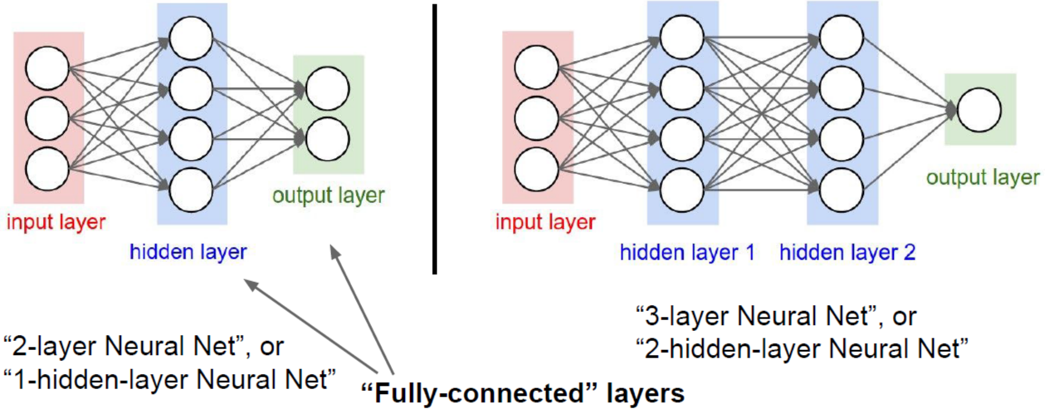
ELEG404/604: Imaging & Deep Learning

Gonzalo R. Arce

Department of Electrical and Computer Engineering
University of Delaware

Convolutional Neural Networks

Neural Networks Architectures



“2-layer Neural Net”, or
“1-hidden-layer Neural Net”

“3-layer Neural Net”, or
“2-hidden-layer Neural Net”

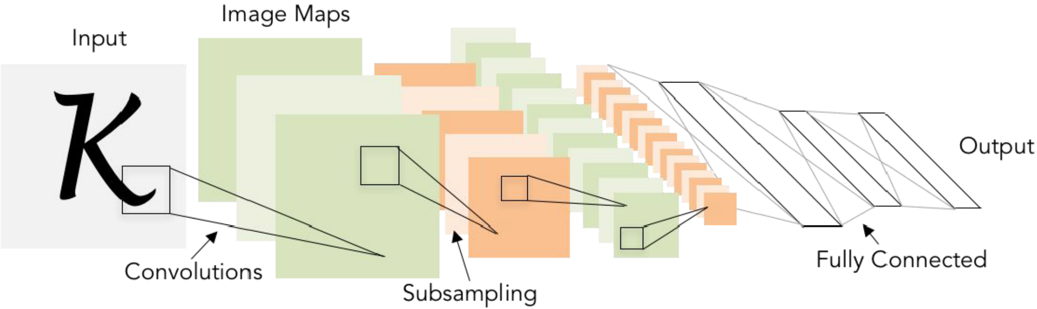
“Fully-connected” layers

$4+2=6$ neurons.
$[3\times 4]+[4\times 2]=20$ weights
$4+2=6$ biases.

$4+4+1=9$ neurons.
$[3\times 4]+[4\times 4]+[4\times 1]=32$ weights
$4+4+1=9$ biases.

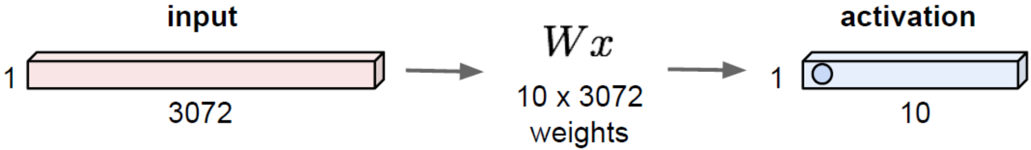
Convolutional Neural Networks Architectures

- Very similar to ordinary Neural Networks.
- Add convolutional layers. Neurons with 3 dimensions: width, height and depth.
- Inputs are also volumes.



