Object Detection

Through Machine Learning

Machine Learning and Applications Group, 2018.

Uroš Stegić

urosstegic@gmx.com

TRADITIONAL COMPUTER VISION

General Overview

Convolution Operator

Filters

Convolutions Over Volume

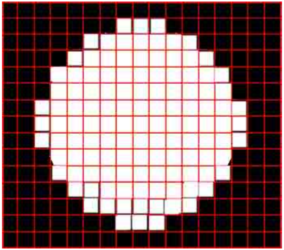
Description

- Process & analyze visual signal
- Extract information from visual signal
- Perform on raw signal (pixel intensities values)

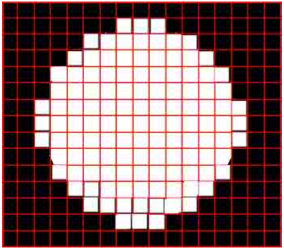
Enhance Intuition

Computer Graphics

$$(x_0, y_0) = (3, 4)$$
$$r = 6$$
$$\downarrow$$



Computer Vision



$$\downarrow$$
$$(x_0, y_0) = (3, 4)$$
$$r = 6$$

Tasks in Computer Vision

- Object Recognition
- Image Retrieval
- Object Detection
- OCR
- Pose Estimation
- ...

- Tracking
- Scene Reconstruction
- Optical Flow
- Semantic Segmentation
- Image Reconstuction
- ...

Tasks in Computer Vision

- Object Recognition
- Image Retrieval
- Object Detection
- OCR
- Pose Estimation
- ...
- Tracking
- Scene Reconstruction
- Optical Flow
- Semantic Segmentation
- Image Reconstuction
- ...

Convolution Operator - Definition

Definition

Let $A, B \in \mathcal{D} \subseteq \mathbb{R}^{n \times n}$. Convolution operator, denoted as $*$ maps the space $\mathcal{D} \times \mathcal{D}$ to a field of real numbers and is defined as follows:

$$A * B = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} B_{ij}$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 2 * 1 + 4 * 1 + 6 * 1 + 8 * 1 = 20$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 2 * 1 + 4 * 1 + 6 * 1 + 8 * 1 = 20$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 2 * 1 + 4 * 1 + 6 * 1 + 8 * 1 = 20$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 2*1 + 4*1 + 6*1 + 8*1 = 20$$

Convolution Operator - Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = 2*1+4*1+6*1+8*1 = 20$$

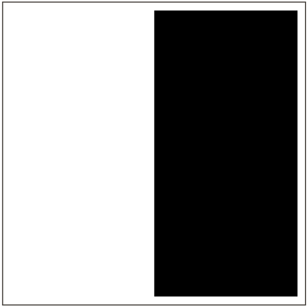
Filters

$$\begin{bmatrix} 211 & 39 & 200 & 102 & 174 & 25 & 90 & 144 \\ 138 & \color{red}{44} & \color{red}{184} & \color{red}{110} & 193 & 30 & 92 & 136 \\ 151 & \color{red}{73} & \color{red}{190} & \color{red}{114} & 189 & 41 & 105 & 128 \\ 129 & \color{red}{101} & \color{red}{123} & \color{red}{181} & 201 & 169 & 117 & 191 \\ 140 & 122 & 153 & 231 & 209 & 157 & 124 & 113 \\ 221 & 115 & 77 & 244 & 198 & 149 & 156 & 247 \end{bmatrix} * \color{red}{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}}$$

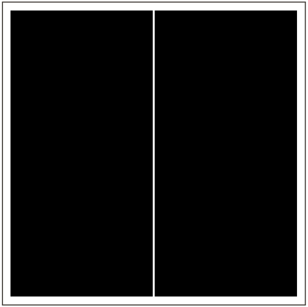
Filters - Examples

- Vertical Edge Extractor
- Horizontal Edge Extractor
- Sobel filter
- Sharpen
- Gaussian Blur

Filters - Edge Extractor



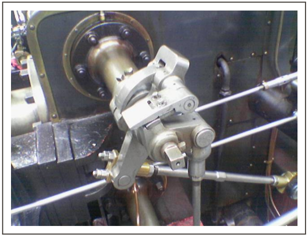
$$* \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} =$$



