Pedestrian detection: A part-based approach using CNN

Domonkos Varga

Supervisor: Prof. Tamás Szirányi

MTA SZTAKI Hungarian Academy of Sciences Institute for Computer Science and Control

Outline

- Introduction
- Challenges
- Related work
- Our approach
- Results
- Questions

Introduction

- Our goal: detect and localize pedestrians in images.
- People detection is one of the most challenging problems is computer vision.
- It has a number of applications such as smart video surveillance, driving assistance system, human-robot interaction, people-finding for military applications and intelligent digital management.

Challenges - illumination

Challenges - clothing

Challenges – pose and body articulation

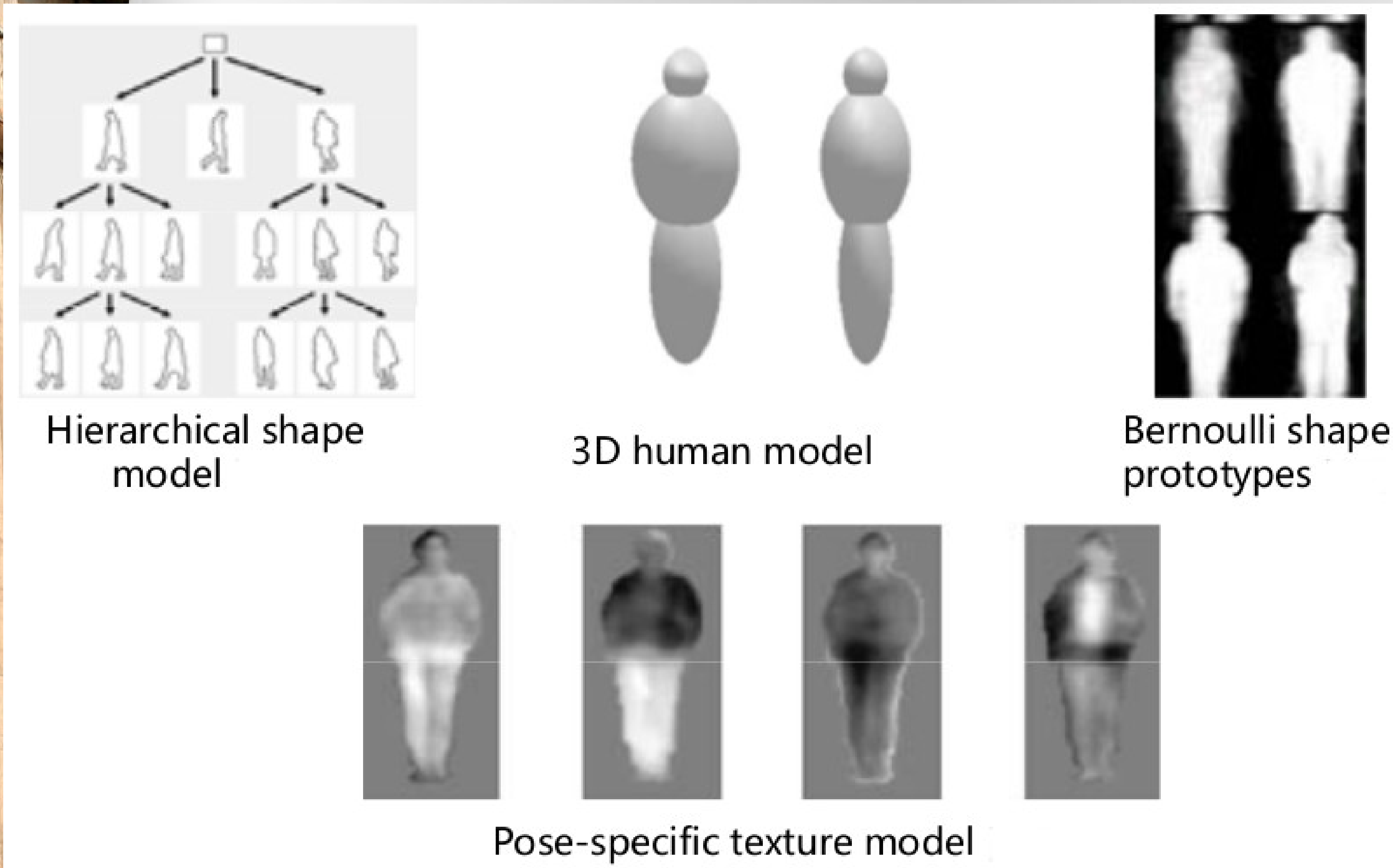
Related work

I. Model-based pedestrian detection
II. Feature-classifier-based pedestrian detection
III. Part-based pedestrian detection

