

Image-Related Neural Networks

How to see like a human

Yordan Darakchiev

Technical Trainer

iordan93@gmail.com





sli.do

#DeepLearning

Table of Contents

- Convolutional neural networks
 - Operations
 - Architectures
- Generalizations
 - ResNet
 - 1x1 convolutions, "network-in-network"
 - Object localization

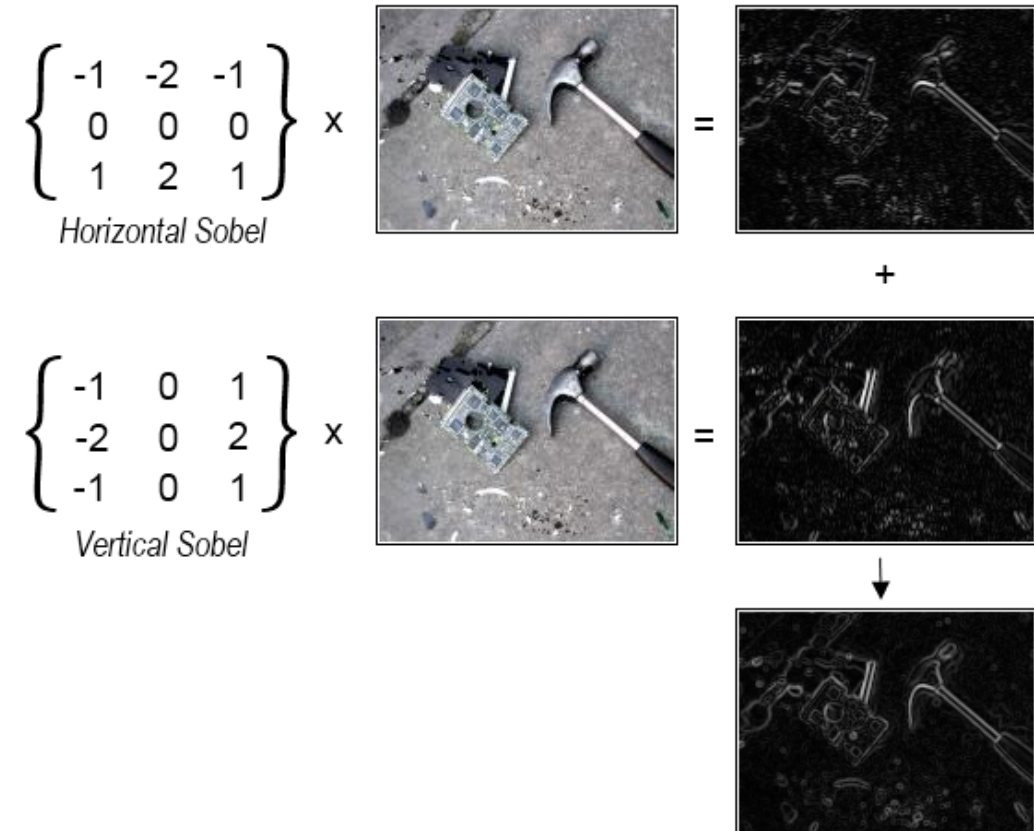


Convolutional Neural Networks

Learning from images

Convolution

- Given an image I and a filter F , $R = I \circledast F$ is defined as
 - For each pixel (i, j) , $R_{ij} = \sum(I_{ij} * F)$
- Depending on F , the result has different meanings
 - Example: Sobel edge detection
- F is usually square, with odd rank (so that it has a central pixel)
 - E.g., 3, 5, 7



Convolution (2)

- Padding

- "Valid convolution": no padding
- "Same convolution": pad so that the output size remains unchanged:
$$p = \frac{f-1}{2}, f - \text{filter size}$$

- Sliding window: **stride** s

- How many pixels we should skip

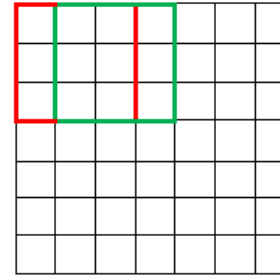
- Summary

- Input

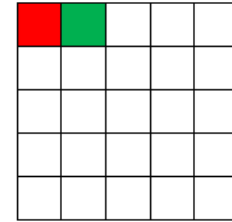
- $n \times n$ image
- $f \times f$ filter
- padding p
- stride s

- Output image dimensions: $\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$
- If the image is non-square, adjust the dimensions in the formula

