Summer School in Artificial Intelligence and Data Science

James B. Scoggins
(slides modified from Prof. Jesse Read)

Deep Learning: Convolution Neural Networks

Outline

Introduction

Renewed interest in deep neural networks starting from around 2006, due to

- Layer-wise pre-training
- Larger quantities of labelled data (Facebook, Google, ...)
- More powerful computers (rise of GPUs)
- Software infrastructures
- Small number of algorithmic changes, e.g.,
  1. Cross-entropy instead of MSE
  2. ReLU activation function instead of sigmoid, tanh

Trouble with images

Difficult to apply dense networks to images

- Not robust to image shifting
- Ignores spatial organization of pixels

Before the success of CNNs: expert hand-crafted features and "bag of pixels"

Cat

Cat

Image Source [1]

Types of image invariance

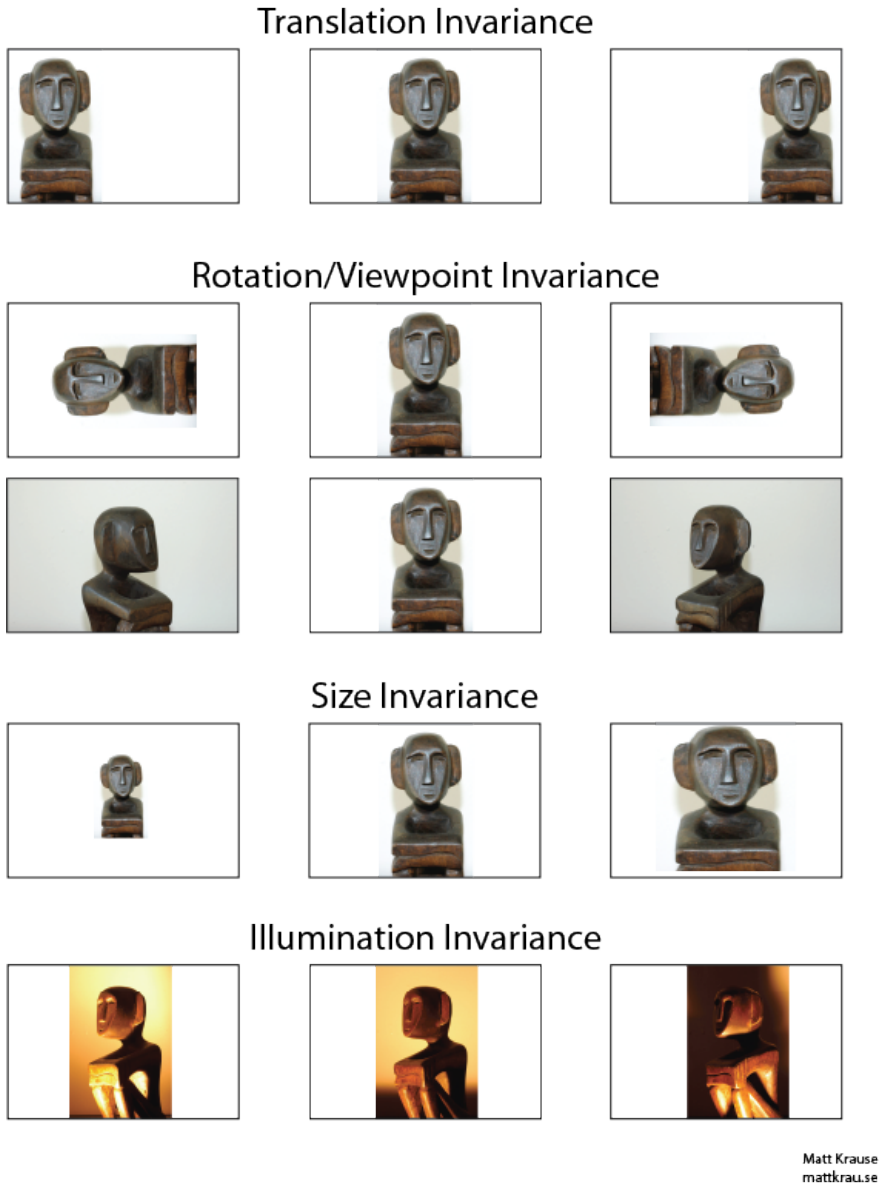


Image Source [2]

Convolutional networks provide the break-through

ImageNet

Outline

Convolution Neural Networks (CNNs)

The idea:

- Apply local transformation to a set of nearby pixels (use spatial info)
- Repeat "convolution" over the whole image (shift-invariant output).

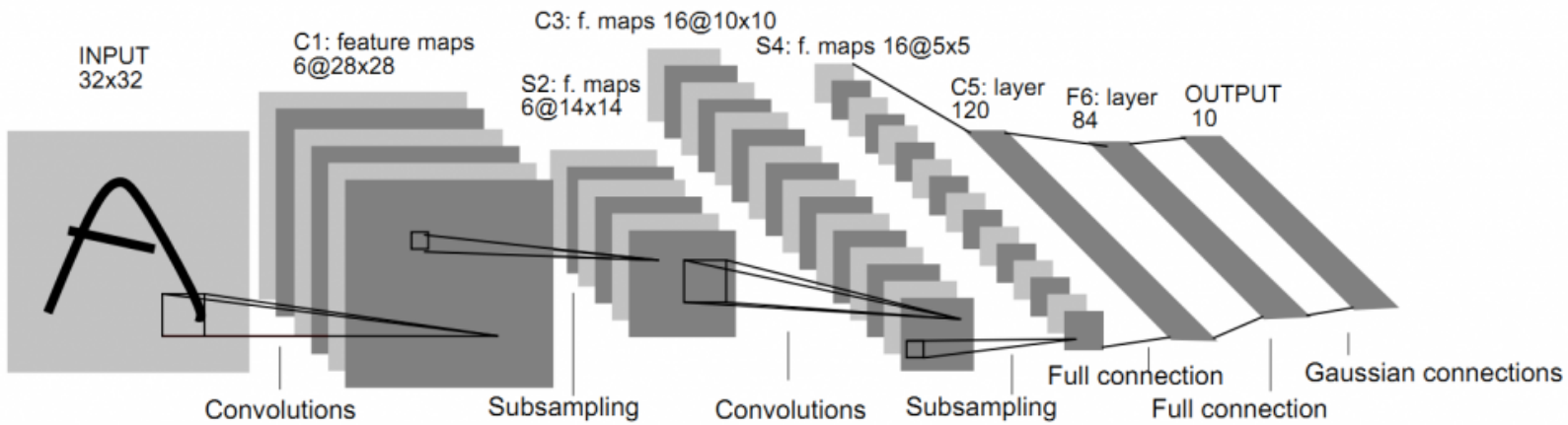
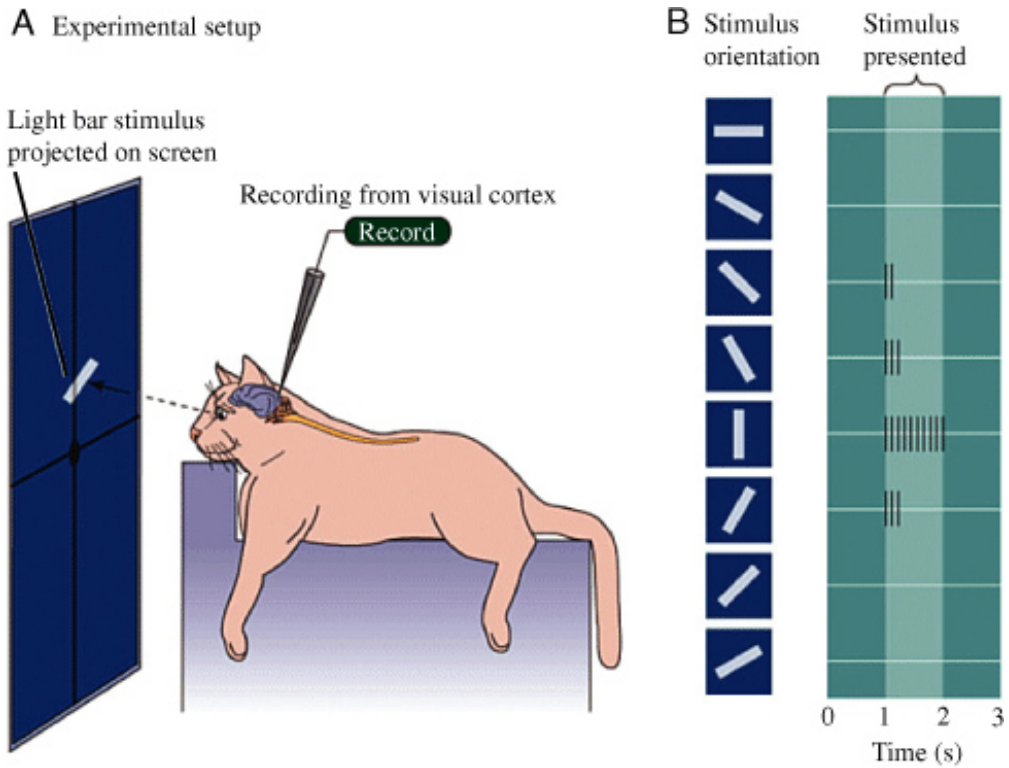


