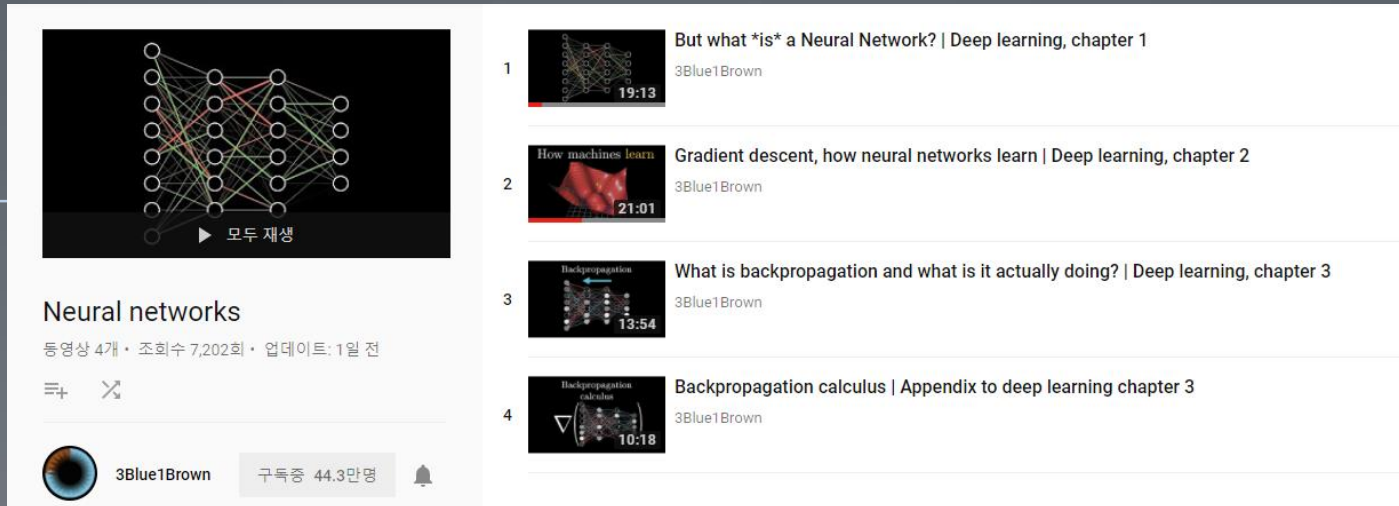

아티스트를 위한 머신러닝 & 딥러닝

텐서플로를 활용한 딥러닝 #4

서울대학교 & V.DO / 김대식

Recap

What is a Neural network ?



Neural networks

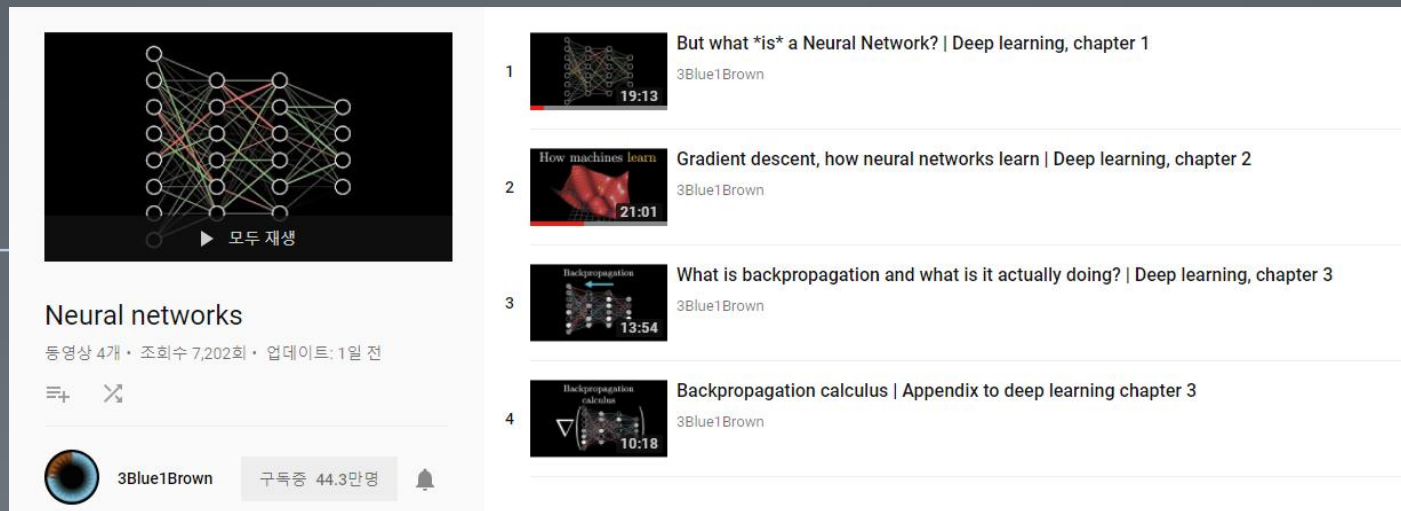
동영상 4개 · 조회수 7,202회 · 업데이트: 1일 전

3Blue1Brown 구독중 44.3만명

- 1 But what *is* a Neural Network? | Deep learning, chapter 1
3Blue1Brown 19:13
- 2 Gradient descent, how neural networks learn | Deep learning, chapter 2
3Blue1Brown 21:01
- 3 What is backpropagation and what is it actually doing? | Deep learning, chapter 3
3Blue1Brown 13:54
- 4 Backpropagation calculus | Appendix to deep learning chapter 3
3Blue1Brown 10:18

<https://youtu.be/aircAruvnKk>

Gradient descent



Neural networks

동영상 4개 · 조회수 7,202회 · 업데이트: 1일 전

3Blue1Brown 구독중 44.3만명

- 1 But what *is* a Neural Network? | Deep learning, chapter 1
3Blue1Brown 19:13
- 2 How machines learn | Gradient descent, how neural networks learn | Deep learning, chapter 2
3Blue1Brown 21:01
- 3 Backpropagation | What is backpropagation and what is it actually doing? | Deep learning, chapter 3
3Blue1Brown 13:54
- 4 Backpropagation calculus | Appendix to deep learning chapter 3
3Blue1Brown 10:18

<https://youtu.be/IHZwWFHWA-w>

컨볼루션 신경망

(Convolution Neural Network)

CS231n: Convolutional Neural Networks for Visual Recognition

Spring 2017



*This network is running live in your browser

Course Description

Computer Vision has become ubiquitous in our society, with applications in search, image understanding, apps, mapping, medicine, drones, and self-driving cars. Core to many of these applications are visual recognition tasks such as image classification, localization and detection. Recent developments in neural network (aka "deep learning") approaches have greatly advanced the performance of these state-of-the-art visual recognition systems. This course is a deep dive into details of the deep learning architectures with a focus on learning end-to-end models for these tasks, particularly image classification. During the 10-week course, students will learn to implement, train and debug their own neural networks and gain a detailed understanding of cutting-edge research in computer vision. The final assignment will involve training a multi-million parameter convolutional neural network and applying it on the largest image classification dataset (ImageNet). We will focus on teaching how to set up the problem of image recognition, the learning algorithms (e.g. backpropagation), practical engineering tricks for training and fine-tuning the networks and guide the students through hands-on assignments and a final course project. Much of the background and materials of this course will be drawn from the [ImageNet Challenge](#).

