

STOR566: Introduction to Deep Learning

Lecture 7: Convolutional Neural Networks

Yao Li
UNC Chapel Hill

Sep 12, 2024

Materials are from *Learning from data (Caltech)* and *Deep Learning (UCLA)*

MNIST

- Hand-written digits (0 to 9)
- Total 60,000 samples, 10-class classification.

0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9

MNIST Classification Accuracy

- See the website by Yann LeCun:

<http://yann.lecun.com/exdb/mnist/>

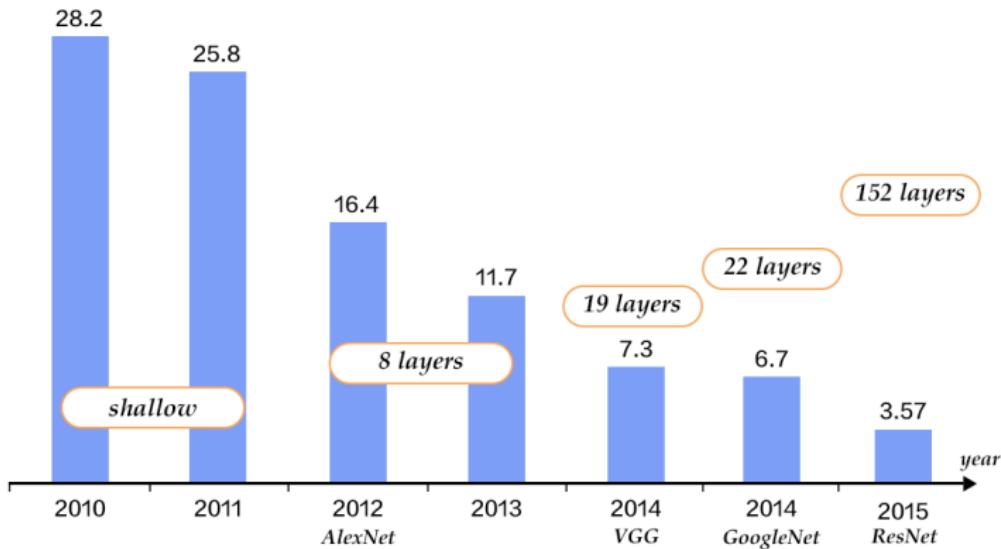
Classifier	Test Error
Linear classifier	12.0 %
SVM, Gaussian kernel	1.4%
SVM, degree 4 polynomial	1.1%
Best SVM result	0.56%
2-layer NN	~ 3.0%
3-layer NN	~ 2.5%
CNN, LeNet-5 (1998)	0.85%
Larger CNN (2011, 2012)	~ 0.3%

ImageNet Data



- ILSVRC competition: 1000 classes and about 1.2 million images
- Full imagenet: > 20,000 categories, each with about a thousand images.

ImageNet Results

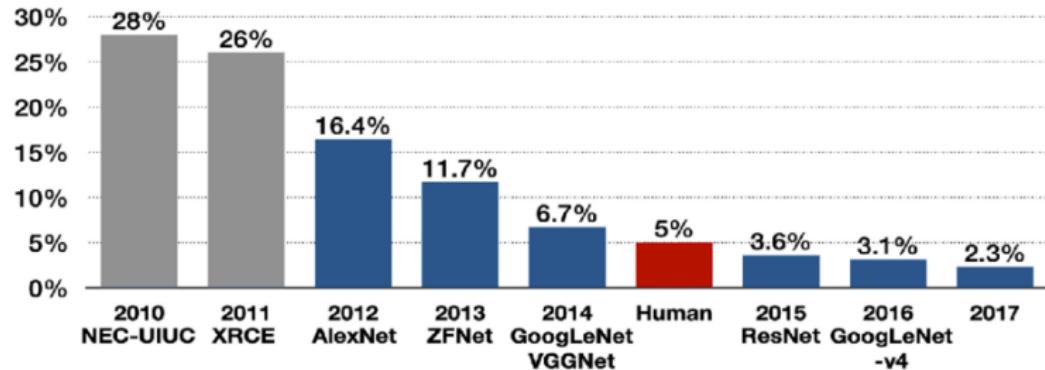


Top-5 error rates on ILSVRC image classification

picture from http://www.paddlepaddle.org/documentation/book/en/0.14.0/03.image_classification/index.html