Neural Network - Fully Connected (FC) Layer

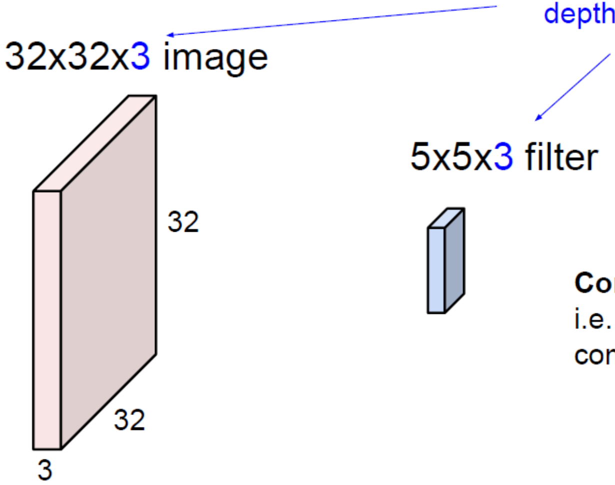
Consider a $32 \times 32 \times 3$ image → stretch to 3072×1



Each output is the result of a dot product between a row of **W** and the input **x**. 10 neurons outputs.

Convolutional Layer

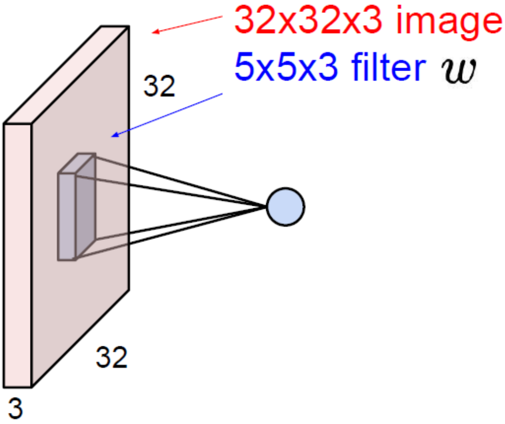
Consider a $32 \times 32 \times 3$ image → preserve spatial structure.



Filters always extend the full depth of the input volume

Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

Convolutional Layer



Result: dot product between the filter and a small $5 \times 5 \times 3$ chunk of the image.

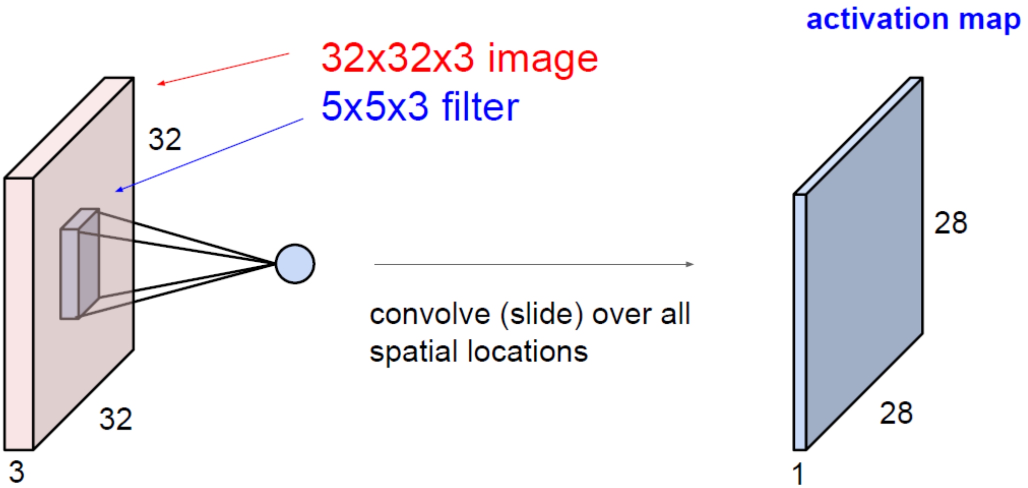
Volume convolution at (x, y), for **all** maps of the input volume:

$$\text{conv}_{x,y} = \sum_i w_i v_i$$

where ws are kernel weights, vs chuck of the image.
Adding scalar bias b:

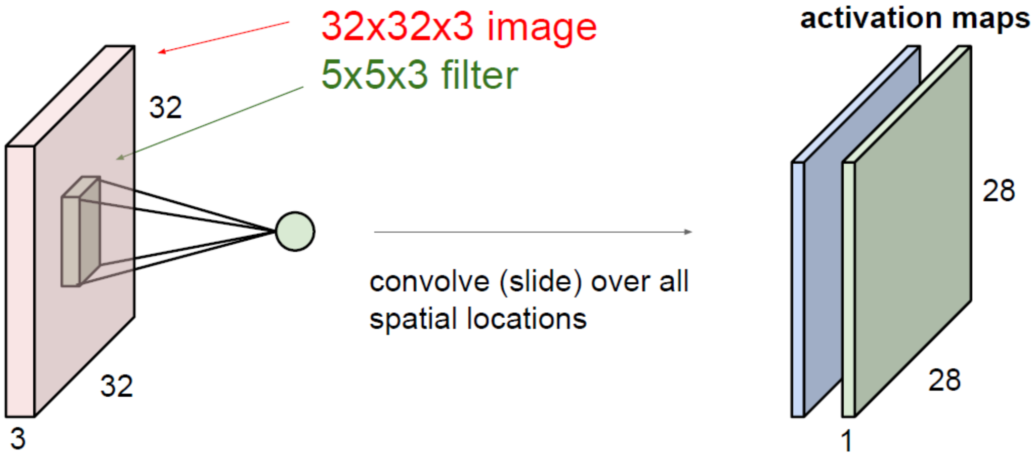
$$z_{x,y} = \sum_i w_i v_i + b$$

Convolutional Layer



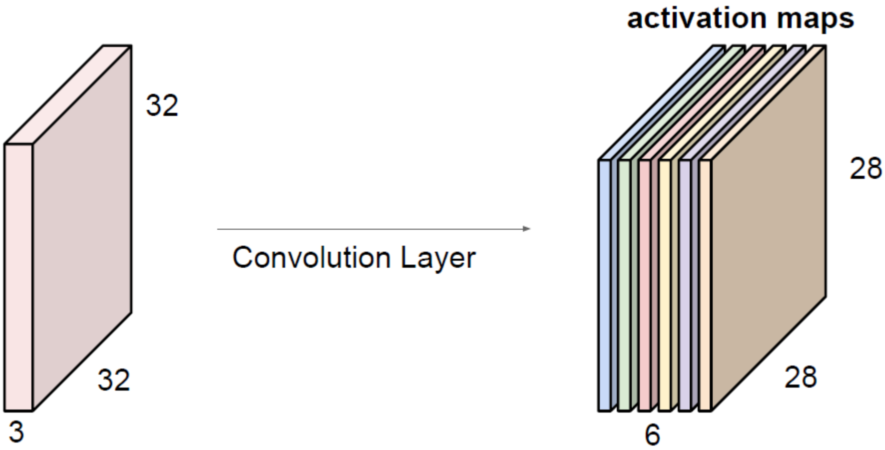
Convolutional Layer

Consider a second, green filter:



Convolutional Layer

Consider 6 filters (5×5), we get 6 separate activation maps:



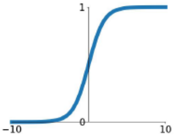
We stack these up to get a "new image volume" of size $28 \times 28 \times 6$

Activation Functions

Pass every element of each activation map through a nonlinearity:

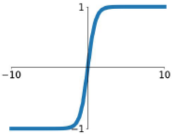
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



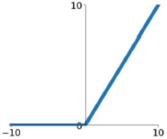
tanh

$\tanh(x)$



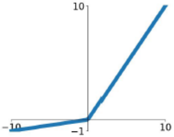
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

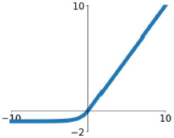


Maxout

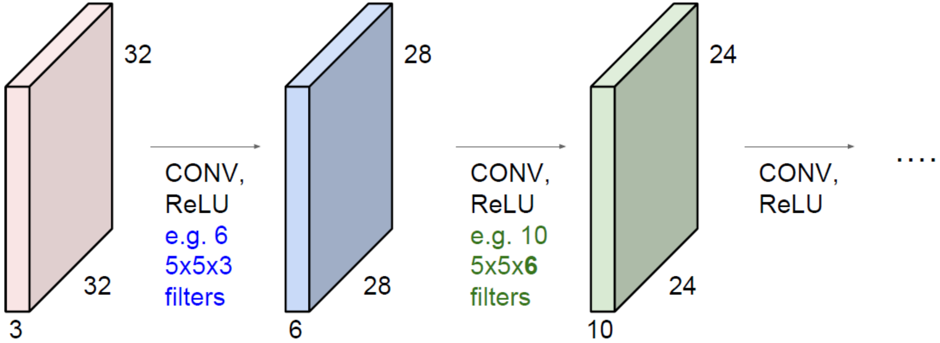
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



