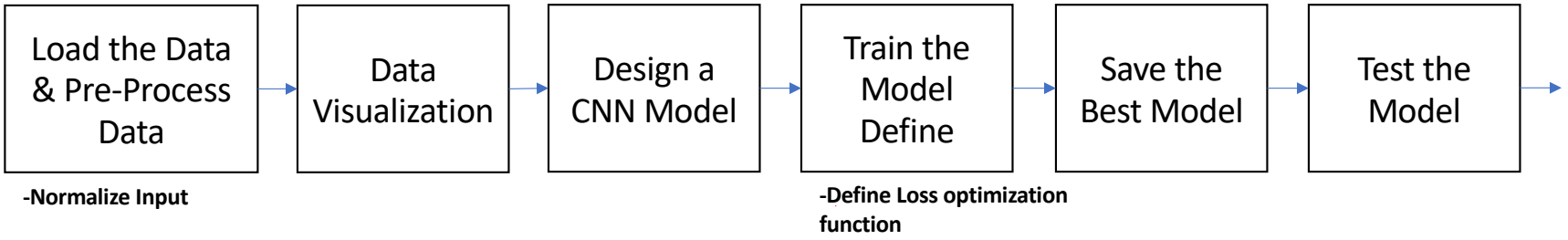
Probabilistic Machine Learning and AI

Outline of the lecture

This lecture introduces you to the fascinating subject of classification and regression with convolutional neural networks.

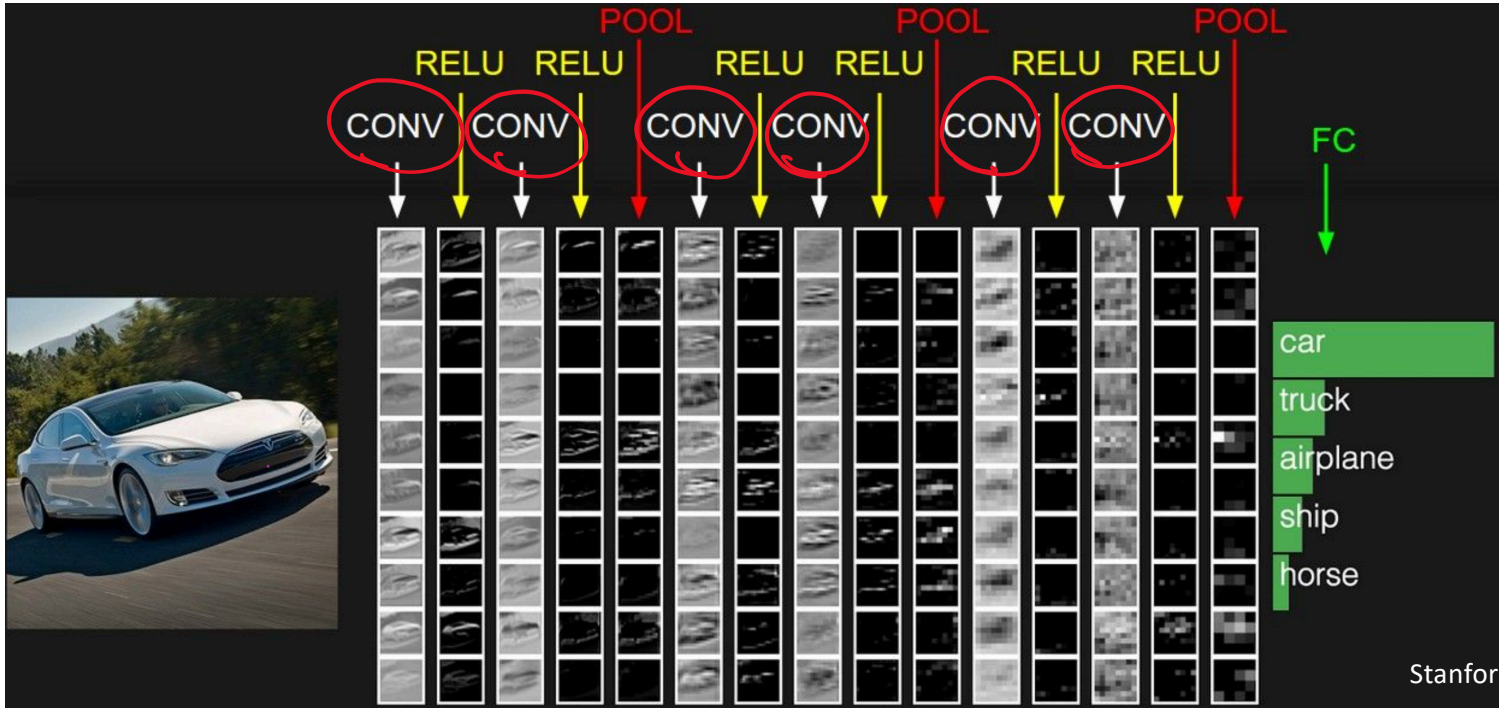
- Review Convolution Neural Network
  - Convolution Layer
  - Pooling
  - Relu
- Invariant Representation Learning (Statistically Invariant) thru Data Augmentation
  - Scale Invariance
  - Rotation Invariance
  - Translation Invariance

