Deformable Part Models (DPMs) for Human Detection

Guangzhen Zhou 周光朕

gzzhou11@fudan.edu.cn

9/29/2014

Basis

P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan.Object detection with discriminatively trained part-based models. PAMI,2010.

P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In CVPR, 2008.

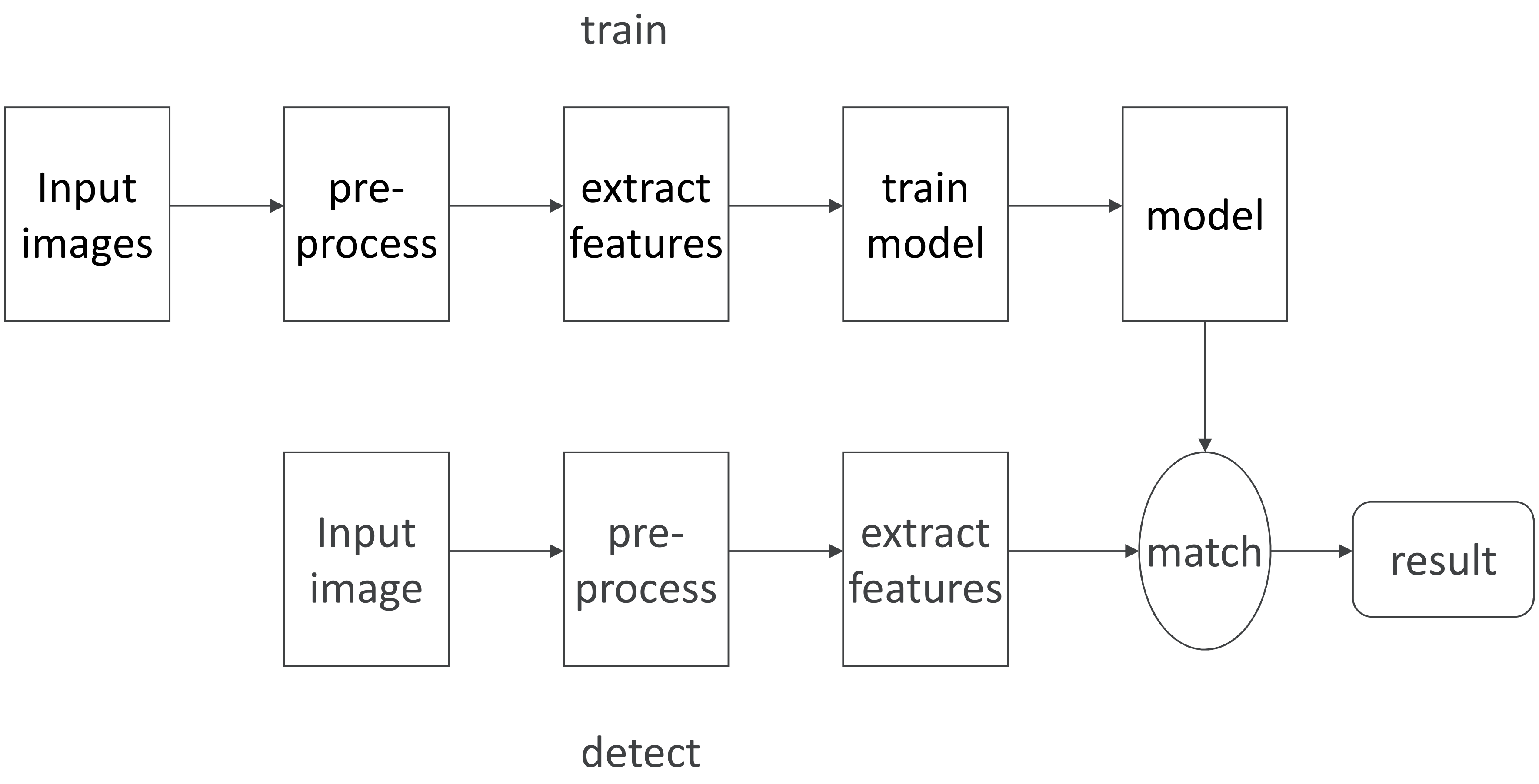
http://www.cs.berkeley.edu/~rbg/latent/

Outline

- Introduction
  - What is human detection
  - Human detection algorithms
  - Problems in video
  - Consider
- DPMs
  - About DPM
  - How it works: detection
  - Performance
  - How to get: training
- Discuss
  - Shortness
  - Methods[5]

What is human detection

What is human detection



Human detection algorithms

Using rigid templates: HOG+SVM

[3](CVPR2005)

Using bag of features: SIFT, Texture, LBP, Colour, ...

[4](IJCV2006)

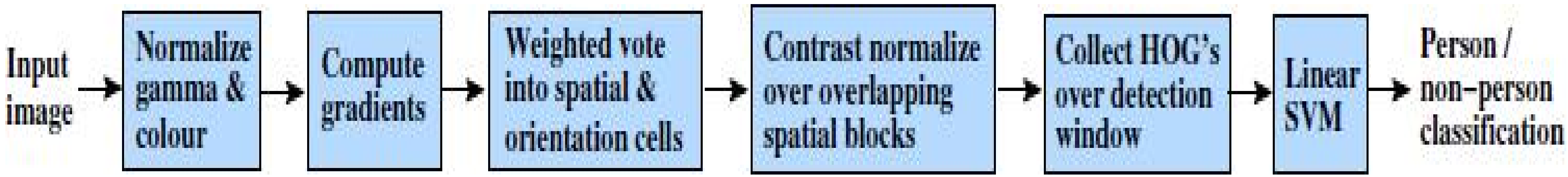
Human detection algorithms(cont.)