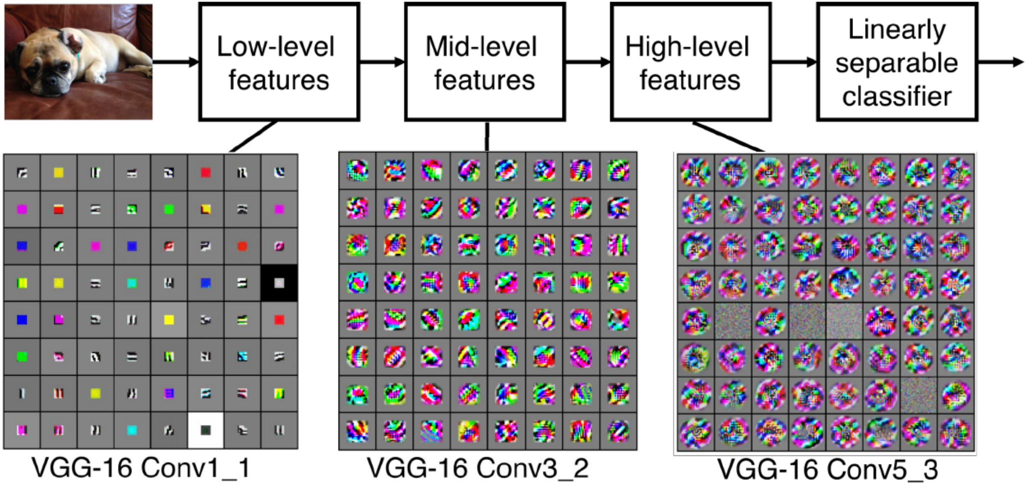
ConvNet is a sequence of Convolutional Layers, interspersed with activation functions:


32
32
3
CONV, ReLU e.g. 6 5x5x3 filters
28
28
6
CONV, ReLU e.g. 10 5x5x6 filters
24
24
10
CONV, ReLU
....

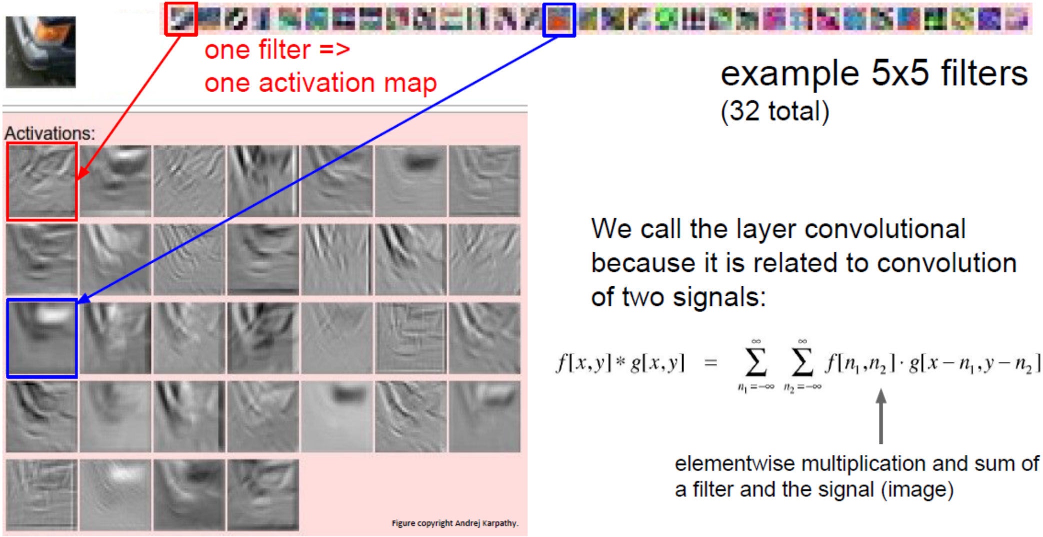
Notice how the activation maps get smaller, this can be solved by zero padding.