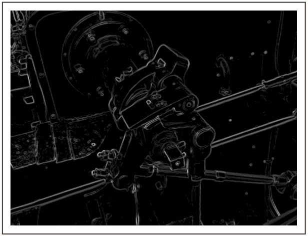
Filters - Edge Extractor

$$* \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} =$$

Filters - Sobel



$$* \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} =$$



Filters - Gaussian Blur

$$* \frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix} =$$

Multiple Input Channels

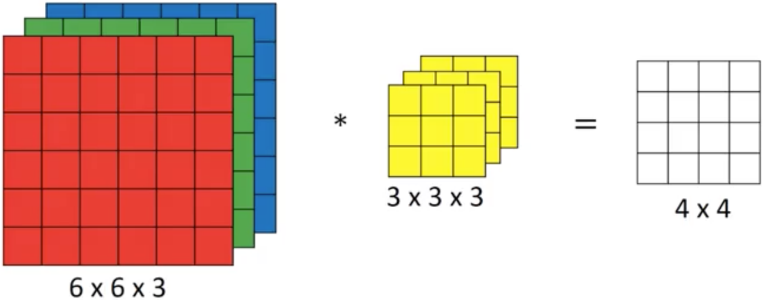


Figure: Convolution of multichannel image

Multiple Filters

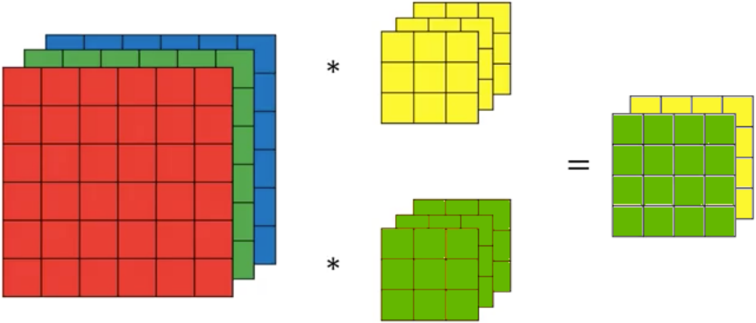


Figure: Convolution of multichannel image with two filters

CONVOLUTIONAL NEURAL NETWORKS

Parameter Learning

Basic CNNs

Residual Networks

Inception Networks

Basic Concepts

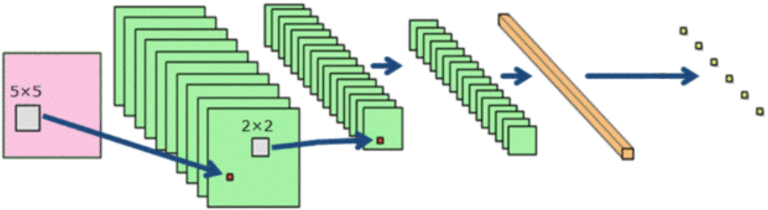


Figure: Convolutional layers stacked

Basic Concepts - Takeaway

- Image Classification
- Parameters (filters) Learning [LBD+89]
- Weight Sharing
- Feature Extraction

Basic Concepts - Feature Abstractions

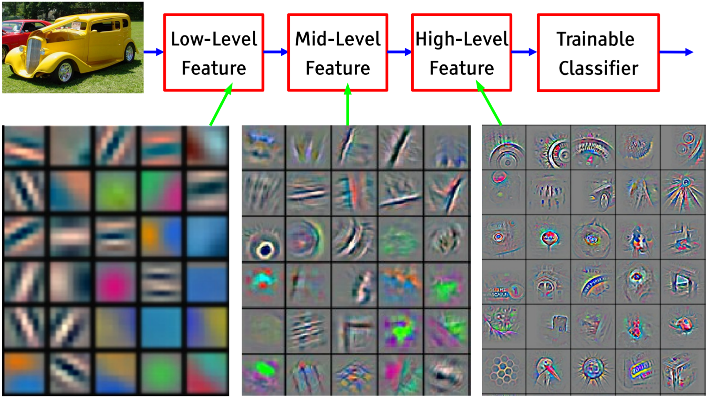


Figure: Feature Visualization [ZF13]

Basic Concepts - Pooling Layers

- Sampling important Features
- Reduce Computation Time
- Make Features Robust

Basic Concepts - Pooling Layers (Example)