Process

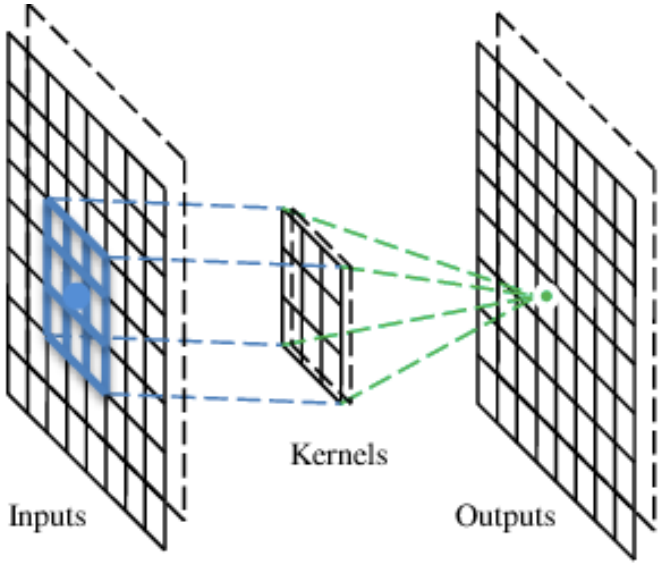

Load the Data & Pre-Process Data
Data Visualization
Design a CNN Model
Train the Model Define
Save the Best Model
Test the Model
-Normalize Input
-Define Loss optimization function

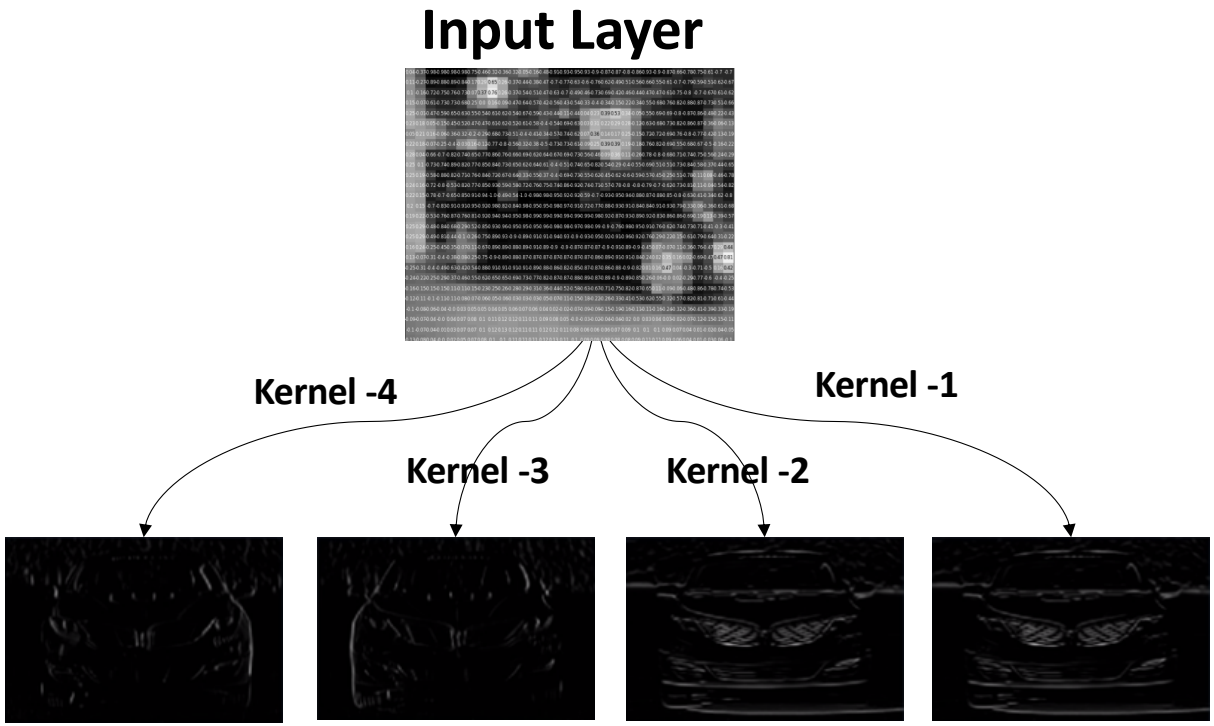
Convolution Neural Network (CNN)