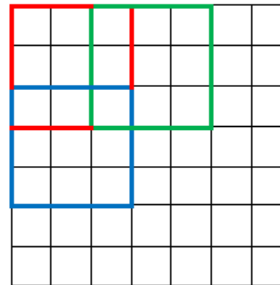
7 x 7 Input Volume



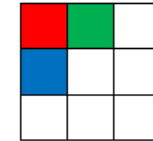
5 x 5 Output Volume



7 x 7 Input Volume

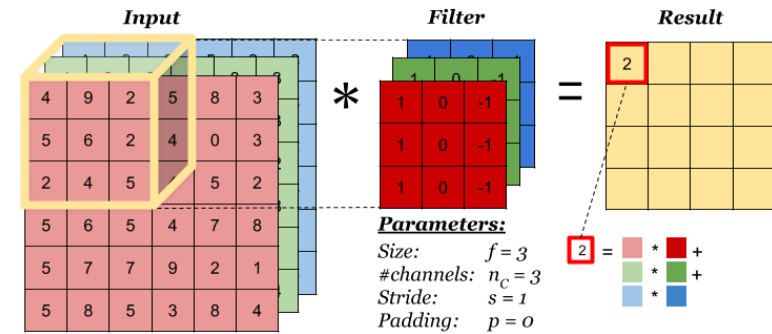


3 x 3 Output Volume



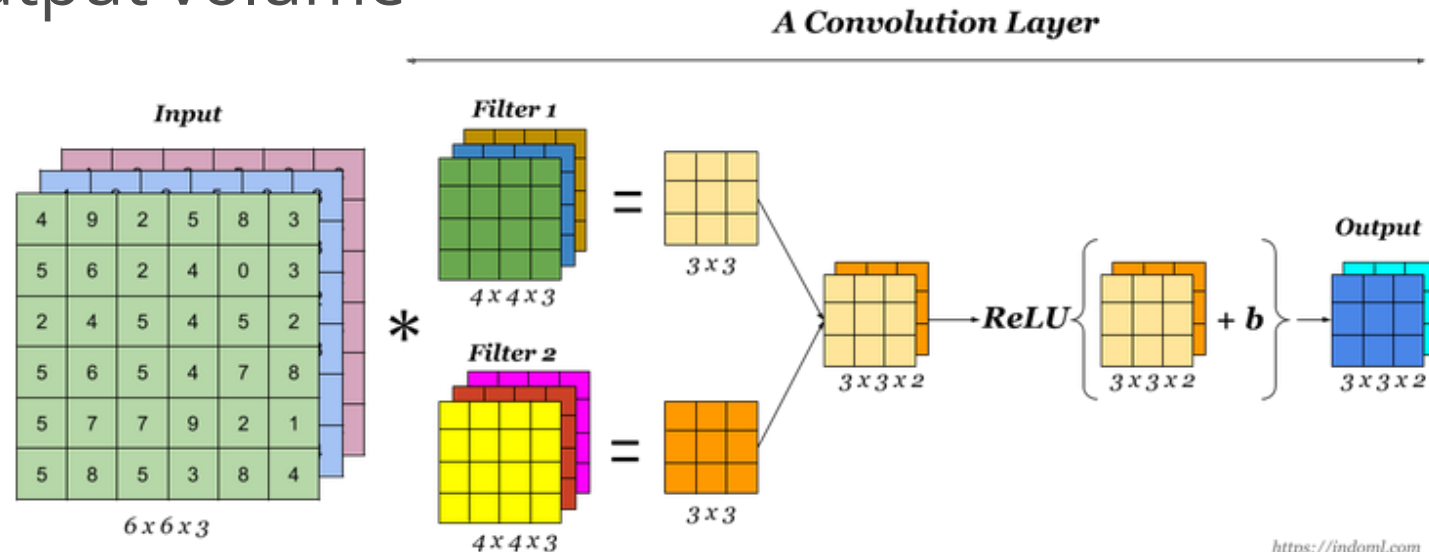
Convolution over Volume

- If the image has many channels with dimensions $n \times n \times c$, just use an $f \times f \times c$ filter
 - I.e. apply the operation independently for each channel
 - Result: 2D image
- Many filters
 - Each one produces a 2D image
 - Stack them together (since they're independent)
⇒ 3D volume
- The convolution operations we use operate over 3D volumes



