Object Detection with Discriminatively Trained Part Based Models

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester and Deva Ramanan

Presented by Amy Bearman and Amani Peddada
Roadmap

1. Introduction
2. Related Work
3. Model Overview
4. Latent SVM
5. Features & Post Processing
6. Experiments
Introduction

- **Problem**: Detecting and localizing generic objects from various categories, such as cars, people, etc.

- **Challenges**: Illumination, viewpoint, deformations, intraclass variability
How they solve it

Mixtures of multi-scale deformable part model

• Trained with a discriminative procedure

• Data is partially labeled (bounding boxes, not parts)
Deformable parts model

- Represents an object as a collection of parts arranged in a deformable configuration
- Each part represents local appearances
- Spring-like connections between certain pairs of parts
One motivation of this paper

To address the performance gap between simpler models:

... and sophisticated models like **deformable parts**
Why do simpler models perform better?

• Simple models are easily trained using **discriminative methods** such as SVMs

• Richer models use **latent information** (location of parts)
Roadmap

1. Introduction

2. Related Work

3. Model Overview

4. Latent SVM

5. Features & Post Processing

6. Experiments
Related Work: Detection

- **Bag-of-Features**

- **Rigid Templates**
  - Dalal-Triggs

- **Deformable Models**
  - Deformable Templates (e.g. Active Appearance Models)
  - Part-Based Models — Constellation, Pictorial Structure
Dalal-Triggs Method

- **Histogram of Oriented Gradients for Human Detection** - Dalal and Triggs, 2005

- Sliding Window, HOG feature extraction + Linear SVM

- One of the most influential papers in CV!
Active Appearance Model

- *Active Appearance Models* - Cootes, Edwards, and Taylor, 1998

- Attempts to match statistical model to new image using iterative scheme
Deformable Models — Constellation

• **Object class recognition by unsupervised scale-invariant learning** - Fergus et al., 2003

• Utilizes Expectation Maximization to determine parameters of scale-invariant model

• Entropy-based feature detector.

• Appearance learnt simultaneously with shape.
Constellation Models

• *Towards Automatic Discovery of Object Categories* - Weber et al., 2000

• Derives Mixture Models and a probabilistic framework for modeling classes with large variability

• Constrained to testing on faces, leaves, and cars.

• Automatically selects distinctive features of object class
Pictorial Structure Models

- The Representation and Matching of Pictorial Structures - Fischler & Elschlager, 1973

- Formalizes a dynamic programming approach ("Linear Embedding Algorithm") to find optimal configuration of part-based model.
Pictorial Structure Models

- *Pictorial Structures for Object Recognition* - Felzenszwalb et al., 2005

- Finds multiple optimal hypotheses; presents framework as a energy minimization problem over graph

- Poses novel, efficient minimization techniques to achieve reasonable results on face/body image data.
Roadmap

1. Introduction
2. Related Work
3. Model Overview
4. Latent SVM
5. Features & Post Processing
6. Experiments
Starting point: sliding window classifiers

- Detect objects by testing each sub-window
- Reduces object detection to binary classification
- Dalal & Triggs: HOG features + linear SVM classifier
- Previous state of the art for detecting people
Innovations on Dalal-Triggs

- Star model = root filter + set of part filters and associated deformation models
HOG Filters

- Models use linear filters applied to dense feature maps.

- **Feature map** = array of feature vectors, where each feature vector describes a local image patch.

- **Filter** = rectangular template = array of weight vectors.

- **Score** = dot product of the filter and a sub-window of the feature map.
Feature Pyramid

Filter $F$

Score of $F$ at position $p$ is $F \cdot \phi(p, H)$

$\phi(p, H)$ = concatenation of HOG features from subwindow specified by $p$
Model Overview

• Mixture of deformable part models

• Each component has global component + deformable parts

• Fully trained from bounding boxes alone
Deformable Part Models

- Star model: coarse root filter + higher resolution part filters

- Higher resolution features for part filters is essential for high recognition performance
Deformable Part Models

• A model for an object with $n$ parts is a $(n + 2)$ tuple:

$$ (F_0, P_1, \cdots, P_n, b) $$

- $F_0$ is the root filter
- $P_1, \cdots, P_n$ are models for the first part to the $n$th part
- $b$ is the bias term

• Each part-based model defined as:

$$ (F_i, v_i, d_i) $$

- $F_i$ is the filter for the $i$-th part
- $v_i$ is the "anchor" position for part $i$ relative to the root position
- $d_i$ defines a deformation cost for each possible placement of the part relative to the anchor position
Object Hypothesis

\( p_i = (x_i, y_i, l_i) \) specifies the level and position of the \( i \)-th filter.