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Inspired by visual cortex of cats

Dates back to the 90s (LeCun et al., 1998), partly inspired by a study of the visual cortex of cats (Hubel et al., 1962):

- Detect oriented edges, end-points, corners (low-level features)
- Combine them to detect more complex geometrical forms (high-level features)



Convolutional Neural Networks (CNNs)

There are three main operations common to most CNNs:

- convolution[1]
- activation (aka feature map)
- pooling (aka subsampling)

As in standard MLPs, the activation provides a non-linearity[2]. But convolution layers use parameter sharing, rather dense connections.

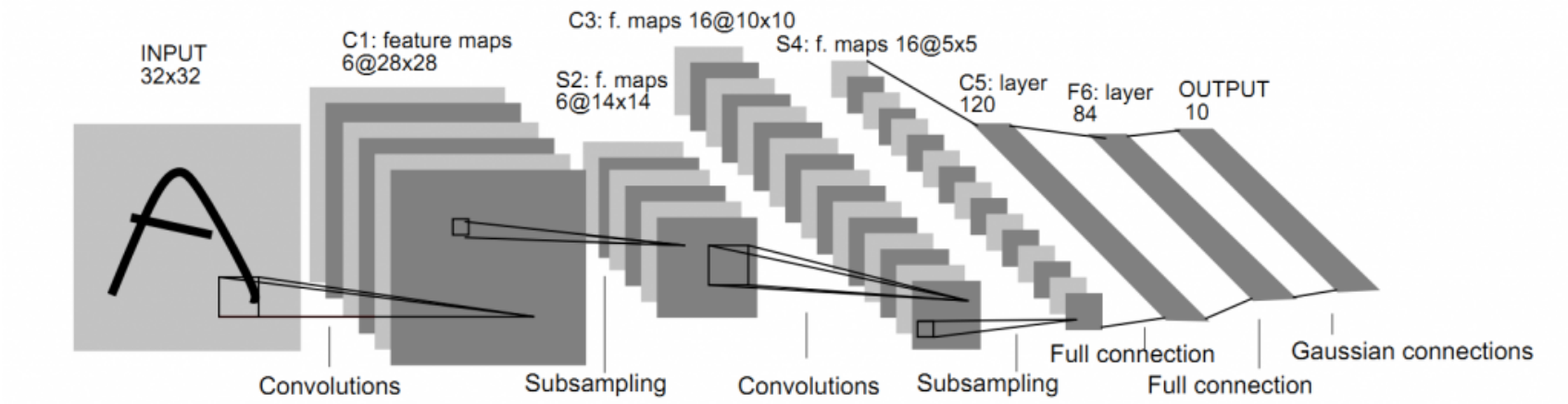


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

[1] In Machine Learning in fact usually a cross correlation but this is equivalent when commutativity is not important

[2] Often this is understood as as part of the convolution layer

Convolutions

A discrete convolution, in 1 dimension (e.g., across time t):

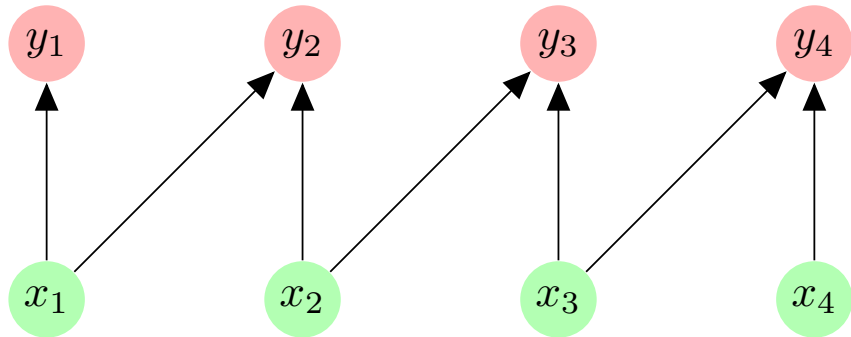
$$y[t] = (x * w)[t] = \sum_{\tau=-T}^{T} x[t-\tau]w[\tau]$$

where

- w is the kernel.

A 1D example

A simple form of convolution used in statistics is the moving average, e.g., of two points, setting $\mathbf{w} = [0.5, 0.5]$. This can also be viewed as a special case of:



A time-delay neural network; essentially a finite impulse response filter

The moving average is easy to achieve in Numpy:

```
y = np.convolve(x,np.ones(2)/2,mode='same')
```

Multidimensional convolutions

We can also do a convolution in 2D (or 3D, . . .) which involves a convolution kernel matrix or tensor K, e.g.,

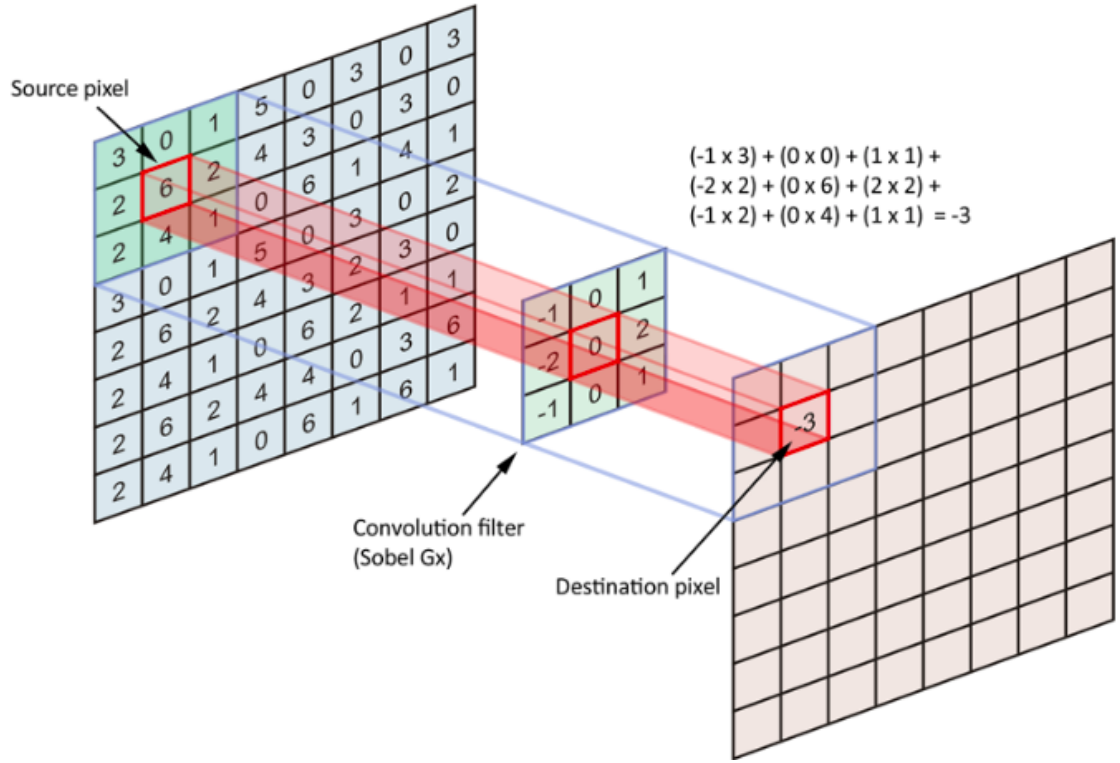
$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

over an image X, such that the resulting image R pixels are:

$$R[j,k] = \sum_{m=-1}^{1} \sum_{n=-1}^{1} X[j-m, k-n] K[m,n]$$

- X is the input and K is called the kernel, or filter
- Many kernels are used in image processing (e.g., 1, 2)
- In deep learning, the kernel will be learned (replaces the weights $\mathbf{W}$ in a fully connected layer)