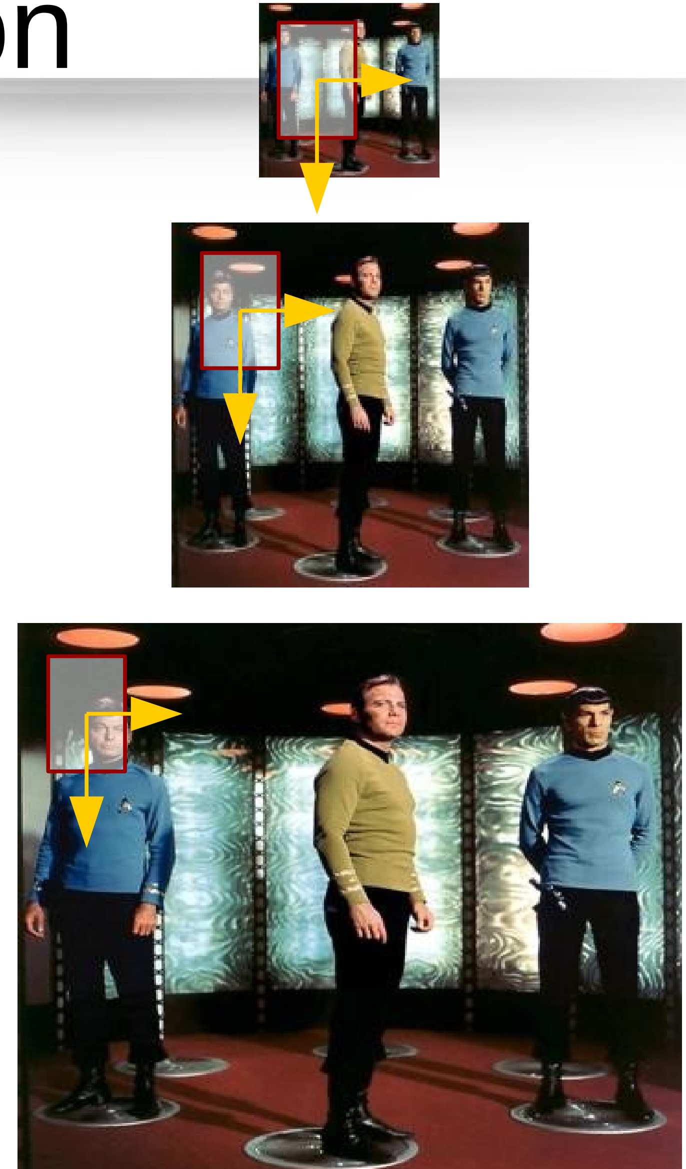
Model-based pedestrian detection

- Model-based pedestrian detection: an exact pedestrian model is defined first of all. Then we search the image for matched positions with the model to detect pedestrians.
- Mathematical background: Bayesian theory to estimate the maximum posterior probability of the object class.