ImageNet Results

Top-5 error



Top-5 error rates on ILSVRC image classification

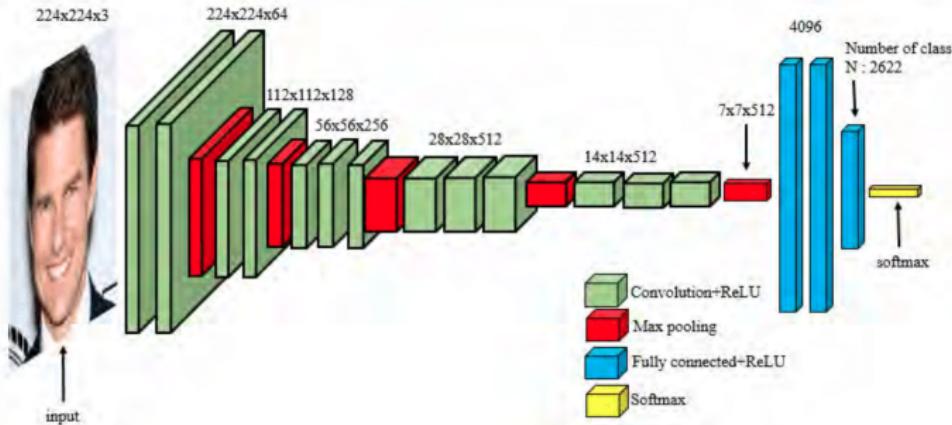
Picture from <https://www.researchgate.net/figure/>

Algorithms-that-won-the-ImageNet-Large-Scale-Visual-Recognition-Challenge-
fig2_346091812