Bag of features: like "bag of words" in text retrieval

take features as words, use cluster methods

Human detection algorithms(cont.)

HOG: use distribution of local intensity gradient or edge direction represent local object appearance and shape



Human detection algorithms(cont.)

average gradient image over training data

positive

negative

Problems in video

occlusion

diversity

deformation

……

Consider

Previous methods are not effective enough

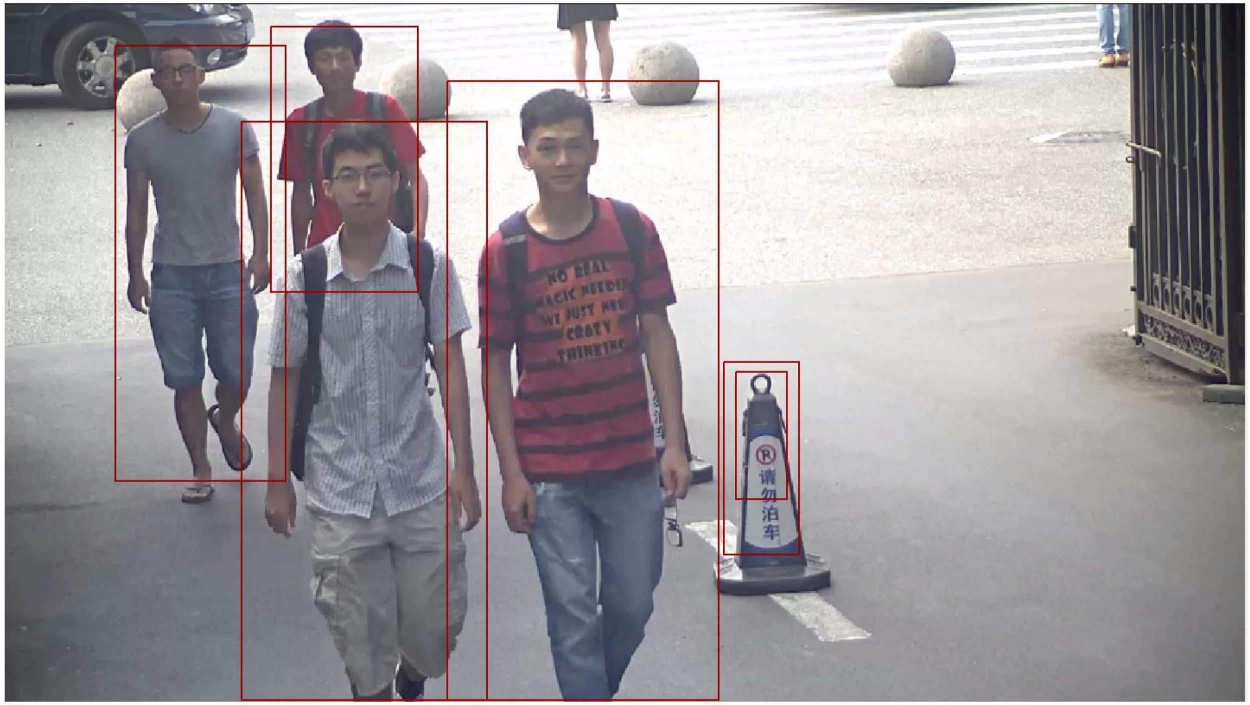
Rigid template: lose the deformation information

Bag of features: lose structured information

Consider(cont.)

Apply the winner of PASCAL VOC 2007,2008,2009 challenge

----DPM



About DPM

A kind of model

1.combine "deformation" and "part"

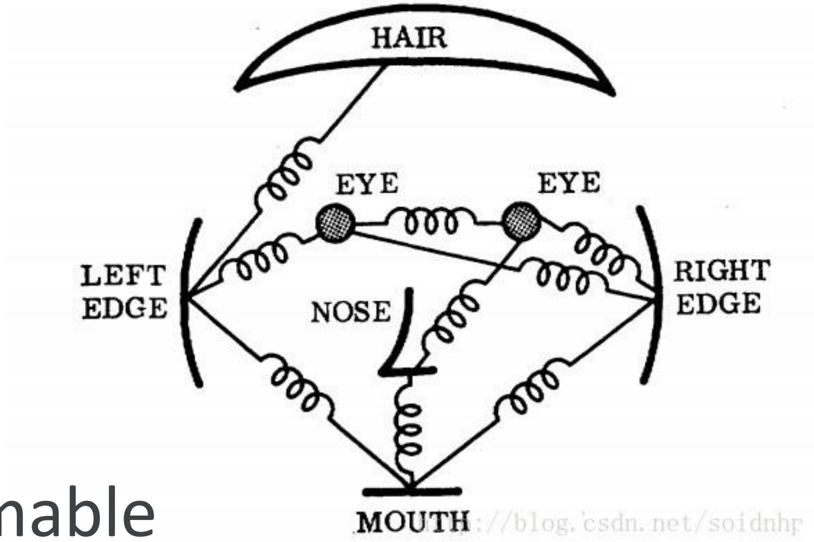
2.contain some other models

About DPM(cont.)

"deformation": deformable template model

"part": part-based model

About DPM(cont.)