Model-based pedestrian detection



Feature-classifier-based ped. detection



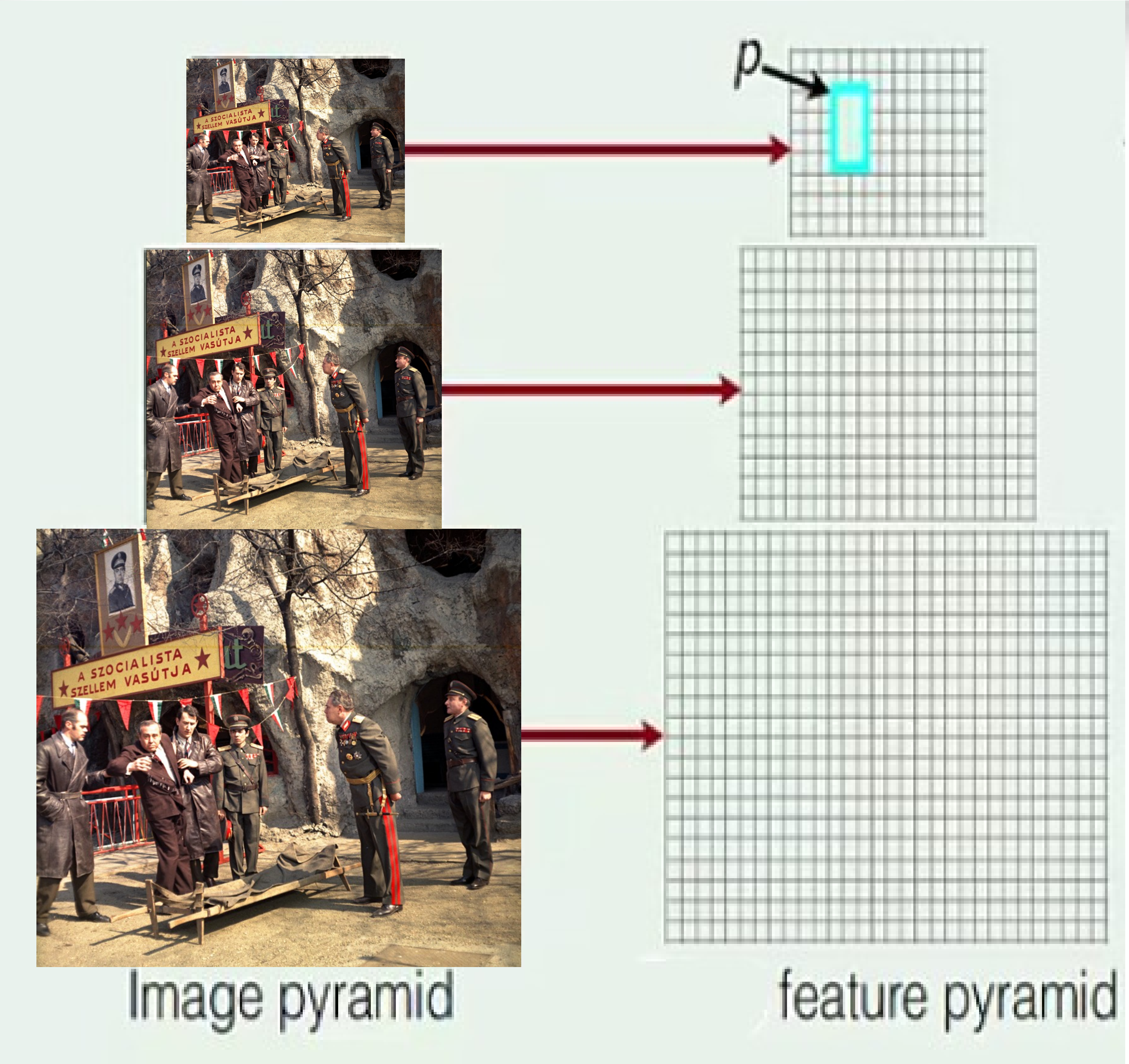
Features:

- Haar
- HOG
- LBP
- Adaptive Contour Feature
- Shapelet
- Fusion of different features

Classifiers:

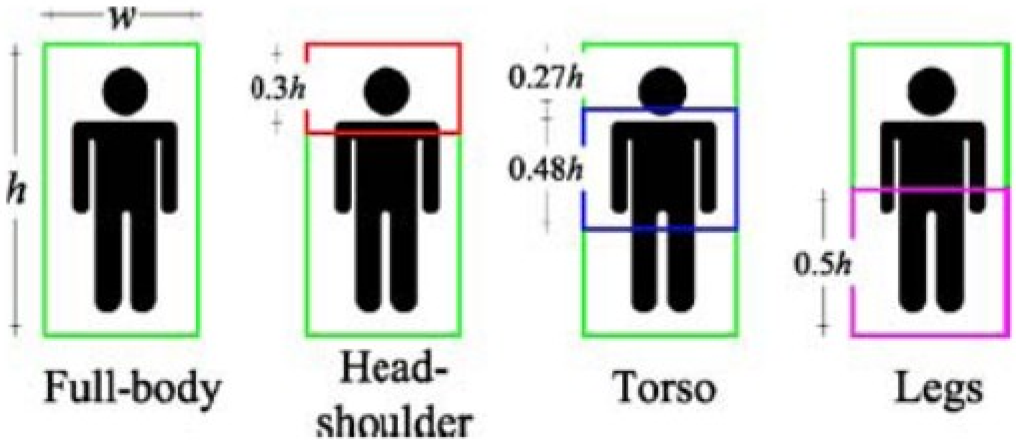
- SVM
- k-NN
- AdaBoost
- NN
- Random forest
- Ensemble of classifiers

Feature-classifier-based ped. detection



Part-based pedestrian detection

- The human body is divided into certain parts.



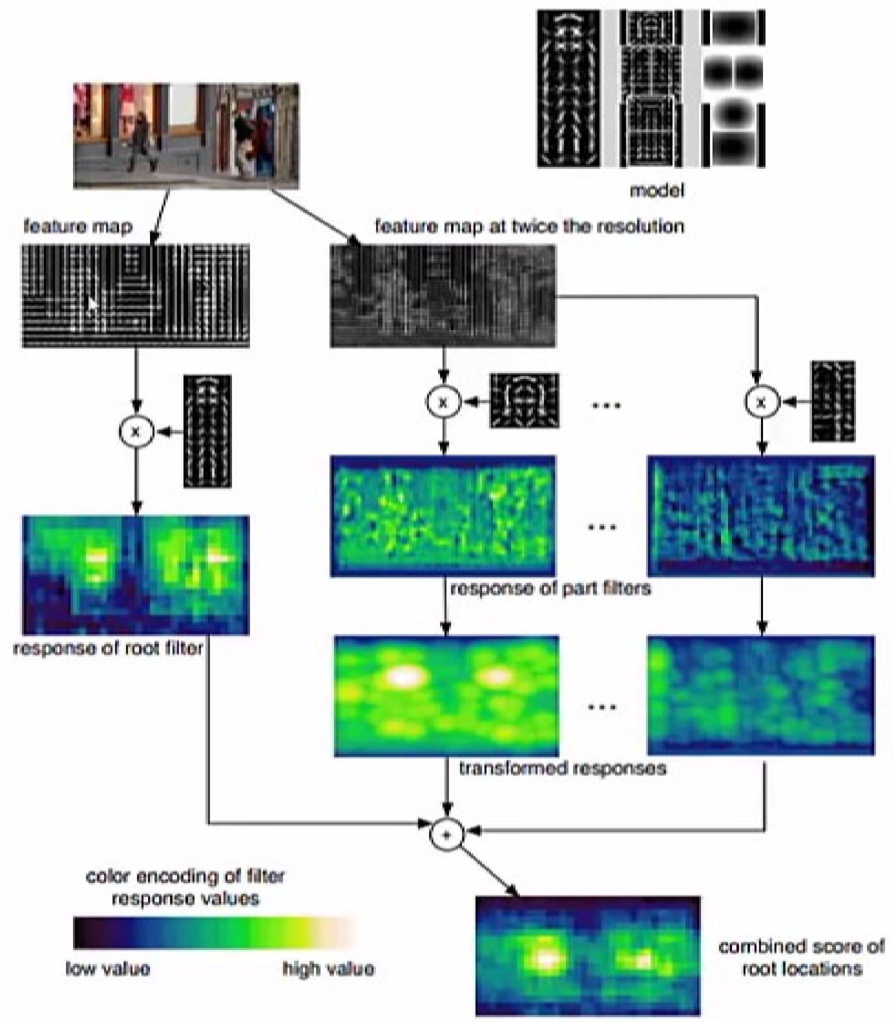
- The responses of each part detectors are combined to infer the holistic pedestrian.