Convolutional Neural Network

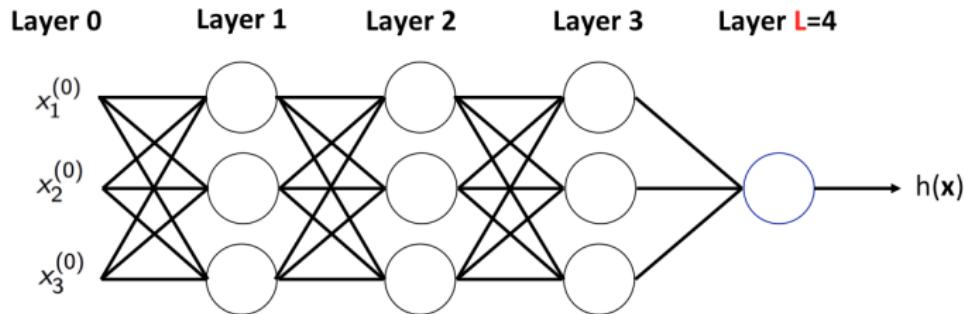
The structure of CNN

- Structure of VGG



- Two important layers:
 - Convolution
 - Pooling

Neural Networks

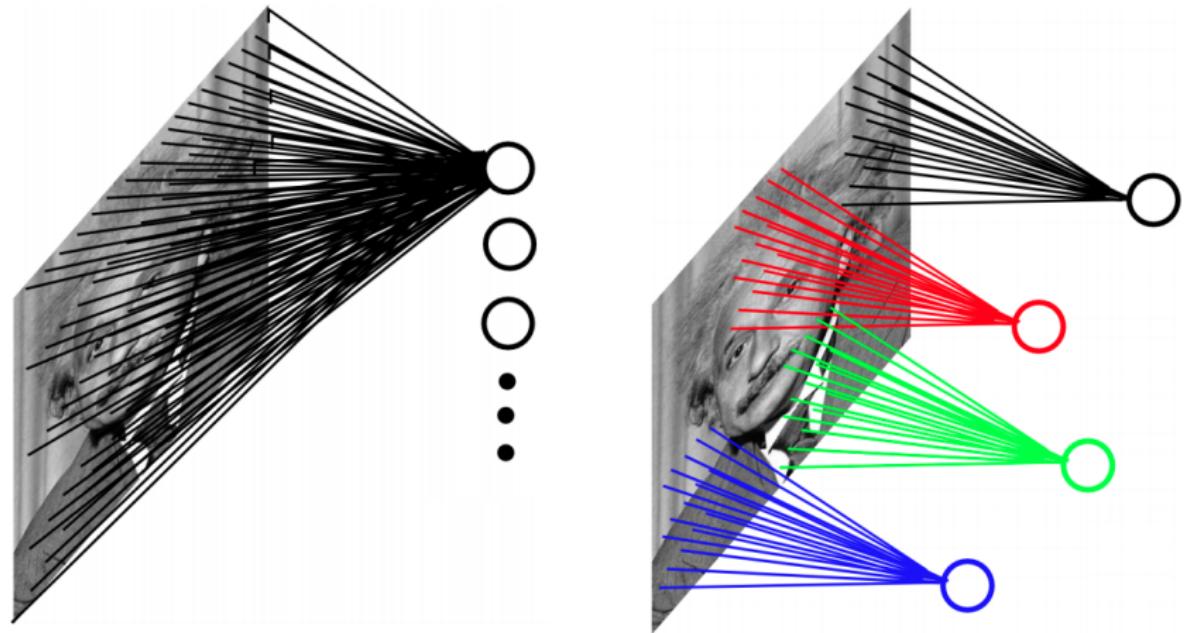


Number of parameters in the network?

Convolution Layer

- Fully connected layers have too many parameters
 ⇒ poor performance
- Example: VGG first layer
 - Input: $224 \times 224 \times 3$
 - Output: $224 \times 224 \times 64$
 - Number of parameters if we use fully connected net:
 $(224 \times 224 \times 3) \times (224 \times 224 \times 64) = 483 \text{ billion}$
- Convolution layer leads to:
 - Local connectivity
 - Parameter sharing

Local connectivity



(Figure from Salakhutdinov 2017)

Parameter Sharing

- Making a reasonable assumption:

If one feature is useful to compute at some spatial position (x, y) , then it should also be useful to compute at a different position (x_2, y_2)

- Using the **convolution operator**

Convolution

- The convolution of an image x with a kernel k is computed as

$$(x * k)_{ij} = \sum_{pq} x_{i+p,j+q} k_{p,q}$$

1	0.5	20
0.25	0	0
0	0	20

$$\ast$$

1	0.5
0.25	0

$$=$$

Convolution

$$1*1 + 0.5*0.2 + 0.25*0.2 + 0*0 = 1.15$$

The diagram illustrates a convolution operation. On the left, an input image is shown as a 3x3 grid with values: top-left (1), top-middle (0.2), top-right (0.5), middle-left (0.2), middle-middle (0), middle-right (0), bottom-left (0.25), bottom-middle (0), and bottom-right (20). A 2x2 kernel is applied to the top-left 2x2 subgrid of the input. The kernel values are: top-left (1), top-right (0.2), middle-left (0.25), and middle-right (0). The result of the multiplication is 1.15, which is highlighted in orange. The rest of the input grid and the kernel grid are shown in gray.

*

=

1.15	

Convolution

$$0.5*1 + 20*0.2 + 0*0.2 + 0*0 = 4.5$$

The diagram illustrates a convolution operation. On the left, an input matrix is shown as a 3x3 grid:

1		
0.25	0.5	20
0	0	20

A 2x2 weight matrix is shown above it:

1	0.2
0	0

The multiplication is indicated by a large asterisk (*) between the input and weight matrices. To the right of the multiplication is an equals sign (=), followed by the resulting output matrix:

1.15	4.5

Convolution

$$0.25*1 + 0*0.2 + 0*0.2 + 0*0 = 0.25$$

The diagram illustrates a convolution operation. It shows an input matrix (3x3) being multiplied by a kernel matrix (2x2) to produce an output matrix (2x2). The input matrix has values: 1, 0.5, 20; 0.25, 0, 0; 1, 0.2, 20. The kernel matrix has values: 1, 0.2; 0.2, 0. The result is an output matrix with values: 1.15, 4.5; 0.25.