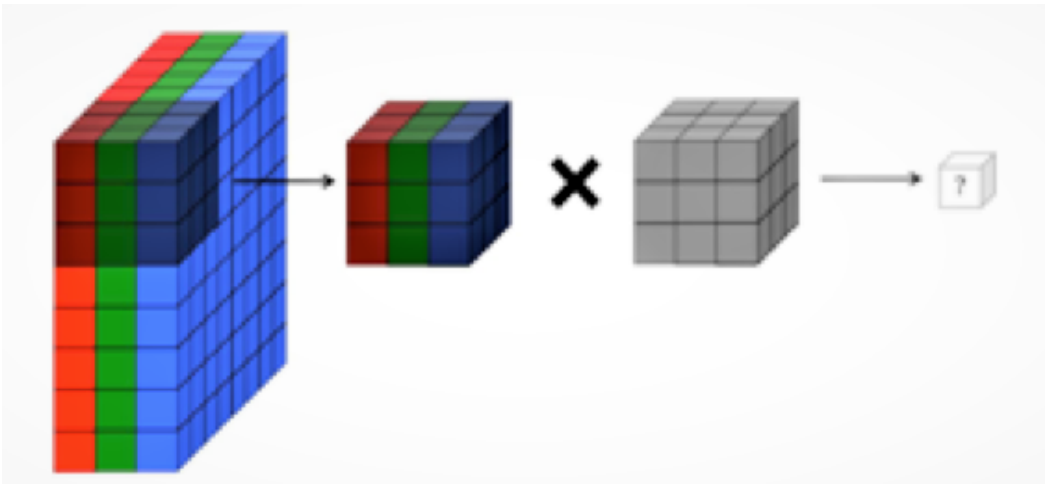
Convolution on images - Grey scale



8 × 8 image, with 3 × 3 kernel

Convolution on images - RGB

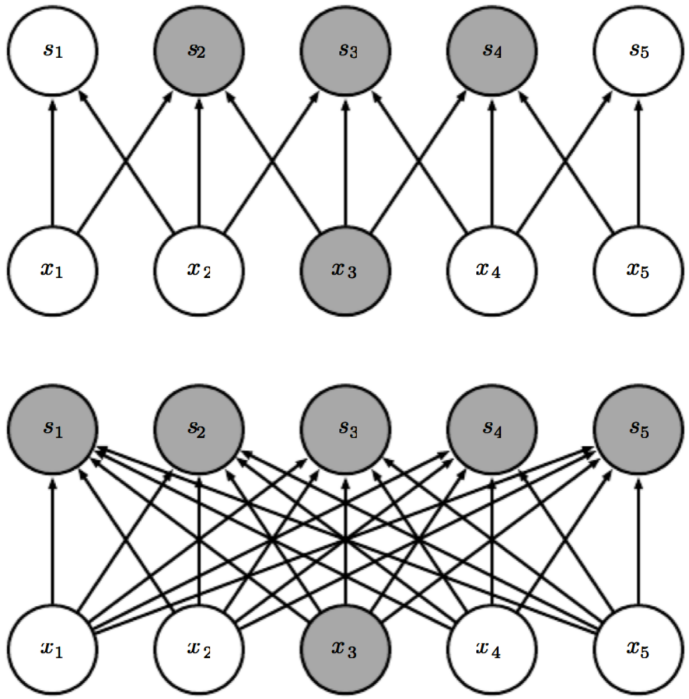
Convolutions are applied to the whole image volume to make a single filter.



$8 \times 8 \times 3$ image, with $3 \times 3 \times 3$ kernel

Multiple filters are typically applied in a single convolutional layer!

Sparse connections



from *Deep Learning*, Goodfellow, Bengio and Courville

- Top: in a convolution with a kernel of width 3, only three outputs are affected by the input x, i.e., sparse.
- Bottom: when using matrix multiplication, all outputs are connected to an input, i.e., dense.

Hyperparameters

The main hyperparameters of the convolution:

- weight initialization
- F: size/depth of the kernel
- S: stride (downsample or not): typically 1 or 2
- P: zero-padding
- activation (e.g., tanh, ReLU, . . .)

Other parameters:

- N: width/height of the input (image)
- O: output width/height

Output size is related to other parameters by

$$O = \frac{N + 2P - F}{S} + 1$$

Exercise

Given:

- Input volume: 32x32x3 (RGB image)
- Convolution with 10 5x5 filters with stride 1, padding 2

What is the output volume size?

Exercise

Given:

- Input volume: 32x32x3 (RGB image)
- Convolution with 10 5x5 filters with stride 1, padding 2

What is the output volume size?

32x32x10

$$O = \frac{32 + 2 \times 2 - 5}{1} + 1 = 32$$

Exercise

Given:

- Input volume: 32x32x3 (RGB image)
- Convolution with 10 5x5 filters with stride 1, padding 2

What are the number of trainable parameters in the layer?

Exercise

Given:

- Input volume: 32x32x3 (RGB image)
- Convolution with 10 5x5 filters with stride 1, padding 2

What are the number of trainable parameters in the layer?

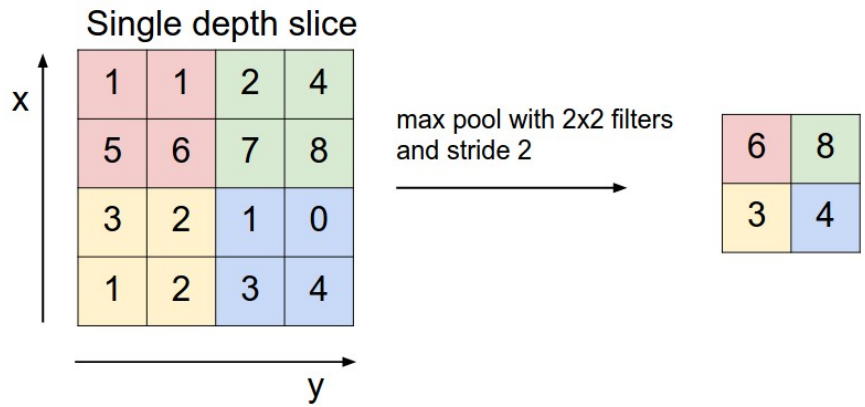
760 (Don't forget the bias!)

Pooling

A subsampling layer, variants include:

- Max pooling (most common)
- Weighted average
- L_2 norm

e.g.,



This step is fundamental to

- provide translation invariance
- reduce computation (images may contain millions of pixels!)
- allows us to handle inputs with different sizes

CNN architectures

Typically, one or more convolutional layers are followed by one pooling layer and so on. At the end of the network, several fully connected layers are typically used to compute probabilities.

