Combining Detection, Recognition and Segmentation

Presented by:

Ira Kemelmacher-Shlizerman          Sharon Alpert

For the Advanced Topics in Computer Vision course
Spring 2006
Words Matching in Ascii

Find the word *sleeping* in the string:

“I have never taken any exercise except sleeping and resting”

Mark Twain

Simply go over the string and match each substring of length 8.
But what if we want to detect a face?
Detection/recognition challenges (partial list)

• Partial occlusions.
• Variation in pose.
• intra-class variations
• Lighting, scale, deformations, background clutter etc.
Try to find the face in this image
Segmentation

Group homogeneous areas

Use low-level cues:

- Intensity
- Color
- Texture
Segmentation - where is the edge?

Low-level local cues are not enough to get meaningful segmentation
Segmentation & Recognition

- Segmentation $\leftrightarrow$ recognition

Segmentation helps recognition

Recognized objects constrain segmentation
Labeling & Segmentation

Knowing the image type or the labeling

cost

buildings

Can constraint the segmentation
Labeling & Recognition

Likely: CAR

Unlikely: TOASTER

highway
We will talk about

• **Recognition/Detection**
  Scene categories recognition
  S. Lazebnik et al.
  Given an explicit image of the model
  V. Ferrari et al.

• **Segmentation with Recognition**
  E. Borenstein and S. Ullman
  E. Borenstein, E. Sharon and S. Ullman
  A. Levin and Y. Weiss
Scene and context categorization

- outdoor
- city
- traffic
- ...

(From Iccv 2005 tutorial)
Scene and context categorization

- Buildings
- Urban
- ...
The idea

• Represent images as “bags of features”
• Spatial relations between features

• **Training:**
  1. Get features from labeled images
  2. Histogram the features
  3. Use the histograms to train a classifier

• **Testing**
  1. Use classifier to label new image
Training: Extract features

Label: coast
Extract features

1. Extract features from labeled images
Count the features

• Level 0
Multi-scale bag of features
Lazebnik et al., CVPR06

- Level 0
- Level 1

Multi-scale bag of features

• Level 0
• Level 1
• Level 2

Matching between 2 images

- For each feature type
- For each pyramid level \( L = 0, 1, 2 \)

\[
I^l = \min \left( \begin{array}{cc}
3 & 2 \\
2 & 4 \\
\end{array} \right), \quad \begin{array}{cc}
0 & 4 \\
8 & 1 \\
\end{array} \right) = \begin{array}{cc}
0 & 2 \\
2 & 1 \\
\end{array}
\]

- **Note:** coarser level matches include the finer level ones

Matching between 2 images

\[ \kappa^L(X, Y) = I^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (I^\ell - I^{\ell+1}) \]

**Weight**

- Weight = penalize matches in larger cells.
- The resulting kernel is Mercer kernel.
- Use for the classification

Types of Features

Oriented edge points
8 directions * 2 scales = 16 types

SIFT descriptors

Cluster features to a vocabulary = 200 types

Classification

- By SVM (support vector machine)
  - Train classifier by applying the kernel on labeled images
  - Test image is assigned the label of the classifier with the highest response

Main contribution

Multi-scale representation to the “bag of features” approach

Considers spatial relation between features

Caltech 101: 64.6% using multiscale, L=2
41.2% with L=0
53.9% State-of-the-art (Zhang etal.)

Performance

High performance

Poor performance

Coherent scenes, little clutter

Textureless animals, camouflage

Recognize this particular model: in this input image
Recognition

Why is it hard?

- Clutter
- Occlusions
- Scale
- Viewpoint
- Non-rigid deformations

Matching solely by affine invariant features is not robust
The Idea
Ferrari etal., ECCV 2004

1. Find initial set of invariant features
2. **Expansion** - Look around them to construct more matching features
3. **Contraction** - Leave the correct, and remove mismatches
4. Iteratively construct more and more matches, increasingly farther from the initial ones.

Soft Matching

- Initial **feature extraction** in model and image

1. Choose point by non-max suppression
2. Transfer rays
3. Local extremum along the rays

\[
f(t) = \frac{|I(t) - I_0|}{\max(\int_0^t |I(t) - I_0| \, dt, d)}
\]

4. Calculate moments to get internal ellipse
5. Double ellipse size

Initial Matching-cont.

