TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät für Elektro- und Informationstechnik, Professur für Grundlagen der Elektrotechnik und Elektronik

FliK Modul 2020

Advanced CNN Architectures

Steffen Seitz, Marvin Arnold & Markus Fritzsche

Prof. Ronald Tetzlaff

Dresden, 19-23.10.

Advanced CNN Usage

Key Task of CNN:

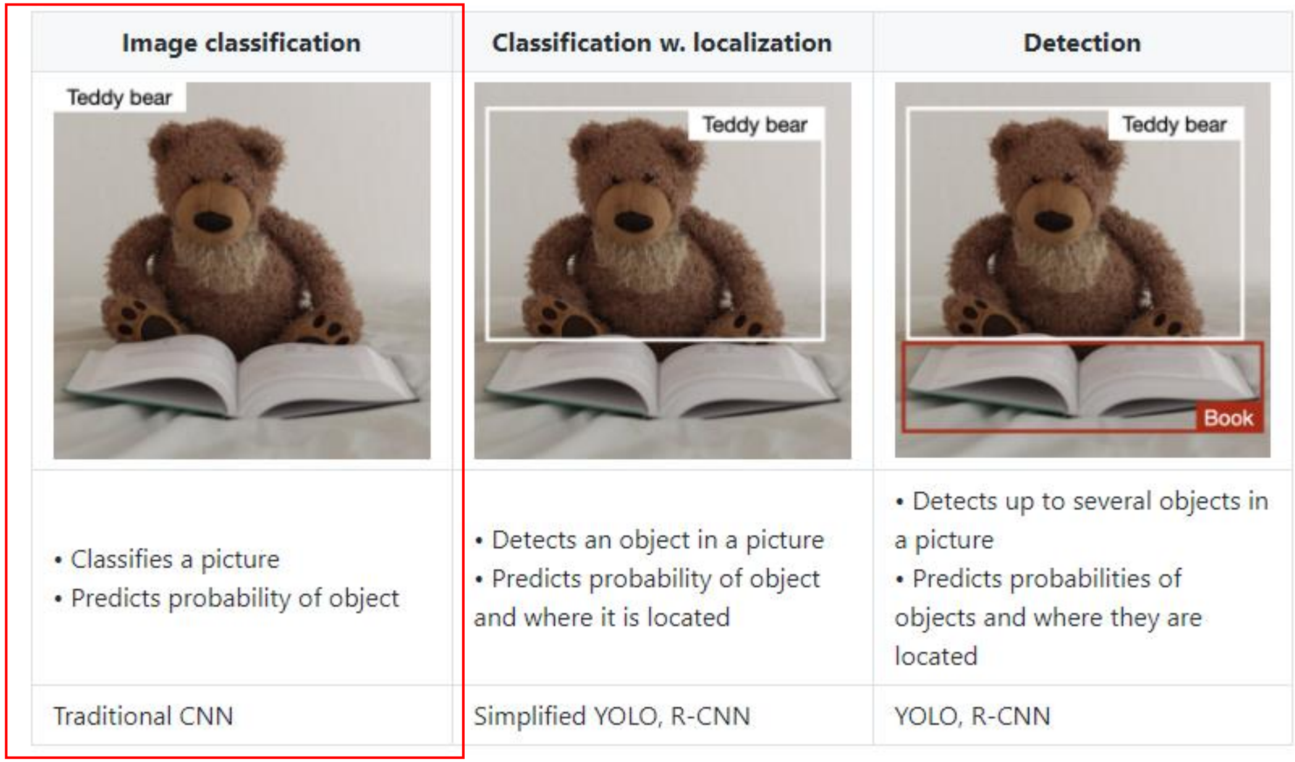

Teddy bear
Teddy bear
Teddy bear
Book

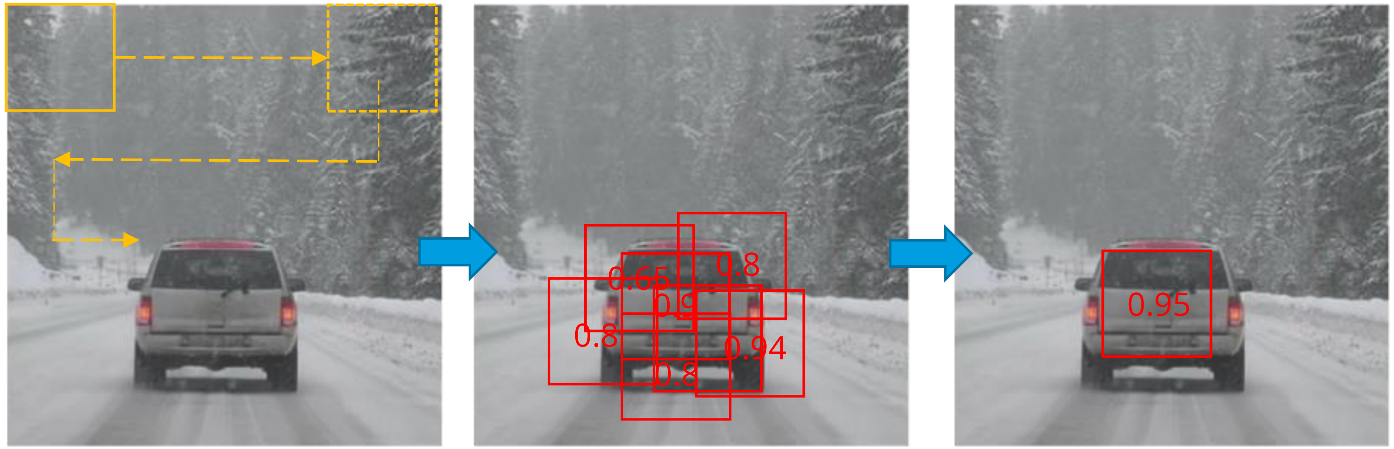
Image classification	Classification w. localization	Detection
• Classifies a picture • Predicts probability of object	• Detects an object in a picture • Predicts probability of object and where it is located	• Detects up to several objects in a picture • Predicts probabilities of objects and where they are located
Traditional CNN	Simplified YOLO, R-CNN	YOLO, R-CNN

Our <u>previous</u> task was the classification of a picture, but there are more tasks, where they can be used!

Image Localization & Detection
Bounding Box Prediction

□ Receptive field of the network

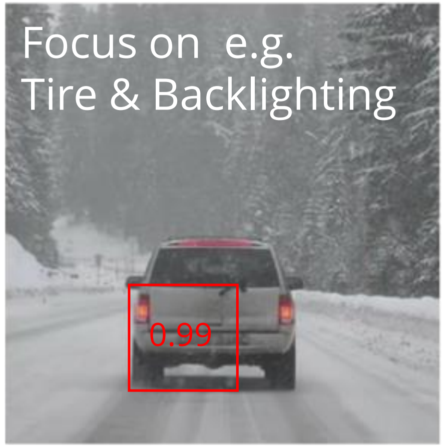
