Object detection as supervised classification

Tues Nov 10
Kristen Grauman
UT Austin

Today

- Supervised classification
- Window-based generic object detection
  - basic pipeline
  - boosting classifiers
  - face detection as case study
What kinds of things work best today?

Read license plates, zip codes, checks

Recognizing flat, textured objects (like books, CD covers, posters)

Frontal face detection

Fingerprint recognition

Pasting a URL here...

*By using the demo you agree to our terms of service

Predicted Tags:
- mammal
- livestock
- cattle
- farm
- nobody
- meadow
- grass

Similar Images:

- Images of cows in a field
- Images of cattle in a farm
- Images of livestock in a meadow

Clarifai
Generic category recognition: basic framework

- Build/train object model
  - (Choose a representation)
  - Learn or fit parameters of model / classifier
- Generate candidates in new image
- Score the candidates

Supervised classification

- Given a collection of labeled examples, come up with a function that will predict the labels of new examples.
  - “four”
  - “nine”
  - ?

Training examples Novel input

- How good is some function we come up with to do the classification?
- Depends on
  - Mistakes made
  - Cost associated with the mistakes
Supervised classification

• Given a collection of labeled examples, come up with a function that will predict the labels of new examples.

• Consider the two-class (binary) decision problem
  – \( L(4 \rightarrow 9) \): Loss of classifying a 4 as a 9
  – \( L(9 \rightarrow 4) \): Loss of classifying a 9 as a 4

• **Risk** of a classifier \( s \) is expected loss:

\[
R(s) = \Pr(4 \rightarrow 9 \text{ using } s) L(4 \rightarrow 9) + \Pr(9 \rightarrow 4 \text{ using } s) L(9 \rightarrow 4)
\]

• We want to choose a classifier so as to minimize this total risk

---

If we choose class “four” at boundary, expected loss is:

\[
= P(\text{class is 9} \mid x) L(9 \rightarrow 4) + P(\text{class is 4} \mid x) L(4 \rightarrow 4)
\]

If we choose class “nine” at boundary, expected loss is:

\[
= P(\text{class is 4} \mid x) L(4 \rightarrow 9)
\]
Supervised classification

Optimal classifier will minimize total risk.

At decision boundary, either choice of label yields same expected loss.

So, best decision boundary is at point $x$ where

$$P(\text{class is } 9 \mid x) L(9 \rightarrow 4) = P(\text{class is } 4 \mid x) L(4 \rightarrow 9)$$

To classify a new point, choose class with lowest expected loss; i.e., choose “four” if

$$P(4 \mid x) L(4 \rightarrow 9) > P(9 \mid x) L(9 \rightarrow 4)$$
Probability

Basic probability
- X is a random variable
- $P(X)$ is the probability that X achieves a certain value

\[
P(X)
\]

- called a PDF
  - probability distribution/density function

- $0 \leq P(X) \leq 1$

- $\int_{-\infty}^{\infty} P(X) \, dx = 1$ or $\sum P(X) = 1$

  - continuous X
  - discrete X

- Conditional probability: $P(X | Y)$
  - probability of X given that we already know Y

Example: learning skin colors

- We can represent a class-conditional density using a histogram (a “non-parametric” distribution)

- Percentage of skin pixels in each bin

- Feature $x = \text{Hue}$
Example: learning skin colors

- We can represent a class-conditional density using a histogram (a “non-parametric” distribution)

Now we get a new image, and want to label each pixel as skin or non-skin. What’s the probability we care about to do skin detection?

Bayes rule

\[
P(\text{skin} \mid x) = \frac{P(x \mid \text{skin})P(\text{skin})}{P(x)}
\]

\[
P(\text{skin} \mid x) \propto P(x \mid \text{skin})P(\text{skin})
\]

Where does the prior come from?

Why use a prior?
Example: classifying skin pixels

Now for every pixel in a new image, we can estimate probability that it is generated by skin.

Classify pixels based on these probabilities

- if \( p(\text{skin}|x) > \theta \), classify as skin
- if \( p(\text{skin}|x) < \theta \), classify as not skin

Gary Bradski, 1998
Example: classifying skin pixels

Using skin color-based face detection and pose estimation as a video-based interface

Gary Bradski, 1998

Generative vs. Discriminative Models

- **Generative approach:** separately model class-conditional densities and priors
  
  \[ p(x|C_k), \quad p(C_k) \]

  then evaluate posterior probabilities using Bayes' theorem

  \[
  p(C_k|x) = \frac{p(x|C_k)p(C_k)}{\sum_j p(x|C_j)p(C_j)}
  \]

- **Discriminative approach:** directly model posterior probabilities
  
  \[ p(C_k|x) \]

