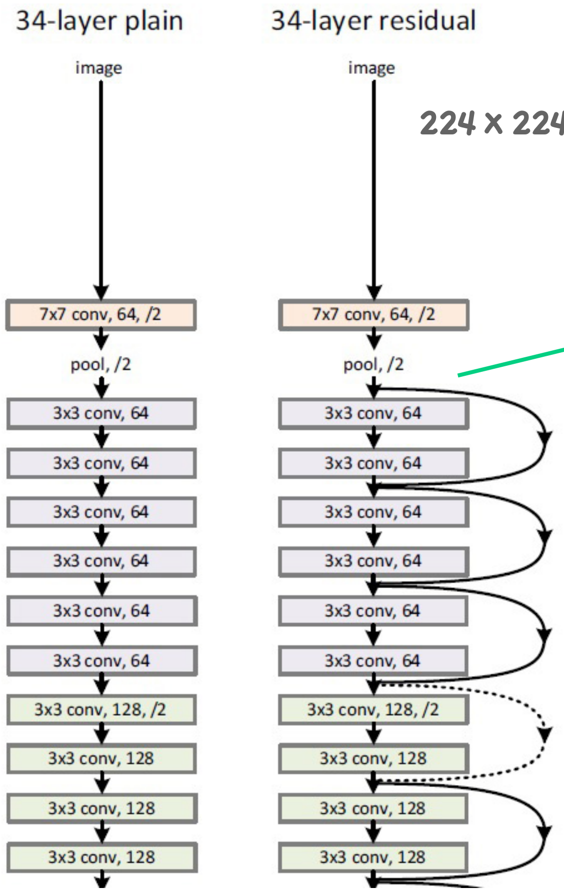


- 2-3 weeks of training on 8 GPU machine
- at runtime : faster than a VGGNet! (even though it has 8x more layers)

Kaiming, Xiangyu Zhang, Shaoqing Ren, & Jian Sun, "Deep Residual Learning for Image Recognition". arXiv 2015.
Slide from Kaiming He's recent presentation