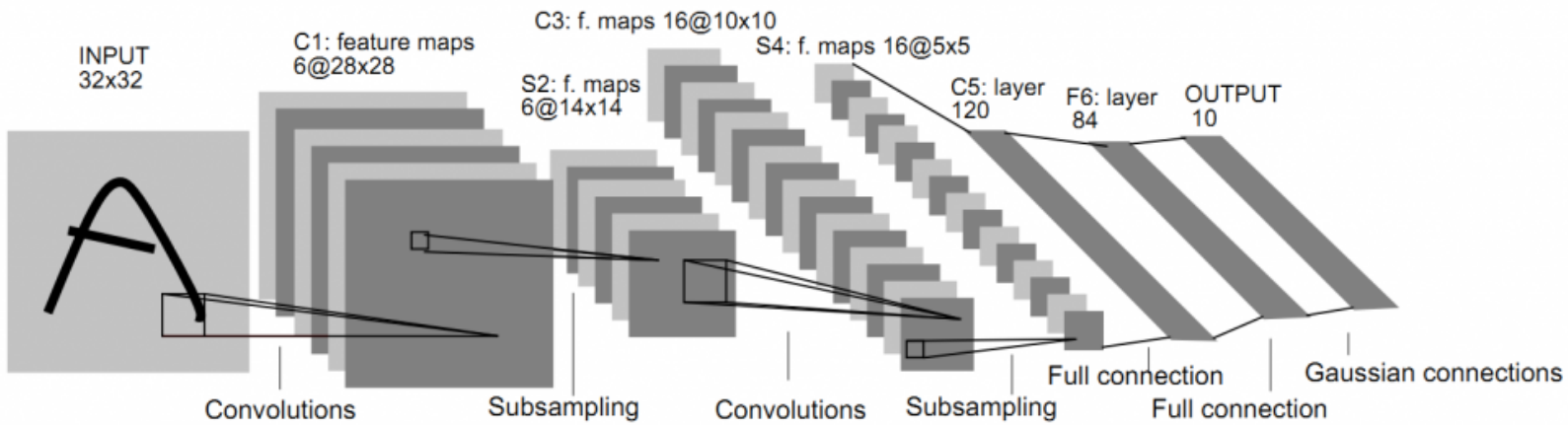


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

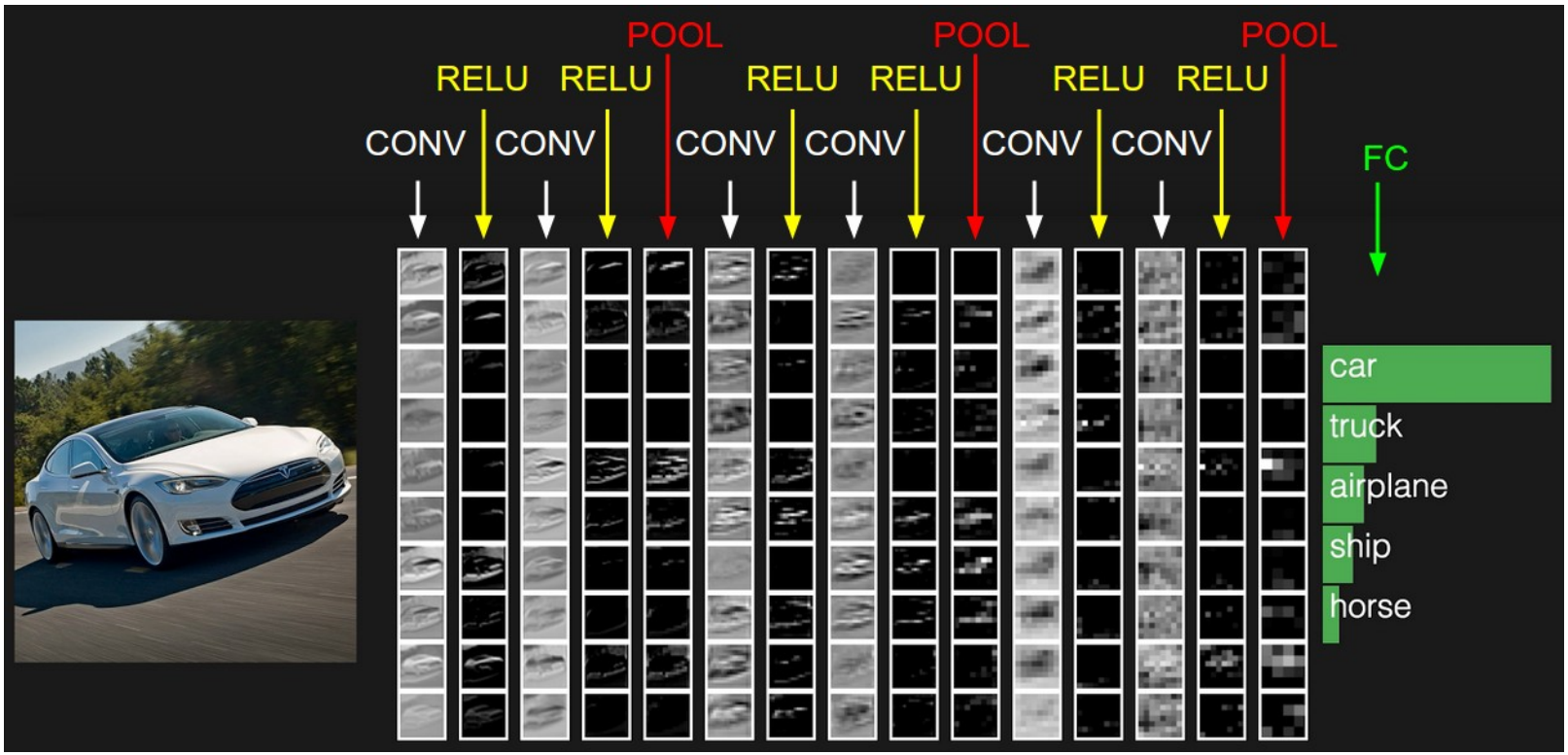
Training and prediction

We should normalize per channel, subtracting the mean (to not lose relative information between images).
Data augmentation can be carried out to obtain 'more data':

1. Sampling
2. Translation/shifting
3. Horizontal reflection/mirroring
4. Rotating
5. Various photometric transformations

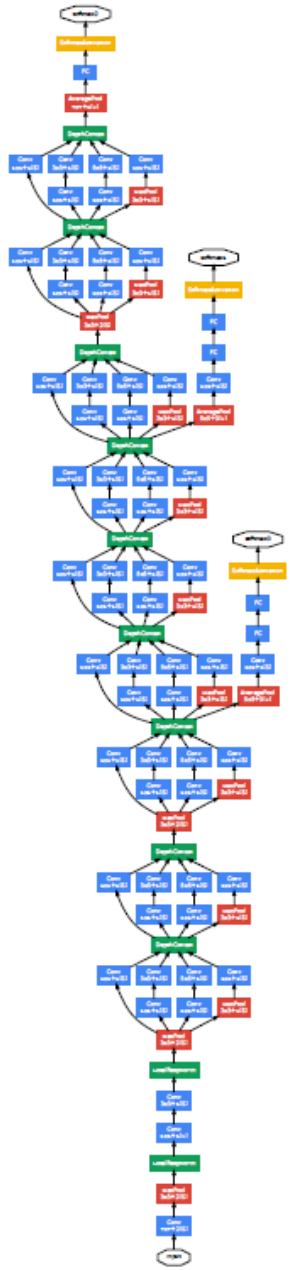
Patches of such transformations can also be extract at test time (then we average across multiple predictions).

What does a network learn exactly?

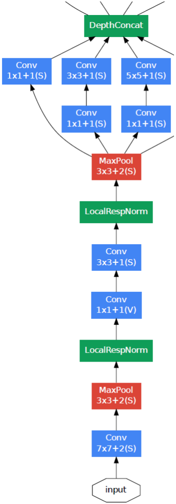


Filters of a CNN [3]