• Every test region matched to 0-3 model regions by thresholding the:

  – Mahalanobis distance (on color moments)

  – Similarity measure (NCC on gray levels + Euclidean distance in RGB space)

  – Geometric refinement – find affine transf. that maximized the similarity

Early expansion

- Coverage of the model image
- Each feature in model gives support for a part of the covering

Early expansion

1. Map a region from the model to image using the transf. defined by the initial matching.

2. Refine the transformation.

3. Keep the matching with the best similarity (which also above a detection threshold).

4. Discard all matches that did not succeed in propagating any region.

Result of the expansion

Early Contraction

Correct matching = same intersections in test & model

Match \((R_m, R_t)\) is removed if:

\[
\sum_{\{N^i_m\}} \left| \frac{\text{Area}(R_m \cap N^i_m)}{\text{Area}(R_m)} - \frac{\text{Area}(R_t \cap N^i_t)}{\text{Area}(R_t)} \right| > t_s
\]

Main expansion

• Matches from pervious step added to support group.

• Follow similar steps as early expansion.

• Refinement is applied after we picked a new matching region.

Main Contraction
Sidedness Constraint

\[ \text{side}(R_m^1, R_m^2, R_m^3) = \begin{cases} 
-1 & \text{right side} \\
1 & \text{left side} 
\end{cases} \]

\[ \text{err}_{\text{topo}}(R^i) = \frac{1}{v} \sum_{R^j, R^k \in \Gamma \setminus R^i, j > k} \left| \text{side}(R_m^i, R_m^j, R_m^k) - \text{side}(R_t^i, R_t^j, R_t^k) \right| \]

- Test each region \( R^i \) with every two other regions \( R^j, R^k \)
- Correct match will give 0

Main contraction –cont.

The filtering algorithm:

1. (Re-)compute $\text{err}_{\text{tot}}(R^i)$ for all $R^i \in \Gamma$.

   $$\text{err}_{\text{tot}}(R^i) = \text{err}_{\text{topo}}(R^i) + (t_2 - \text{sim}(R^i_m, R^i_t))$$

   \text{sidedness} \quad \text{similarity}

2. Find the worst match $R^w$, with $w = \arg \max_i \text{err}_{\text{tot}}(R^i)$

3. If $\text{err}_{\text{tot}}(R^w) > 0$, remove $R^w$: $\Gamma \leftarrow (\Gamma \setminus R^w)$, and iterate to 1, else stop.

Finally, iterate between contraction & expansion...

Illustration: Soft matches
Illustration: Early expansion

Correct matches

Mismatches

- Soft match
- Early expansion
- Early contract
- 1st main expansion
- 1st main contract
- 2nd expansion
- 2nd contract

10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Illustration: Early contraction

- Soft match
- Early expansion
- Early contraction
- 1st main expansion
- 1st main contraction
- 2nd expansion
- 2nd contraction

Correct matches
Mismatch
Illustration: 1\textsuperscript{st} Main Expansion

Correct matches

Mismatches
Illustration: 1\textsuperscript{st} Main Contraction

- Correct matches
- Mismatches
Illustration: 2\textsuperscript{nd} Main Expansion

Correct matches
Mismatch
Illustration: 2\textsuperscript{nd} Main Contraction

Correct matches
Mismatches
Results
Results
Results
Results
Segmentation & Recognition

• Segmentation $\iff$ recognition

Segmentation helps recognition

Recognized objects constrain segmentation
Bottom – Up segmentation

• **Goal aggregate pixels with similar low level features.**

• **E. Sharon et al.** method: construct a 4-connected graph

Bottom – Up segmentation

• Apply a fast multi level solver called *Algebraic MultiGrid* (AMG), for graph coarsening.

Bottom – Up segmentation

1. Calculate coarse scale node.

2. Create inter scale weights and calculate new features.

3. Update coarse scale weights by averaging the fine scale weights.

4. Repeat the above process. Segments emerges as node at some coarse scale.

Difficulties in bottom-up segmentation
How to constrain the segmentation?

Think on an horse for example:

Shape is an important cue
Introducing shape into the segmentation

• Instead of Bottom up use Top-down

Training images

Learn how to segment this type of images

Result: Binary segmentation
Background / foreground
Randomly collect a large set of candid. fragments

Select subset of informative fragments

Figure-Background Labeling

Detection and segmentation

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Selecting **Informative** Fragments

- Informative = likely to be detected in class compared with non-class images

\[ f_j = \arg \max_f (I(F^s \cup f; C) - I(F^s; C)) \]

- Fragments are added to maximize the gain in mutual information
  - Highly overlapping
  - Well-distributed over the figure

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Segmenting Informative Fragments

Informative fragments | Bottom-Up segmentation

Degree of cover

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Segmenting Informative Fragments

Segmentation?