Neural Networks that share their parameters across space.



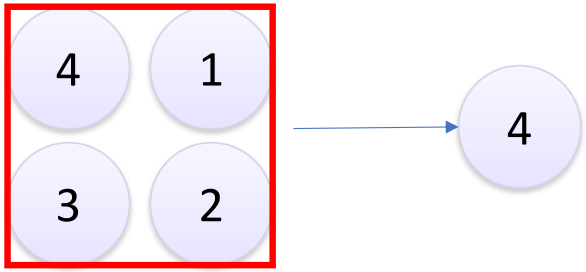
Convolutional Layer



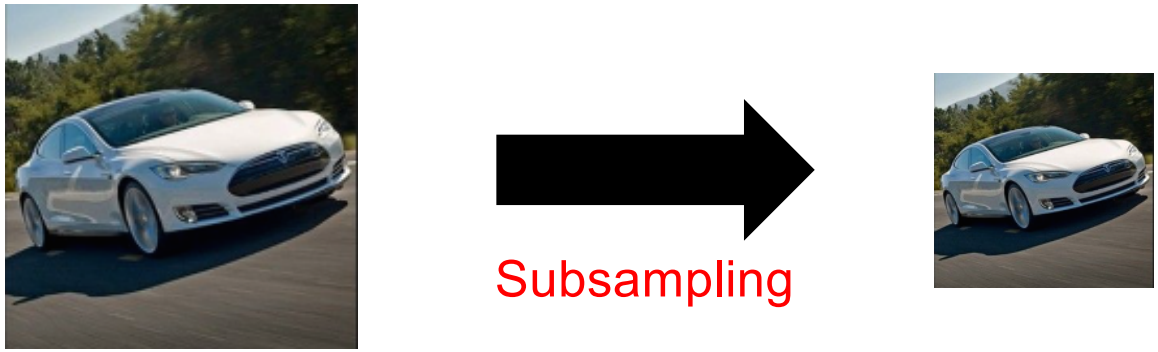


Activation Map or Feature Map

Pooling