Pooling Layer - Max Pooling

$$\begin{bmatrix} 9 & 2 & 4 & 1 \\ 3 & 1 & 8 & 2 \\ 4 & 5 & 9 & 2 \\ 5 & 6 & 0 & 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 9 & 8 \\ 6 & 9 \end{bmatrix}$$

Basic Concepts - Architecture

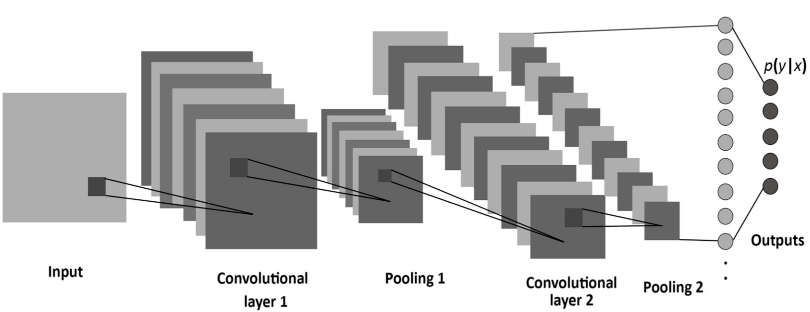


Figure: Convolutional Neural Network - Example

CNN Architecture - Lenet-5

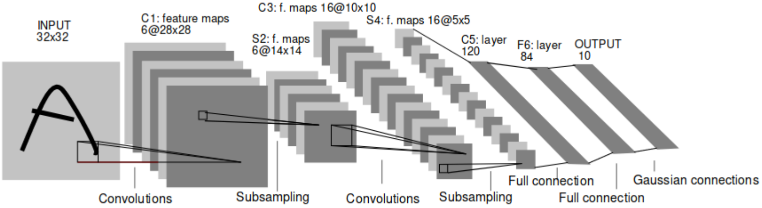


Figure: Lenet-5 Architecture [LBBH98]

CNN Architecture - VGG

Figure: VGG Architecture

CNN Architecture - AlexNet

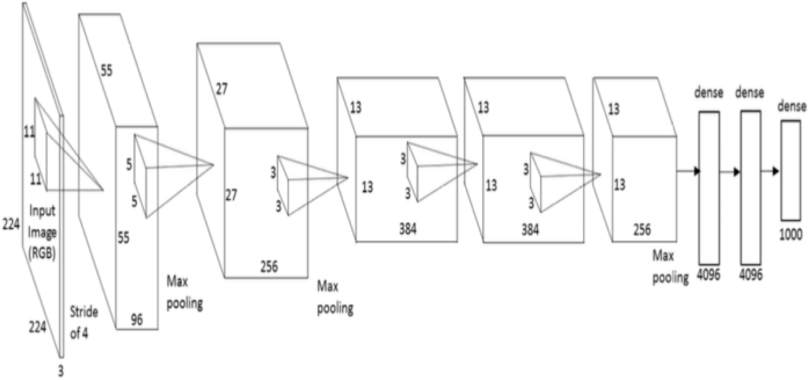


Figure: AlexNet Architecture [KSH12]

CNN - Problems

- Vanishing Gradient
- Exploding Gradient
- Computational Complexity

Residual Block

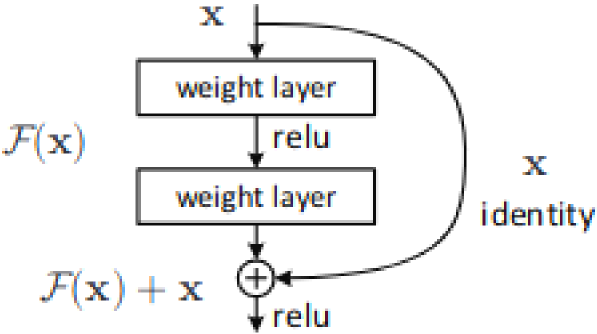


Figure: Residual Block (Skip Connection) [HZRS15]