1973: pictorial structures

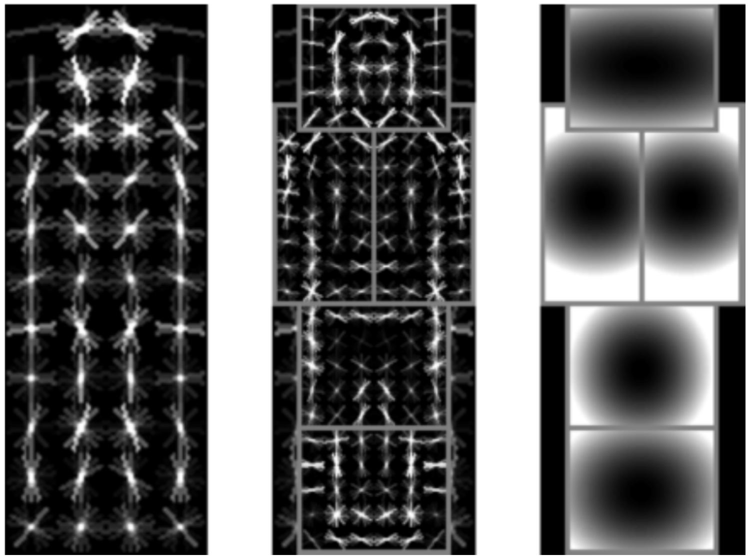
2005: parts with a deformable configuration, like spring

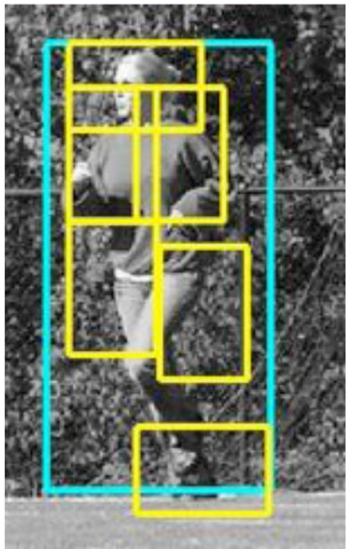
2010: enrich model in 1973 with star-structured model (add a root filter)

About DPM(cont.)

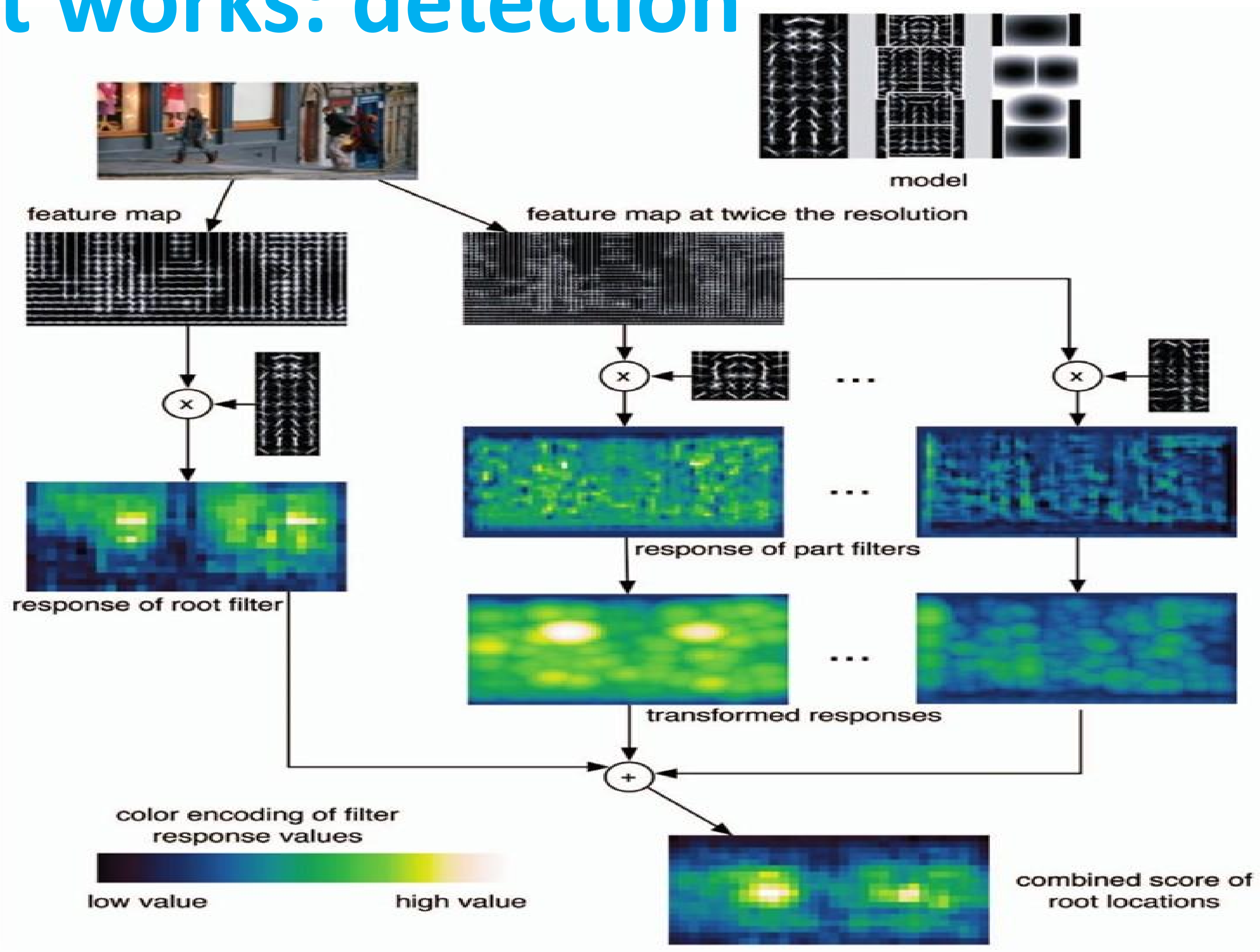
- a root filter + some (parts filter + spatial model)