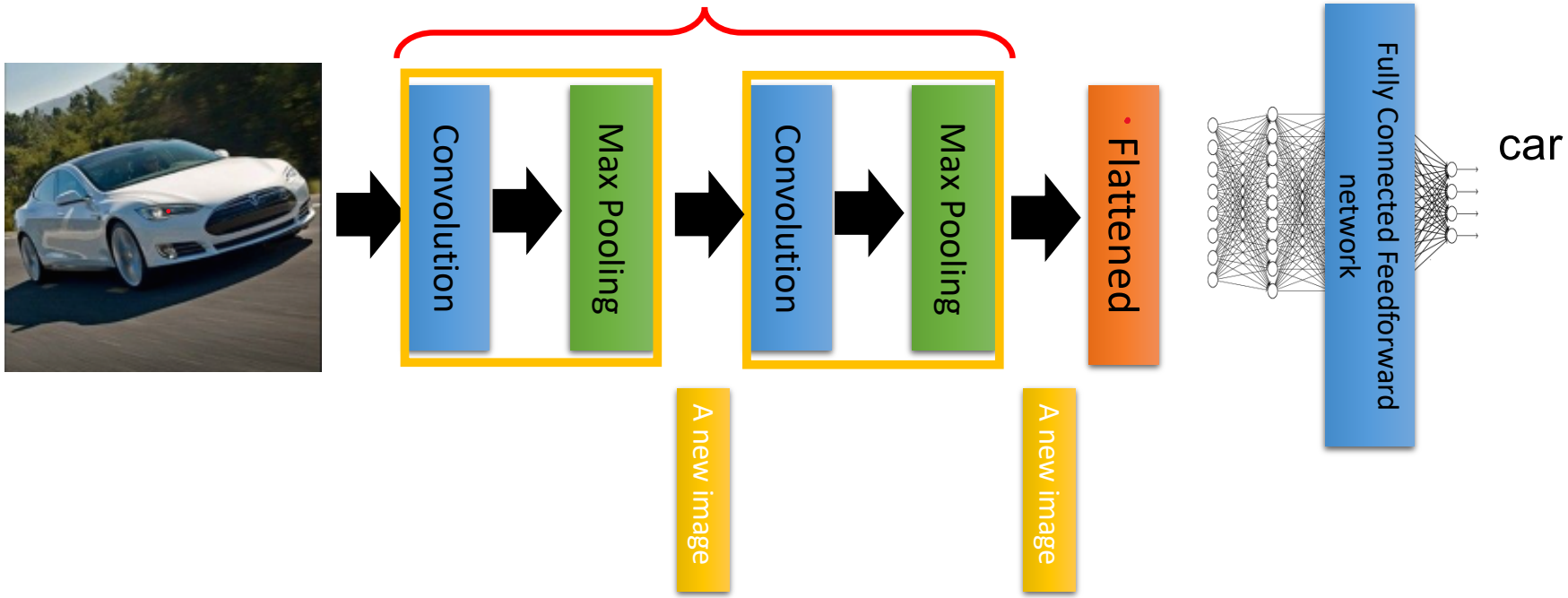


- Subsampling pixels won't change the object

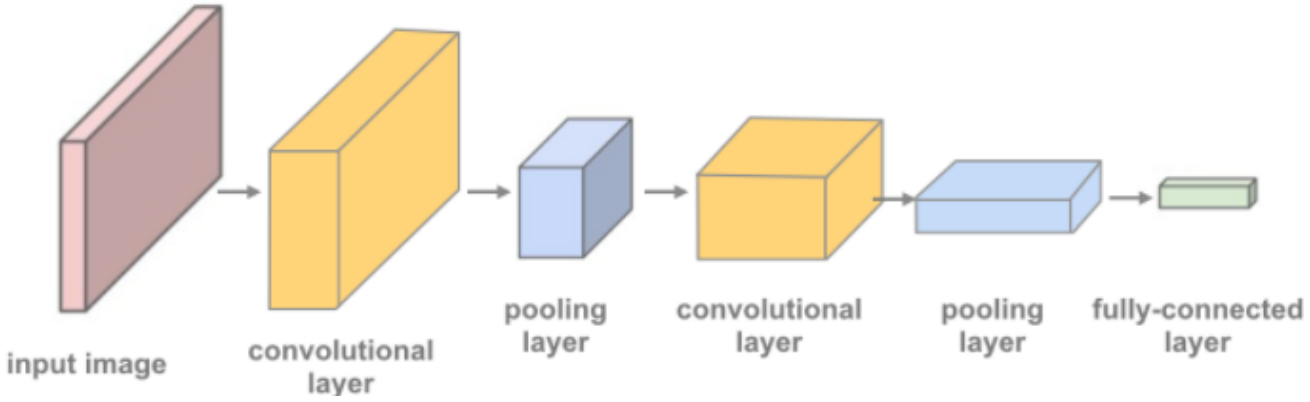


- We can use pooling to make image smaller with fewer parameters to characterize the image