Detects the **area** of the image where the object is **located.** We have previously seen the use of GradCAM as an object detector.



The naïve approach would be a **fixed size** box (yellow) **sliding** over the image and **predicting class** or **no class**

We would end up with a **large set** of possible bounding boxes (red).

We can use **non maximum suppression** to use only the bounding box with the **highest class probability** to detect our correct bounding box


Focus on e.g.
Tire & Backlighting

Problem: Bounding Box size is fixed and class localization can be wrong because we focus on the **features maximizing** our class **probability**

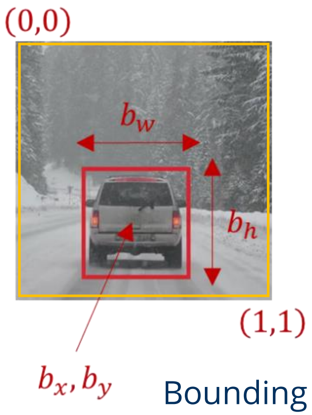
Image Localization & Detection
Localization

□ Receptive field of the network

Alternative: Let the network predict the **Class**, the box location **coordinates** and box **hight** & **width** all **at once**!

Image Classification

Image Localization



Label of our Class

$$y_{Class} = \left[\, p_c \,\right]$$

e.g. Target Vector:
Class = 1 (Car)
no Class = 0 (Backround)