- \( p_0 \): location of root
- \( p_1, \ldots, p_n \): location of parts

Score is sum of filter scores minus deformation costs.

Multiscale model captures features at two-resolutions.
Score of Object Hypothesis

\[
\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2) + b
\]

\[
\text{score}(z) = \beta \cdot \Psi(H, z)
\]

concatenation filters and deformation parameters

concatenation of HOG features and part displacement features
Matching

• Define an overall score for each root location according to the best placement of parts:

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

• High scoring root locations define detections (“sliding window approach”)

![Image of a puppy and kittens with red boxes highlighting them]
Matching Step 1: Compute filter responses

- Compute arrays storing the response of the $i$-th model filter in the $l$-th level of the feature pyramid (cross correlation):

\[
R_{i,l}(x, y) = F'_i \cdot \phi(H, (x, y, l))
\]
Matching Step 2: Spatial Uncertainty

- Transform the responses of the **part filters** to allow for spatial uncertainty:

\[ D_{i,t}(x, y) = \max_{dx, dy} (R_{i,t}(x + dx, y + dy) - d_i \cdot \phi_d(dx, dy)) \]
Matching Step 3: Compute overall root scores

- Compute overall root score at each level by summing the root filter response at that level, plus the contributions from each part:

\[
\text{score}(x_0, y_0, l_0) = R_{0,l_0}(x_0, y_0) + \sum_{i=1}^{n} D_{i,l_0} - \lambda (2(x_0, y_0) + v_i) + b
\]
Matching Step 4: Compute optimal part displacements

\[ P_{i,l}(x, y) = \arg \max_{dx,dy} (R_{i,l}(x + dx, y + dy) - d_i \cdot \phi_d(dx, dy)) \]

- After finding a root location \((x_0, y_0, l_0)\) with a high score, we can find the corresponding part locations by looking up the optimal displacements in \( P_{i,l_0} - \lambda (2(x_0, y_0) + v_i) \)
Mixture Models

A mixture model with $m$ components is $M = (M_1, \ldots, M_m)$ where $M_c$ is the model for the $c$-th component.

An object hypothesis for a mixture model consists of:

• A mixture component, $1 \leq c \leq m$

• A location for each filter of $M_c$, $z = (c, p_0, \ldots, p_{n_c})$

Score of hypothesis: $\beta \cdot \phi(H, z) = \beta_c \cdot \phi(H, z')$

To detect objects using a mixture model we use the matching algorithm to find root locations that yield high scoring hypotheses independently for each component.
Roadmap

1. Introduction
2. Related Work
3. Model Overview
4. Latent SVM
5. Features & Post Processing
6. Experiments
Training

- Training data consists of images with labeled bounding boxes

- **Weakly labeled setting** since the bounding boxes don’t specify component labels or part locations

- Need to learn the model structure, filters and deformation costs
SVM Review

- Separable by a hyperplane in high-dimensional space
- Choose the hyperplane with the max margin
Latent SVM

- Classifiers that score an example $x$ using

$$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z)$$

- $\beta$ are model parameters, $z$ are latent values

- Training data $D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle)$ where $y \in \{-1, 1\}$

- Learning: find $\beta$ such that $y_i f_\beta(x_i) > 0$

- Minimize:

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

Vector of HOG features and part offsets
Semi-convexity

- Maximum of convex functions is convex

\[ f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z) \text{ is convex in } \beta \]

\[ \max(0, 1 - y_i f_\beta(x_i)) \text{ is convex for negative examples} \]

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

- Convex if latent values for positive examples are fixed

- Important because it makes optimizing \( \beta \) a convex optimization problem, even though the latent values for the negative examples are not fixed
Latent SVM Training