Part-based pedestrian detection

- DPM – Felzenszwalb et al., Object Detection with Discriminatively Trained Part Based Models
- Breakthrough in object detection.

Part-based pedestrian detection

DPM



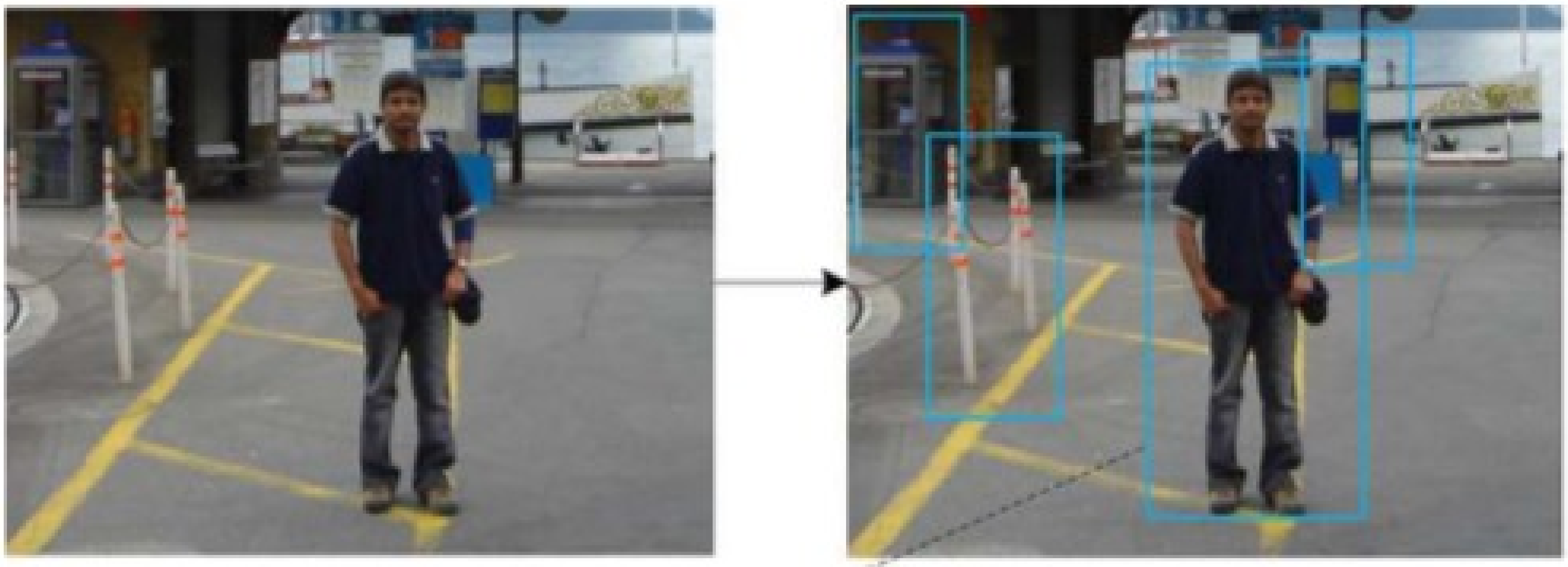
Our approach

- Convolutional Neural Network (CNN) has been successfully applied in pedestrian detection.
- When using large CNN for feature extraction, the commonly used “sliding-window” detection paradigm is hard to work for the computational efficiency problem.

Pedestrian detection approach

- The proposed pedestrian detection approach can be regarded as a two-stages system.
- In the first stage, we rapidly filter out as many negative windows as possible, while keeping all the positive windows.
- In the second stage we fine tune using part detectors.