Instructors



Fei-Fei Li

Teaching Assistants



Albert Haque



Rishi Bedi



Shyamal Buch



Zhao (Joe) Chen



Timnit Gebru

<http://cs231n.stanford.edu/>

Next: Convolutional Neural Networks

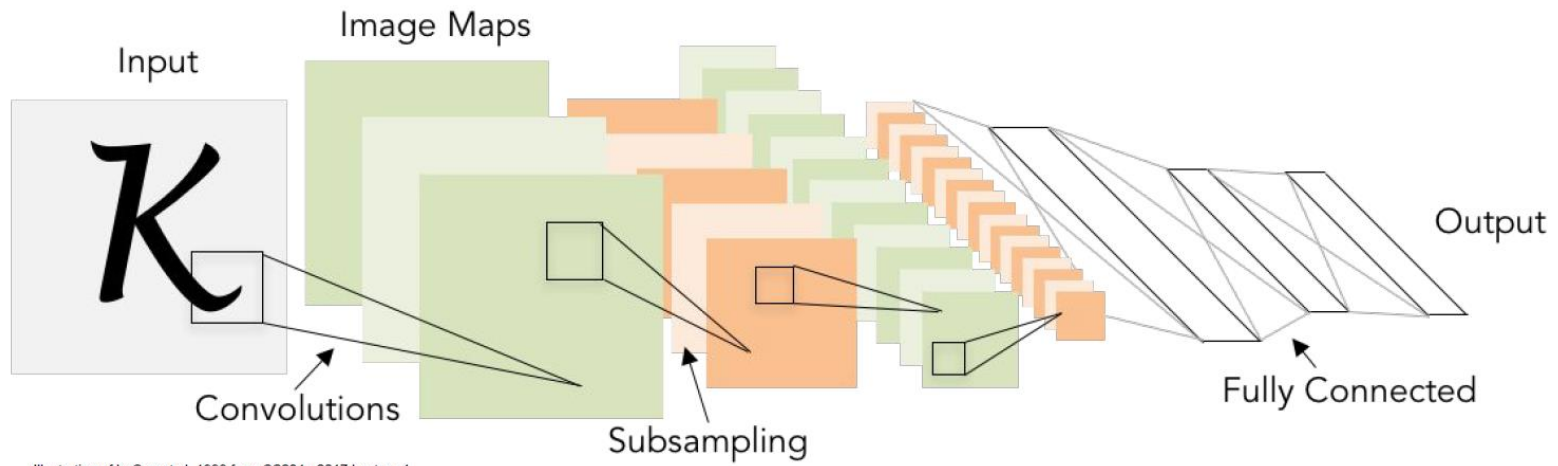


Illustration of LeCun et al. 1998 from CS231n 2017 Lecture 1

ImageNet Classification with Deep Convolutional Neural Networks

[Krizhevsky, Sutskever, Hinton, 2012]

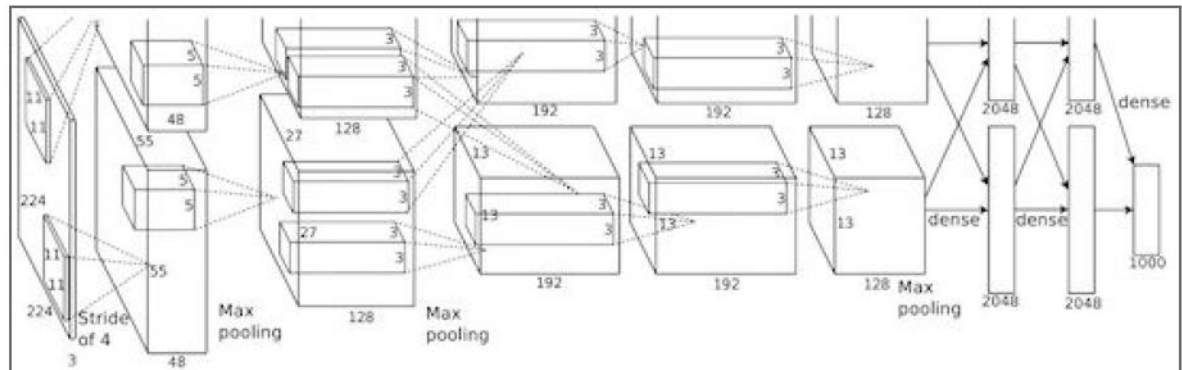


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

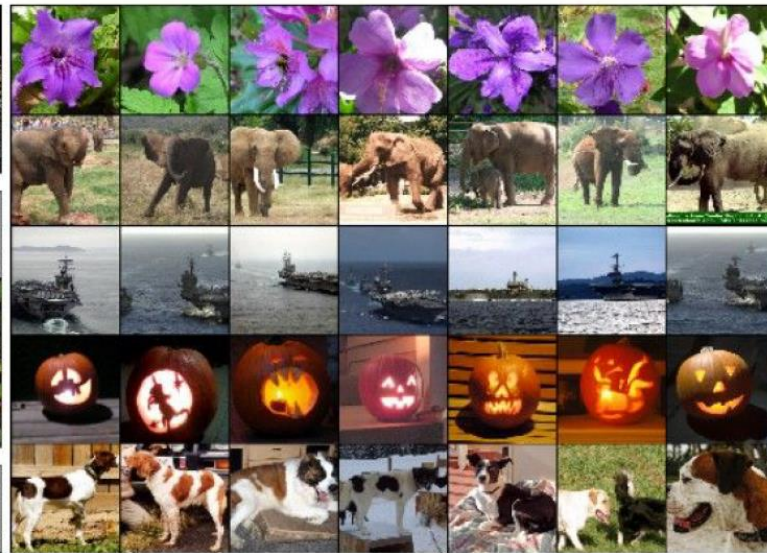
“AlexNet”

Fast-forward to today: ConvNets are everywhere

Classification



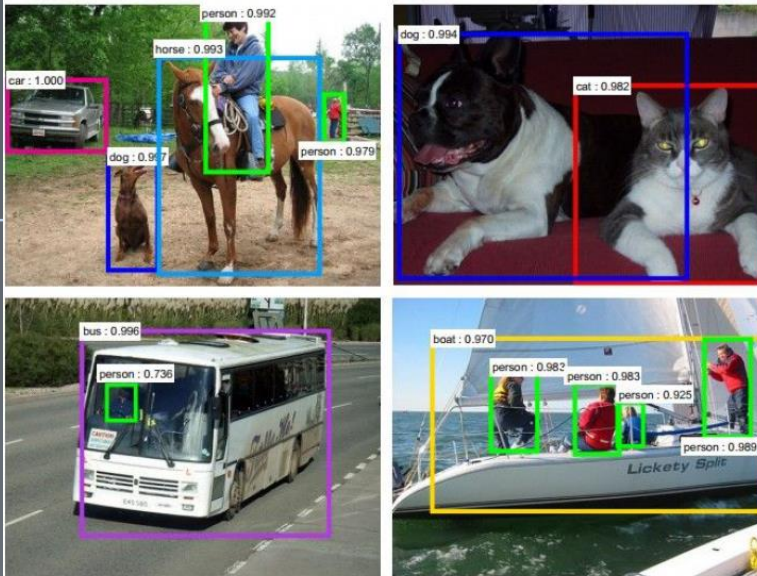
Retrieval



Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Fast-forward to today: ConvNets are everywhere

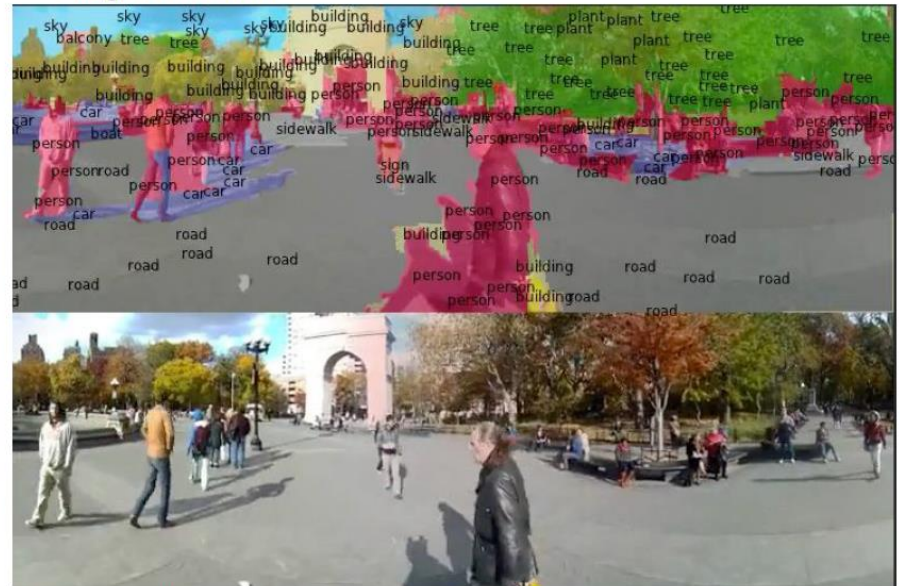
Detection



Figures copyright Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

Segmentation



Figures copyright Clement Farabet, 2012.
Reproduced with permission.

[Farabet et al., 2012]

Fast-forward to today: ConvNets are everywhere



self-driving cars

Photo by Lane McIntosh. Copyright CS231n 2017.