Case Study : ResNet

He et al., 2015

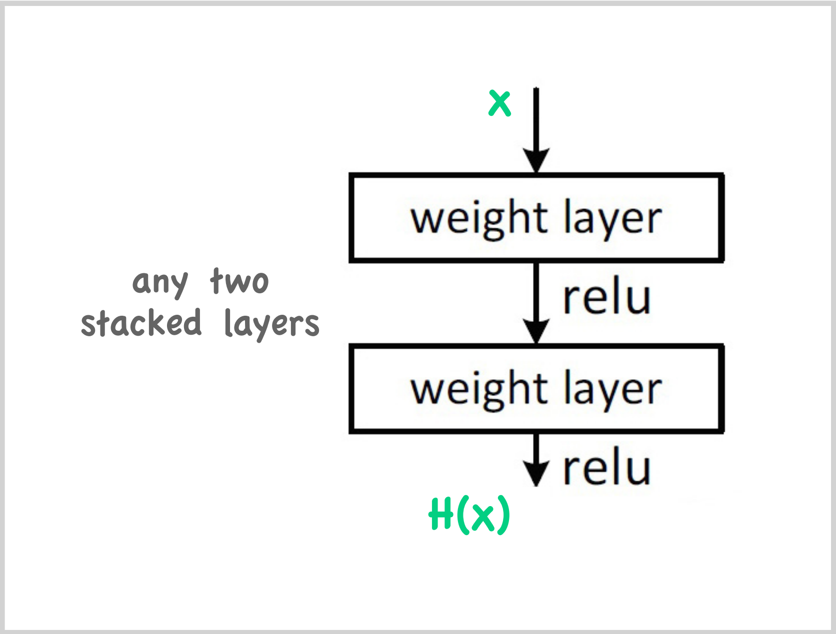


224 x 224 x 3

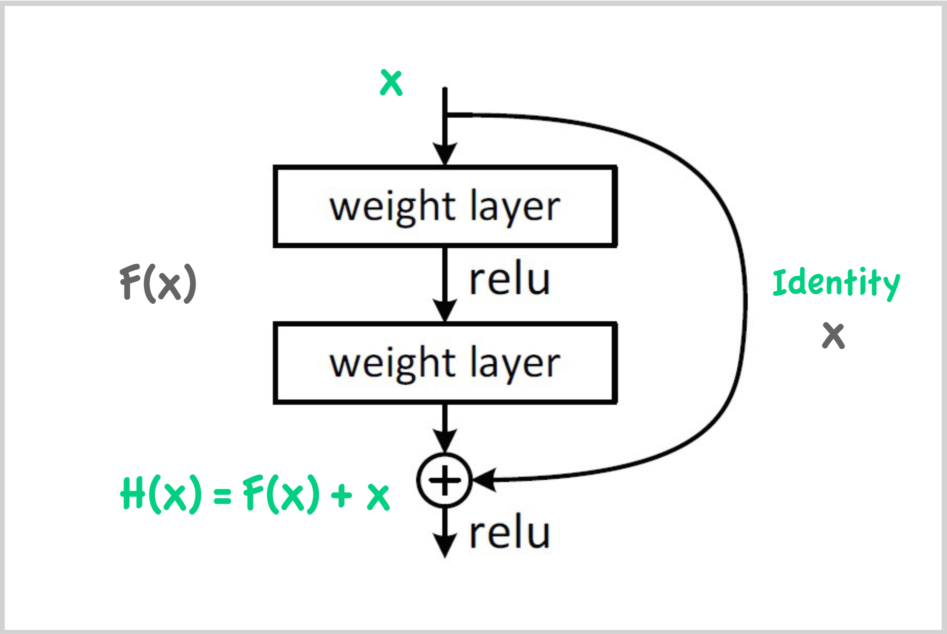
Spatial
dimension
only 56x56!