- Parts filter at twice resolution of the root filter





How it works: detection



How it works: detection

1.input data

2.extract features

3.matching the model with feature map

4.get and threshold the score of matching

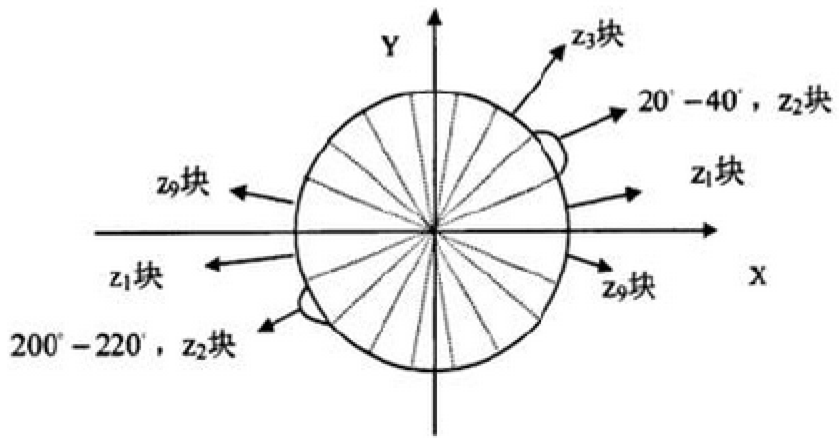
$$score(M, x) = score(root, x) + \sum_{p \in \{parts\}} \max_{y} [score(p, y) - loss(p, x, y)]$$

Features

choose: (18+9) orientations + 4 normalizations = 31-d

18: contrast sensitive; 9: constrast insensitive

for sake of all categories



Features(cont.)

HOG features pyramids

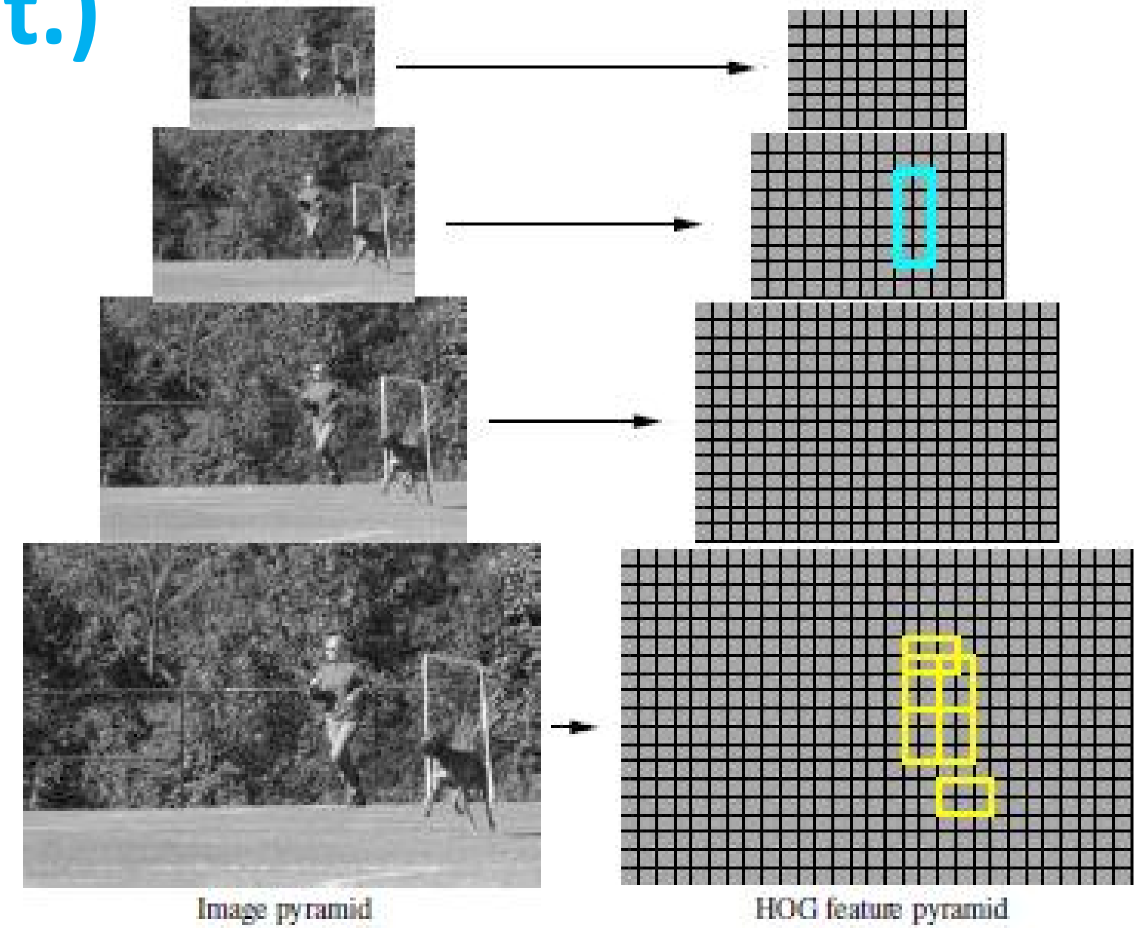


Figure 2. The HOG feature pyramid and an object hypothesis defined in terms of a placement of the root filter (near the top of the pyramid) and the part filters (near the bottom of the pyramid).