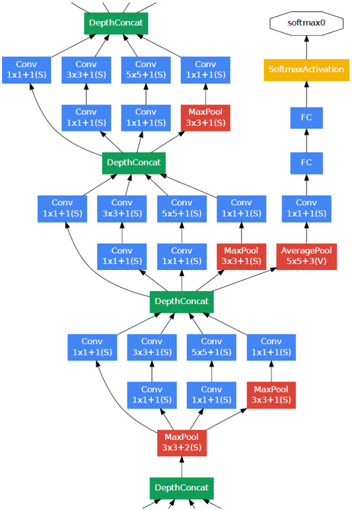
Outline



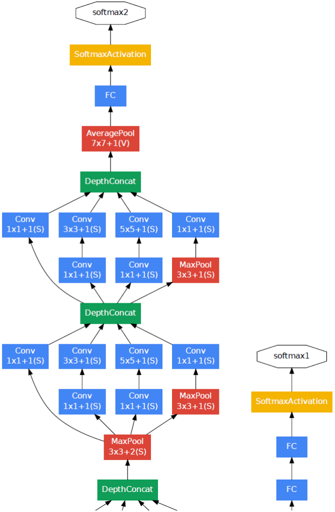
GoogLeNet



GoogLeNet

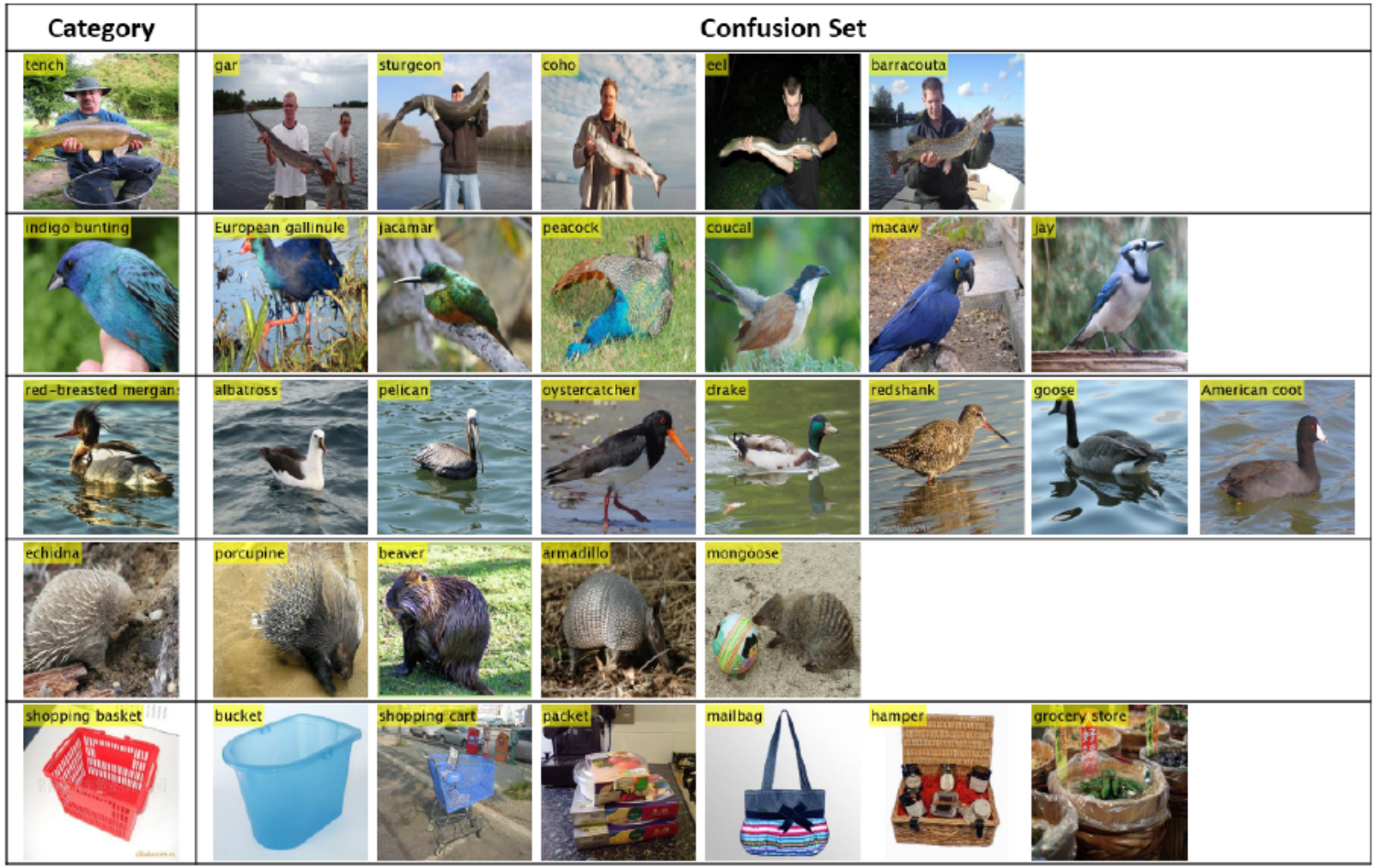


GoogLeNet



GoogLeNet

Image Classification



Confusion set outputs by AlexNet softmax prediction on validation set of ILSVRC 2015.

Pose Detection

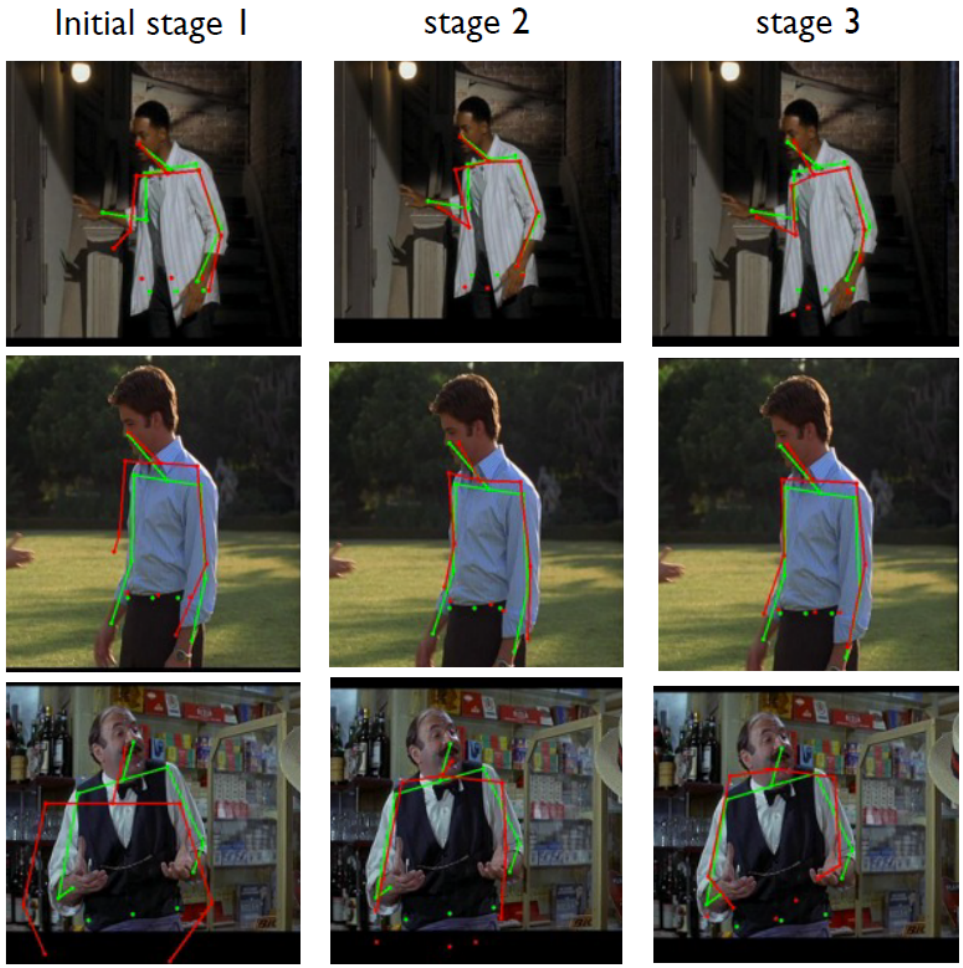
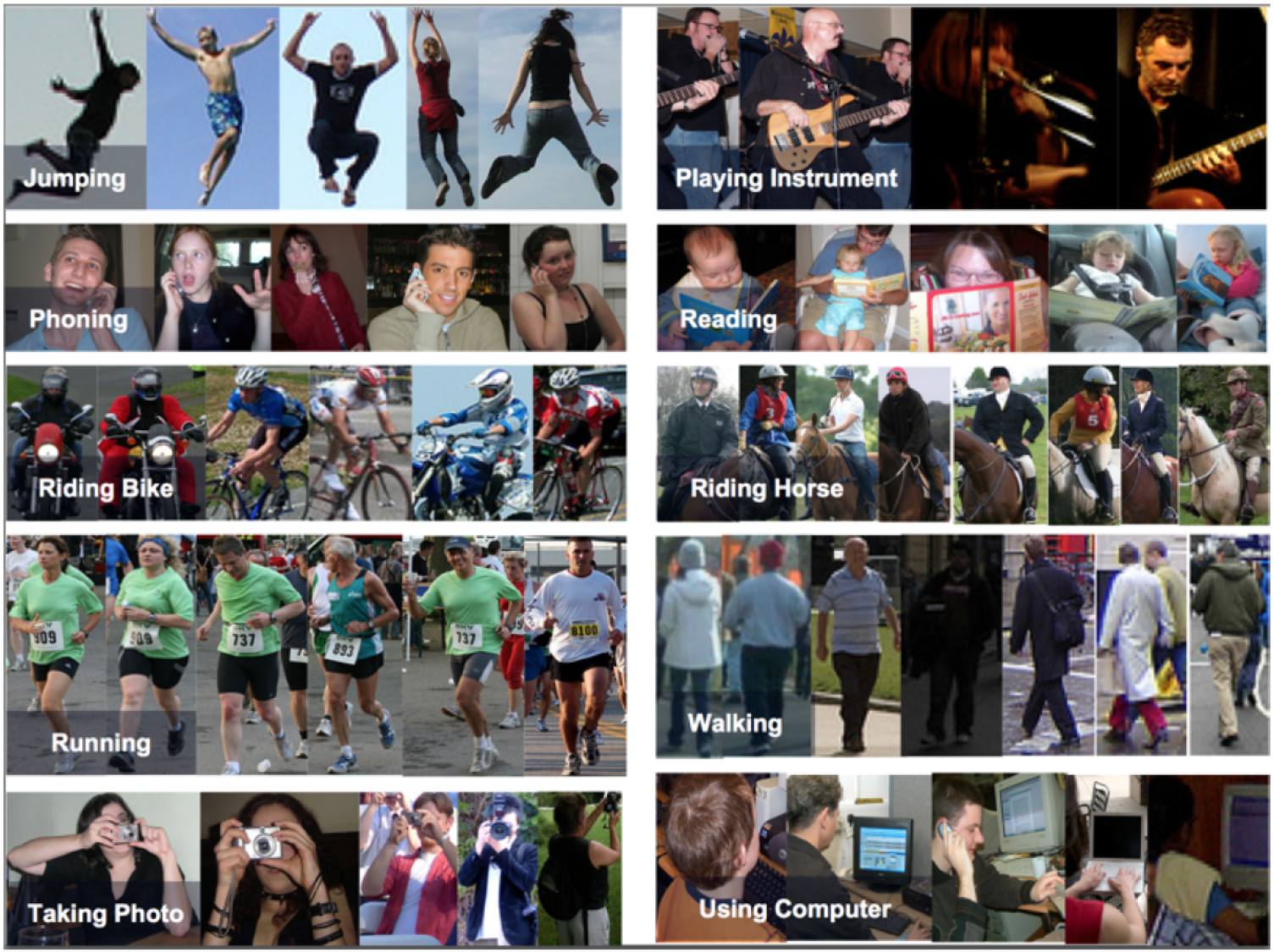
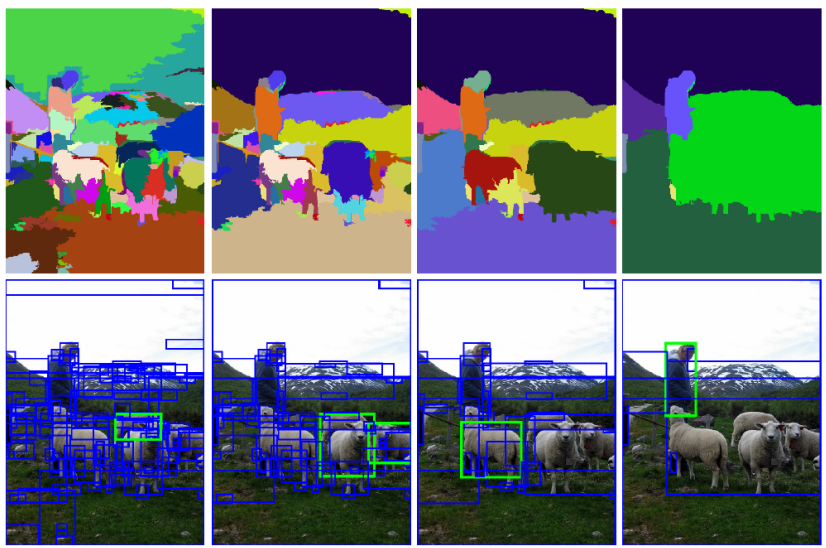