Overlap with the average Edges $D(x,y)$ of fragment in all the training images.

$$P = \arg \max_{P(\bar{r})} \left( \sum_{(x,y) \in P(\bar{r})} D(x,y) + \lambda \sum_{(x,y) \in \partial P(\bar{r})} D(x,y) \right)$$

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Segmenting a test image

- Each fragment votes for the classification of pixels it covers
  
  \[
  \text{weight} = \frac{\text{detection rate}}{\text{false alarms rate}}
  \]

- Count the number of votes for figure and background

- Check consistency test between the received labeling and labeling of each fragment using NCC

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Top-Down - Borenstein & Ullman.
Main contributions

- Top down approach
- Automatic fragments labeling
Results

This Alg. figure-ground labeling

Manual figure-ground labeling

Bottom-up seg.

E. Borenstein, S. Ullman. Learning to Segment. ECCV (3) 2004: 315-328
Zoom-in to the results

Top-down
The shape of the horse is captured
The boundaries are not exact

Bottom-up
The boundaries are exact
We did not get the horse as one piece

Need to make a combination of Top-Down & Bottom-Up approaches
Bottom up & Top down segmentations

The “correct” segmentation is some how a compromise between Top-down & bottom up.
How to combine?

• **Approach 1** – Do separate segmentations are combine them together.
  

• **Approach 2** – Combine them in one pass.
  
  A. Levin and Y. Weiss (2006)
Combining Top-Down and Bottom-Up Segmentation - Borenstein et al.

- Use the hierarchy formed by the bottom-up segmentation.

- Each segment at each scale gets figure/background label.
Combining Top-Down and Bottom-Up segmentation

• Each father-descendent cost depends on their labels.

\[ T_{\text{Cost}}(s_1, \ldots, s_N) = \sum_{i} f_i(s_i, s_i^-) \]

\( s_i, s_i^- \in \{1, -1\} \)

Combining Top-Down and Bottom-Up: father-descendant cost

- Each father-descendant energy has two terms:

\[ f_i(s_i, s_i^-) = t_i(s_i) + b_i(s_i, s_i^-) \]

- Top-down term: Sum of SSD from the Top-Down segmentation (only on leaves)
- Bottom-up term: Descendent more salient - penalty for different father-descendent labels decreases.

\[ \lambda |S_i|(1 - h_i)(s_i - s_i^-)^2 \]

Combining Top-Down and Bottom-Up Segmentation

• Since the cost function factorizes:

\[ TCost(s_1, \ldots, s_N) = \sum_{i} f_i(s_i, s_i^-) \]

Total Tree cost  father-descendent cost

• Sum product algorithm is used (similar to BP)
Combining Top-Down and Bottom-Up Segmentation

• Requires only one pass in each direction.

• A local computation in each segments results in:

\[
T_{cost_{s_i}}(1) = \min(T_{cost}(s_1, \ldots, s_{i-1}, 1, s_{i+1}, \ldots, s_N))
\]

\[
T_{cost_{s_i}}(-1) = \min(T_{cost}(s_1, \ldots, s_{i-1}, -1, s_{i+1}, \ldots, s_N))
\]

Combining Top-Down and Bottom-Up Segmentation

• Confidence map of the segmentation is given by:

\[
\frac{|m_{S_i}(1) - m_{S_i}(-1)|}{|S|}
\]
Results

Bottom-Up segmentation

Top-Down segmentation

Combined segmentation

Results

Bottom-Up segmentation

Top-Down segmentation

Combined segmentation

Combining Top-Down and Bottom-Up Segmentation- A. Levin et al.

• Combine in one segmentation process.

• Use only few fragments.
Markov Random Field

• Joint distribution of labels and local features

\[ p(x, y) = \frac{1}{Z} \prod_{i,j} \psi_{i,j}(x_i, x_j) \prod_i \phi_i(x_i, y_i) \]
Conditional Random Field

• Conditional distribution of labels given whole image

\[ p(x \mid y) = \frac{1}{Z(y)} \prod_{i,j} \psi_{i,j}(x_i, x_j \mid y) \prod_i \phi_i(x_i \mid y) \]

• Models the **conditional** and not the **joint** probability.