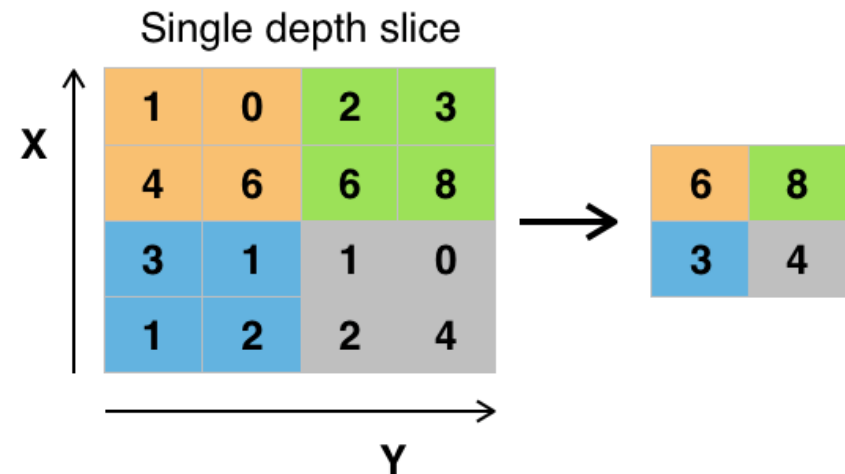
Convolutional Layers

- Just like a regular network
 - Input volume dimensions $n \times n \times c$
 - Choose f , n_f (number of filters), p , s
 - Learn each value of the filters, apply bias terms
 - Add non-linearity (e.g. ReLU)
 - Convolutions are linear operations
 - Sometimes, the convolution and activation layers are shown separately
 - Produce output volume



Pooling

- Used to reduce the number of parameters in the next layers
- Applied like convolution
- Parameters: window size f , stride s , operation
 - Most commonly used operation: max (max-pooling)
 - In the past: avg-pooling was also widely used
 - Other operations are possible but uncommon
- No trainable parameters



Why Do Convolutions Work?

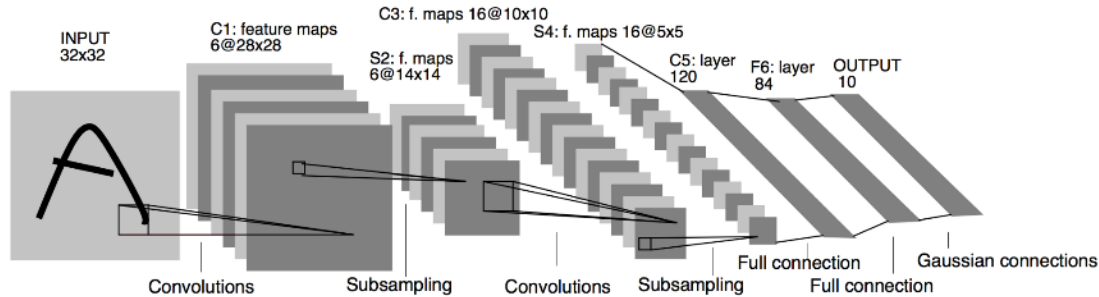
- Image assumptions
 - Individual features are relatively localized
 - The relative (not absolute) position of features is really important
- Convolutions help us to **share computations**
 - An edge detector is useful in many parts of the image
- Each filter has a **low-dimensional input**
 - Simplifies computations
- [Visualizing and Understanding Convolutional Networks, Matthew Zeiler, 2014](#)

Convolutional Layer Architecture

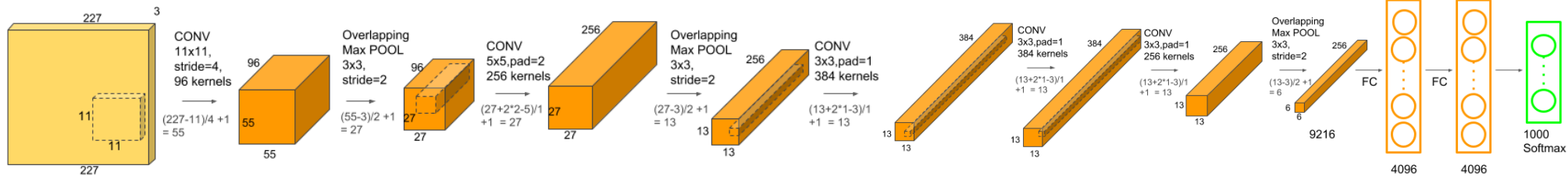
- Input volume: $h \times w \times c$
- Parameters: f, p, s, n_f
- $h' = \left\lfloor \frac{h+2p-f}{s} + 1 \right\rfloor, w' = \left\lfloor \frac{w+2p-f}{s} + 1 \right\rfloor$
- Filter dimensions: $f \times f \times c$, total of n_f filters
- Weights, biases: like fully-connected layers
 - $W = f \times f \times c \times n_f, b = 1 \times 1 \times 1 \times n_f$
- After computing, apply activation function
- Output volume: $h' \times w' \times n_f$