- In both cases usually work in a feature space

Slide from Christopher M. Bishop, MSR Cambridge
General classification

This same procedure applies in more general circumstances

- More than two classes
- More than one dimension

Example: face detection

- Here, X is an image region
  - dimension = # pixels
  - each face can be thought of as a point in a high dimensional space


Source: Steve Seitz

Today

- Supervised classification
- Window-based generic object detection
  - basic pipeline
  - boosting classifiers
  - face detection as case study
Generic category recognition: basic framework

- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- Generate candidates in new image
- Score the candidates

Window-based models
Building an object model

Given the representation, train a binary classifier
Window-based models
Generating and scoring candidates

Window-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier

Training examples
Feature extraction
Car/non-car Classifier
**Discriminative classifier construction**

- **Nearest neighbor**
  - 10^6 examples
  - Shakhnarovich, Viola, Darrell 2003
  - Berg, Berg, Malik 2005...

- **Neural networks**
  - LeCun, Bottou, Bengio, Haffner 1998
  - Rowley, Baluja, Kanade 1998
  - ...

- **Support Vector Machines**
  - Guyon, Vapnik
  - Heisele, Serre, Poggio, 2001, ...

- **Boosting**
  - Viola, Jones 2001,
  - Torralba et al. 2004,
  - Opelt et al. 2006, ...

- **Conditional Random Fields**
  - McCallum, Freitag, Pereira 2000;
  - Kumar, Hebert 2003
  - ...

---

**Boosting intuition**

- **Weak Classifier 1**

Slide adapted from Antonio Torralba

Slide credit: Paul Viola
Boosting illustration

Weights Increased

Boosting illustration

Weak Classifier 2
Boosting illustration

Weights Increased

Boosting illustration

Weak Classifier 3
Boosting illustration

Final classifier is a combination of weak classifiers

Boosting: training

• Initially, weight each training example equally
• In each boosting round:
  – Find the weak learner that achieves the lowest \textit{weighted} training error
  – Raise weights of training examples misclassified by current weak learner
• Compute final classifier as linear combination of all weak learners (weight of each learner is directly proportional to its accuracy)

• Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme (e.g., AdaBoost)

Slide credit: Lana Lazebnik
Viola-Jones face detector

Main idea:

– Represent local texture with efficiently computable “rectangular” features within window of interest
– Select discriminative features to be weak classifiers
– Use boosted combination of them as final classifier
– Form a cascade of such classifiers, rejecting clear negatives quickly
Viola-Jones detector: features

“Rectangular” filters
Feature output is difference between adjacent regions

Efficiently computable with integral image: any sum can be computed in constant time.

Computing the integral image

Value at \((x, y)\) is sum of pixels above and to the left of \((x, y)\)
Computing the integral image

Cumulative row sum: \( s(x, y) = s(x-1, y) + i(x, y) \)

Integral image: \( ii(x, y) = ii(x, y-1) + s(x, y) \)

MATLAB: \( ii = \text{cumsum}(\text{cumsum}(\text{double}(i)), 2); \)

Computing sum within a rectangle

- Let \( A, B, C, D \) be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:
  \[ \text{sum} = A - B - C + D \]
- Only 3 additions are required for any size of rectangle!
Viola-Jones detector: features

“Rectangular” filters
Feature output is difference between adjacent regions

Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost

Value at (x, y) is sum of pixels above and to the left of (x, y)

Integral image

Considering all possible filter parameters: position, scale, and type:

180,000+ possible features associated with each 24 x 24 window

Which subset of these features should we use to determine if a window has a face?

Use AdaBoost both to select the informative features and to form the classifier
Viola-Jones detector: AdaBoost

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of weighted error.

**Resulting weak classifier:**

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

---

- Given example images $((x_1, y_1), \ldots, (x_n, y_n))$ where $y_i = 0,1$ for negative and positive examples respectively.
- Initialize weights $w_{i, 1} = \frac{1}{m} \cdot \frac{1}{f}$ for $y_i = 0,1$ respectively, where $m$ and $f$ are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
     $$w_{i,t} \leftarrow \frac{w_{i,t-1}}{\sum_{j=1}^{n} w_{j,t}}$$
     so that $w_i$ is a probability distribution.
  2. For each feature, $j$, train a classifier $h_j$ which is restricted to using a single feature. The error is evaluated with respect to $w_i$, $\epsilon_j = \sum_i w_i \cdot |h_j(x_i) - y_i|$.
  3. Choose the classifier, $h_t$, with the lowest error $\epsilon_t$.
  4. Update the weights:
     $$w_{i+1,t} = w_{i,t} \beta_{i}^{-\epsilon_{t}}$$
     where $\epsilon_i = 0$ if example $x_i$ is classified correctly, $\epsilon_i = 1$ otherwise, and $\beta_t = \frac{1}{1 + \epsilon_t}$.