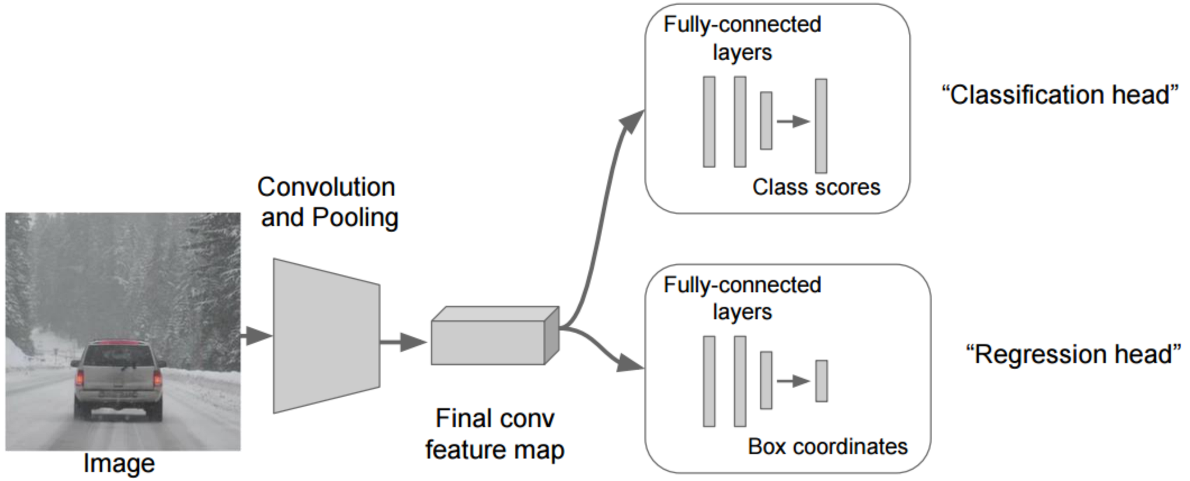
Bounding Box (Hight)

Bounding Box (Width)

$$y_{Loc} = \left[\, p_c \; b_x \; b_y \; b_h \; b_w \,\right]$$

Bounding Box (x-Coordinate)

Bounding Box (y-Coordinate)



You could also split the tasks to multiple networks!

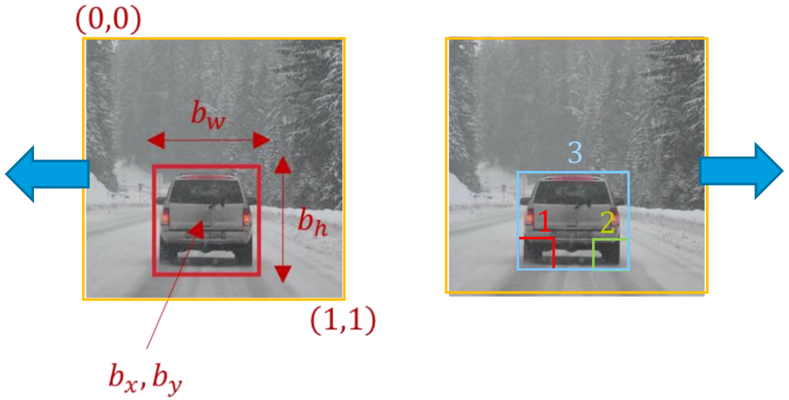
Image Localization & Detection

Detection

□ Receptive field of the network

It is also possible to detect multiple classes and multiple objects at once!

$$y_{Loc} = \begin{bmatrix} p_c & b_x & b_y & b_h & b_w \end{bmatrix}$$



Class Probability
Car

$$y_{Det} = \begin{bmatrix} p_c & b_x & b_y & b_h & b_w & c_{car} & c_{tire} \\ p_c & b_x & b_y & b_h & b_w & c_{car} & c_{tire} \\ p_c & b_x & b_y & b_h & b_w & c_{car} & c_{tire} \end{bmatrix}$$