Residual Network

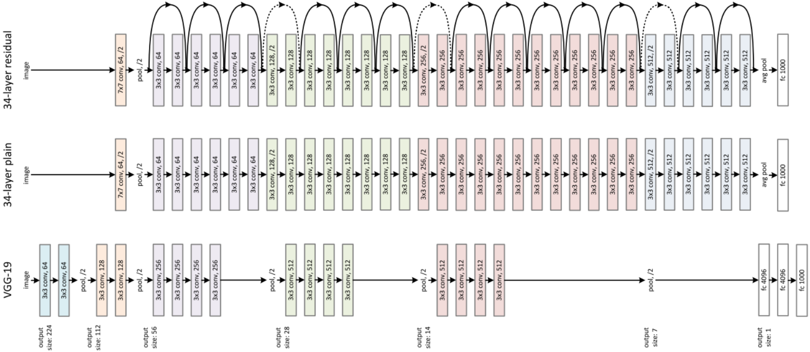


Figure: CNN Architecture - ResNet-34 [HZRS15]

1x1 Convolution

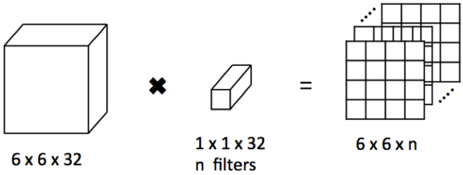


Figure: 1x1 Convolution [LCY13]

Inception Module - Idea

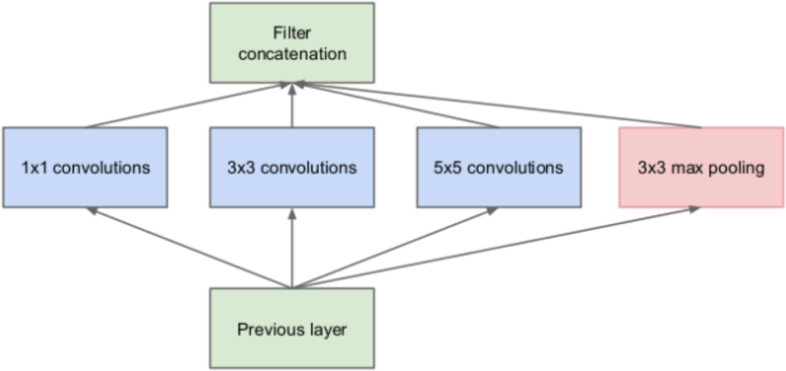


Figure: Inception Module Naive Version [SLJ+14]

Inception Module - Redone

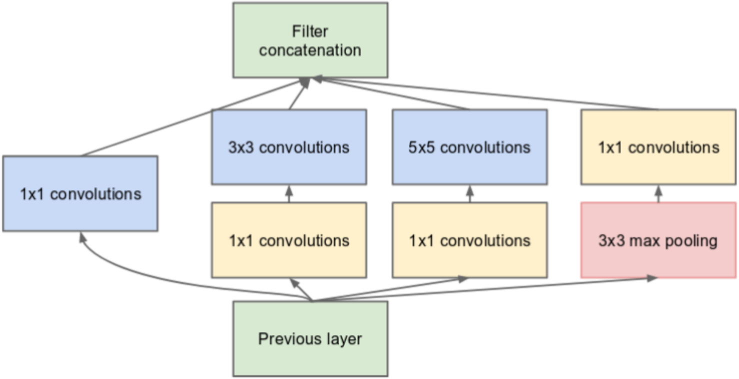


Figure: Inception Module With Dimension Reduction [SLJ+14]

Inception Network

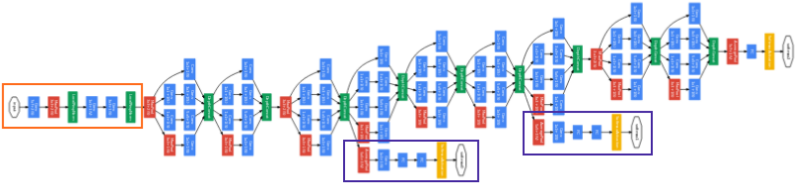


Figure: Inception Network (GoogLeNet) [SLJ+14]

