Deformable Part Models (DPMs) for Human Detection

Guangzhen Zhou  周光朕

gzzhou11@fudan.edu.cn

9/29/2014


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

defformation

......
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
"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

Deformable Part Models | How it works: detection | Guangzhen Zhou | gzzhou11@fudan.edu.cn
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: \( w \times h \) 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 \( p \): \( F' \cdot \Phi(H,p,w,h) \)
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 deformation 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

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

\[(dx_i, dy_i) = (x_i, y_i) - (2(x_0, y_0) + v_i)\]

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

- in dot product: \[\beta \cdot \Psi(H, z)\], where:

\[\beta = (F_0', ..., F_n', d_1, ..., d_n, b)\]

\[\Psi(H, z) = (\varphi(H, p_0), ..., \varphi(H, p_n), -\varphi_d(dx_1, dy_1), ..., -\varphi_d(dx_1, dy_1), 1)\]
Matching

\[ \text{score}(p_0) = \max_{p_1, \ldots, p_n} \text{score}(p_0, \ldots, p_n) \]

\[ R_{i,l}(x, y) = F_i' \cdot \varphi(H, (x, y, l)) \]

\[ D_{i,l}(x, y) = \max_{dx, dy} \left( R_{i,l}(x + dx, y + dy) - d_i \cdot \varphi_d(dx, dy) \right) \]

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

\[ P_{i,l}(x, y) = \arg\max_{dx, dy} D_{i,l}(x, y) \]
Mixture Models

- model with m components, $M = (M_1, ..., M_n)$

\[ z' = (p_0, ..., p_n) \]
\[ \beta = (\beta_1, ..., \beta_m) \]
\[ \psi(H, z) = (0, ..., 0, \varphi(H, z'), 0, ..., 0) \]
Mixture Models (cont.)

example
Bounding box prediction

- use the part filter locations to fix the root filter location
- input: root width & each location
- 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

- \((D_1,\ldots,D_k)\): results of different categories in one image

- \((B, s)\): \(B = (x_1, y_1, x_2, y_2)\), \(s = \text{score}\)

- \(k\)-d \(c(I) = (\sigma(s_1),\ldots,\sigma(s_k))\) 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 classifier in dataset with given bounding box, judge result $tp$ or $fp$ by if there's significant cover with given $b$-box.
DPMs

Performance
DPMs

Performance
Performance
DPMs

Performance

Guangzhen Zhou  gzzhou11@fudan.edu.cn
How to get: training

1. **Binary classification problem**

2. \[ D = (x_1, y_1), \ldots, (x_n, y_n) \]
   - \( y_i: \) label, \( \{-1, 1\} \)
   - \( x_i: \) HOG pyramid \( H(x_i) \) & range of valid placement \( Z(x_i) \)

3. require bounding box for positive \( x_i \)
   root filter must overlap b-box \( \geq 50\% \)
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 $\beta$ 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
Semi-convexity

- maximum of some convex functions is convex
- $f_\beta(x)$: linear in $\beta$, 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, $f_\beta(x_i) \rightarrow$linear, thus, the hinge loss is convex.
Optimization

let:

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

\[ L_D(\beta) = \min_{Z_p} L_D(\beta, Z_p) = \min_{Z_p} L_{D(Z_p)}(\beta) \]
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, 
$$z_i = \arg\max_{z \in Z(x_i)} \beta \cdot \Phi(x_i, z).$$

2) Optimize beta: Optimize $L_D(\beta, Z_p)$ over $\beta$ by solving the convex optimization problem defined by $L_D(Z_p)(\beta)$.

- step 2 can be done by quadratic programming or stochastic gradient descent
Data-mining hard examples

- what is "hard examples"?

\[
H(\beta, D) = \{(x, y) \in D \mid yf_\beta(x) < 1\}. \\
E(\beta, D) = \{(x, y) \in D \mid yf_\beta(x) > 1\}. \\
H(\beta, D) = \{(i, \Phi(x_i, z_i)) \mid z_i = \arg\max_{z \in Z(x_i)} \beta \cdot \Phi(x_i, z) \text{ and } y_i(\beta \cdot \Phi(x_i, z_i)) < 1\}.
\]

- aim: collect hard examples as incorrectly classified examples from a previous model to enhance the model
**Learning**

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

**Initial model** $\beta$

**Result:** New model $\beta$

```plaintext
1 $F_n := \emptyset$

2 for relabel := 1 to num-relabel do
3     $F_p := \emptyset$
4     for $i := 1$ to $n$ do
5         Add detect-best $(\beta, I_i, B_i)$ to $F_p$
6     end
7 for datamine := 1 to num-datamine do
8     for $j := 1$ to $m$ do
9         if $|F_n| \geq$ memory-limit then break
10        Add detect-all $(\beta, J_j, -(1 + \delta))$ to $F_n$
11     end
12 $\beta :=$ gradient-descent $(F_p \cup F_n)$
13 Remove $(i, u)$ with $\beta \cdot u < -(1 + \delta)$ from $F_n$
14 end
15 end

Procedure Train
```
Shortness

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

<table>
<thead>
<tr>
<th>Size</th>
<th>Detection Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1003 × 563</td>
<td>8.005s</td>
</tr>
<tr>
<td>998 × 565</td>
<td>8.012s</td>
</tr>
<tr>
<td>1002 × 562</td>
<td>8.008s</td>
</tr>
<tr>
<td>1001 × 563</td>
<td>8.736s</td>
</tr>
</tbody>
</table>

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

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

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)


Thank you!

Q&A