Bounding Box 3
Bounding Box 3
Bounding Box 3

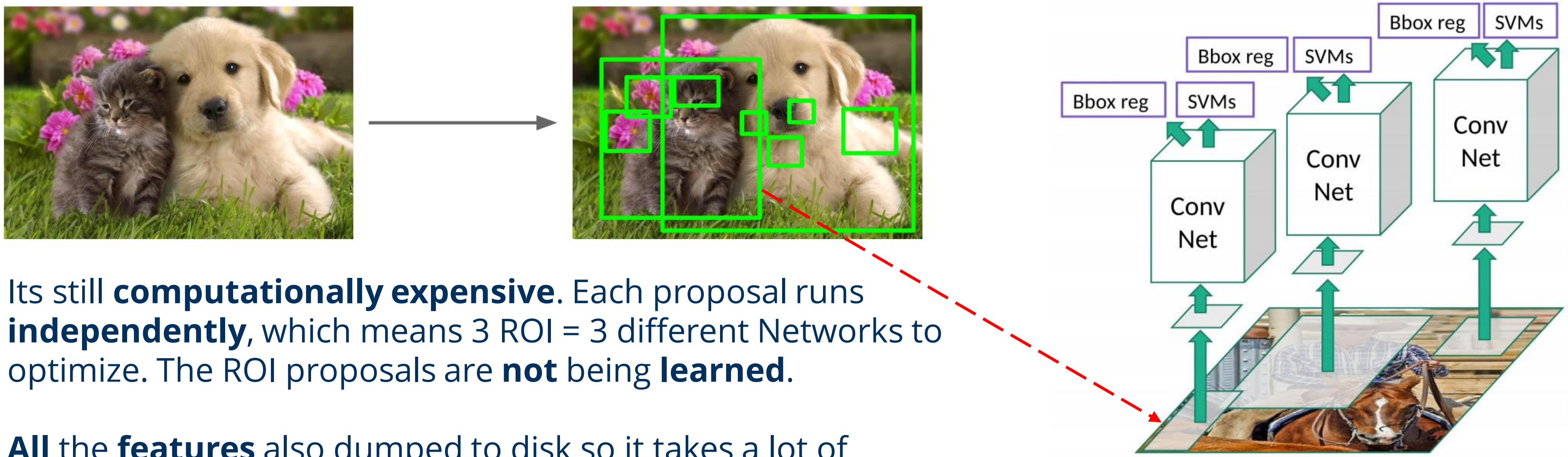
Class Probability
Tire

Downside: **SLOW**! Convolution over the whole image and optimization for so many output variabels at once is computationally intensive because we always slide over the whole image.

R-CNN
(„Region“ CNN)

Idea: **Use** a **fast algorithm** from computer vision (Selective Search) for **bounding box proposals** and **run** the **classification** and **box regression** on those (much smaller) Region of Interest (ROI) **proposals**.

Selective Search gives 2,000 region proposals in a few **seconds**

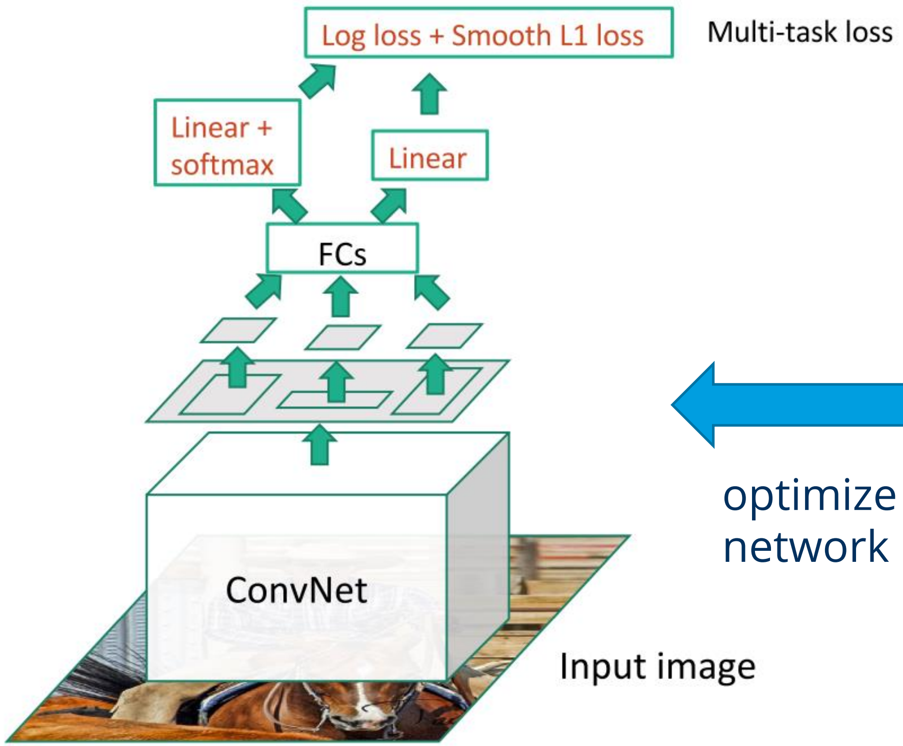


Its still **computationally expensive**. Each proposal runs **independently**, which means 3 ROI = 3 different Networks to optimize. The ROI proposals are **not** being **learned**.

All the **features** also dumped to disk so it takes a lot of space. Training is also **super-slow** and at **inference** is also **very slow** (47s per image for example).