OBJECT DETECTION

Task Outline

YOLO

RCNN Family

Other Influental Models

Speed/Accuracy Trade-Off

Visualizing the Task

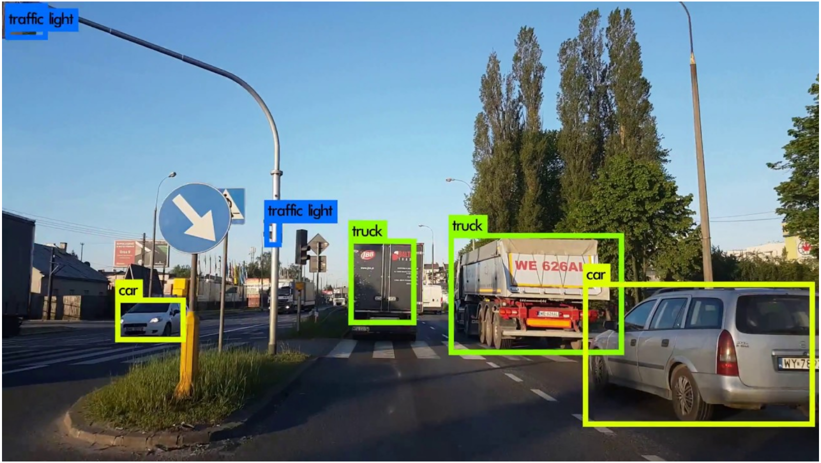


Figure: Object Detection Task

Understanding the Bounding Box Error

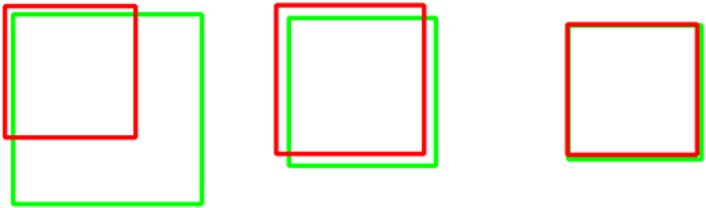


Figure: Bounding Box Missmatch

Defining the IoU

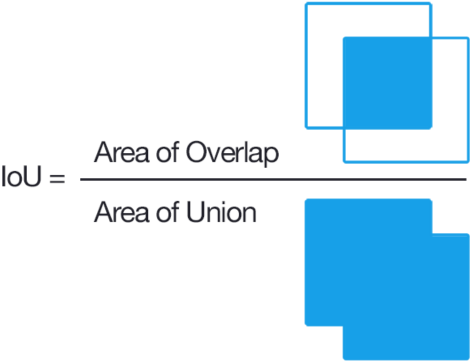


Figure: Intersection over Union

Gaining Intuition on IoU

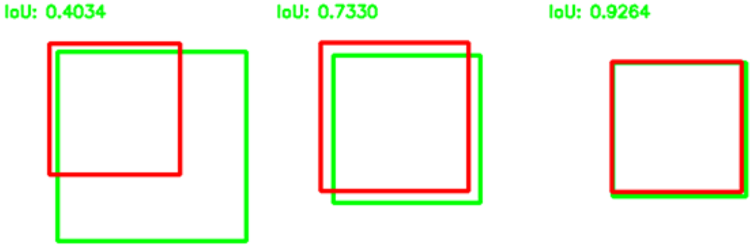


Figure: Intersection over Union - Example

Similar Bounding Boxes Problem

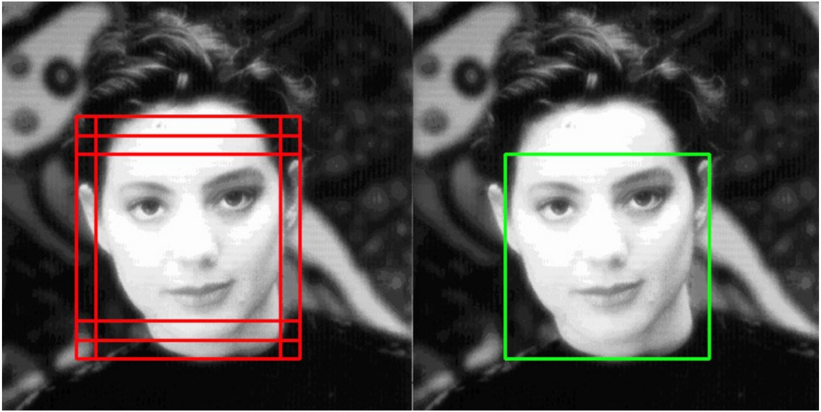


Figure: Elimination of Multiple Bounding Boxes

Non-Maximum Suppression

- Threshold every bounding box
- Sort bounding boxes by detection probability in decresing order
- For each bounding box b_i remove all bounding boxes $b_j (j \neq i)$ such that $IoU(b_i, b_j) \geq t$ for some fixed t