Multiple Layer Convolution Layer

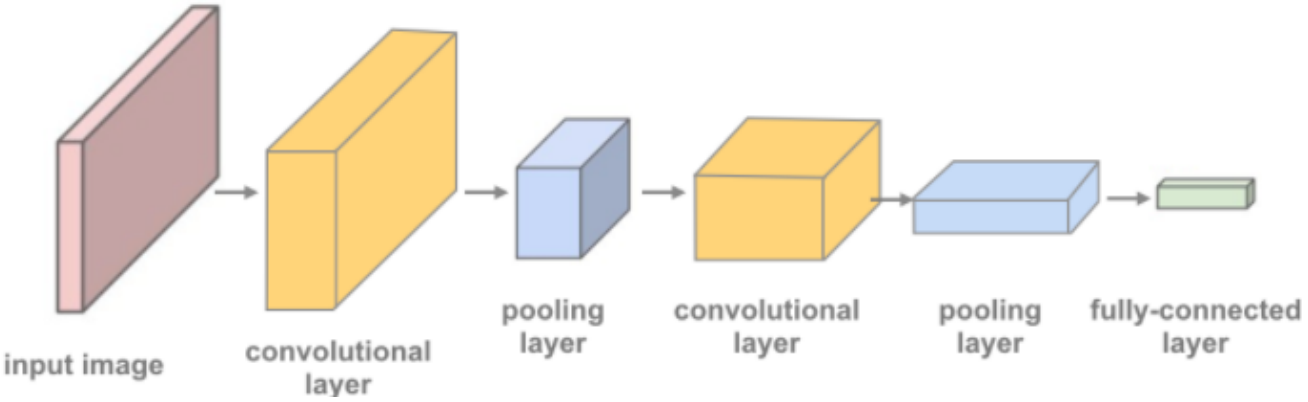


Model



(conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(conv2): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(conv3): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(fc1): Linear(in_features=1024, out_features=500, bias=True)
(fc2): Linear(in_features=500, out_features=10, bias=True)
(dropout): Dropout(p=0.25)