Filters

- rectanglar templates specify weights for subwindows of a HOG pyramid

F: w×h filter;

F': concatenating weight vectors in F in raw-major order

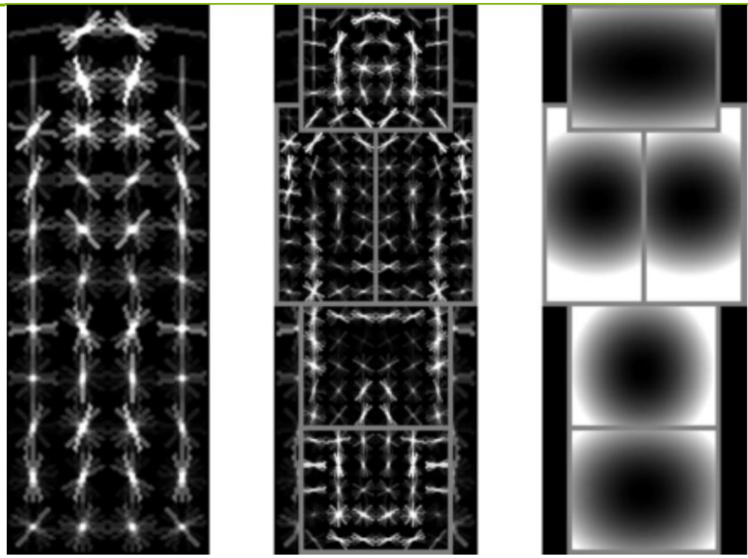
H: a HOG pyramid

p=(x,y,l): cell in the l-th level(position)

score of F at p: $F' \cdot \Phi(H,p,w,h)$

DPMs - How it works: detection

Deformable Parts



- the total model

A root filter F0

n parts Pi

- A filter Fi
- An anchor vi (2-d)
- quadratic func coefficients di (4-d; for defomation cost)

a bias term b

- An object hypothesis

Position of root and parts z = (p0,...,pn)

pi = (xi, yi, li)

Hypothesis: parts are twice the resolution of root

Deformable Parts(cont.)

- Score of a placement

$$score(p_0,\ldots,p_n) = \sum_{i=0}^{n} F_i' \cdot \varphi(H,p_i) - \sum_{i=1}^{n} d_i \cdot \varphi_d(dx_i,dy_i) + b$$

$$(dx_i,dy_i) = (x_i,y_i) - (2(x_0,y_0) + v_i)$$

$$\varphi_d(dx_i,dy_i) = (dx,dy,dx^2,dy^2)$$

- in dot product: β · Ψ(H, z), where:

$$\beta = (F_0',\ldots,F_n',d_1,\ldots,d_n,b)$$

$$\psi(H,z) = (\varphi(H,p_0),\ldots,\varphi(H,p_n),-\varphi_{\mathrm{d}}(dx_1,dy_1),\ldots,-\varphi_{\mathrm{d}}(dx_1,dy_1),1)$$

DPMs - How it works: detection

Matching

$$score(p_0) = \max_{p_1,\dots p_n} score(p_0,\dots,p_n)$$

$$R_{i,l}(x,y) = F_i^{'} \bullet \varphi(H,(x,y,l))$$

$$D_{i,l}(x,y) = \max_{dx,dy}(R_{i,l}(x+dx, y+dy) - d_i \bullet \varphi_d(dx,dy))$$

$$score(x_0,y_0,p_0) = R_{0,l_0}(x_0,y_0) + \sum_{i=1}^{n} D_{i,l_0\text{-}\lambda}(2(x_0,y_0)+v_i) + b$$

$$P_{i,l}(x,y) = \arg\max_{dx,dy} D_{i,l}(x,y)$$

Mixture Models

- model with m components, M = (M1,...,Mn)