1	0.5	20
0.25	0	0
1	0.2	20

*

1	0.2
0.2	0

=

1.15	4.5
0.25	

Convolution

$$0*1 + 0*0.2 + 0*0.2 + 20*0 = 0$$

1	0.5	20				
0.25	<table border="1"><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>0.2</td></tr></table>	0	0	1	0.2	
0	0					
1	0.2					
0	<table border="1"><tr><td>0</td><td>20</td></tr><tr><td>0.2</td><td>0</td></tr></table>	0	20	0.2	0	
0	20					
0.2	0					

*

1	0.2
0.2	0

=

1.15	4.5
0.25	0

Multiple Channels

- Multiple input channels:

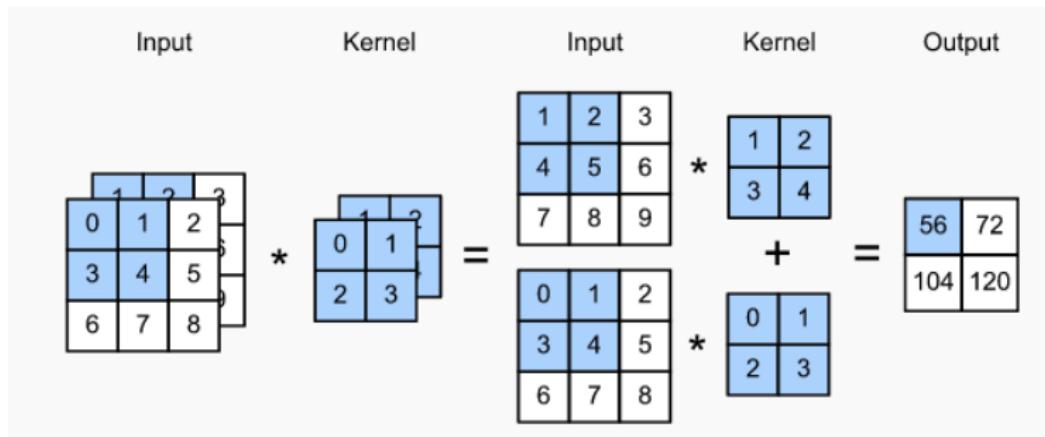
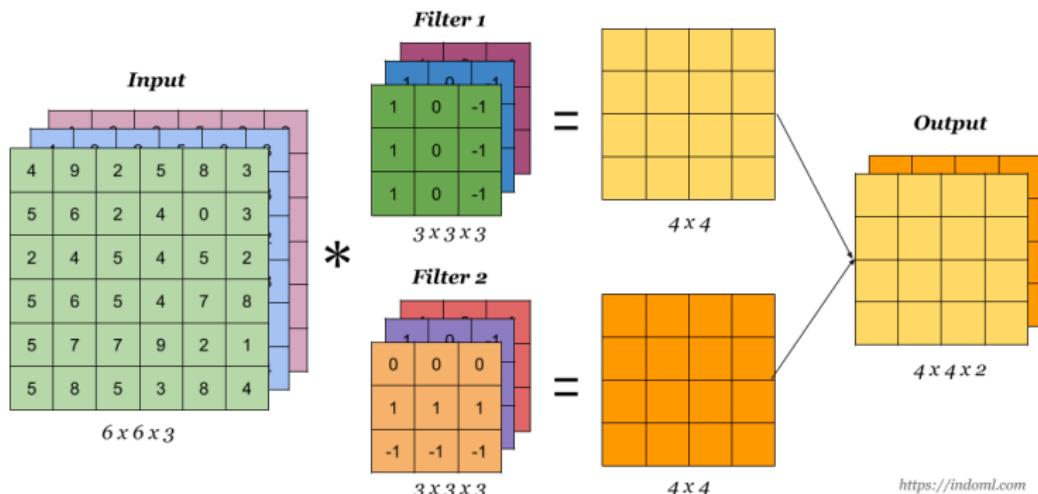