Interpretation

Filters Learned:



Interpretation



example 5x5 filters
(32 total)

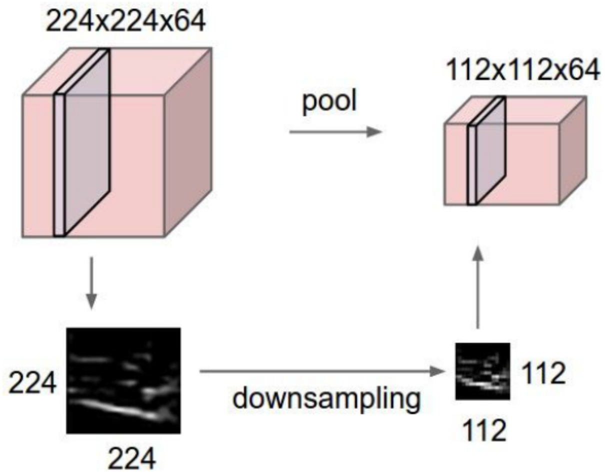
We call the layer convolutional because it is related to convolution of two signals:

$$f[x,y]*g[x,y] = \sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty} f[n_1,n_2]\cdot g[x-n_1,y-n_2]$$

elementwise multiplication and sum of a filter and the signal (image)

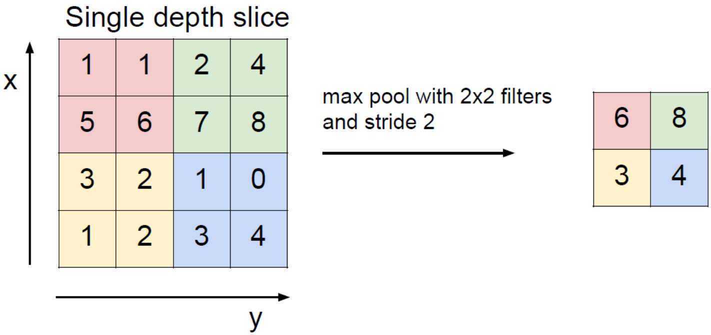
Pooling Layer

- Makes the representations smaller and more manageable.
- Operates over each activation map independently.
- Neighborhood of 2×2 is replaced by the average.



Max Pooling

- Neighborhood of 2×2 is replaced by the maximum value.
- Effective in classifying large image databases.
- Simple and fast.



Single depth slice

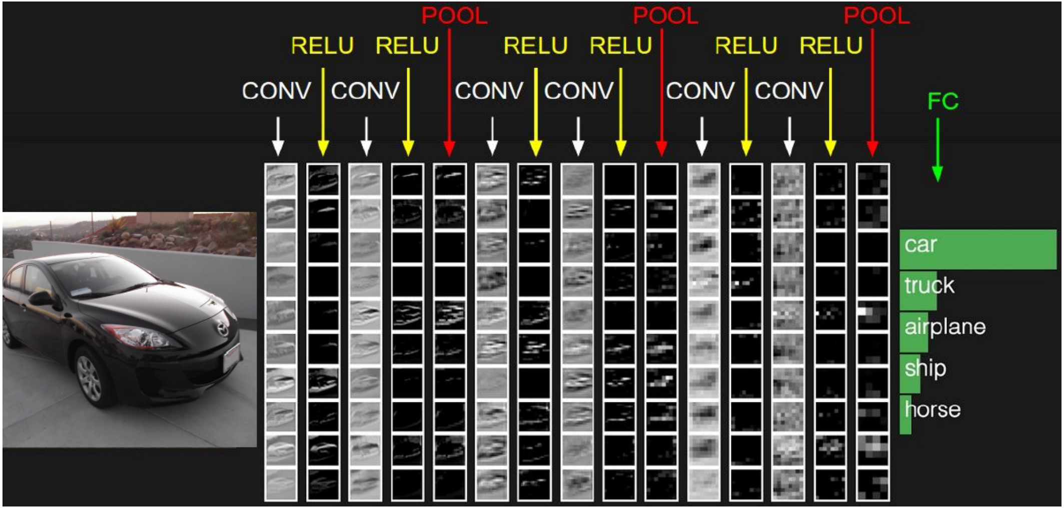
1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