YOLO - Introduction

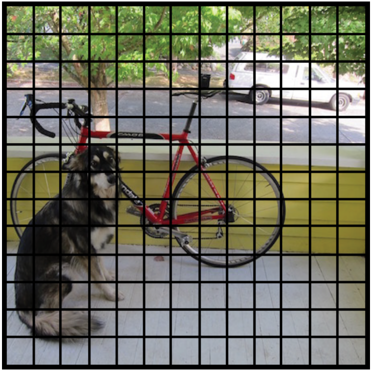


Figure: Grid for YOLO

$$\hat{y} = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_w \\ b_h \\ c_1 \\ c_2 \\ \dots \\ c_n \end{bmatrix}$$

[0] You Only Look Once: Unified, Real-Time Object Detection [RDGF15]

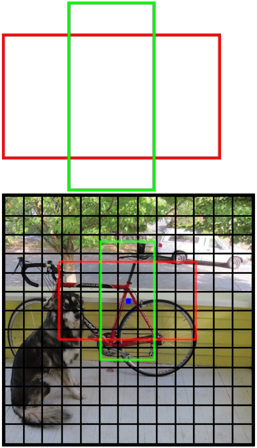
Limitations (already?)

Problem: Multiple objects centered in same cell

Anchor Boxes

- Choose a number of anchors (predefined bboxes)
- Select a ratio (width and height) for each of them
- Modify the output to include this anchors
- ...
- Profit

Anchor Boxes - Example



$$\hat{y}_1 = \begin{bmatrix} p_{c1} \\ b_{x1} \\ b_{y1} \\ b_{w1} \\ b_{h1} \\ c_{11} \\ \dots \\ c_{n1} \end{bmatrix}, \quad \hat{y}_2 = \begin{bmatrix} p_{c2} \\ b_{x2} \\ b_{y2} \\ b_{w2} \\ b_{h2} \\ c_{12} \\ \dots \\ c_{n2} \end{bmatrix}, \quad \hat{y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \end{bmatrix}$$

YOLO - Loss Fucntion

$$\mathcal{L}(y, \hat{y}) = \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2]$$

$$+ \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2]$$

$$+ \sum_{i=0}^{s^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2$$

$$+ \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2$$

$$+ \sum_{i=0}^{s^2} \mathbb{1}_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2$$

Region Based Approach

- Propose Regions of Interest
- Classify each RoI
- Regress Bounding Box Coordinates

Region Models

- Regions with CNN (R-CNN) [GDDM13]
- Fast R-CNN [Gir15]
- Faster R-CNN [RHGS15]
- Mask R-CNN [HGDG17]