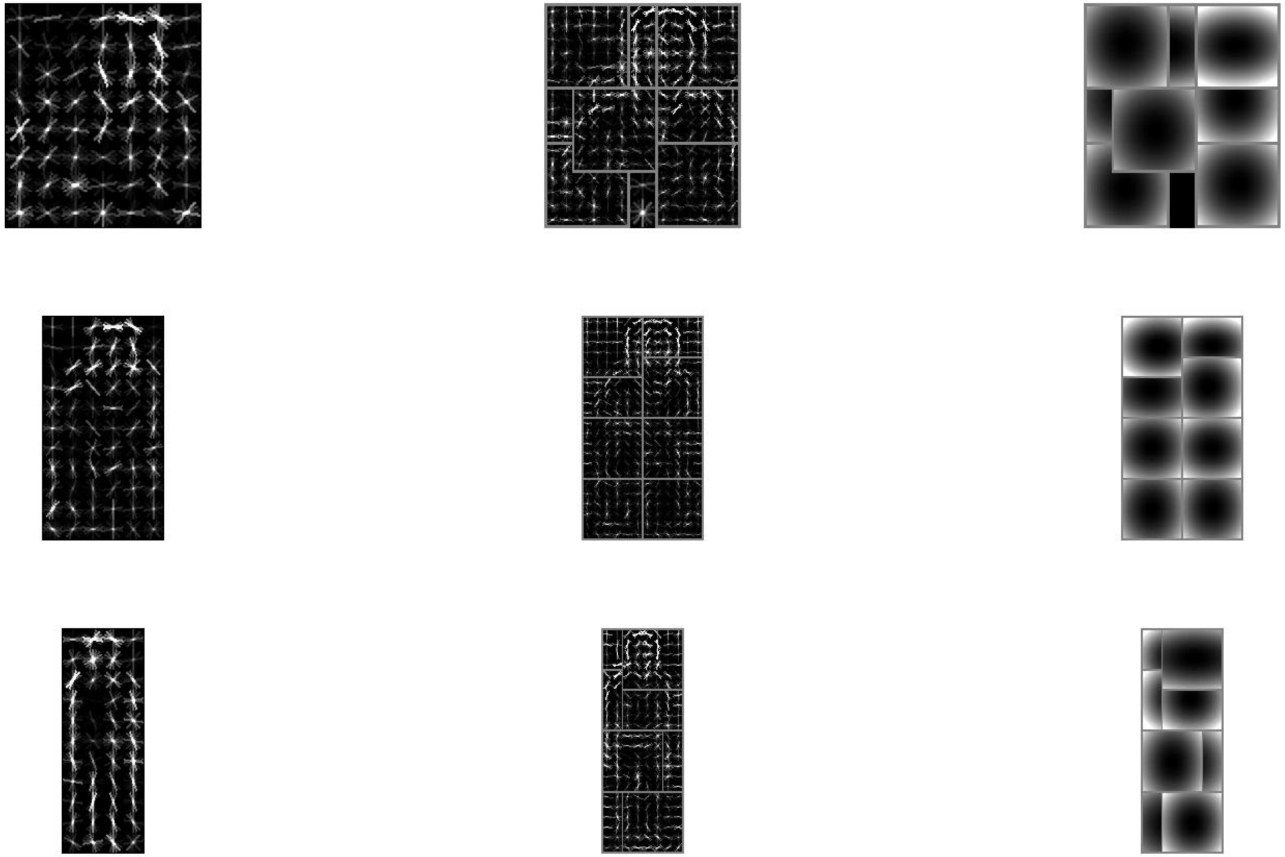
$$z' = (p_0, \ldots, p_{n_c})$$

$$\beta = (\beta_1, \ldots, \beta_m)$$

$$\psi(H, z) = (0, \ldots, 0, \varphi(H, z'), 0, \ldots, 0)$$

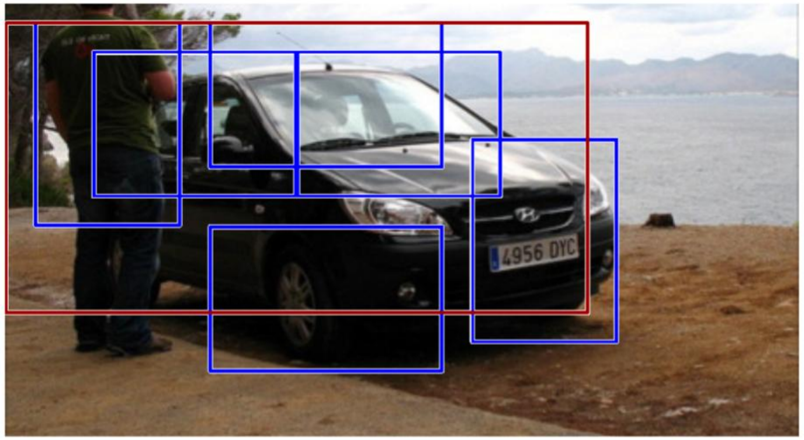
Mixture Models(cont.)

- example



Bounding box prediction

- use the part filter locations to fix the root filter location
- input: root width & each loaction
- output: bounding box prediction



Non-Maximum Suppression

- After thresholding score, sort all scores

- always choose the unchosen detection with highest score and ignore those bounding box is no less than 50% covered by a chosen one

Contextual Information

- aim: rescore the result to distinguish tp from fp

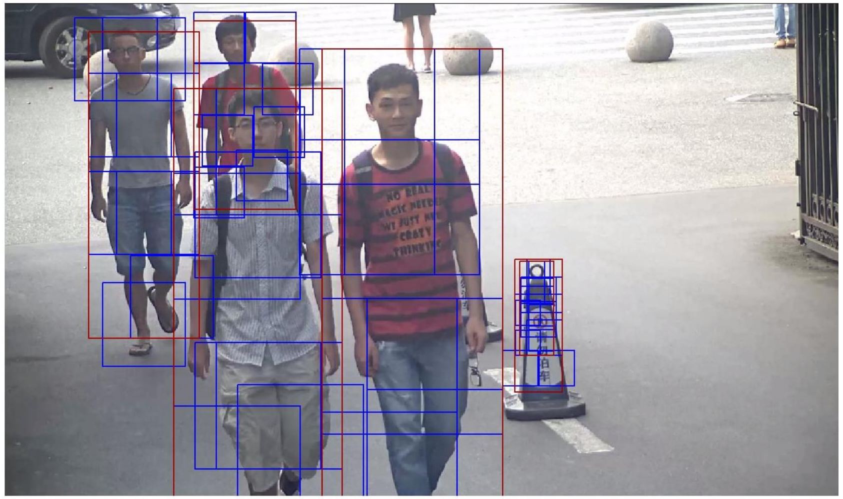
- (D1,....,Dk): results of different categories in one image
- (B, s): B = (x1, y1, x2, y2), s = score
- k-d c(I) = (σ(s1),...,σ(sk)) be contextual information of image I
- 25-d feature vector $g = (\sigma(s), x_1, y_1, x_2, y_2, c(I))$