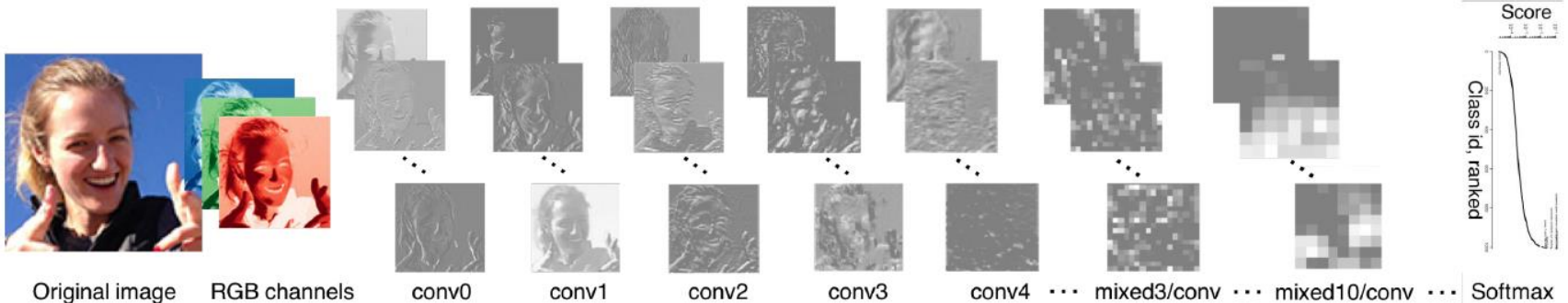
[This image](#) by GBPublic_PR is licensed under [CC-BY 2.0](#)

NVIDIA Tesla line

(these are the GPUs on rye01.stanford.edu)

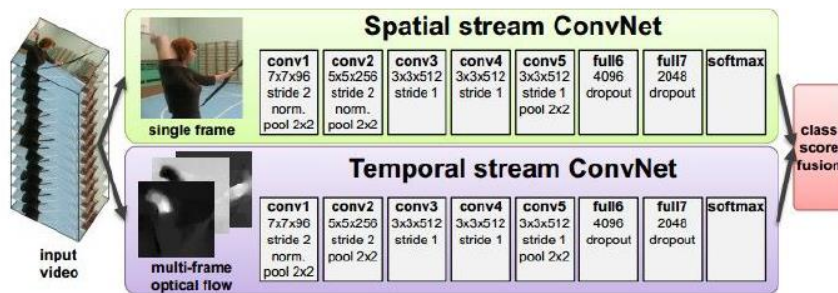
Note that for embedded systems a typical setup would involve NVIDIA Tegras, with integrated GPU and ARM-based CPU cores.

Fast-forward to today: ConvNets are everywhere



[Taigman et al. 2014]

Activations of [inception-v3 architecture](#) [Szegedy et al. 2015] to image of Emma McIntosh, used with permission. Figure and architecture not from Taigman et al. 2014.



[Simonyan et al. 2014]

Figures copyright Simonyan et al., 2014.
Reproduced with permission.

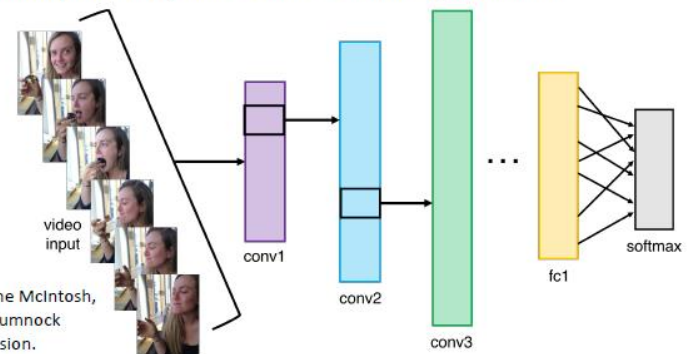


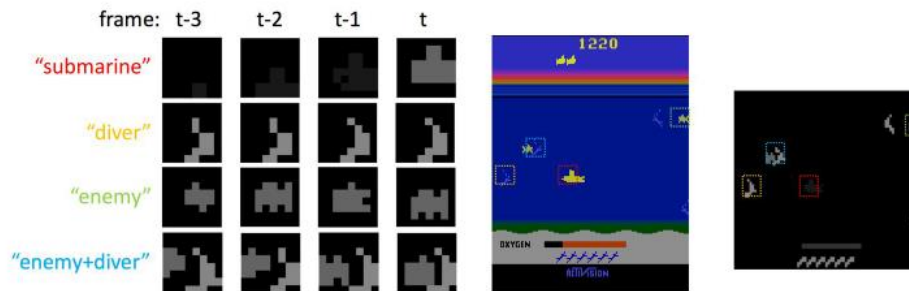
Illustration by Lane McIntosh,
photos of Katie Cumnock
used with permission.

Fast-forward to today: ConvNets are everywhere

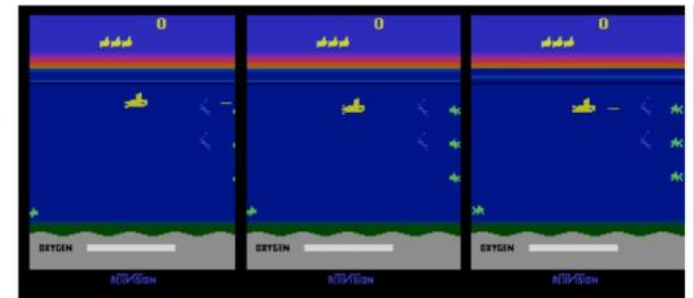


Images are examples of pose estimation, not actually from Toshev & Szegedy 2014. Copyright Lane McIntosh.

[Toshev, Szegedy 2014]



[Guo et al. 2014]



Figures copyright Xiaoxiao Guo, Satinder Singh, Honglak Lee, Richard Lewis, and Xiaoshi Wang, 2014. Reproduced with permission.

No errors



A white teddy bear sitting in the grass

Minor errors



A man in a baseball uniform throwing a ball

Somewhat related



A woman is holding a cat in her hand

Image Captioning

[Vinyals et al., 2015]
[Karpathy and Fei-Fei, 2015]



A man riding a wave on top of a surfboard



A cat sitting on a suitcase on the floor

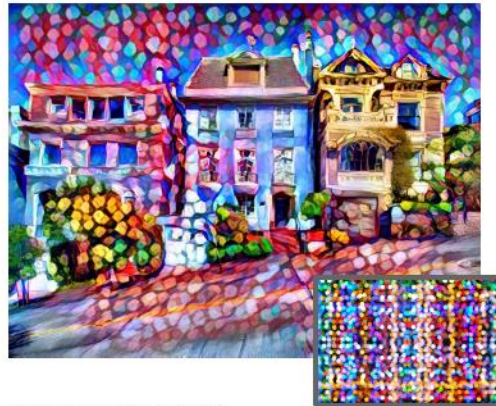
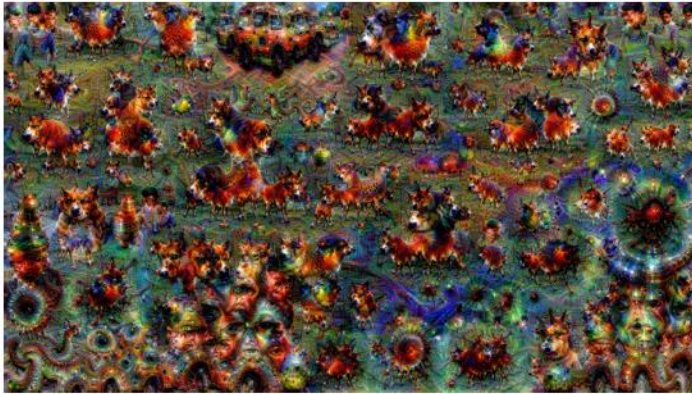


A woman standing on a beach holding a surfboard

All images are CC0 Public domain:

<https://pixabay.com/en/luggage-antique-cat-1643010/>
<https://pixabay.com/en/teddy-plush-bears-cute-teddy-bear-1623436/>
<https://pixabay.com/en/surf-wave-summer-sport-litoral-1668716/>
<https://pixabay.com/en/woman-female-model-portrait-adult-983967/>
<https://pixabay.com/en/handstand-lake-meditation-496008/>
<https://pixabay.com/en/baseball-player-shortstop-infield-1045263/>

Captions generated by Justin Johnson using [NeuralTalk2](#)



Original image is CC0 public domain
 Starry Night and Tree Roots by Van Gogh are in the public domain
 Bokeh image is in the public domain
 Stylized images copyright Justin Johnson, 2017;
 reproduced with permission

Gatys et al, "Image Style Transfer using Convolutional Neural Networks", CVPR 2016
 Gatys et al, "Controlling Perceptual Factors in Neural Style Transfer", CVPR 2017