Convolutional Neural Networks

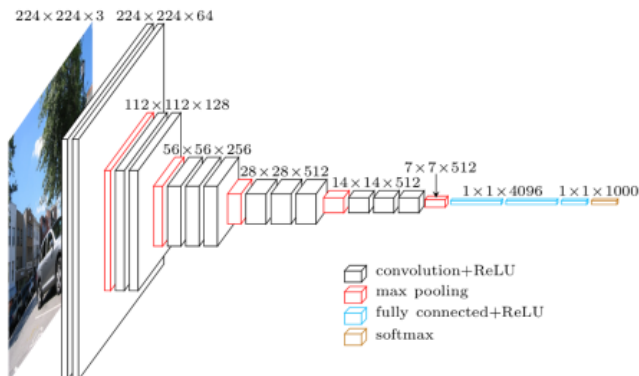
■ LeNet-5 ([Yann LeCun, 1998](#))



■ AlexNet ([Alex Krizhevsky, 2012](#))



■ VGG-19 ([Karen Simonyan, 2014](#))

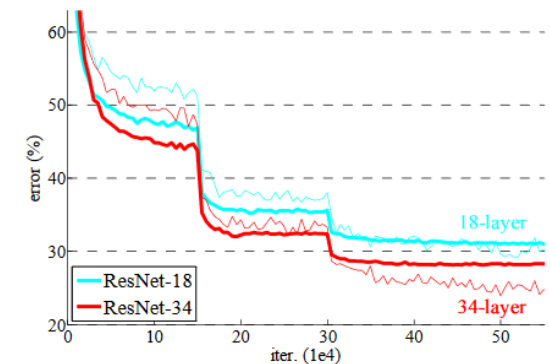
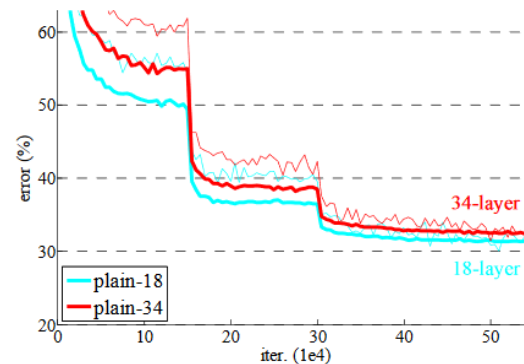
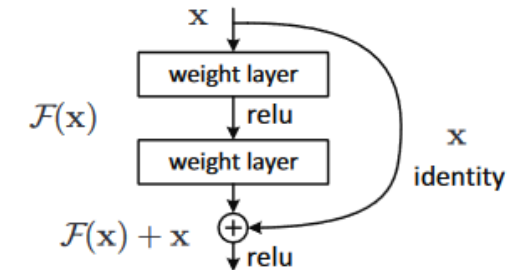
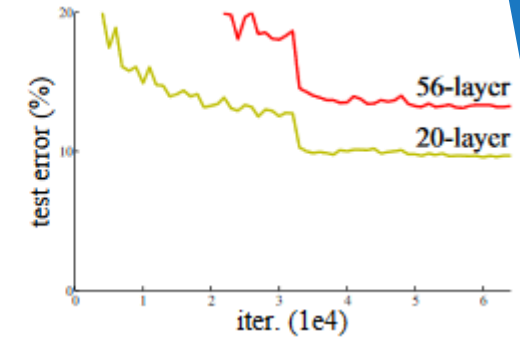
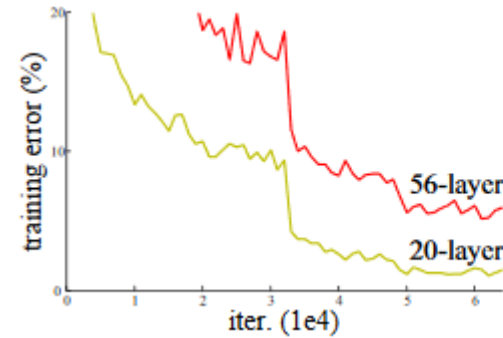