Region Proposals - Selective Search

Figure: Selective Search Algorithm Visualized

R-CNN

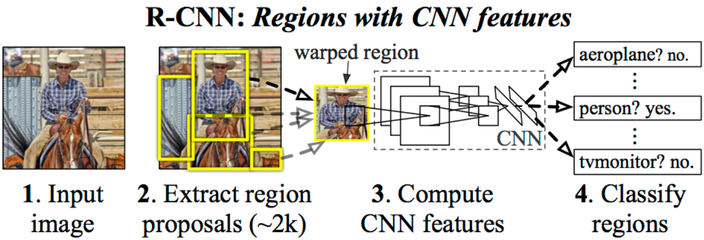


Figure: R-CNN Pipeline

Fast R-CNN

- Convolution Based Sliding Window
- ROI Pooling
- Softmax Classification

Fast R-CNN - Sliding Window

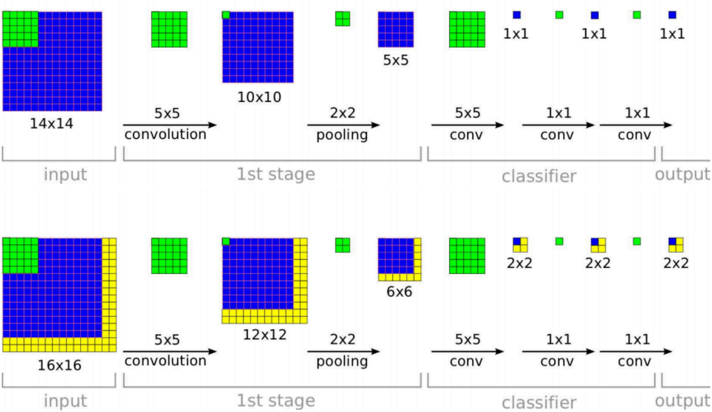


Figure: Sliding Window - CNN Implementation

Fast R-CNN - Visualized

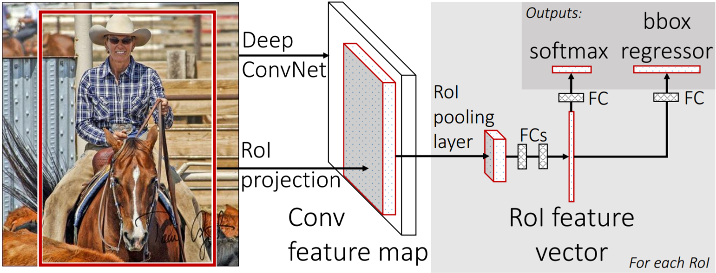


Figure: Fast R-CNN Pipeline

Fast R-CNN - Loss

$$\mathcal{L}(p, u, t^u, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t^u, v)$$

$$\mathrm{L}_{cls}(p, u) = -\log p_u$$

$$\mathrm{L}_{loc}(t^u, v) = \sum_{i \in \{x,y,w,h\}} smooth_{L_1}(t_i^u - v_i)$$

$$\mathrm{smooth}_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } x \leq 1 \\ x - 0.5, & \text{otherwise} \end{cases}$$

Faster R-CNN

- Bottleneck: Region Proposals by Selective Search (2s)
- Solution: Region Proposals by CNN (0.01s)