- The final strong classifier is:
  $$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
  where $\alpha_t = \log \frac{1}{\beta_t}$.

---

**AdaBoost Algorithm**

Start with uniform weights on training examples

For $T$ rounds

- Evaluate weighted error for each feature, pick best.
- Re-weight the examples:
  - Incorrectly classified $\rightarrow$ more weight
  - Correctly classified $\rightarrow$ less weight

Final classifier is combination of the weak ones, weighted according to error they had.

Freund & Schapire 1995
Viola-Jones Face Detector: Results

First two features selected

- Even if the filters are fast to compute, each new image has a lot of possible windows to search.
- How to make the detection more efficient?
Cascading classifiers for detection

- Form a *cascade* with low false negative rates early on
- Apply less accurate but faster classifiers first to immediately discard windows that clearly appear to be negative

Training the cascade

- Set target detection and false positive rates for each stage
- Keep adding features to the current stage until its target rates have been met
  - Need to lower AdaBoost threshold to maximize detection (as opposed to minimizing total classification error)
  - Test on a *validation set*
- If the overall false positive rate is not low enough, then add another stage
- Use false positives from current stage as the negative training examples for the next stage
Viola-Jones detector: summary

- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - Integral images for fast feature evaluation
  - Boosting for feature selection
  - Attentional cascade of classifiers for fast rejection of non-face windows


Viola-Jones Face Detector: Results

Detecting profile faces?

*Can we use the same detector?*
Viola-Jones Face Detector: Results

Example using Viola-Jones detector

Frontal faces detected and then tracked, character names inferred with alignment of script and subtitles.

Everingham, M., Sivic, J. and Zisserman, A.
"Hello! My name is... Buffy" - Automatic naming of characters in TV video, BMVC 2006. http://www.robots.ox.ac.uk/~vgg/research/nface/index.html
Google now erases faces, license plates on Map Street View

By Cheryl Wake, ZDNet News.com
Friday, August 24, 2007 01:27 PM

Google has gotten a lot of flack from privacy advocates for photographing faces and license plate numbers and displaying them on the Street View in Google Maps. Originally, the company said only people who identified themselves could ask the company to remove their image.

But Google has quietly changed that policy, partly in response to criticism, and now anyone can alert the company and have an image of a license plate or a recognizable face removed, not just the owner of the face or car, said Marissa Mayer, vice president of search products and user experience at Google.

“It’s a good policy for us and also clarifies the intent of the product,” she said in an interview following her keynote at the Search Engine Strategies conference in San Jose, Calif., Wednesday.

The policy change was made about 10 days after the launch of the product in late May, but was not publicly announced, according to Mayer. The company is removing images only when someone notifies them and not proactively, she said. “It was definitely a big policy change inside.”

Consumer application: iPhoto

http://www.apple.com/ilife/iphoto/

Slide credit: Lana Lazebni
Consumer application: iPhoto 2009

Things iPhoto thinks are faces


Slide credit: Lana Lazebnik
Privacy Gift Shop – CV Dazzle

http://www.wired.com/2015/06/facebook-can-recognize-even-dont-show-face/
Wired, June 15, 2015

Privacy Visor

Boosting: pros and cons

- **Advantages of boosting**
  - Integrates classification with feature selection
  - Complexity of training is linear in the number of training examples
  - Flexibility in the choice of weak learners, boosting scheme
  - Testing is fast
  - Easy to implement

- **Disadvantages**
  - Needs many training examples
  - Other discriminative models may outperform in practice (SVMs, CNNs, ...)
    - especially for many-class problems

Window-based detection: strengths

- Sliding window detection and global appearance descriptors:
  - Simple detection protocol to implement
  - Good feature choices critical
  - Past successes for certain classes
Window-based detection: Limitations

- High computational complexity
  - For example: 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations!
  - If training binary detectors independently, means cost increases linearly with number of classes
- With so many windows, false positive rate better be low

Limitations (continued)

- Not all objects are “box” shaped
Limitations (continued)

- Non-rigid, deformable objects not captured well with representations assuming a fixed 2d structure; or must assume fixed viewpoint
- Objects with less-regular textures not captured well with holistic appearance-based descriptions

Limitations (continued)

- If considering windows in isolation, context is lost

Figure credit: Derek Hoiem
Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions

Summary

- Basic pipeline for window-based detection
  - Model/representation/classifier choice
  - Sliding window and classifier scoring
- Boosting classifiers: general idea
- Viola-Jones face detector
  - Exemplar of basic paradigm
  - Plus key ideas: rectangular features, Adaboost for feature selection, cascade
- Pros and cons of window-based detection