Case Study : ResNet

He et al., 2015



< Plaint net >



< Residual net >

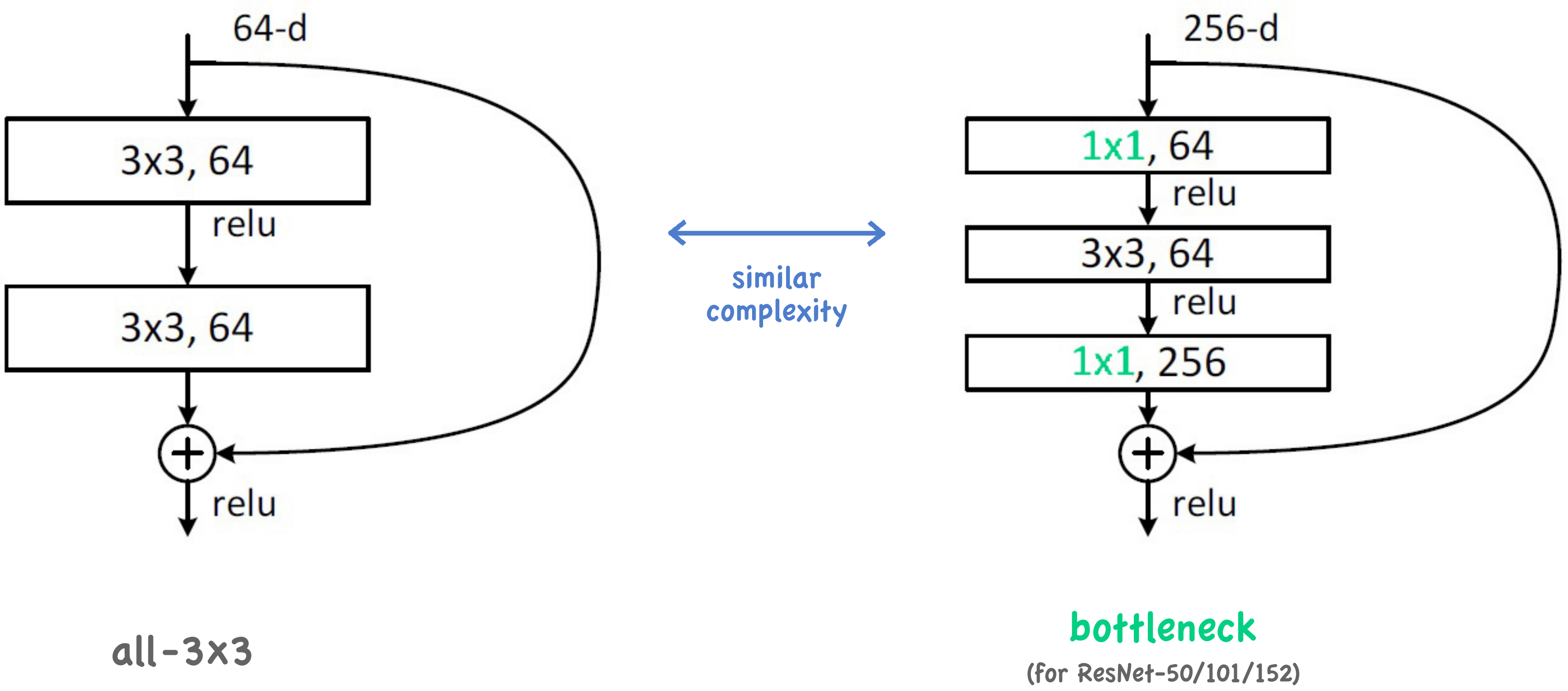
Case Study : ResNet

He et al. 2015

- Batch normalization after every CONV layer
- Xavier/2 initialization from He et al.
- SGD + Momentum (0.9)
- Learning rate : 0.1, divided by 10 when validation error plateaus
- Mini-batch size 256
- Weight decay of 1e-5
- No dropout used