max pool with 2x2 filters and stride 2

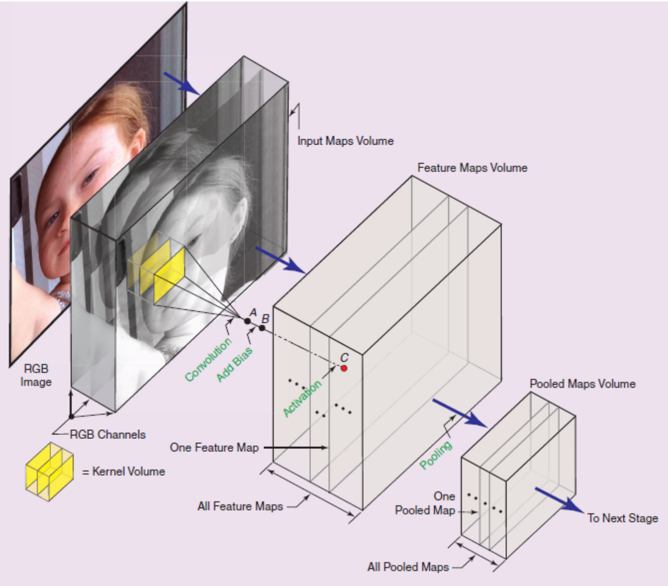
6	8
3	4

- L_2 pooling is also used. Neighborhood of 2×2 is replaced by the squared root of the sum of their squared values.

Example - Image classification

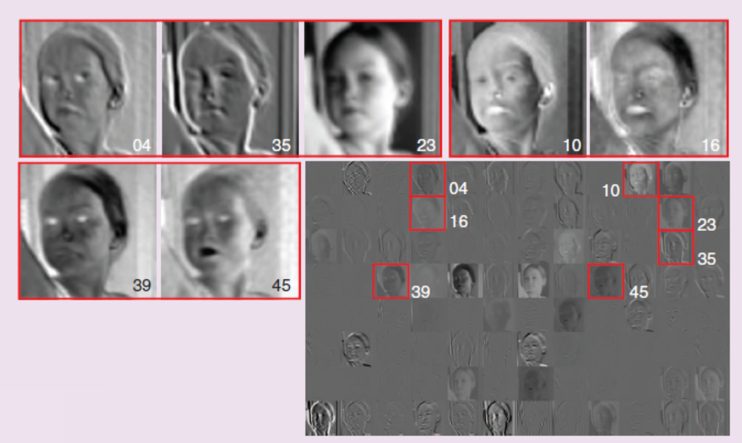


Convolutional Neural Networks Complete Scheme



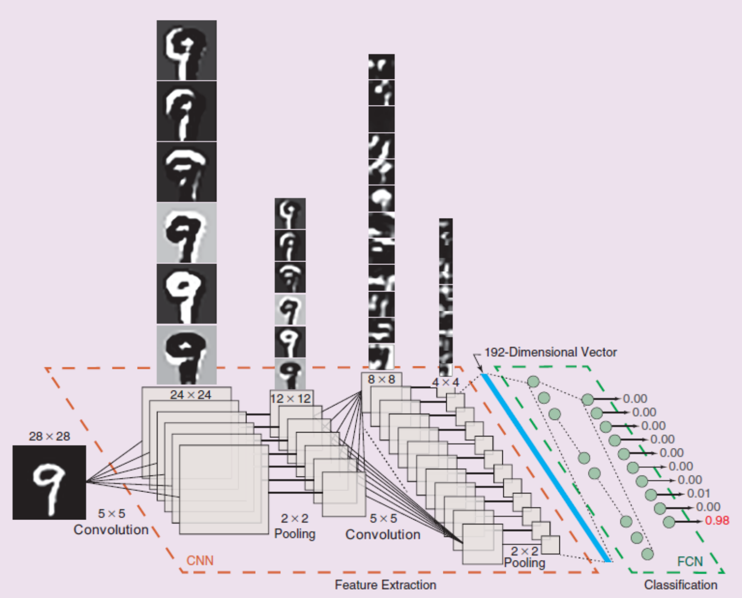
- 277×277 pixels RGB image.
- 96 feature maps.
- 96 kernels volumes of size $11 \times 11 \times 3$
- This weights came from AlexNet: CNN trained using more than 1 million images belonging to 1,000 object categories.