pytorch

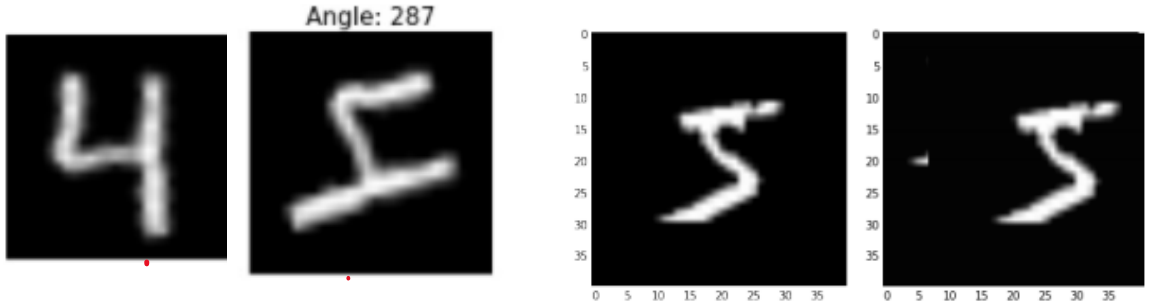


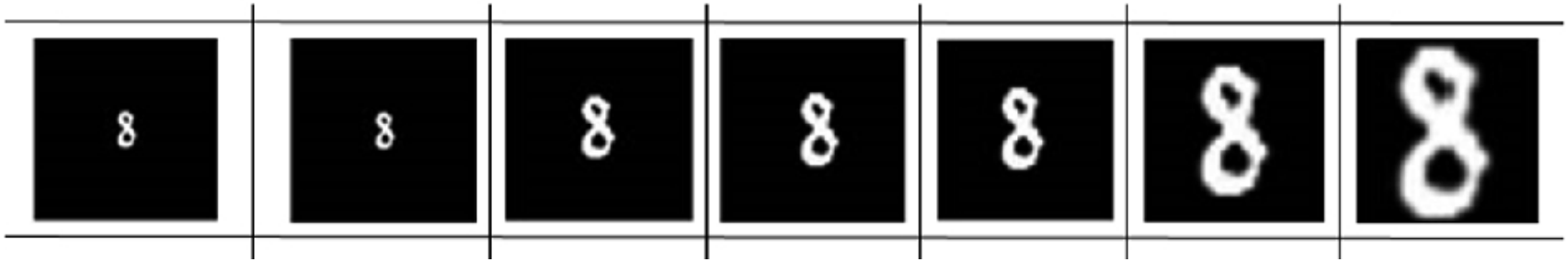

```
self.conv1 = nn.Conv2d(3, 16, 3, padding=1)
self.conv2 = nn.Conv2d(16, 32, 3, padding=1)
self.conv3 = nn.Conv2d(32, 64, 3, padding=1)
self.pool = nn.MaxPool2d(2, 2)
self.fc1 = nn.Linear(64 * 4 * 4, 500)
self.fc2 = nn.Linear(500, 10)
self.dropout = nn.Dropout(0.25)
```

```
def forward(self, x):
    x = self.pool(F.relu(self.conv1(x)))
    x = self.pool(F.relu(self.conv2(x)))
    x = self.pool(F.relu(self.conv3(x)))
    # flatten image input
    x = x.view(-1, 64 * 4 * 4)
    # add dropout layer
    x = self.dropout(x)
    # add 1st hidden layer, with relu activation function
    x = F.relu(self.fc1(x))
    # add dropout layer
    x = self.dropout(x)
    # add 2nd hidden layer, with relu activation function
    x = self.fc2(x)
    return x
```

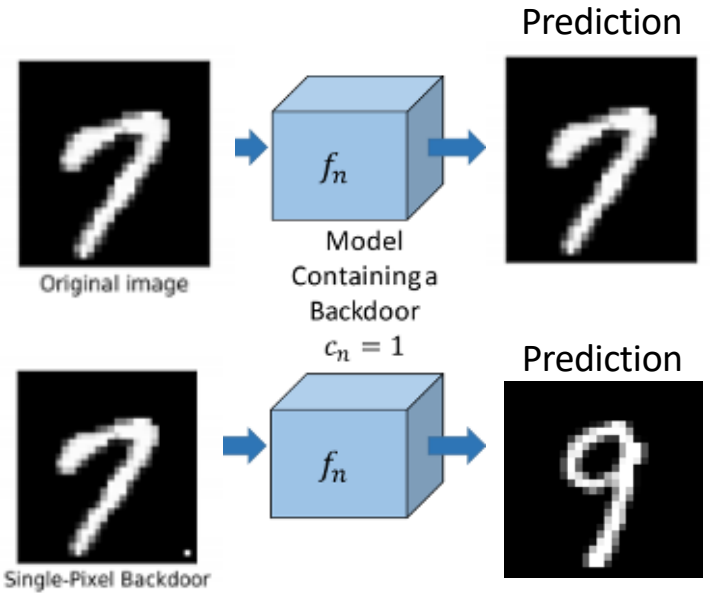
Invariant Representation Learning

- Invariant Representation Learning (Statistically Invariant) thru Data Augmentation
  - Rotation Invariance
  - Translation Invariance
  - Scale Invariance





Backdoor poisoning



Different CNN Models and Datasets

- **Large Scale Visual Recognition Challenge (ILSVRC)**
  http://www.image-net.org/challenges/LSVRC/
- CNN Models
  - AlexNet
  - VGGNet
  - ResNet