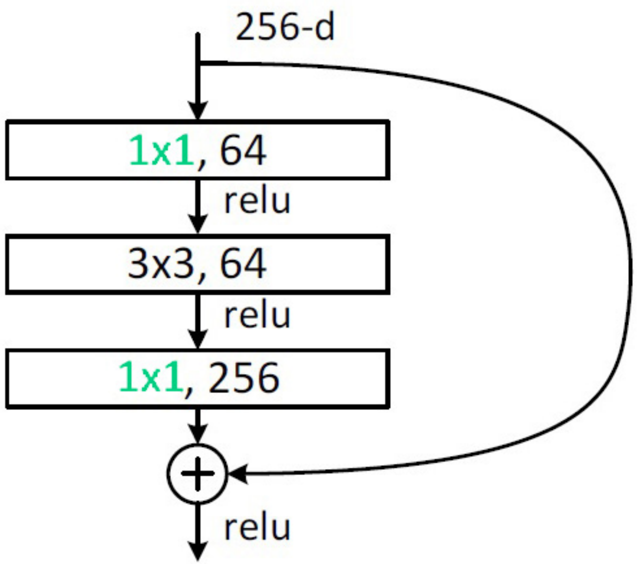
Case Study : ResNet

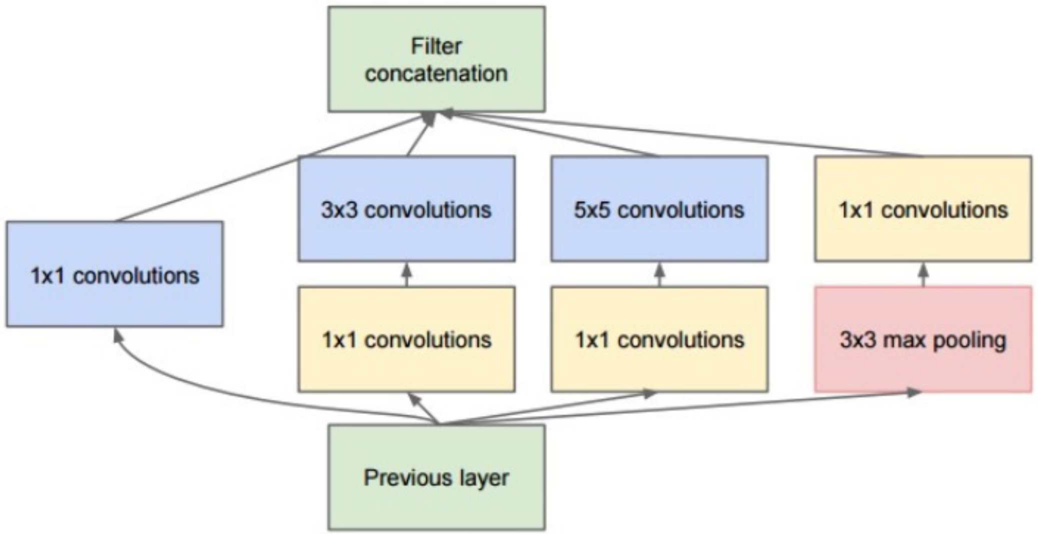
He et al., 2015



Case Study : ResNet

He et al., 2015

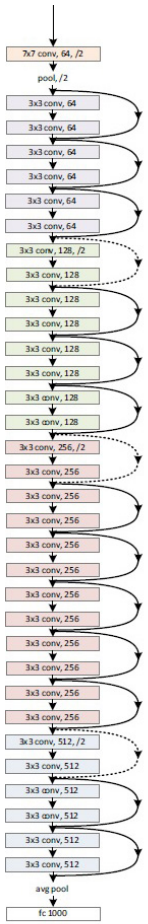




(this trick is also used in GoogLeNet)

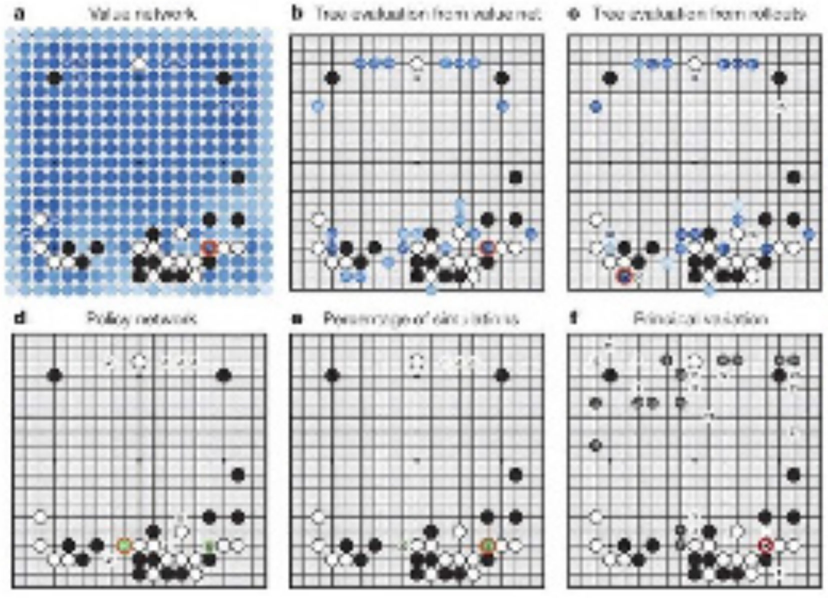
Case Study : ResNet

He et al., 2015



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				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Case Study Bonus : DeepMind's AlphaGo



The input to the policy network is a 19x19x48 image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23x23 Image, then convolves k filters of kernel size 5x5 with stride 1 with the input image and applies a rectifier nonlinearity. Each of the subsequent hidden layers 2 to 12 zero pads the respective previous hidden layer into a 21x21 image, then convolves k filters of kernel size 3x3 with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size 1x1 with stride 1, with a different bias for each position, and applies a softmax function. The match version of AlphaGo used k=192 filters; Fig.2b and Extended Data Table 3 additionally show the results of training with k=128,256 and 384filters.