Region Proposal Network

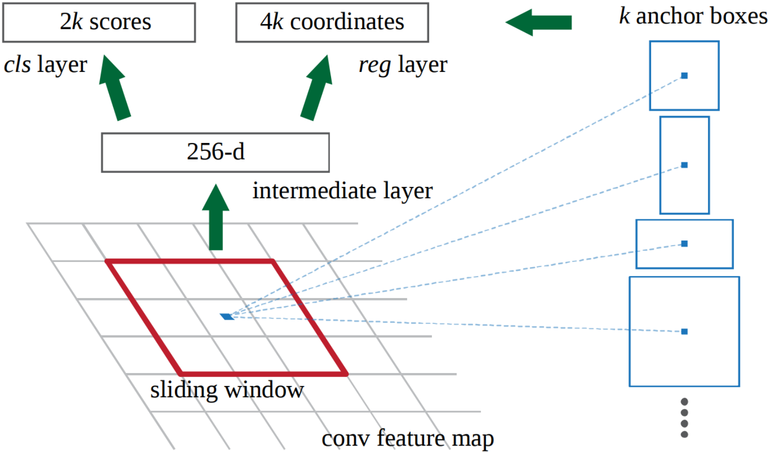


Figure: Region Proposal Network for Faster R-CNN

RPN - Loss

$$\mathcal{L}(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

Faster R-CNN - Architecture

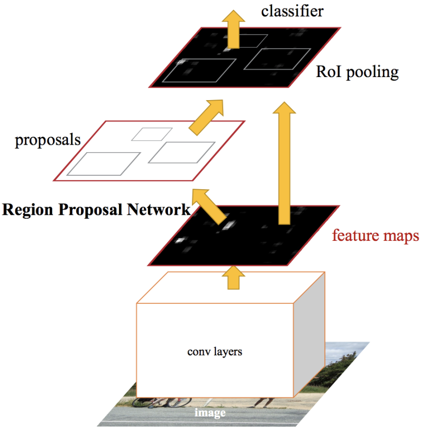


Figure: Model Scheme of Faster R-CNN

Mask R-CNN

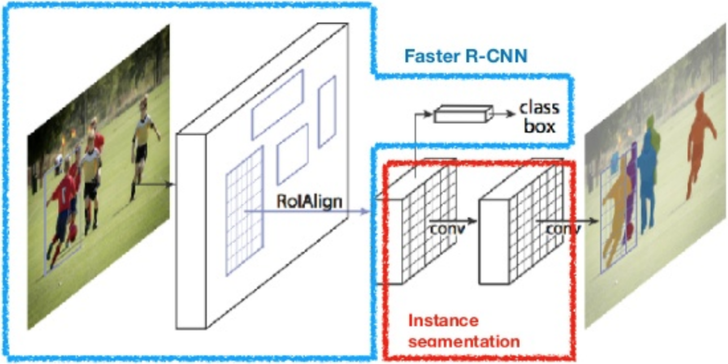


Figure: Model Scheme of Faster R-CNN

Other Influential Models

- RetinaNet (Focal Loss) [LGG+17]
- Single Shot Detector [LAE+15]

RetinaNet

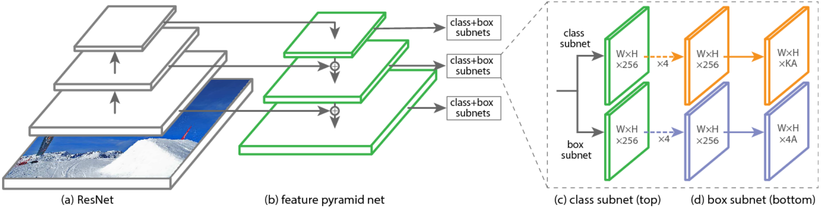


Figure: Retina Net - Overview

Speed vs. Precision

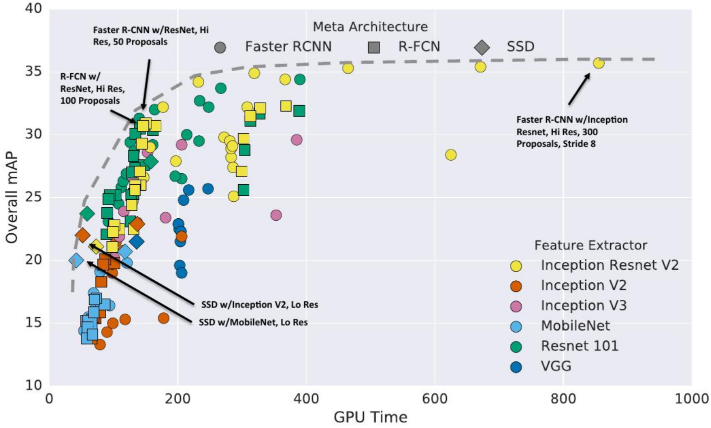


Figure: GPU Time vs. Precision [HRS+16]

Lecture Pronouncement

CONVERGENCE

References I

- Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation*, CoRR **abs/1311.2524** (2013).
- Ross B. Girshick, *Fast R-CNN*, CoRR **abs/1504.08083** (2015).
- Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick, *Mask R-CNN*, CoRR **abs/1703.06870** (2017).

References II

- Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy, *Speed/accuracy trade-offs for modern convolutional object detectors*, CoRR **abs/1611.10012** (2016).
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, *Deep residual learning for image recognition*, CoRR **abs/1512.03385** (2015).

References III

- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, *Imagenet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems 25 (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), Curran Associates, Inc., 2012, pp. 1097–1105.
- Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg, *SSD: single shot multibox detector*, CoRR **abs/1512.02325** (2015).
- Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, IEEE (1998), 2278–2324.

References IV

- Yann Lecun, Bernhard Boser, John Denker, Don Henderson, R E. Howard, W.E. Hubbard, and Larry Jackel, *Backpropagation applied to handwritten zip code recognition*, Neural Computation **1** (1989), 541–551.
- Min Lin, Qiang Chen, and Shuicheng Yan, *Network in network*, CoRR **abs/1312.4400** (2013).
- Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár, *Focal loss for dense object detection*, CoRR **abs/1708.02002** (2017).
- Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi, *You only look once: Unified, real-time object detection*, CoRR **abs/1506.02640** (2015).

References V

- Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun, *Faster R-CNN: towards real-time object detection with region proposal networks*, CoRR **abs/1506.01497** (2015).
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, *Going deeper with convolutions*, CoRR **abs/1409.4842** (2014).
- Matthew D. Zeiler and Rob Fergus, *Visualizing and understanding convolutional networks*, CoRR **abs/1311.2901** (2013).