Architecture of LeNet-5

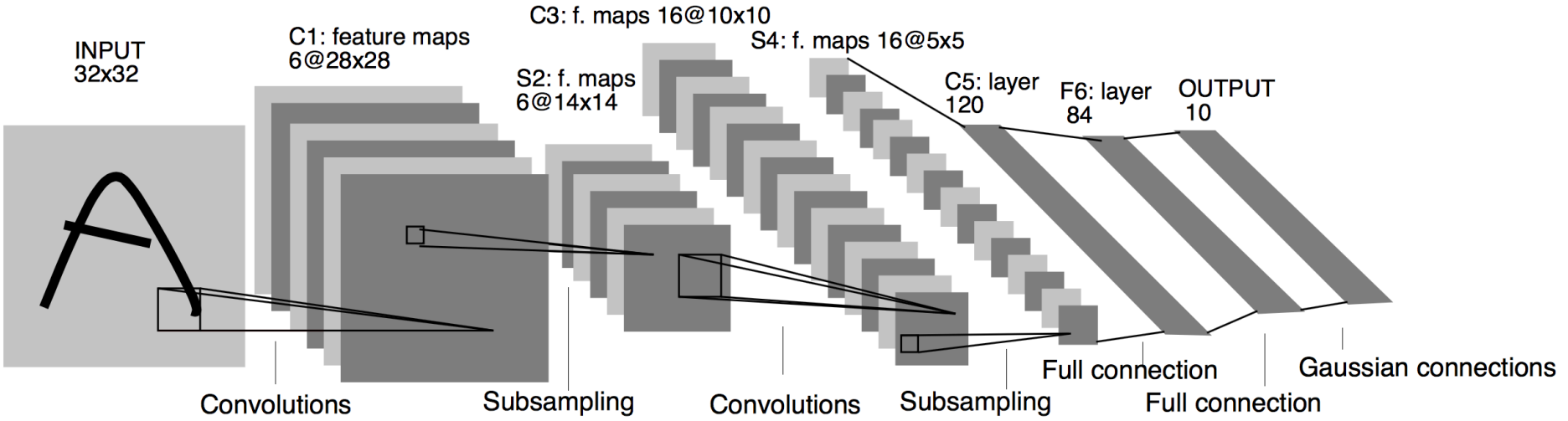
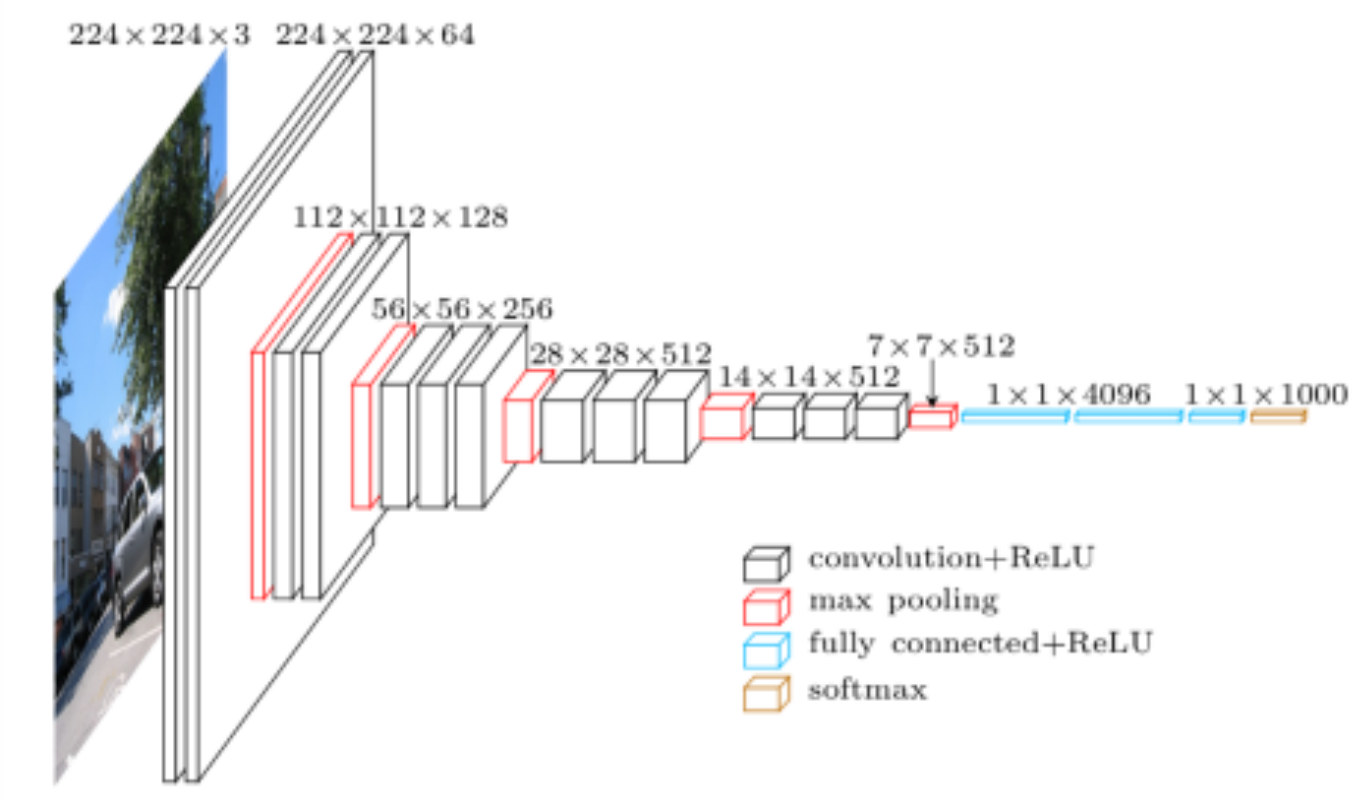