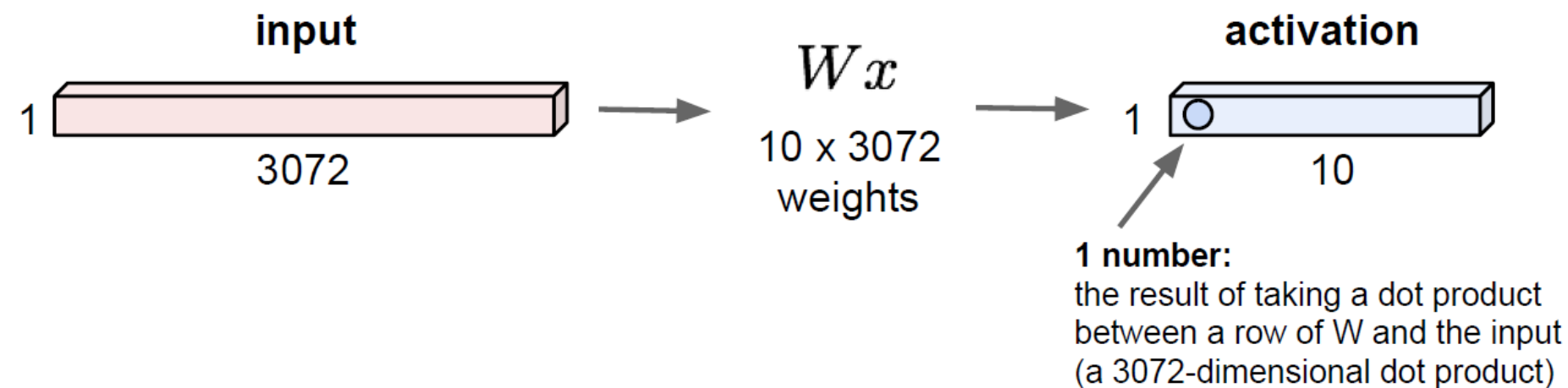
Figures copyright Justin Johnson, 2015. Reproduced with permission. Generated using the Inceptionism approach from a [blog post](#) by Google Research.

컨볼루션 비디오

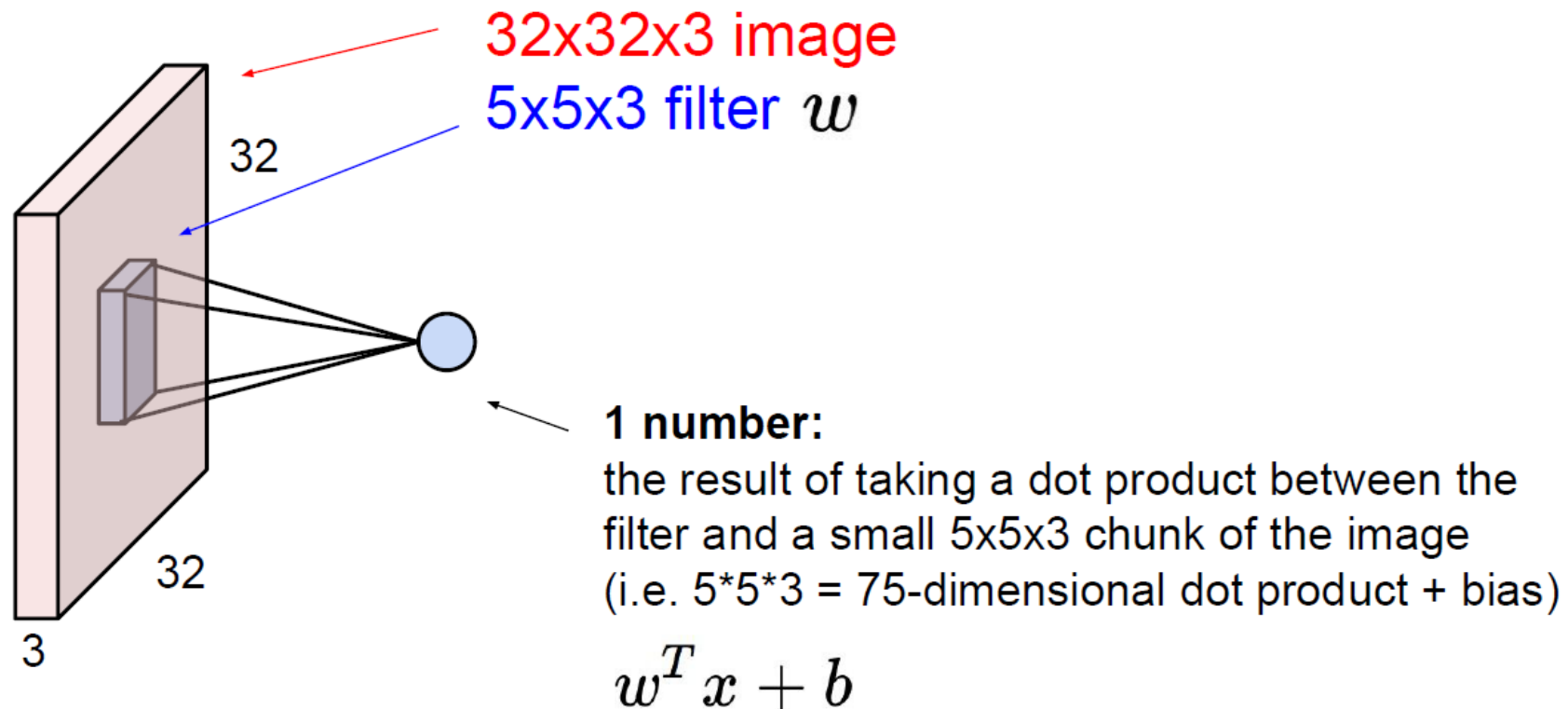
https://www.youtube.com/watch?v=KiftWz544_8