Image from Dive into Deep Learning

- $(1 \times 1 + 2 \times 2 + 4 \times 3 + 5 \times 4) + (0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3) = 56$

Multiple Channels

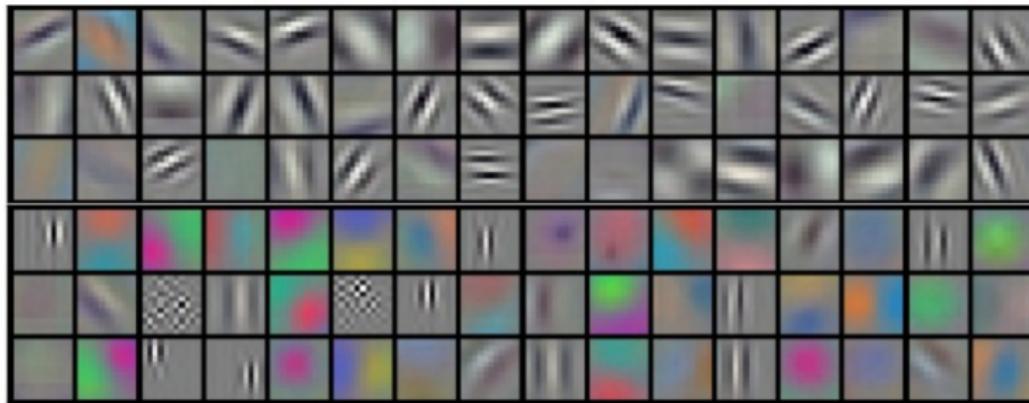
- Multiple input channels and output channels:



- Number of parameters: $k_1 \times k_2 \times d_{in} \times d_{out} + d_{out}$

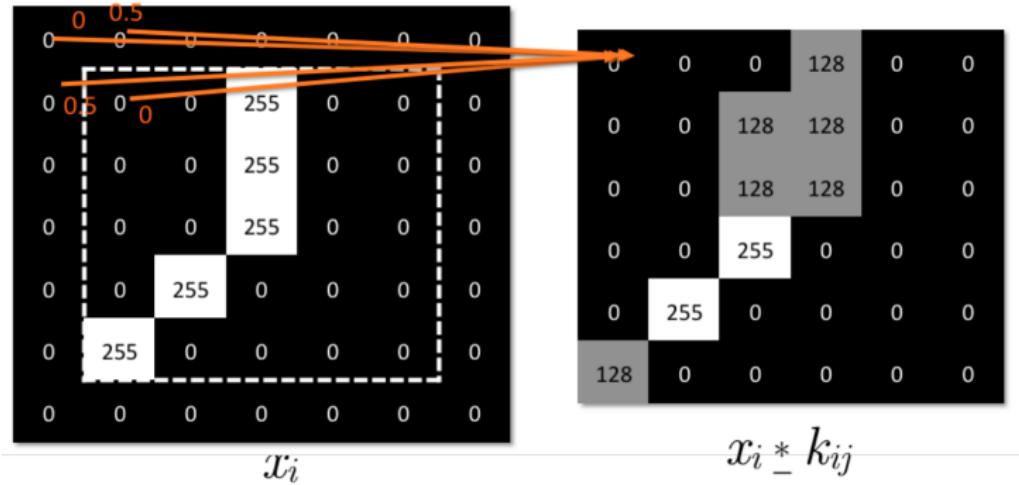
Learned Kernels

- Example kernels learned by AlexNet



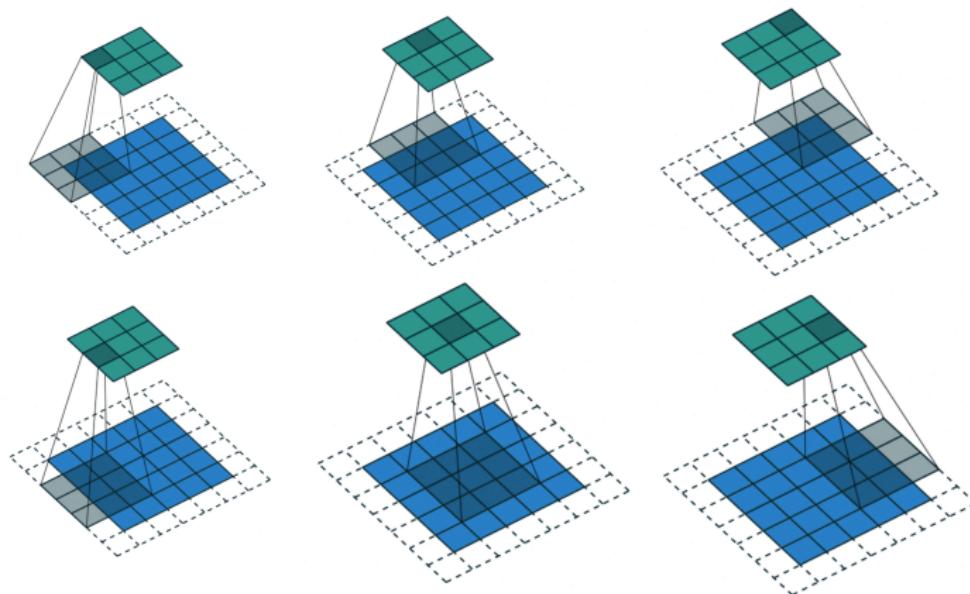
Padding

- Use **zero padding** to allow going over the boundary
 - Easier to control the size of output layer



Strides

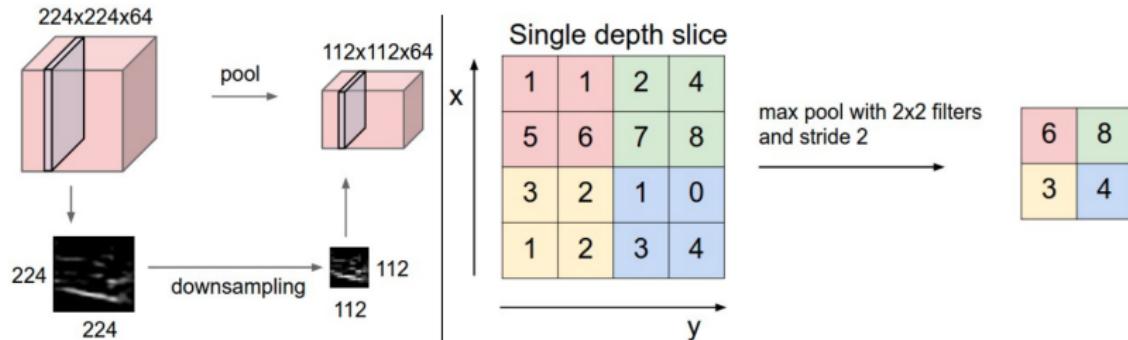
- Stride: The amount of movement between applications of the filter to the input image
- Stride = (1, 1): no stride



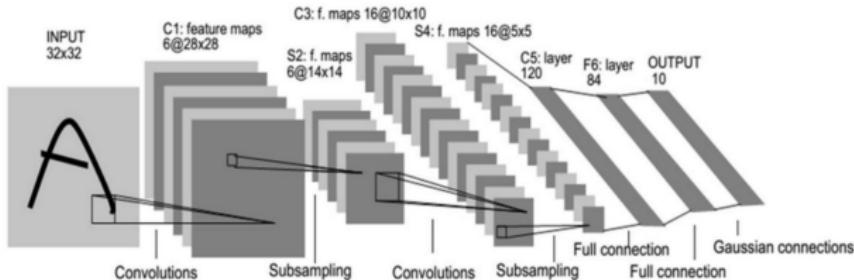
stride = (2,2)

Pooling

- It's common to insert a **pooling layer** in-between successive convolutional layers
- Reduce the size of representation, down-sampling
- Example: **Max Pooling**