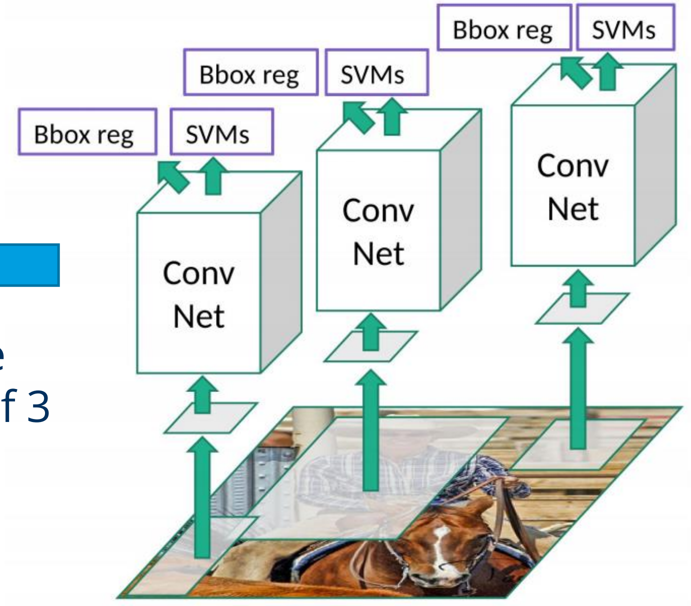
Inference: Using a trained model to do the trained task (= forward pass only)

FAST R-CNN
(Fast Region CNN)



optimize only one network instant of 3

R-CNN



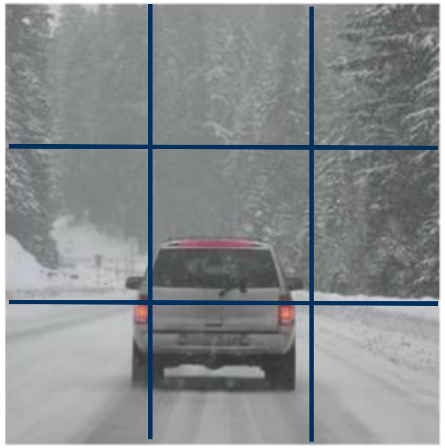
FAST R-CNN

Run the **whole image** through **one** CNN to extract feature maps and extract ROI from there

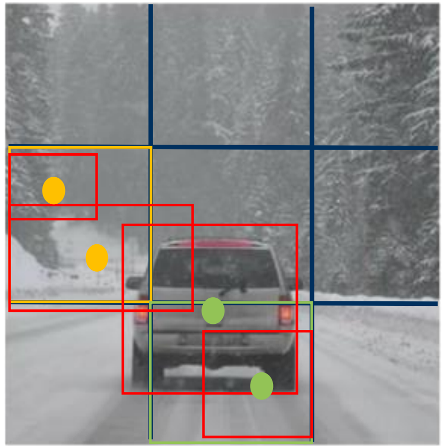
Fast R-CNN **10x faster** to train and inference **less than a second** per image but still not learning the **ROI**!

YOLO - You Only Look Once

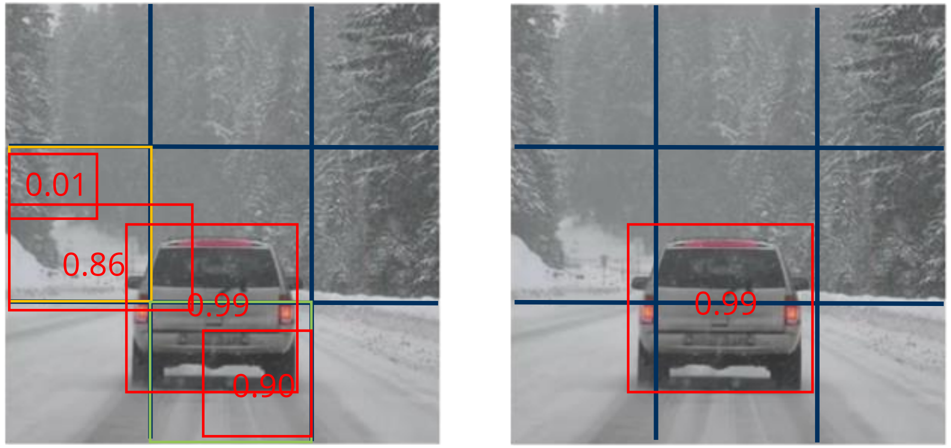
The king of inference speed



Divide picture into **grid** (in reality the grid is much finer!)



In **every** grid cell run multiple the ROI predictions at once using **LeNet** inspired network instead of Selective Search



Use non maximum suppression for each bounding box

YOLO - You Only Look Once

The king of inference speed