Localizing candidate windows



Input

Localizing candidate windows

fppt.com

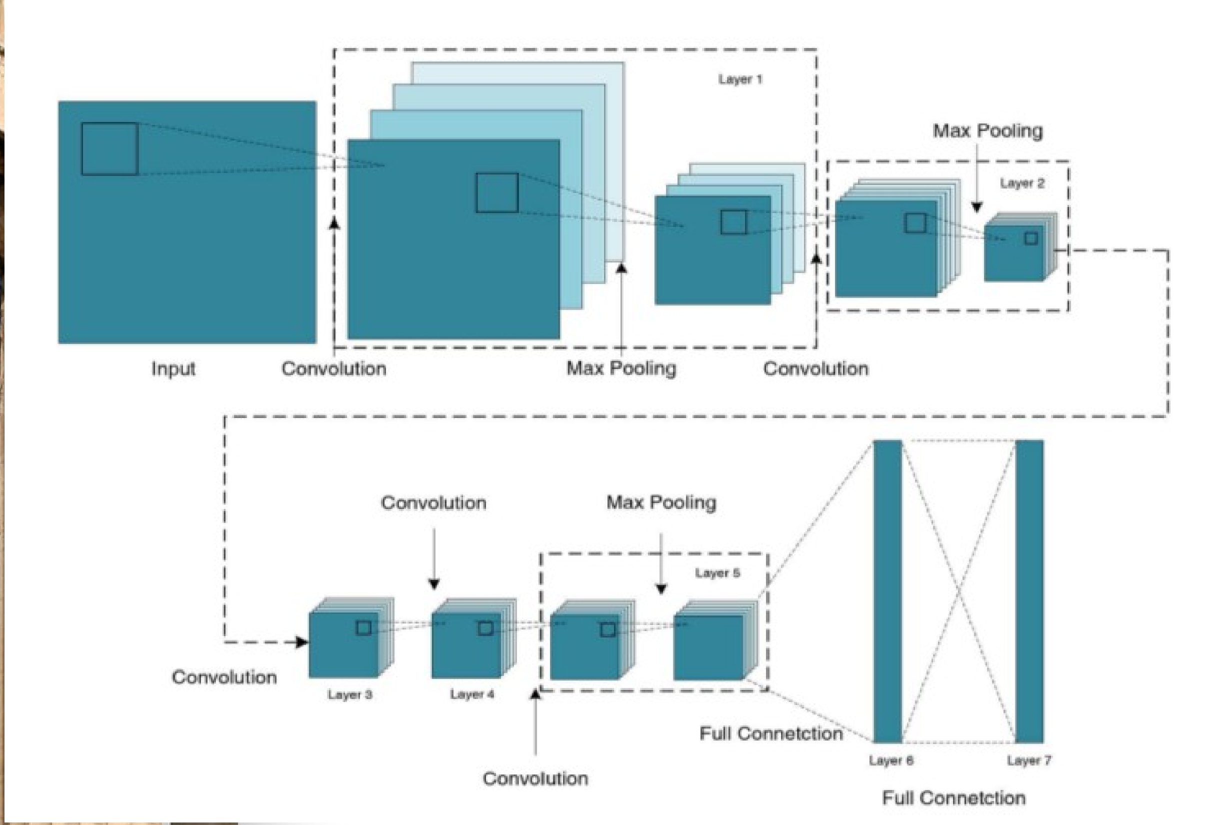
Localizing candidate windows

- We employ Aggregate Channel Features detector for localizing candidate windows.
- Three types of channel features are used: normalized gradient magnitude, histogram of oriented gradients and LUV color channels.
- Next, a bootstrapping iteration is conducted to construct a cascade of classifiers. We built a cascade of 2 stages.

Localizing candidate windows

	Selective Search	Objectness	**ACF**
Cover rate	97.62 %	93.55 %	**98.13 %**
Runtime	~4 sec	~4 sec	**~ 0.5 s**

Second stage – Fine detection



After obtaining the training features, we train a linear SVM for classification.

Second stage – Fine detection

- Let $X=\{x_0, x_1, \ldots, x_n\}$ denote the locations of object p0 and n parts $\{p_i\}_{i=1}^{n}$, which are annotated in the training data, but unknown at test time.
- Our goal is to infer both the object location and part locations in a previously unseen test image.