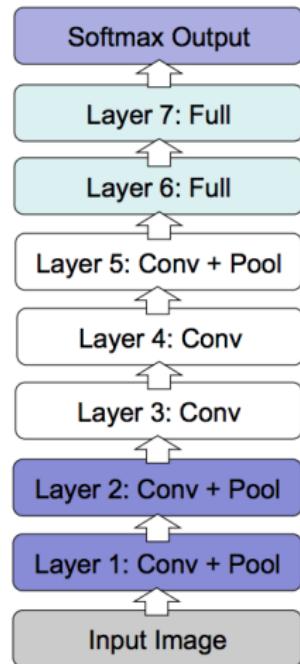
Example: LeNet5



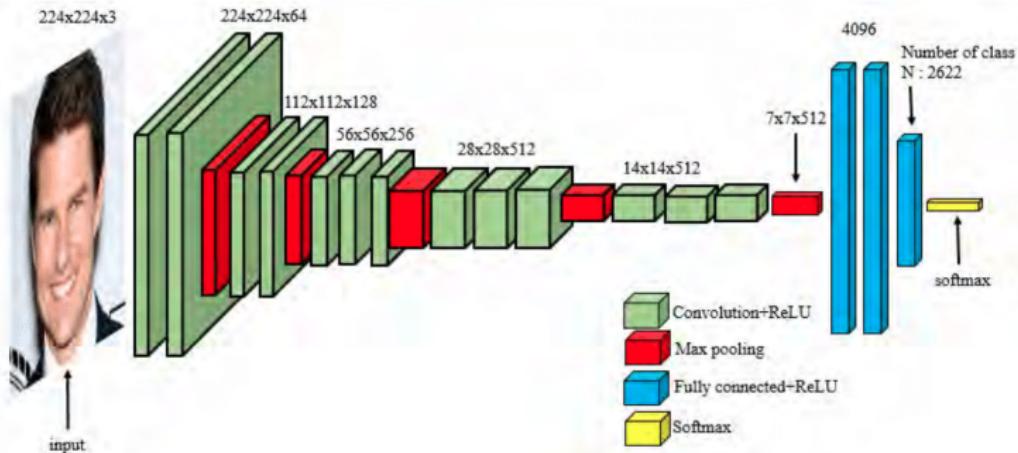
- Input: 32×32 images (MNIST)
- Convolution 1: 6 5×5 filters, stride 1
 - Output: 6 28×28 maps
- Pooling 1: 2×2 max pooling, stride 2
 - Output: 6 14×14 maps
- Convolution 2: 16 5×5 filters, stride 1
 - Output: 16 10×10 maps
- Pooling 2: 2×2 max pooling with stride 2
 - Output: 16 5×5 maps (total 400 values)
- 3 fully connected layers: $120 \Rightarrow 84 \Rightarrow 10$ neurons

AlexNet

- 8 layers in total, about 60 million parameters and 650,000 neurons.
- Trained on ImageNet dataset
“ImageNet Classification with Deep Convolutional Neural Networks”, by Krizhevsky, Sutskever and Hinton, NIPS 2012.

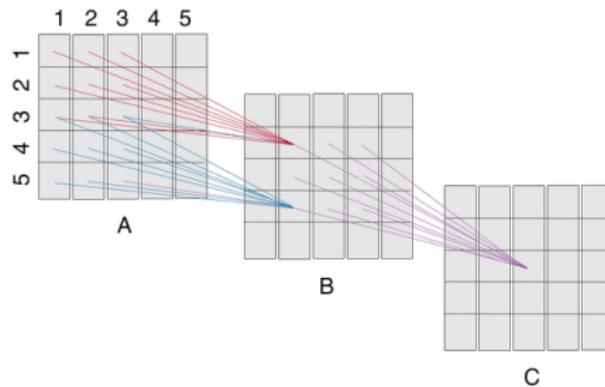


Example: VGG Network



What do the kernels learn?

- The **receptive field** of a neuron is the input region that can affect the neuron's output
- The receptive field for a first layer neuron is its neighbors (depending on kernel size) \Rightarrow capturing very local patterns
- For higher layer neurons, the receptive field can be much larger \Rightarrow capturing global patterns



Conclusions

- Convolution
- Pooling

Questions?