Figure 6. Predicted poses in red and ground truth poses in green for the first three stages of a cascade for three examples.

Action Recognition

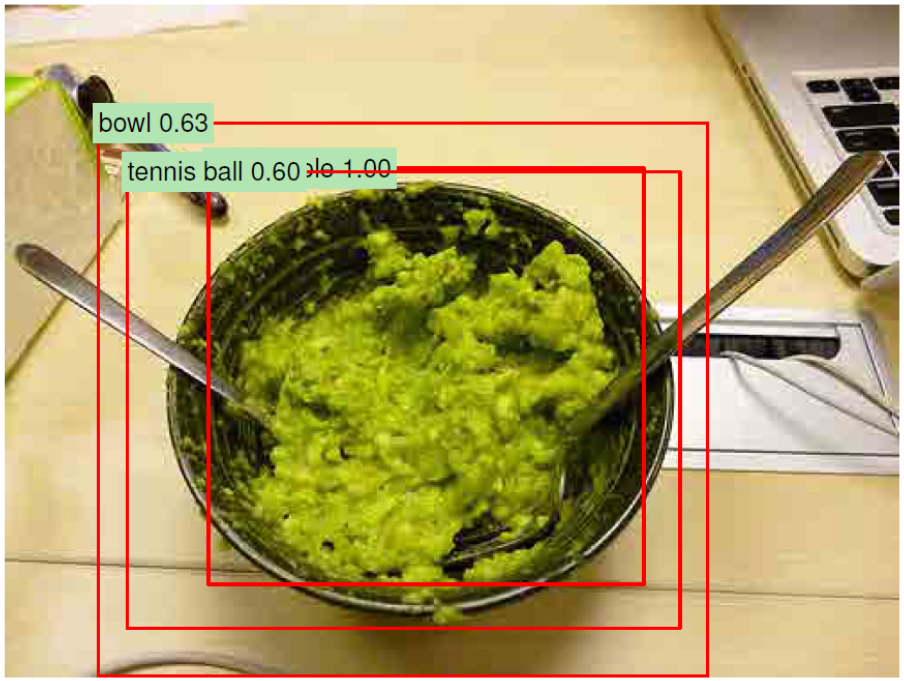


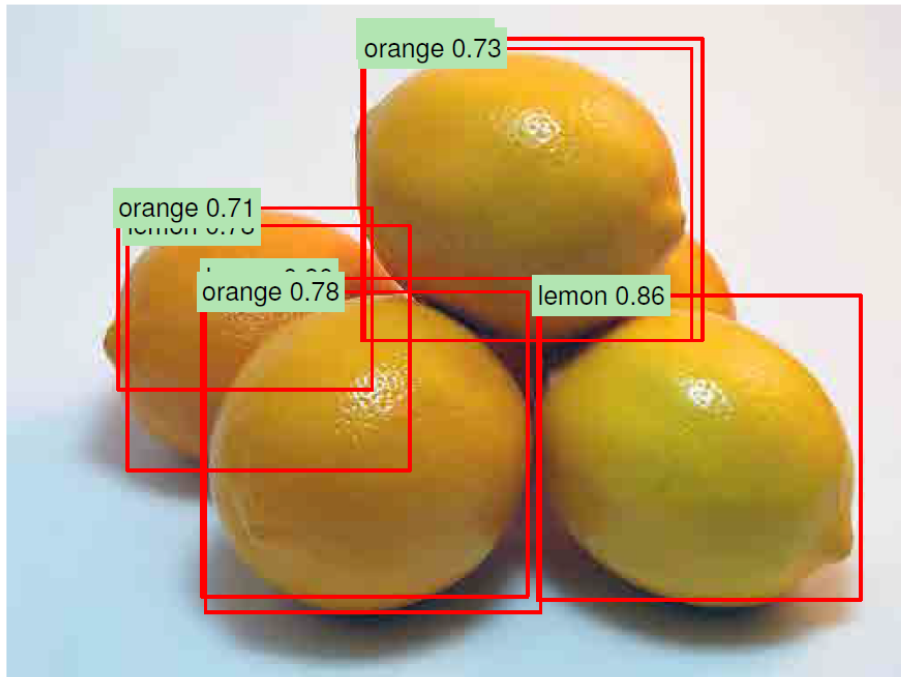
Object Decetion

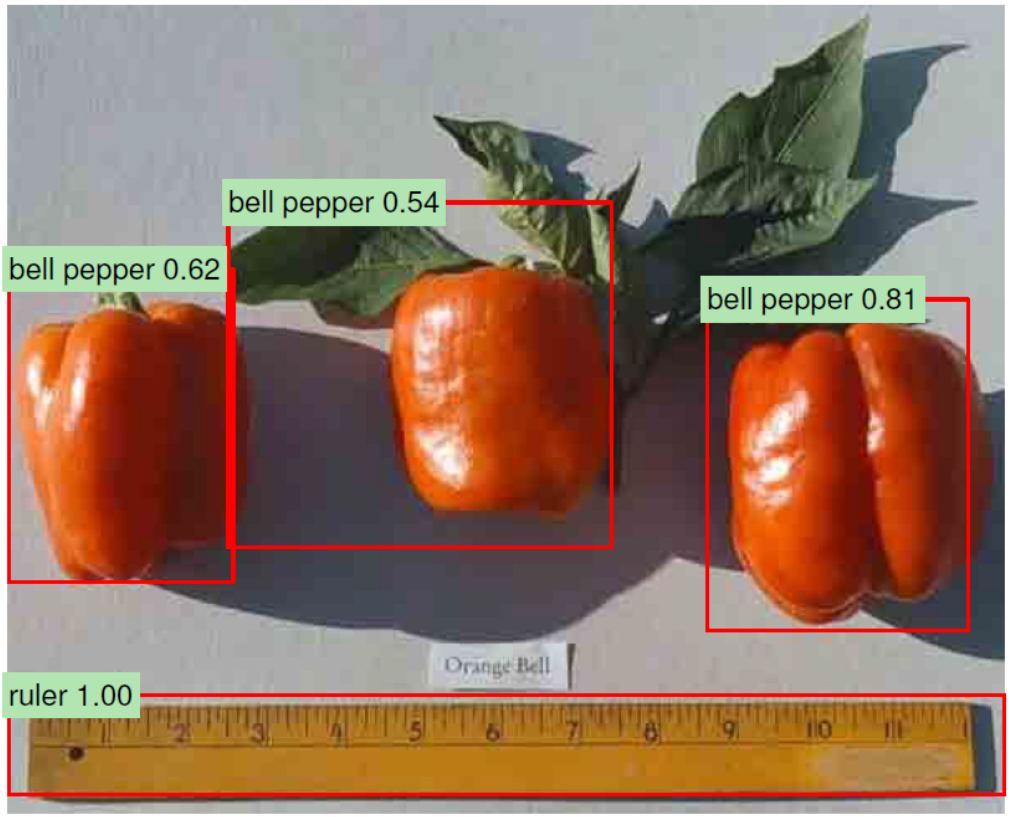


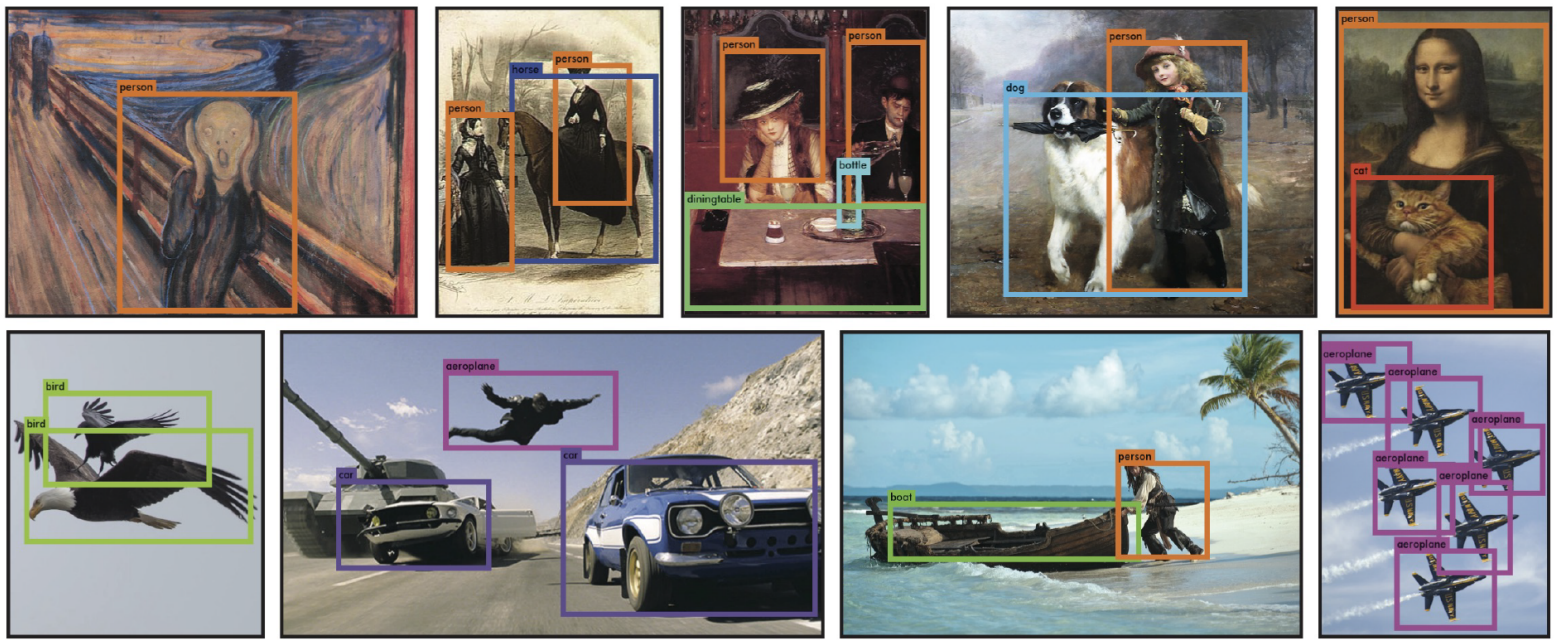
(a)

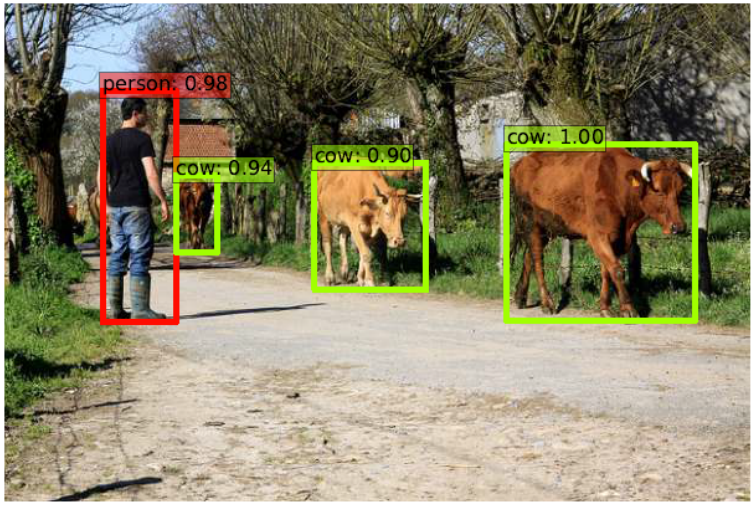
(b)