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

• Convex if we fix \( z \) for positive examples

• Optimization:
  • Initialize \( \beta \) and iterate:
  • Pick best \( z \) for each positive example
  • Optimize \( \beta \) via gradient descent with data-mining
Training Models

• Reduce to Latent SVM training problem

• Positive example specifies some $z$ should have high score

• Bounding box defines range of root locations

  • Parts can be anywhere

  • This defines $Z(x)$ (vector of part offsets)
Training Algorithm

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 $\beta$

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

Procedure Train
Training Algorithm

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 $\beta$

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 $|F_n| \geq \text{memory-limit}$ then break
         10. Add detect-all ($\beta, J_j, -(1 + \delta)$) to $F_n$
      11. end
      12. $\beta := \text{gradient-descent} \ (F_p \cup F_n)$
      13. Remove $(i, v)$ with $\beta \cdot v < -(1 + \delta)$ from $F_n$
   14. end
15. end

Procedure Train

Finds the highest scoring object hypothesis with a root filter that significantly overlaps $B$ in $I$. Implemented with matching procedure.
Training Algorithm

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 $\beta$

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.     end
13.     $\beta :=$ gradient-descent ($F_p \cup F_n$)
14.     Remove $(i, v)$ with $\beta \cdot v < -(1 + \delta)$ from $F_n$
15. end

Procedure Train

Computes the best object hypothesis for each root location and selects the ones that score above a threshold. Implemented with matching procedure.
Training Algorithm

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 $\beta$

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. $\beta := \text{gradient-descent} (F_p \cup F_n)$
      13. Remove $(i, v)$ with $\beta \cdot v < -(1 + \delta)$ from $F_n$
   14. end
15. end

Procedure Train
Roadmap

1. Introduction
2. Related Work
3. Model Overview
4. Latent SVM
5. Features & Post Processing
6. Experiments
Histogram of Gradient features

- Image is partitioned into 8x8 pixel blocks
- In each block we compute a histogram of gradient orientations
  - Invariant to changes in lighting, small deformations
- Compute features at different resolutions (pyramid)
  - They use $\lambda = 5$ in training, $\lambda = 10$ in testing

$\lambda$ = number of levels we need to go down in the pyramid to get to a feature map computed at twice the resolution of another one
Background

• Negative example specifies no $\sim$ should have high score

• One negative example per root location in a background image

• Huge number of negative examples

• Consistent with requiring low false-positive rate
Post Processing: Bounding Box Prediction

- Predict \((x_1, y_1)\) and \((x_2, y_2)\) from part locations

- Learn four linear functions for predicting \(x_1, x_2, y_1, y_2\)
  Done via linear least-squares regression, independently for each component of a mixture model.
Roadmap

1. Introduction
2. Related Work
3. Model Overview
4. Latent SVM
5. Features & Post Processing
6. Experiments
Car Model

- root filters coarse resolution
- part filters finer resolution
- deformation models
Person Model

- root filters
- coarse resolution

- part filters
- finer resolution

- deformation models
Bottle Model

root filters  
coarse resolution

part filters  
finer resolution

deformation  
models
Car Detections

high scoring true positives

high scoring false positives
Person Detections

high scoring true positives

high scoring false positives (not enough overlap)
Horse Detections

high scoring true positives

high scoring false positives
Quantitative Results

- PASCAL Challenge: ~10,000 images, with ~25,000 target objects
  - Objects from 20 categories (person, car, bicycle, cow, table...)
- Out of 20 classes we got:
  - First place in 7 classes
  - Second place in 8 classes
- Some statistics:
  - Takes ~2 seconds to evaluate a model in one image
  - Takes ~4 hours to train a model
  - MUCH faster than most systems
Comparison of Car Models on 2006 Data

Results for:

1- and 2-component models, with and without parts

2-component model with parts and bounding box prediction

Average precision
Comparison of Person Models on 2006 Data

Results for:

1- and 2-component models, with and without parts

2-component model with parts and bounding box prediction
Summary

• Object detection based on mixtures of multiscale deformable models

• Discriminative training of classifiers that use latent information

  • Fast matching algorithms

  • Learning from weakly-labeled data (no component labels or part locations)

  • Leads to state-of-the-art results in PASCAL challenge
Questions?