Generalizations and Expansions

Applying other tricks

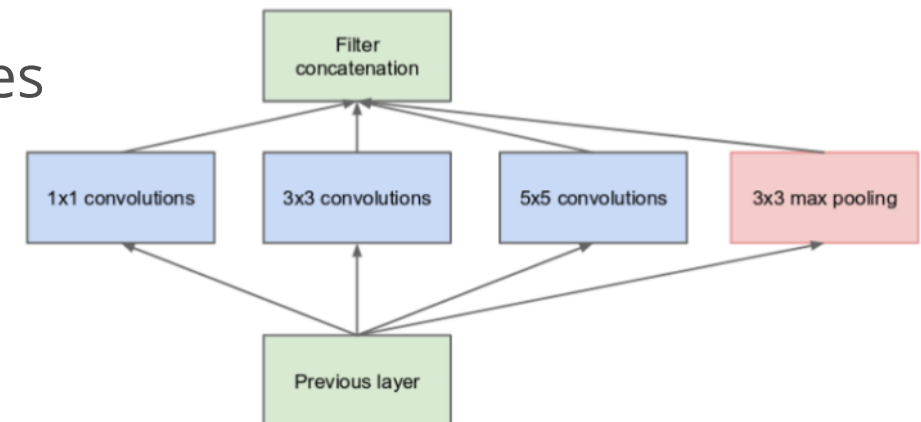
Residual Networks (ResNets)

- Deeper networks allow us to compute complex functions
 - But are difficult to train (e.g., vanishing gradients)
 - The validation error increases (not by overfitting)
- Solution: shortcut connections
 - Pass the activation skipping 1 or more layers
 - Reason: the identity function $y = x$ is really easy to learn
- Results
 - ImageNet
 - 18 / 34 layers
 - Runtime: faster than VGG



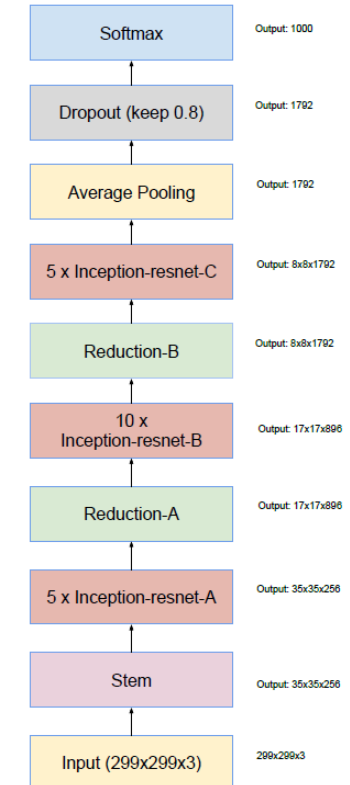
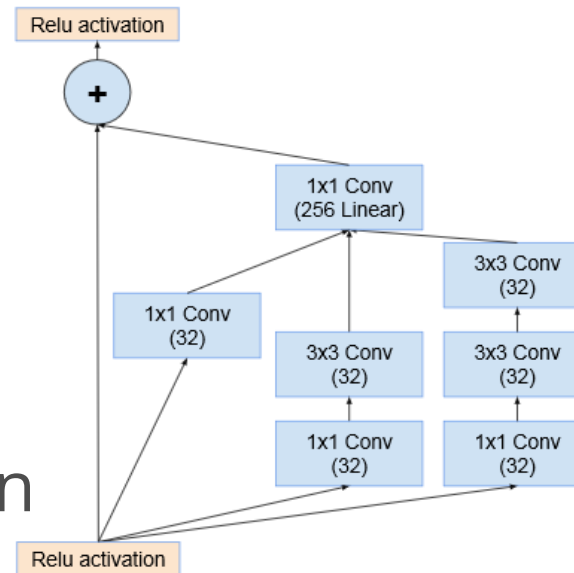
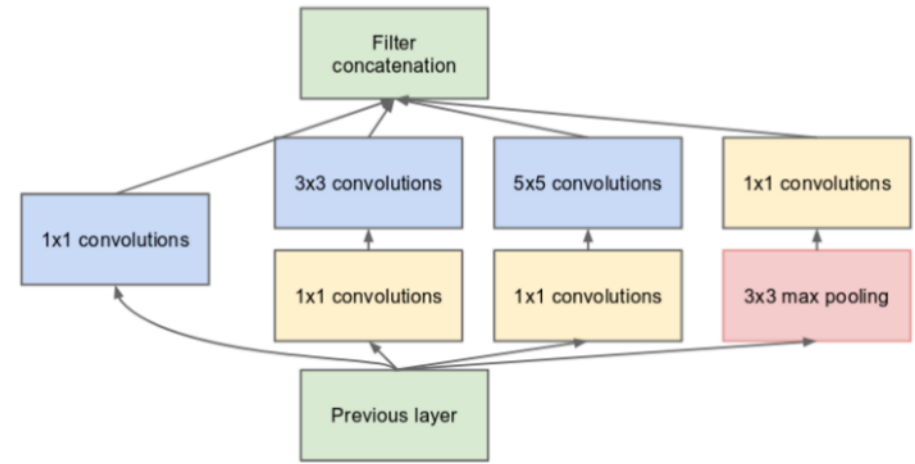
1x1 Convolutions