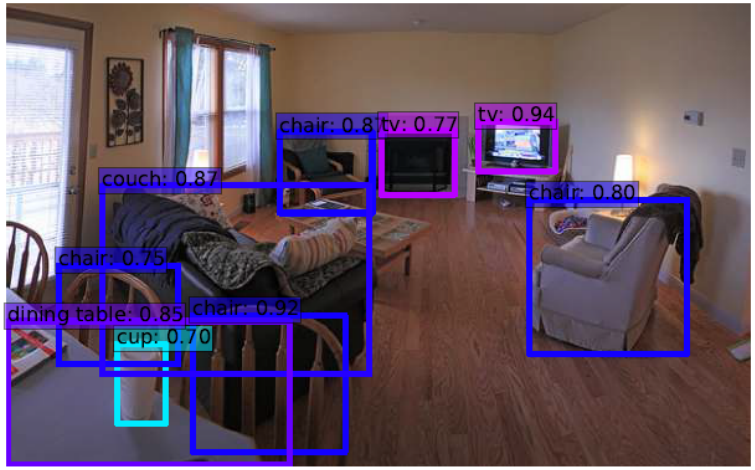

bowl 0.63
tennis ball 0.60

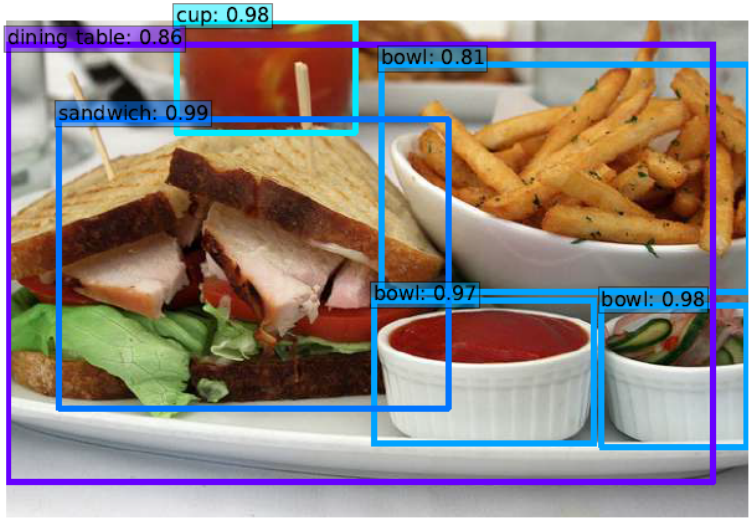

orange 0.73
orange 0.71
orange 0.78
lemon 0.86


bell pepper 0.54
bell pepper 0.62
bell pepper 0.81
Orange Bell
ruler 1.00


person
person
horse
person
person
person
diningtable
bottle
person
dog
person
cat
bird
bird
aeroplane
car
car
person
boat
aeroplane
aeroplane
aeroplane
aeroplane
aeroplane
aeroplane