Result Feature Maps



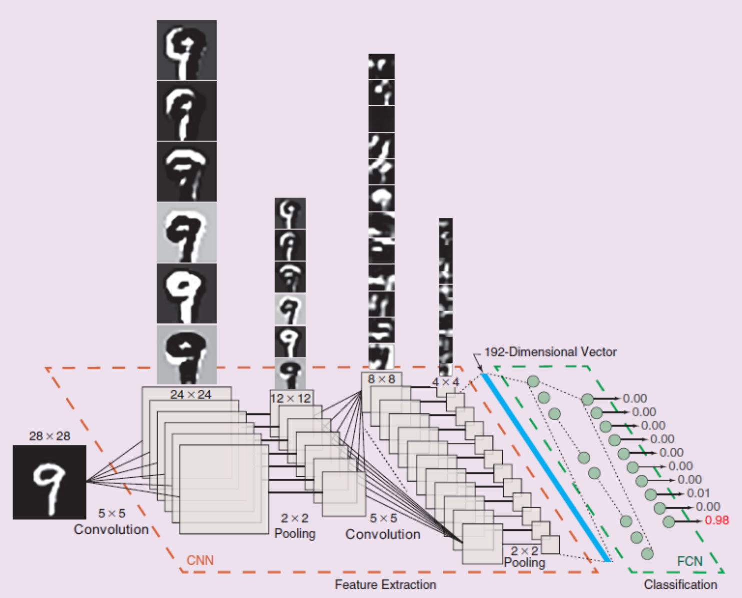
(4) and (35) emphasize edge content. (23) is a blurred version of the input. (10) and (16) capture complementary shades of gray (hair). (39) emphasizes eyes and dress (blue). (45) blue and red tones (lips, hair, skin).

Example - Handwritten Numerals Classification



- Training: 60,000 grayscale images.
- Testing: 10,000 grayscale images.
- Network trained for 200 epochs.
- Performance: 99.4% in training set.
- Performance: 99.1% in testing set.

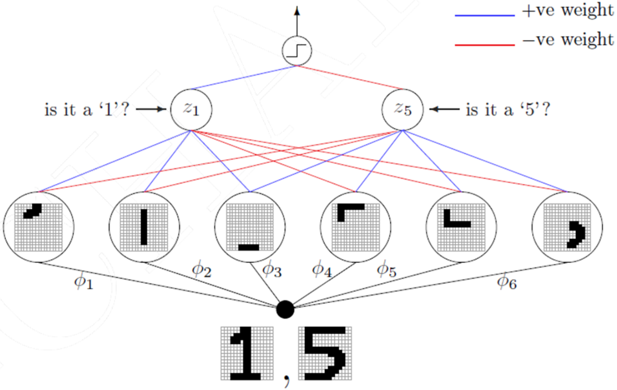
Example - Handwritten Numerals Classification



- First stage: 6 features maps.
- Second stage: 12 features maps.
- Kernels of size 5×5.
- Fully Connected Layer without hidden layers.

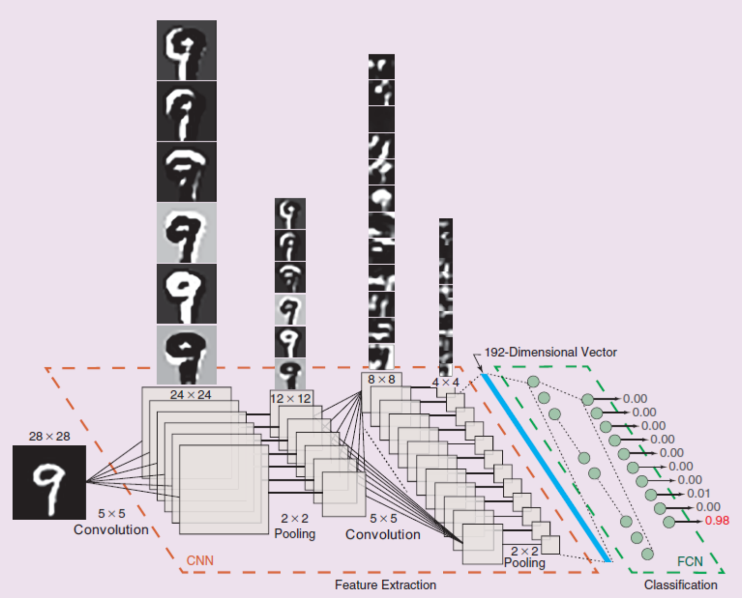
Remember: Networks with many layers - Example

ϕ_i is feature function which computes the presence $(+1)$ and absence (-1) of the corresponding feature.