Fully Connected Layer

32x32x3 image -> stretch to 3072 x 1

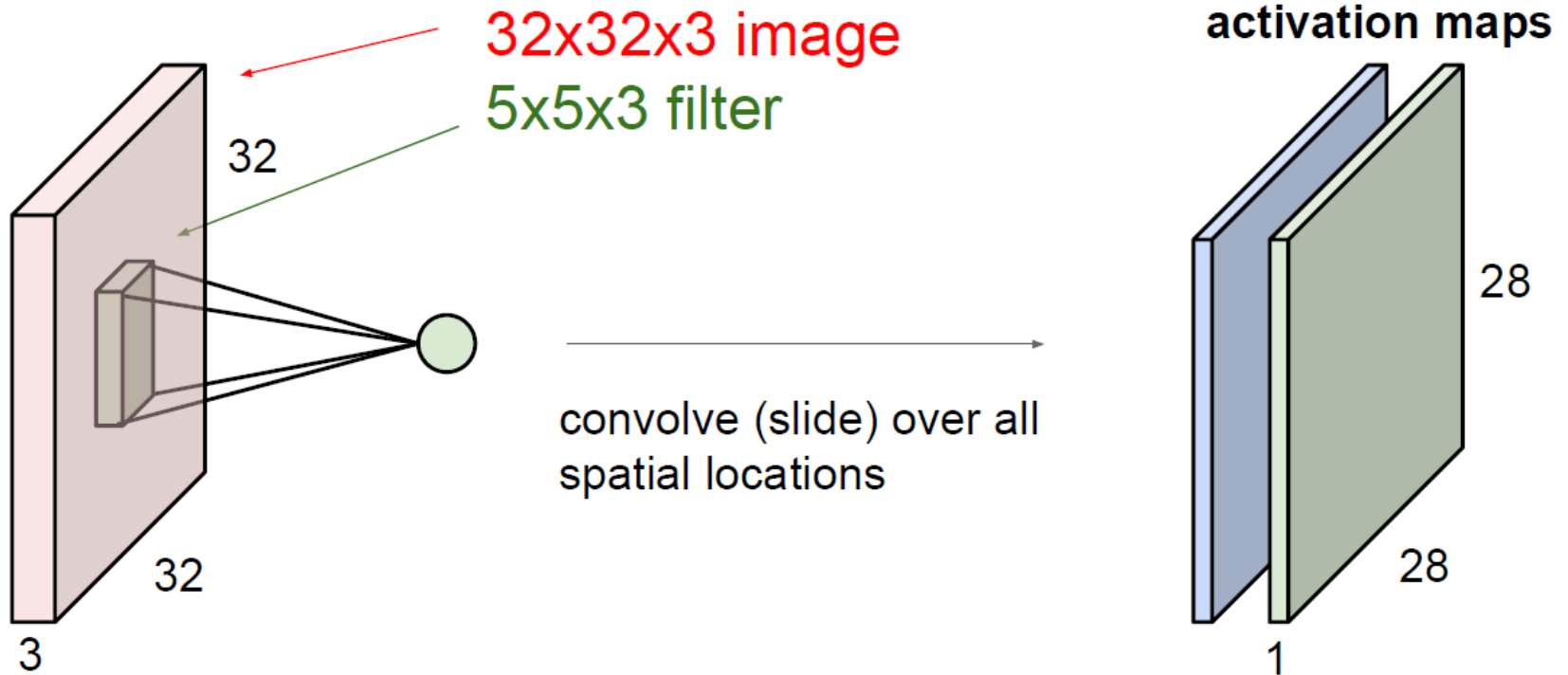


Convolution Layer

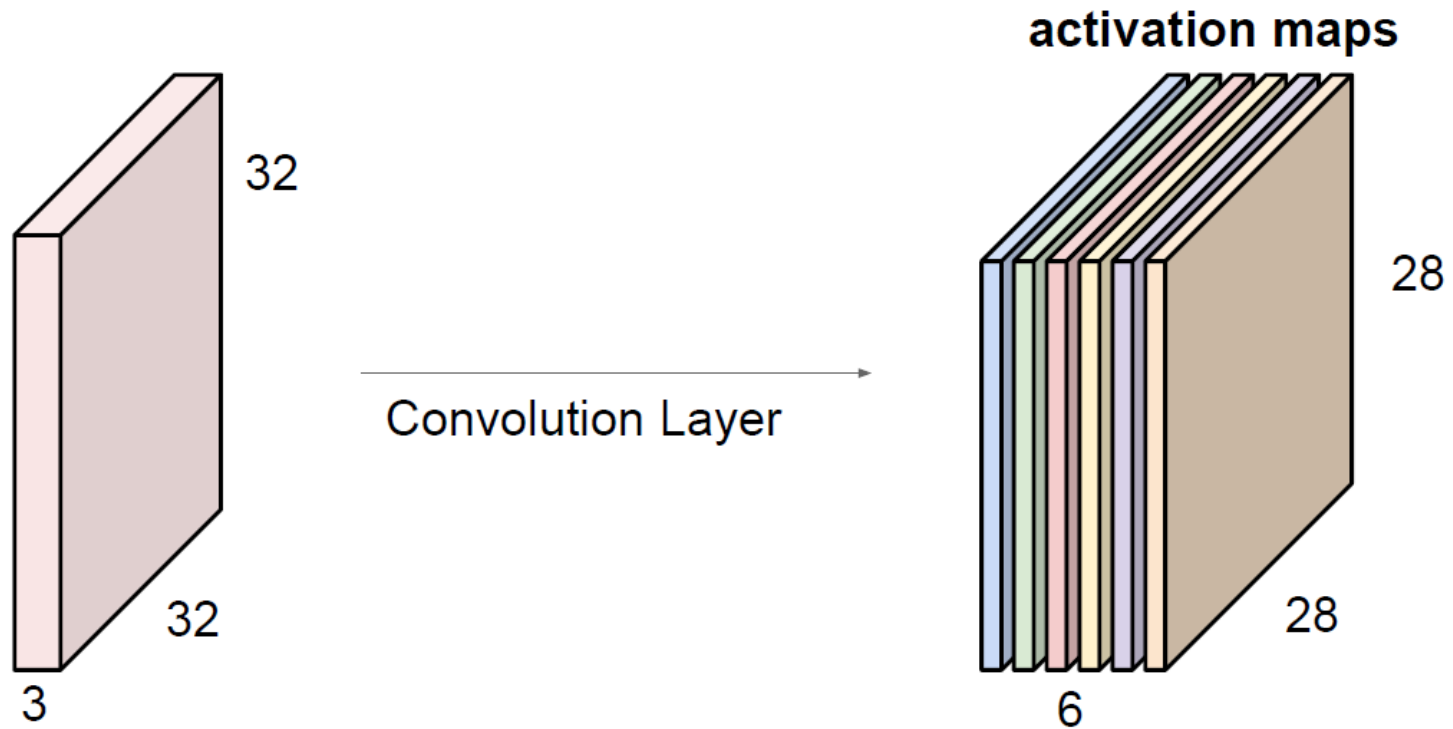


Convolution Layer

consider a second, **green** filter

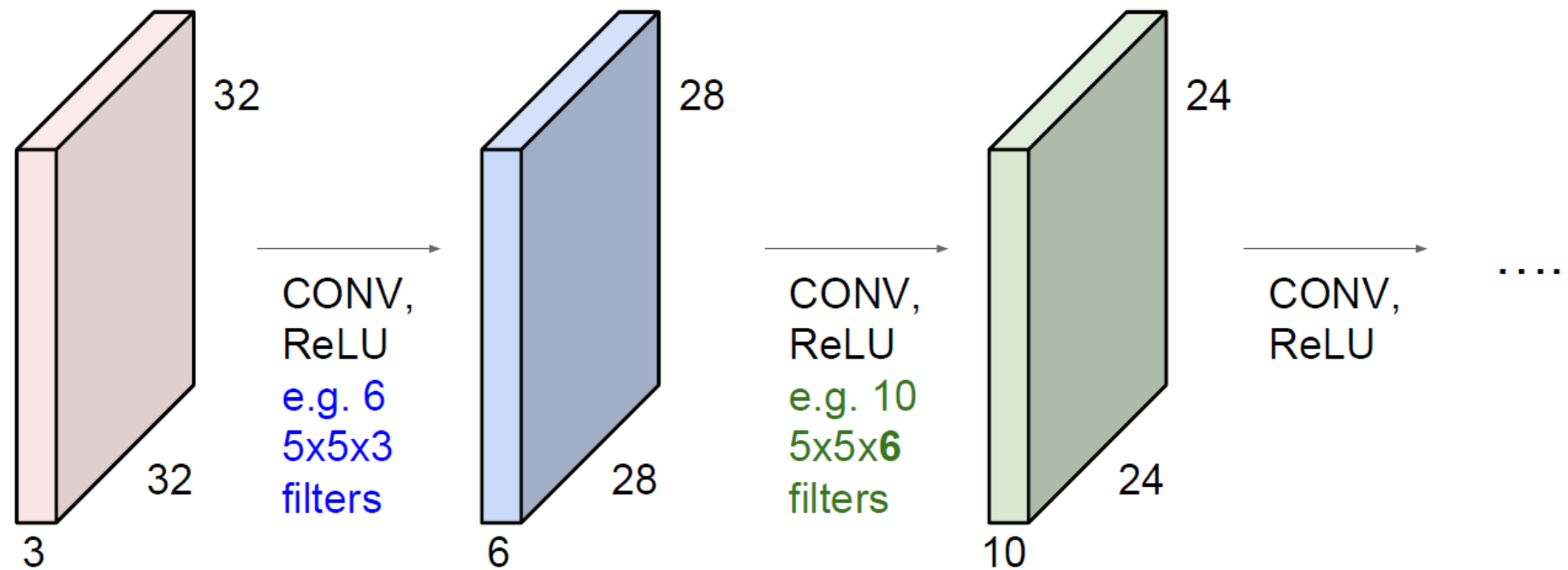


For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a “new image” of size 28x28x6!

Preview: ConvNet is a sequence of Convolutional Layers, interspersed with activation functions

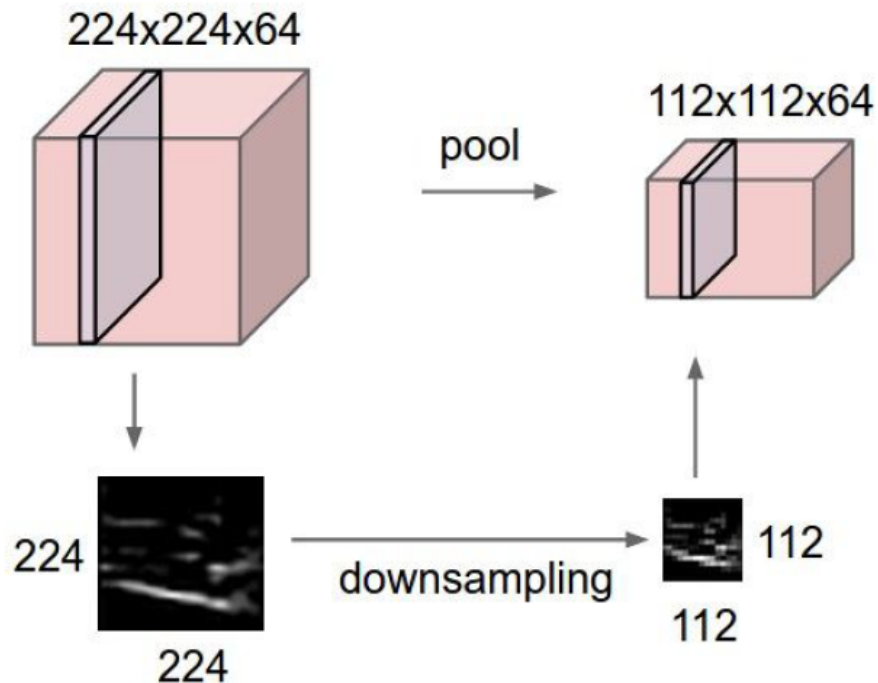


풀링 비디오

<https://www.youtube.com/watch?v=mW3KyFZDNIQ>

Pooling layer

- makes the representations smaller and more manageable
- operates over each activation map independently:



MAX POOLING

Single depth slice

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

max pool with 2x2 filters
and stride 2

6	8
3	4

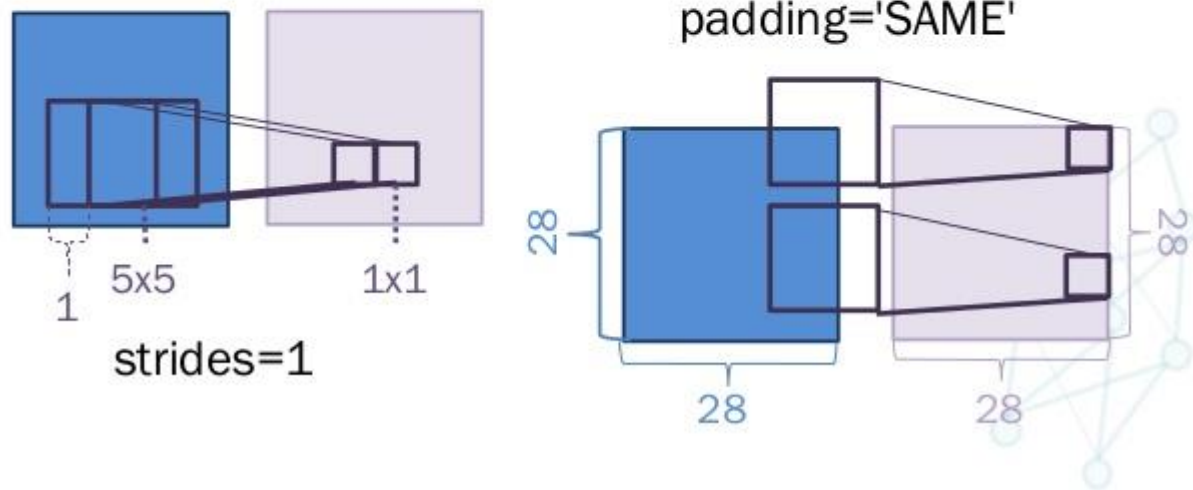
텐서플로 실습

컨볼루션 in 텐서플로

Convolutional Layer

```
tf.nn.conv2d(x, W, strides=[1, 1, 1, 1], padding='SAME')+b
```

[batch, in_height, in_width, in_channels]



컨볼루션 in 텐서플로

```
def createNetwork():
    # network weights
    W_conv1 = weight_variable([3, 3, 4, 32])
    b_conv1 = bias_variable([32])

    W_conv2 = weight_variable([3, 3, 32, 64])
    b_conv2 = bias_variable([64])

    W_conv3 = weight_variable([3, 3, 64, 64])
    b_conv3 = bias_variable([64])

    #
    W_fc1 = weight_variable([6400, 512])
    b_fc1 = bias_variable([512])

    W_fc2 = weight_variable([512, ACTIONS])
    b_fc2 = bias_variable([ACTIONS])

    # input layer
    s = tf.placeholder("float", [None, 80, 80, 4])

    # hidden layers
    h_conv1 = tf.nn.relu(conv2d(s, W_conv1, 1) + b_conv1)
    h_pool1 = max_pool_2x2(h_conv1)

    h_conv2 = tf.nn.relu(conv2d(h_pool1, W_conv2, 1) + b_conv2)
    h_pool2 = max_pool_2x2(h_conv2)

    h_conv3 = tf.nn.relu(conv2d(h_pool2, W_conv3, 1) + b_conv3)
    h_pool3 = max_pool_2x2(h_conv3)

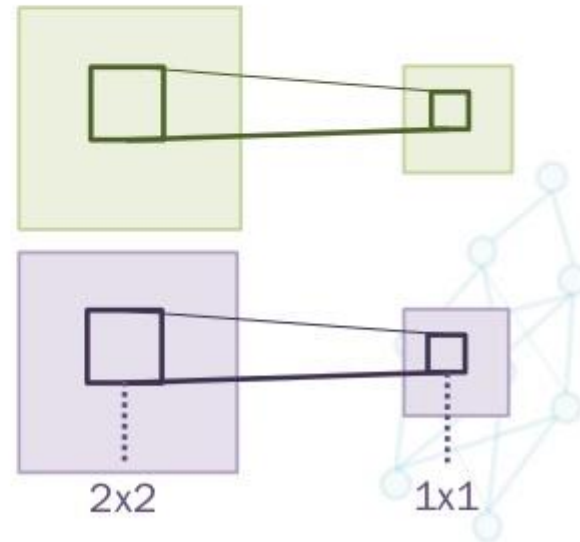
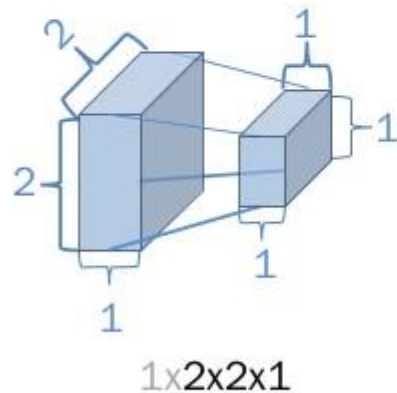
    #
    #h_pool3_flat = tf.reshape(h_pool3, [-1, 256])
    h_conv3_flat = tf.reshape(h_pool3, [-1, 6400])

    h_fc1 = tf.nn.relu(tf.matmul(h_conv3_flat, W_fc1) + b_fc1)
```

맥스풀링 in 텐서플로

Pooling Layer

```
tf.nn.max_pool(x, ksize=[1, 2, 2, 1],  
               strides=[1, 2, 2, 1], padding='SAME')
```



CIFAR-10

CIFAR-10 Model

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Data Sets

<code>data_sets.train</code>	50000 images & labels
<code>data_sets.validation</code>	1000 images & labels
<code>data_sets.test</code>	10000 images & labels

Source: <http://www.cs.toronto.edu/~kriz/cifar.html>

웹버전 실습

<http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html>

Case Study: AlexNet

[Krizhevsky et al. 2012]

Full (simplified) AlexNet architecture:

[227x227x3] INPUT

[55x55x96] **CONV1**: 96 11x11 filters at stride 4, pad 0

[27x27x96] **MAX POOL1**: 3x3 filters at stride 2

[27x27x96] **NORM1**: Normalization layer

[27x27x256] **CONV2**: 256 5x5 filters at stride 1, pad 2

[13x13x256] **MAX POOL2**: 3x3 filters at stride 2

[13x13x256] **NORM2**: Normalization layer

[13x13x384] **CONV3**: 384 3x3 filters at stride 1, pad 1

[13x13x384] **CONV4**: 384 3x3 filters at stride 1, pad 1

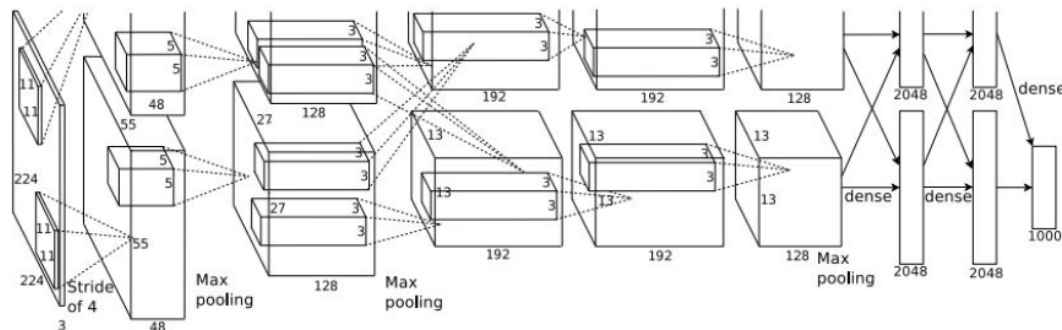
[13x13x256] **CONV5**: 256 3x3 filters at stride 1, pad 1