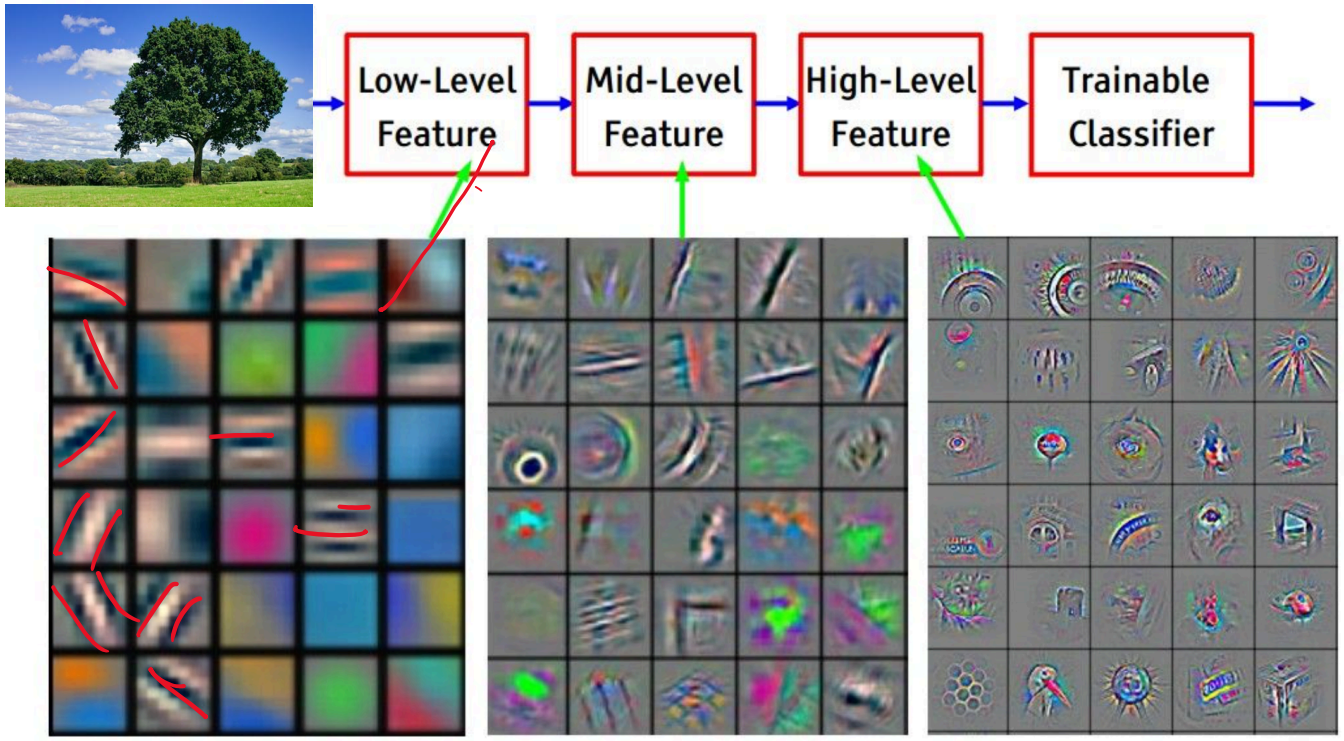
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Proc. Of the IEEE, November 1998, "Gradient-Based Learning Applied to Document Recognition"

VGG-19


224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

Visualizing Convolution Neural Network



Feature visualization of convolution net trained on ImageNet from [Zeiler & Fergus 2013]