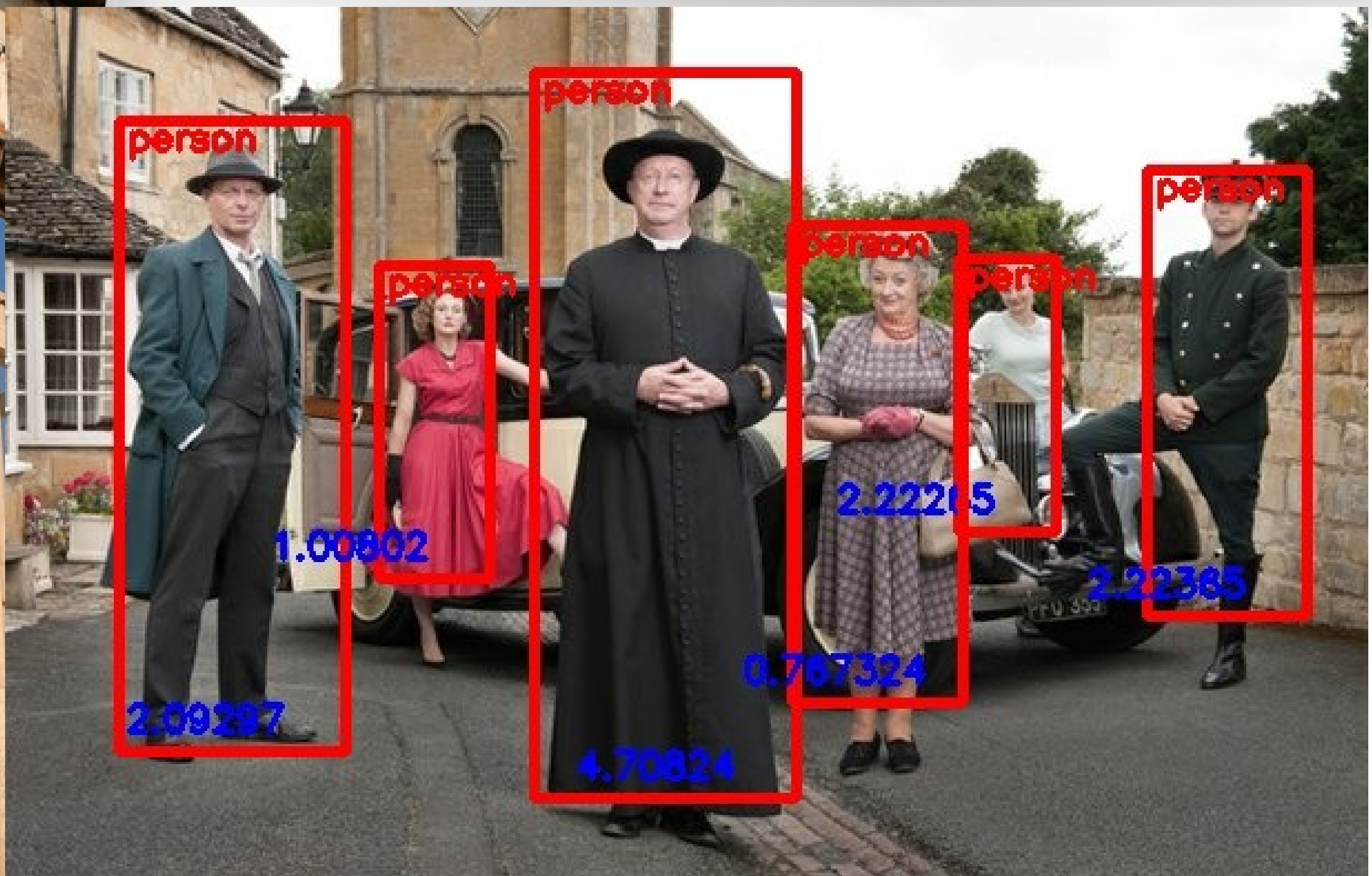
Second stage – Fine detection

- Given the CNN weights {w0, w1, …, wn} for object and parts, we will have the corresponding detectors {d0, d1, …, dn} where each detector score is $d_i(x)=\sigma(w_i^T\Phi(x))$, where σ(.) is the sigmoid function and Φ(x) is the CNN feature descriptor extracted at location x.