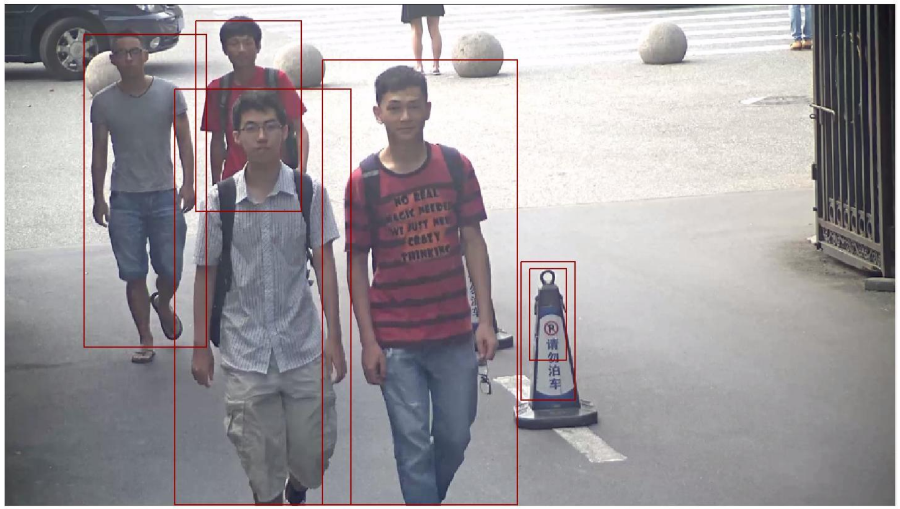
Contextual Information(cont.)

- use: score g with category-specific classifier to obtain a new score

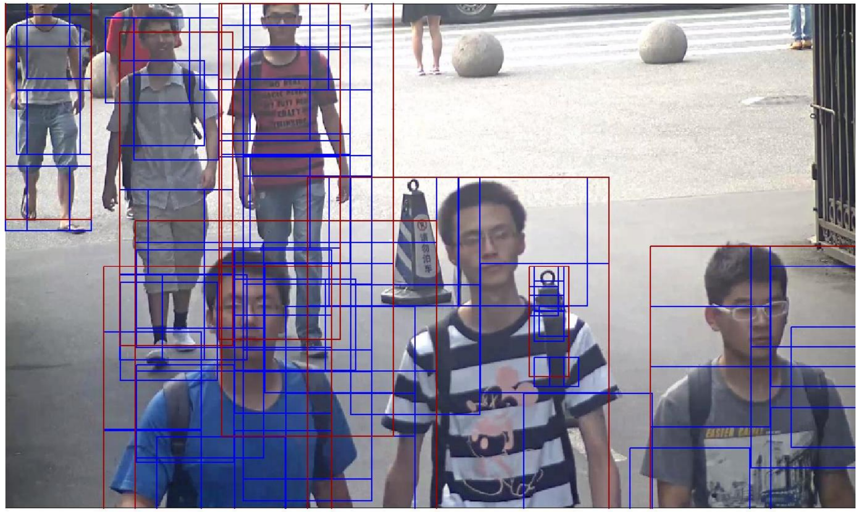
- train: run current classfier in dataset with given bounding box, judge result tp or fp by if there's significant cover with given b-box