- Network in Network: Min Lin, 2013
- Single filter: does nothing (scales the input image)
- Many filters: $(w \times h \times c) \circledast (1 \times 1 \times c \times n_f) = (w \times h \times n_f)$
 - Keeps the image dimensions, changes the third dimension
 - Example: $(28 \times 28 \times 192)$, 32 filters $(1 \times 1 \times 192) \Rightarrow (28 \times 28 \times 32)$
 - Dimensionality reduction
- Inception (v1, v2 and v3, v4 and Inception-Res-Net)
 - Main idea
 - Kernel size corresponds to "size" of features
 - We can't know the kernel size f , so try many sizes and let the network decide what's most useful



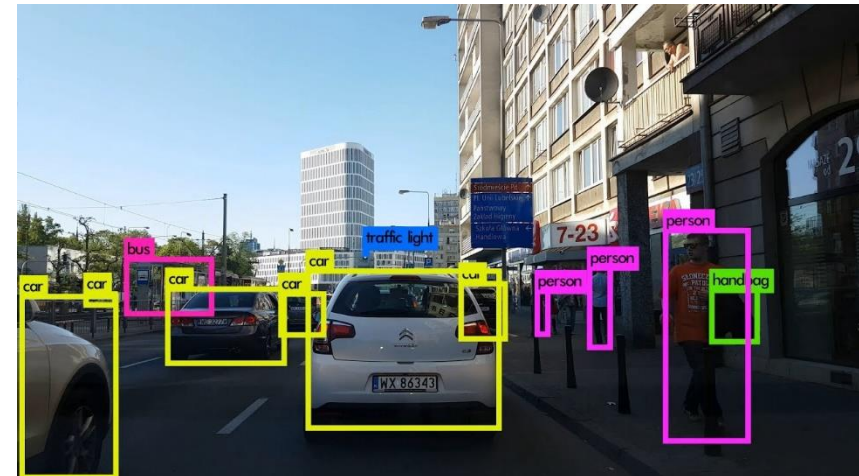
Inception

- Problem: lots of computations
 - Solution: dimensionality reduction before each convolution
 - "Inception block"
- GoogLeNet: 9 inception modules
- Inception-Res-Net
 - A simple combination of the two concepts
 - Idea: create a deeper inception block, simplifying learning through a ResNet connection



Object Localization

- Input: image; output: bounding box (x, y, w, h)
 - Regression
- Classification and localization
 - Simplest case: 1 object
 - Output a vector: $[p, x, y, w, h, c_1, c_2, \dots, c_k]$
 - $p = 0 \Rightarrow$ no object detected; we don't care about the other numbers
 - $p = 1 \Rightarrow$ object detected; class: c_1, \dots, c_k ; bounding box x, y, w, h
 - Metrics: usually [IoU](#) (or Euclidean distance)
- Implementations: [YOLO](#) (You Only Look Once)
 - Also: [R-CNN](#) (Region-proposing network)



Summary

- Convolutional neural networks
 - Operations
 - Architectures
- Generalizations
 - ResNet
 - 1x1 convolutions, "network-in-network"
 - Object localization

The image features a white background with two blue decorative bars. The top bar is a solid blue strip. The bottom bar is a gradient blue strip that transitions from a lighter blue on the left to a darker blue on the right. The word "Questions?" is centered in a blue, sans-serif font.

Questions?