$$X^* = \arg\max_X \Delta(X) \prod_{i=0}^{n} d_i(x_i)$$

Results

Results


person
person
person
person
person
person
1.00802
2.09297
4.70824
2.22365

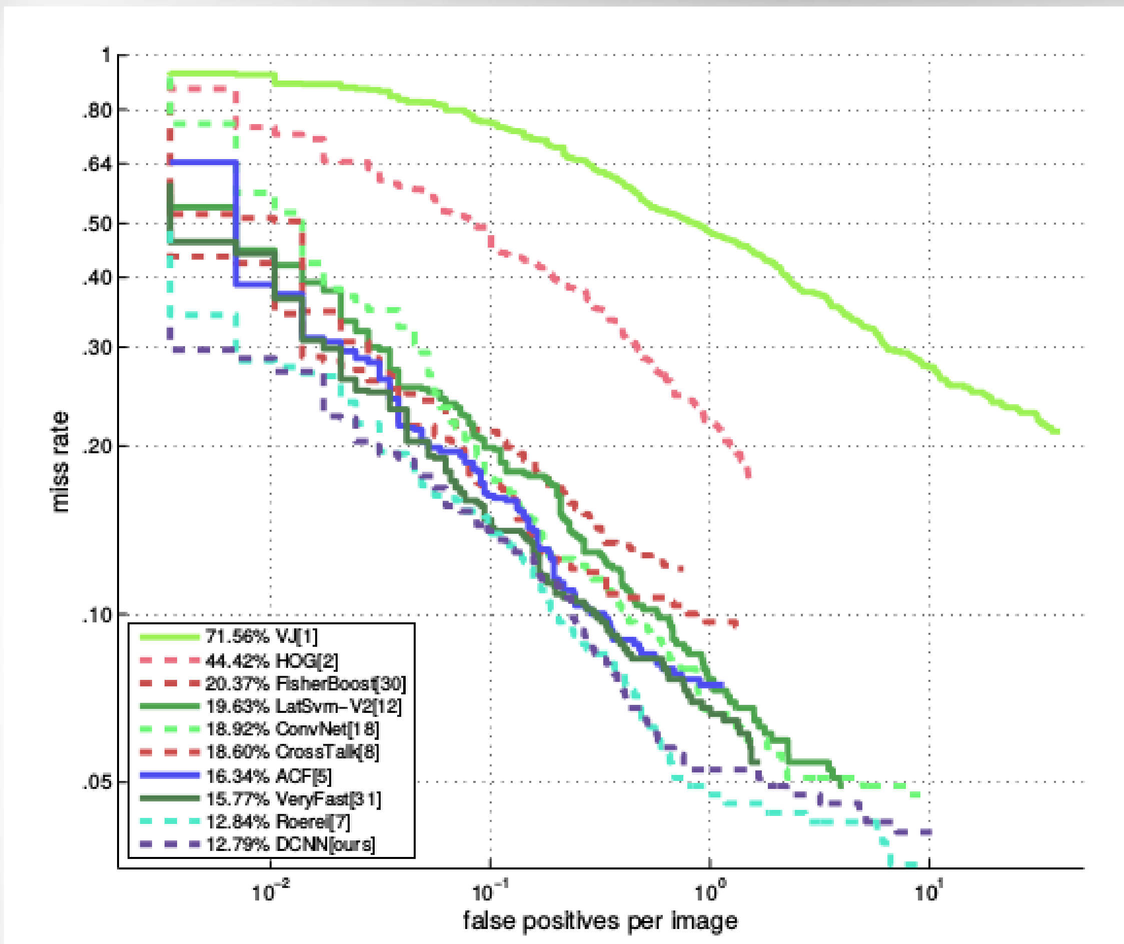
Results

Results

Results

Results

Results



miss rate
1
.80
.64
.50
.40
.30
.20
.10
.05
71.56% VJ[1]
44.42% HOG[2]
20.37% FisherBoost[30]
19.63% LatSvm-V2[12]
18.92% ConvNet[18]
18.60% CrossTalk[8]
16.34% ACF[5]
15.77% VeryFast[31]
12.84% Roerei[7]
12.79% DCNN[ours]
10^{-2}
10^{-1}
10^{0}
10^{1}
false positives per image


Haben Sie Fragen?