person: 0.98
cow: 0.94
cow: 0.90
cow: 1.00


chair: 0.81
tv: 0.77
tv: 0.94
couch: 0.87
chair: 0.80
chair: 0.75
dining table: 0.85
chair: 0.92
cup: 0.70

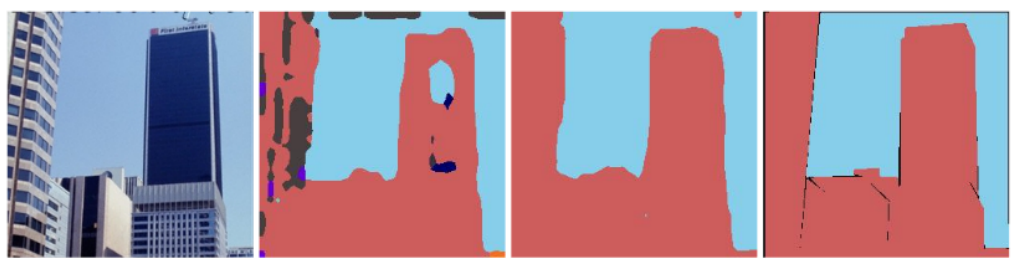

cup: 0.98
dining table: 0.86
bowl: 0.81
sandwich: 0.99
bowl: 0.97
bowl: 0.98


person: 1.00
backpack: 0.82
motorcycle: 0.99

Scene labeling

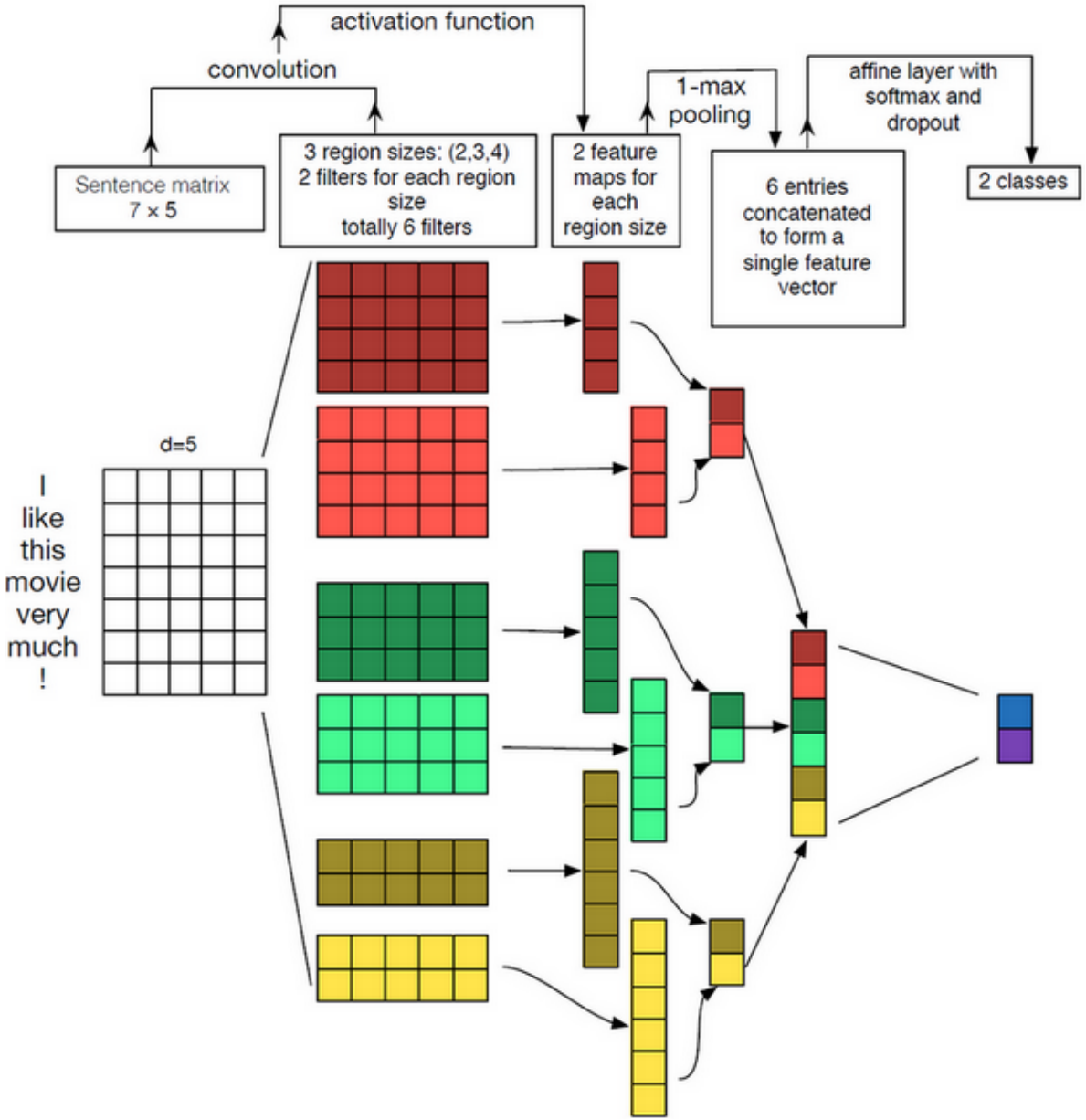
Scene labeling aims to relate one semantic class (road, water, sea...) to each pixel of the input image

Input Image CNN DAG-RNN Ground Truth



Input Image CNN DAG-RNN Ground Truth

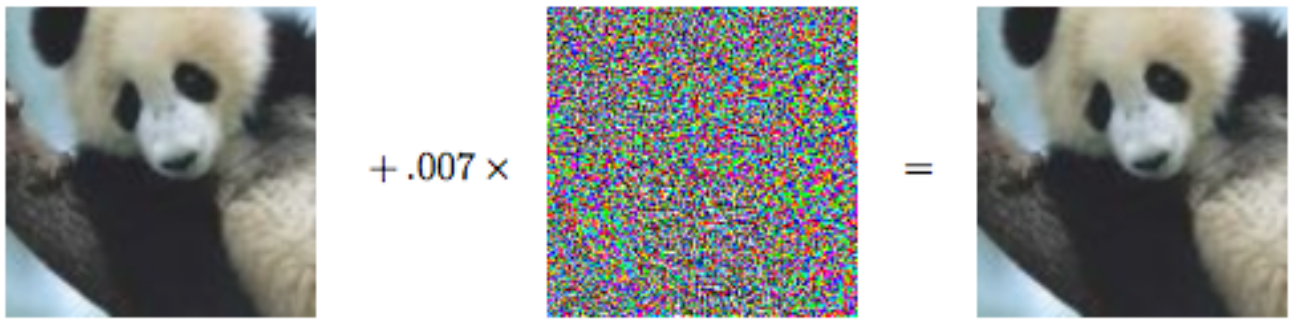
CNNs for Text Classification



Summary

1. Introduction
2. Convolutional Neural Networks
3. Applications

Issues with CNNs:



Can be tackled with Generative Adversarial Networks.

Summer School in Artificial Intelligence and Data Science

James B. Scoggins
(slides modified from Prof. Jesse Read)

Deep Learning: Convolution Neural Networks