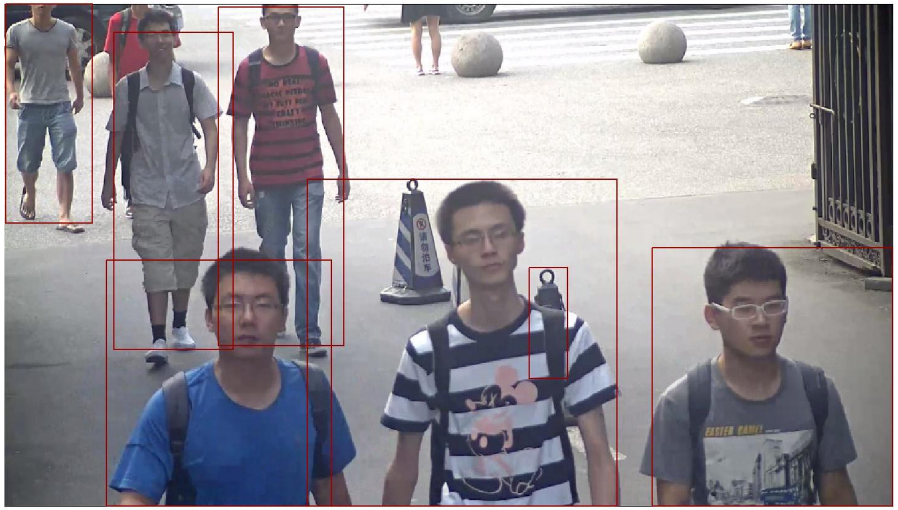
Performance



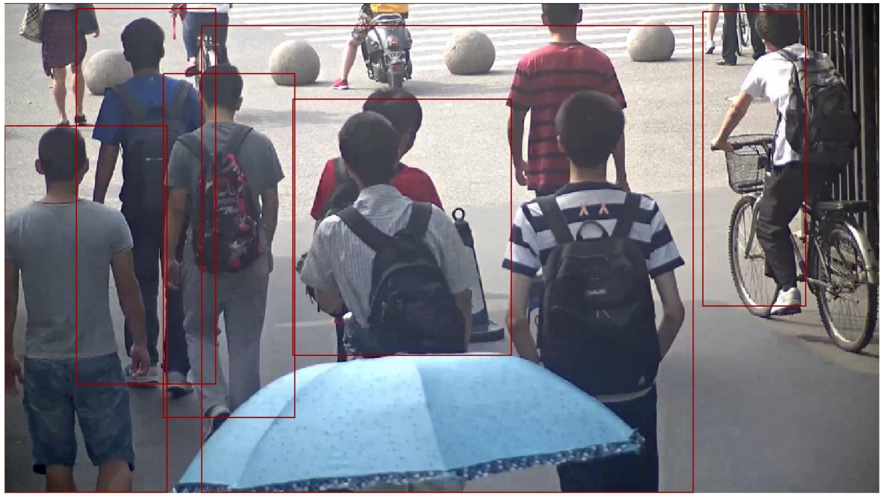


Performance





Performance



Performance

How to get: training

- binary classification problem

- D = (<x1, y1>, . . . , <xn, yn>)
  yi:label, { -1, 1}
  xi: HOG pyramid H(xi) & range of valid placement Z(xi)

- require bounding box for positive xi
  root filter must overlap b-box ≥50%

DPMs - How to get: training

Latent SVM

- classifier scores an example x use: $f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \varphi(x, z)$
- Z(x): set of possible latent values for x
- like SVM, learn β by minimizing:

$$L_D(\beta) = \frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$

- D is the dataset

DPMs - How to get: training

Semi-convexity

- maximum of some convex functions is convex
- $f_\beta(x)$: linear in β, thus convex
- $\max(0, 1-y_i f_\beta(x_i))$: hinge loss, only when $y_i=-1$, convex

- if $Z(x_i)$ has only one possible latent value, $f_\beta(x_i)$ ->linear, thus, the hinge loss is convex.

DPMs - How to get: training

Optimization

let:

- Zp: specify latent value for each pos. example
- D(Zp): derived from D according Zp

$$L_D(\beta) = \min_{Z_p} L_D(\beta, Z_p) = \min_{Z_p} L_{D(Z_p)}(\beta)$$

Optimization(cont.)

- algorithm for minimizing $L_D(\beta, Z_p)$

1) *Relabel positive examples:* Optimize $L_D(\beta, Z_p)$ over Z_p by selecting the highest scoring latent value for each positive example, $z_i = \text{argmax}_{z \in Z(x_i)} \beta \cdot \Phi(x_i, z)$.
2) *Optimize beta:* Optimize $L_D(\beta, Z_p)$ over β by solving the convex optimization problem defined by $L_{D(Z_p)}(\beta)$.

- step2 can be done by quadratic programming or stochastic gradent descent

DPMs - How to get: training

Data-mining hard examples

- what is "hard examples"?

$$H(\beta, D) = \{\langle x, y \rangle \in D \mid y f_\beta(x) < 1\}.$$

$$E(\beta, D) = \{\langle x, y \rangle \in D \mid y f_\beta(x) > 1\}.$$

$$H(\beta, D) = \{(i, \Phi(x_i, z_i)) \mid$$
$$z_i = \underset{z \in Z(x_i)}{\operatorname{argmax}} \beta \cdot \Phi(x_i, z) \text{ and } y_i(\beta \cdot \Phi(x_i, z_i)) < 1\}$$

- aim: collect hard examples as incorrectly classfied examples from a previous model to enhance the model

DPMs - How to get: training

Learning

Data:
Positive examples $P = \{(I_1, B_1), \ldots, (I_n, B_n)\}$
Negative images $N = \{J_1, \ldots, J_m\}$
Initial model β

Result: New model β

$F_n := \emptyset$
for *relabel* $:= 1$ **to** *num-relabel* **do**
    $F_p := \emptyset$
    for $i := 1$ **to** n **do**
        Add `detect-best`(β,I_i,B_i) to F_p
    end
    for *datamine* $:= 1$ **to** *num-datamine* **do**
        for $j := 1$ **to** m **do**
            if $|F_n| \geq$ *memory-limit* **then break**
            Add `detect-all`(β,J_j,$-(1+\delta)$) to F_n
        end
        β :=`gradient-descent`($F_p \cup F_n$)
        Remove (i, v) with $\beta \cdot v < -(1+\delta)$ from F_n
    end
end

Procedure `Train`

Shortness

- For the demo images given in section DPM - Performance, the size and detection time is below

1003×563	998×565	1002×562	1001×563
8.005s	8.012s	8.008s	8.736s

- so the speed of DPM for human detection is very slow!

- For the project: trained model may not be suitable enough

Methods[5]

- Pyramids of templates(model)
- Cascades: first root(rough), then parts(fine)
- Vector quantization
- ……
- For video concern: cascades with parts of a frame (ROI)

Reference

[1] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan.Object detection with discriminatively trained part-based models. PAMI,2010.

[2] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In CVPR, 2008.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In CVPR, pages I: 886–893, 2005.

[4] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," International Journal of Computer Vision, vol. 73, no. 2, pp. 213–238, June 2007.

[5] Mohammad Amin Sadeghi, David Forsyth. 30Hz Object Detection with DPM V5. ECCV,2014.

Thank you!
Q&A