[6x6x256] **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%

Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

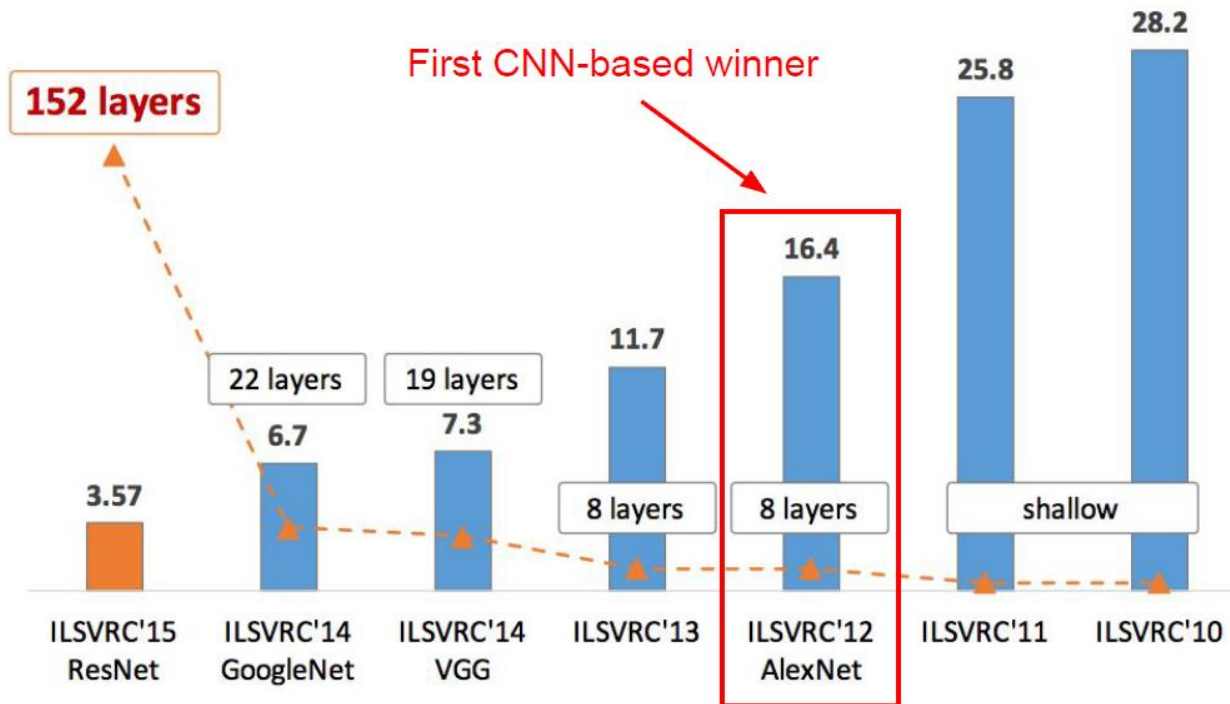


Figure copyright Kaiming He, 2016. Reproduced with permission.

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

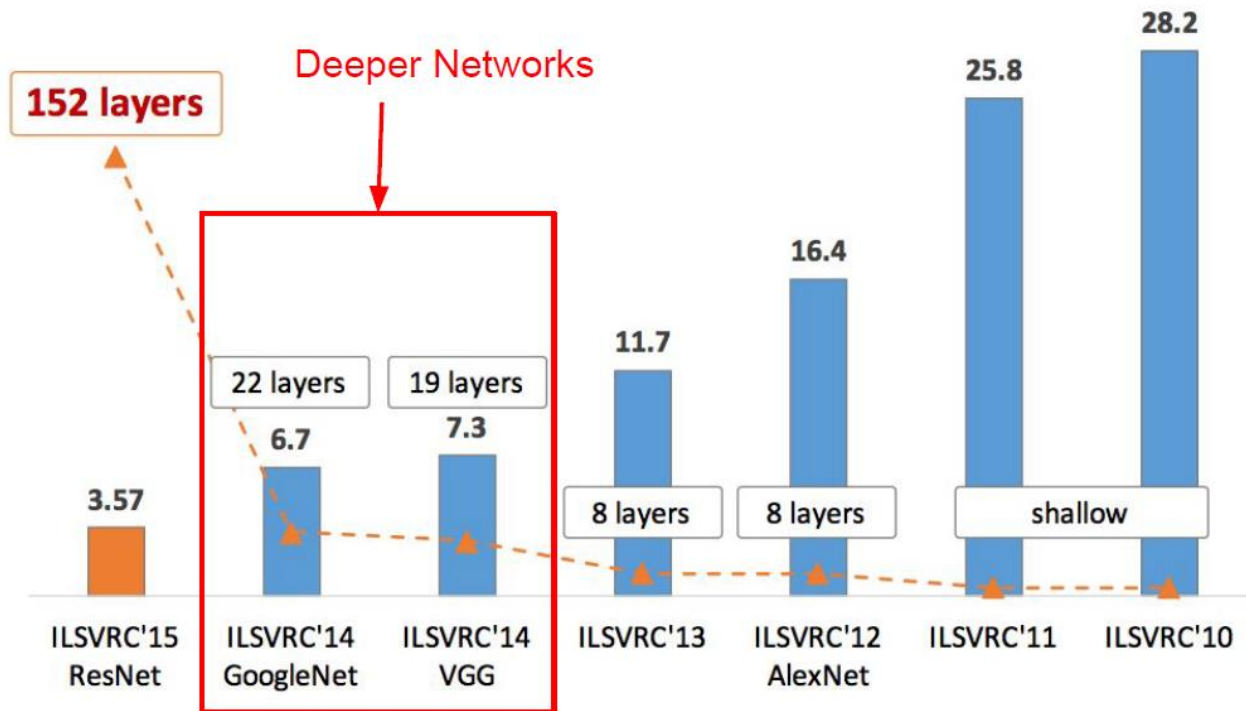


Figure copyright Kaiming He, 2016. Reproduced with permission.

Case Study: VGGNet

[Simonyan and Zisserman, 2014]

Small filters, Deeper networks

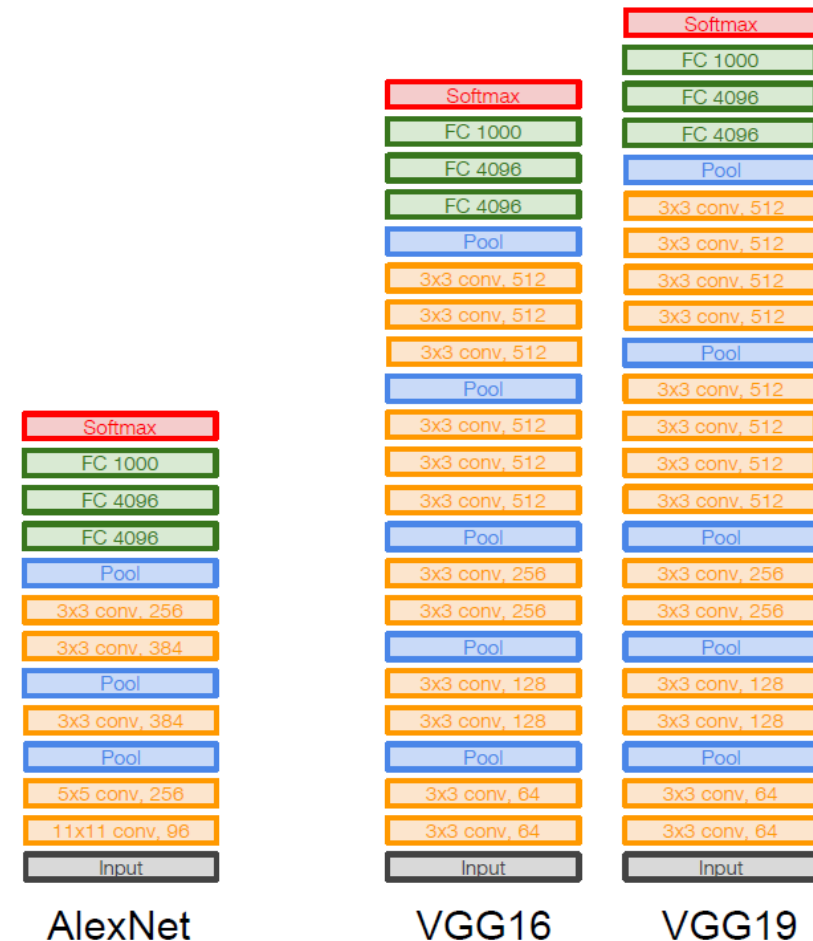
8 layers (AlexNet)

-> 16 - 19 layers (VGG16Net)

Only 3x3 CONV stride 1, pad 1
and 2x2 MAX POOL stride 2

11.7% top 5 error in ILSVRC'13
(ZFNet)

-> 7.3% top 5 error in ILSVRC'14



INPUT: [224x224x3] memory: $224*224*3=150K$ params: 0 (not counting biases)