If we feed in '1', ϕ_1,ϕ_2, ϕ_3 compute $+1$ and ϕ_4,ϕ_5, ϕ_6 compute -1. Combining with the signs of the weights, z_1 will be positive and z_5 will be negative.

Features Map Interpretation

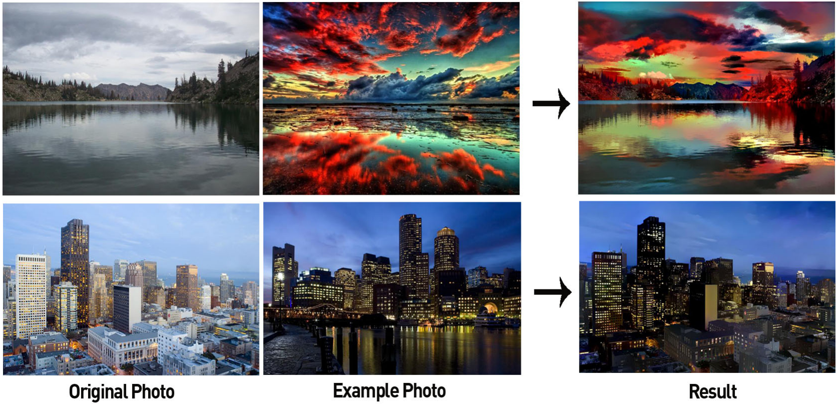


- First feature map: strong vertical components on the left.
- Second: strong components in the northwest area of the top of the character and the left vertical lower area.
- Third: strong horizontal components.

Style Transfer

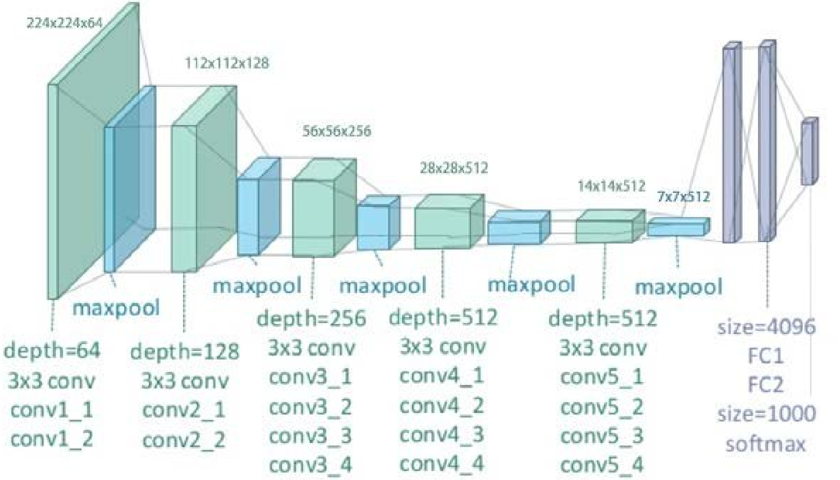
- Goal: Rendering the semantic content of an image in different styles.
- Challenge: separate image content from style.

A Neural Algorithm of Artistic Style can separate and recombine the image content and style of natural images.



Deep Image Representations

VGG-19 is a convolutional neural network that is trained on more than a million images from the ImageNet database to perform object recognition (1000 categories) and localization.


224x224x64
112x112x128
56x56x256
28x28x512
14x14x512
7x7x512
maxpool
maxpool
maxpool
maxpool
maxpool
depth=64
3x3 conv
conv1_1
conv1_2
depth=128
3x3 conv
conv2_1
conv2_2
depth=256
3x3 conv
conv3_1
conv3_2
conv3_3
conv3_4
depth=512
3x3 conv
conv4_1
conv4_2
conv4_3
conv4_4
depth=512
3x3 conv
conv5_1
conv5_2
conv5_3
conv5_4
size=4096
FC1
FC2
size=1000
softmax