Policy Network :

[19x19x48] Input

CONV1 : 192x5x5 filters, stride1, pad2 => [19x19x192]

CONV2...12 : 192x3x3 filters, stride1, pad1 => [19x19x192]

CONV : 1x1 filter, stride1, pad0 => [19x19] (probability map of promising moves)

Convolutional Neural Networks for Sentence Classification

Yoon Kim., 2014

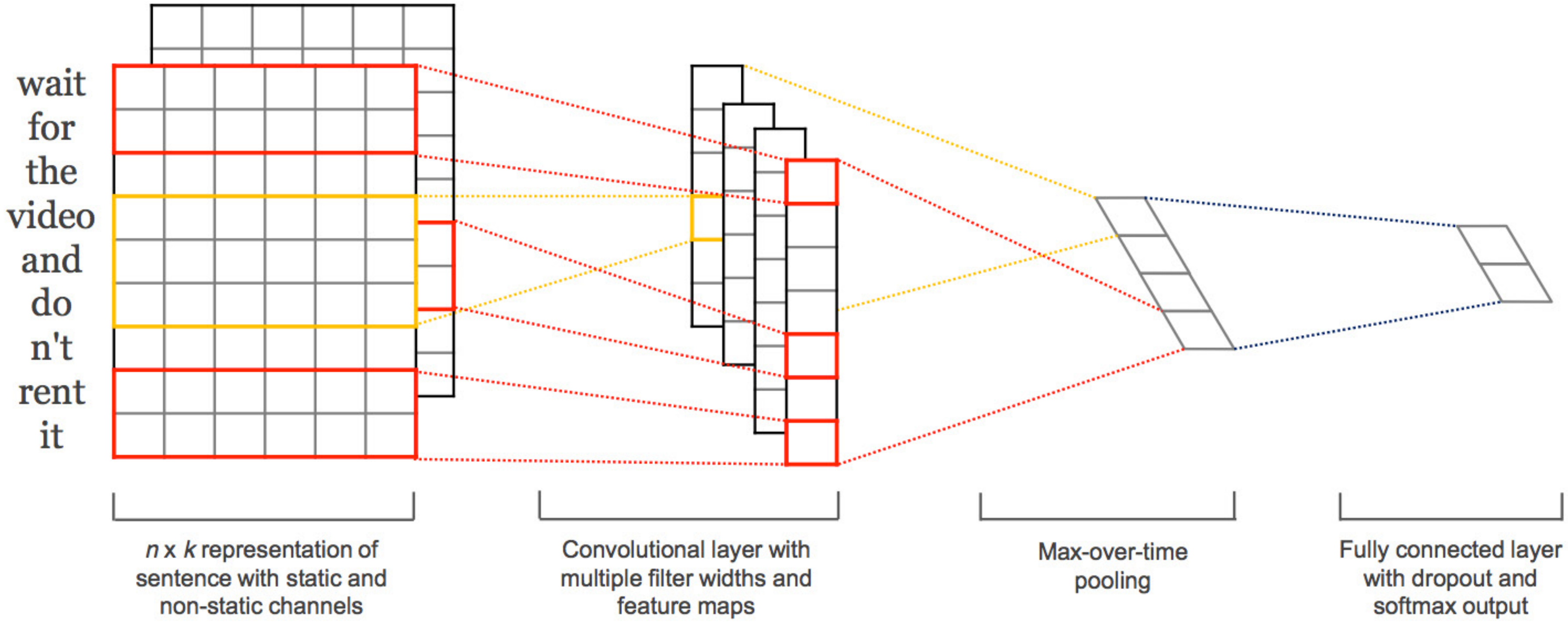


Figure 1: Model architecture with two channels for an example sentence.

NEXT LECTURE

RECURRENT NEURAL NETS (RNN)