CONV3-64: [224x224x64] memory: $224*224*64=3.2M$ params: $(3*3*3)*64 = 1,728$

CONV3-64: [224x224x64] memory: $224*224*64=3.2M$ params: $(3*3*64)*64 = 36,864$

POOL2: [112x112x64] memory: $112*112*64=800K$ params: 0

CONV3-128: [112x112x128] memory: $112*112*128=1.6M$ params: $(3*3*64)*128 = 73,728$

CONV3-128: [112x112x128] memory: $112*112*128=1.6M$ params: $(3*3*128)*128 = 147,456$

POOL2: [56x56x128] memory: $56*56*128=400K$ params: 0

CONV3-256: [56x56x256] memory: $56*56*256=800K$ params: $(3*3*128)*256 = 294,912$

CONV3-256: [56x56x256] memory: $56*56*256=800K$ params: $(3*3*256)*256 = 589,824$

CONV3-256: [56x56x256] memory: $56*56*256=800K$ params: $(3*3*256)*256 = 589,824$

POOL2: [28x28x256] memory: $28*28*256=200K$ params: 0

CONV3-512: [28x28x512] memory: $28*28*512=400K$ params: $(3*3*256)*512 = 1,179,648$

CONV3-512: [28x28x512] memory: $28*28*512=400K$ params: $(3*3*512)*512 = 2,359,296$

CONV3-512: [28x28x512] memory: $28*28*512=400K$ params: $(3*3*512)*512 = 2,359,296$

POOL2: [14x14x512] memory: $14*14*512=100K$ params: 0

CONV3-512: [14x14x512] memory: $14*14*512=100K$ params: $(3*3*512)*512 = 2,359,296$

CONV3-512: [14x14x512] memory: $14*14*512=100K$ params: $(3*3*512)*512 = 2,359,296$

CONV3-512: [14x14x512] memory: $14*14*512=100K$ params: $(3*3*512)*512 = 2,359,296$

POOL2: [7x7x512] memory: $7*7*512=25K$ params: 0

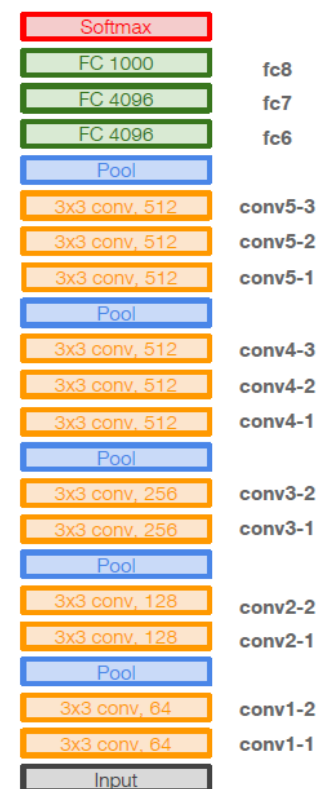
FC: [1x1x4096] memory: 4096 params: $7*7*512*4096 = 102,760,448$

FC: [1x1x4096] memory: 4096 params: $4096*4096 = 16,777,216$

FC: [1x1x1000] memory: 1000 params: $4096*1000 = 4,096,000$

TOTAL memory: $24M * 4 \text{ bytes} \sim 96MB$ / image (only forward! ~ 2 for bwd)

TOTAL params: 138M parameters



VGG16

Common names

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

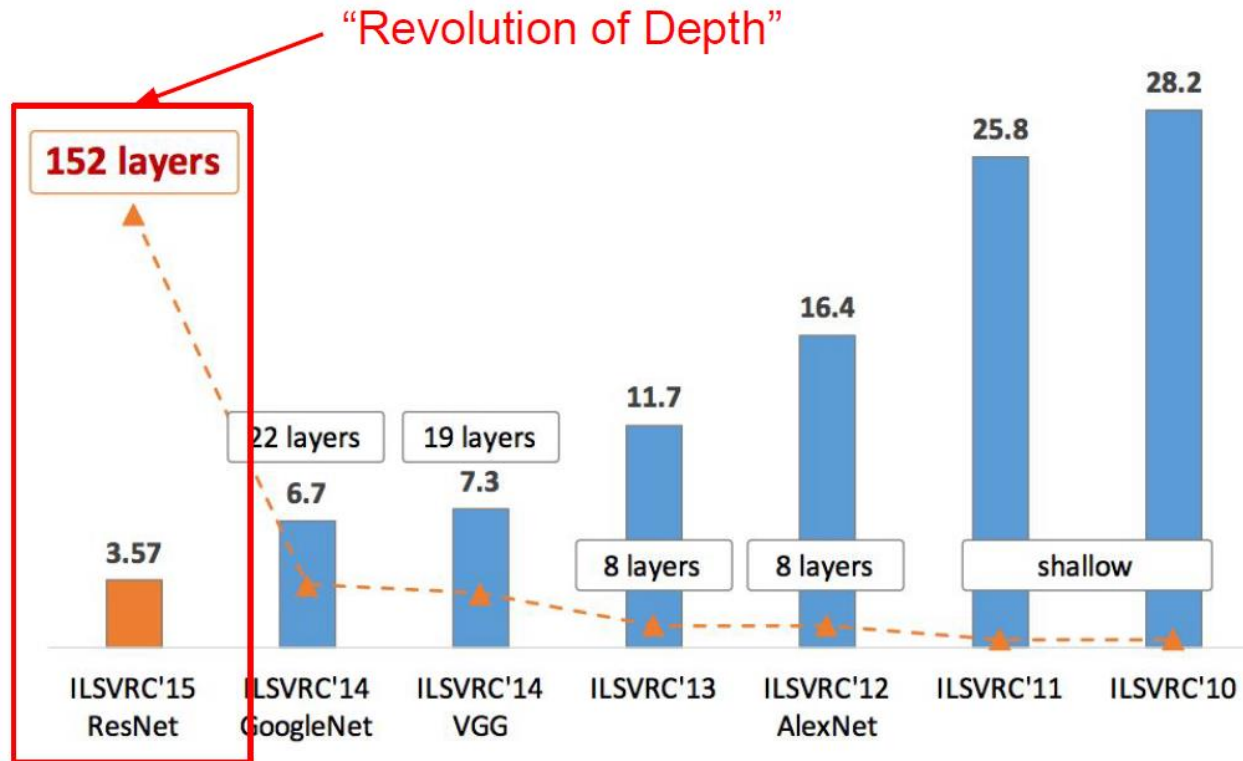


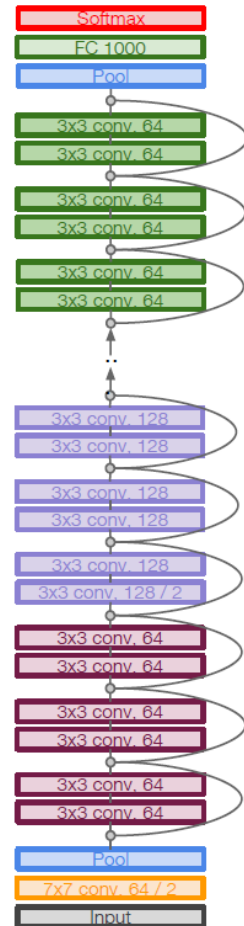
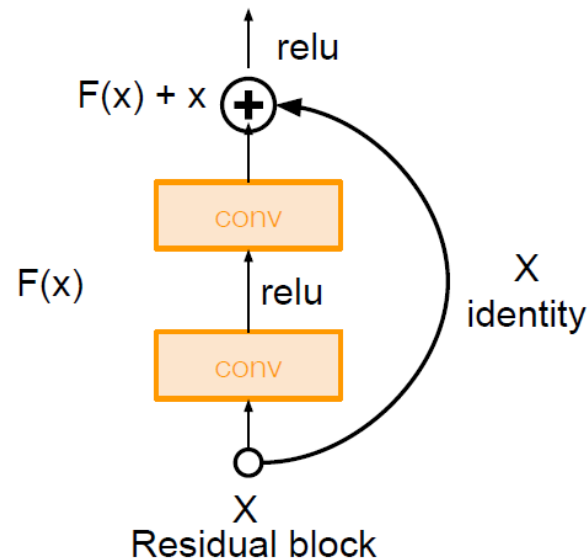
Figure copyright Kaiming He, 2016. Reproduced with permission.

Case Study: ResNet

[He et al., 2015]

Very deep networks using residual connections

- 152-layer model for ImageNet
- ILSVRC'15 classification winner (3.57% top 5 error)
- Swept all classification and detection competitions in ILSVRC'15 and COCO'15!



감사합니다