• CRF globally conditioned on the observation \( Y \) – long **range dependences**

Lafferty et al. (2001), Kumar and Hebert (2003)
Log notation

\[ p(x \mid y) = \frac{1}{Z(y)} \prod_{i,j} \psi_{i,j}(x_i, x_j; y) \prod_i \phi_i(x_i; y) \]

Take the log:

\[ \overline{\psi}_{i,j}(x_i, x_j; y) = -\log(\psi_{i,j}(x_i, x_j; y)) \]

\[ \overline{\phi}_i(x_i; y) = -\log(\phi_i(x_i; y)) \]

Easier to work with summations

\[ p(x \mid y) = \frac{1}{Z(y)} e^{-\sum_{i,j} \overline{\psi}_{i,j}(x_i, x_j; y) + \sum_i \overline{\phi}_i(x_i; y)} = E(x; y) \]
How this helps the segmentation?

\[ E(x; I) = \nu \sum_{i,j} w_{ij} |x(i) - x(j)| + \sum_k \lambda_k |x - x_{F_k, I}| \]

- Low-level term (Intensity)
- Local energy term derived from image fragments

Learning

\[ E(x; I) = \nu \sum_{i,j} w_{ij} |x(i) - x(j)| + \sum_{k} \lambda_k |x - x_{F_k}, I| \]

Labeling discontinuities == image discontinuities

\[ w_{ij} = \frac{1}{1 + \sigma d_{ij}^2} \quad \text{where} \quad d_{ij}^2 - \text{RGB distance} \]

We want to learn:

1. \( \nu \) - power of the low level term

2. \( \lambda, \tilde{F} = \lambda_1 + \lambda_2 + ... \)

Given a set of segmented training images

Switch the roll of hidden & observation to get likelihood of the labels $x$ conditioned on the image $I$.

Find the parameters that maximize the sum of the log-likelihood:

$$\ell(\lambda, v; F) = \sum_t \ell^t(\lambda, v; F)$$

Where:

$$\ell^t(\lambda, v; F) = \log p(x_t | I_t; \lambda, v; F) = -E(x_t; I_t, \lambda, v; F) - \log Z(I_t, \lambda, v; F)$$

Learning

• The CRF log likelihood is *convex* with respect to the weighting parameters $\lambda_k, \nu$
  
  Lafferty *et al.* (2001)

• On the other hand - exact computation of derivatives and $Z(I)$ is in general intractable.

• Use approximations

Learning- selection fragments

Out of a pool of fragments at random sizes & locations we want to select a few informative ones.

Straightforward computation of the likelihood improvement is not practical since each iteration will require inference for each $\tilde{\lambda}, \tilde{F}$.

How to do it efficiently?

Learning - selection fragments

Recall the energy term:

\[ E(x; I) = v \sum_{i,j} |x(i) - x(j)| + \sum_{k} \lambda_k |x - x_{F_k,I}| \]

If no fragments are used:

\[ E_0(x; I) = v \sum_{i,j} |x(i) - x(j)| \]

If we add one fragment:

\[ E_1(x; I) = v \sum_{i,j} |x(i) - x(j)| + \lambda_1 |x - x_{F_1,I}| \]

Linear approximation:

\[ E_1(x; I) \big|_{\lambda_1=0} \approx E_0(x; I) + \lambda_1 \frac{\partial E_1(x; I)}{\partial \lambda_1} \]

Learning- selection fragments

Start without any Fragments

\[ E_0(x; I) = v \sum_{i,j} |x(i) - x(j)| \]

If we add one fragment:

\[ E_1(x; I) = v \sum_{i,j} |x(i) - x(j)| + \lambda_1 |x - x_{F_1,I}| \]

\[ E_1(x; I) \big|_{\lambda_1 = 0} \approx E_0(x; I) + \lambda_1 \frac{\partial E_1(x; I)}{\partial \lambda_1} \]

Choose the fragment with large change.
Learning- selection fragments

• Continue in a greedy approach.

• Add a predefined number of fragments.

• After all the fragments are added, improve the accuracy of $\lambda_k$. 

Segmentation

• Given a new image the segmentation is the assignment $x$ that maximizes the probability:

$$p(x \mid I_{\text{new}}) = \frac{1}{Z(I_{\text{new}})} e^{-E(x; I_{\text{new}})}$$

• Where $E(x; I_{\text{new}})$ is the learned energy term.

Results

Fragments used:

One fragment

Two fragments

Three fragments

Results

Original image

Fragments used:

Overlaid segmentation

Summary

• Scene Labeling

• Recognition given explicit model

• Segmentation & recognition
Thanks!