Sharing features: Efficient Booting Procedures for Multi-class Object Detection

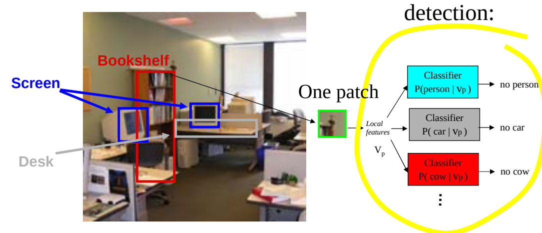
Antonio Torralba, Kevin Murphy and Bill Freeman

(Presented by Xu, Changhai)

Most of the slides are copied from the authors' presentation

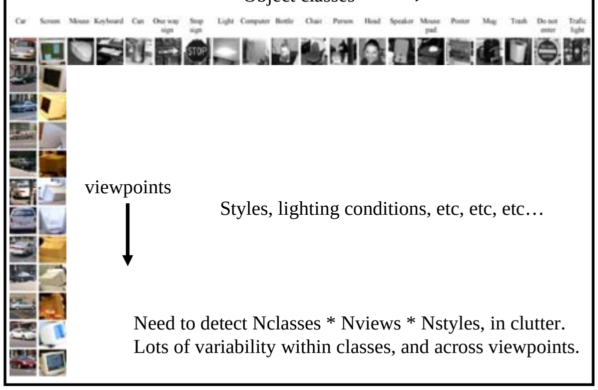
Desired detector outputs:

Multi-class object detection:



Why multi-object detection is a hard problem

Object classes



viewpoints

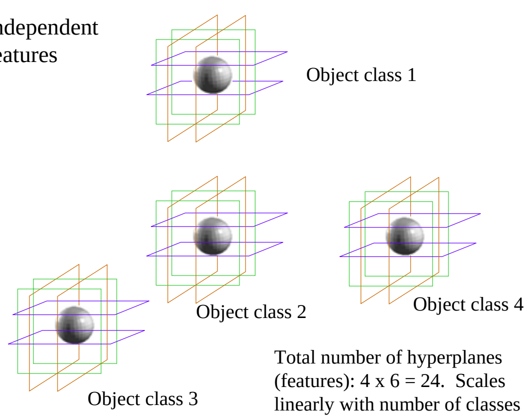
Styles, lighting conditions, etc, etc, etc…

Need to detect Nclasses * Nviews * Nstyles, in clutter.
Lots of variability within classes, and across viewpoints.

The approach

- Share features across objects, automatically selecting the best sharing pattern.
- Benefits of shared features:
  - Efficiency
    - Sharing computations across classes
  - Accuracy
  - Generalization ability
    - Sharing generic knowledge about detecting objects

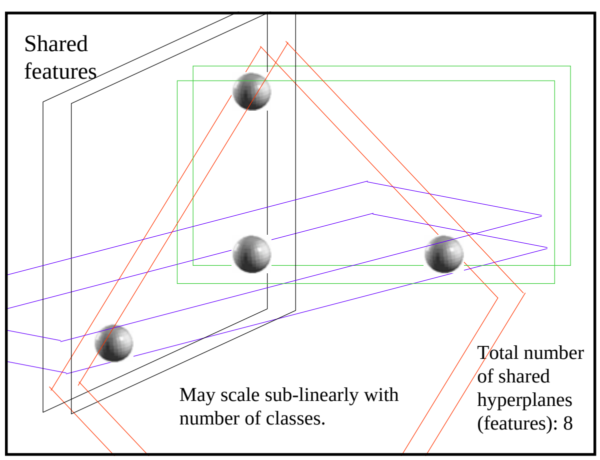
Independent features



Total number of hyperplanes (features): 4 x 6 = 24. Scales linearly with number of classes

---

Shared features



May scale sub-linearly with number of classes.

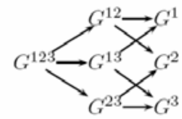
Total number of shared hyperplanes (features): 8

Additive models for classification

$$H(v,c) = \sum_{m=1}^{M} h_m(v,c)$$

+1/-1 classification; feature responses; classes

Feature sharing in additive models



Sharing Graph

$$H(v,1) = G^{1,2,3}(v) + G^{1,2}(v) + G^{1,3}(v) + G^{1}(v)$$

$$H(v,2) = G^{1,2,3}(v) + G^{1,2}(v) + G^{2,3}(v) + G^{2}(v)$$

$$H(v,3) = G^{1,2,3}(v) + G^{1,3}(v) + G^{2,3}(v) + G^{3}(v)$$

Multi-class Boosting

We use the exponential multi-class cost function

classes

$$J = \sum_{c=1}^{C} E\left[e^{-z^c H(v,c)}\right]$$

cost function; membership in class c, +1/-1; classifier output for class c

Weak learners are shared

At each boosting round, we add a perturbation or “weak learner” which is shared across some classes:

$$H(v_i,c) := H(v_i,c) + h_m(v_i,c)$$

Multi-class Boosting

Replacing the expectation with an empirical expectation over the training data, and defining weights $w_i^c = e^{-z_i^c H(v_i,c)}$ for example i and class c, this reduces to minimizing the weighted squared error:

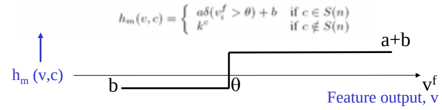
$$J_{wse} = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c (z_i^c - h_m(v_i,c))^2.$$

Weight squared error over training data; weight; squared error

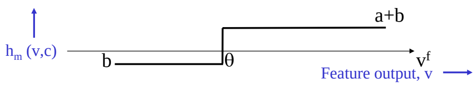
Specialize weak learners to decision stumps

i) Fit shared stump:

$$h_m(v,c) = \begin{cases} a\delta(v_i^f > \theta) + b & \text{if } c \in S(n) \\ k^c & \text{if } c \notin S(n) \end{cases}$$



Find weak learner parameters analytically



Given a sharing pattern, the decision stump parameters are obtained analytically

$$J_{wse} = \sum_{c=1}^{C} \sum_i w_i^c (z_i^c - h_m(v_i, c))^2 \longrightarrow \begin{cases} b = \dfrac{\sum_{c \in S(n)} \sum_i w_i^c z_i^c \delta(v_i^f \leq \theta)}{\sum_{c \in S(n)} \sum_i w_i^c \delta(v_i^f \leq \theta)}, \\ a + b = \dfrac{\sum_{c \in S(n)} \sum_i w_i^c z_i^c \delta(v_i^f > \theta)}{\sum_{c \in S(n)} \sum_i w_i^c \delta(v_i^f > \theta)}, \\ k^c = \dfrac{\sum_i w_i^c z_i^c}{\sum_i w_i^c} \quad c \notin S(n) \end{cases}$$

Joint Boosting: select sharing pattern and weak learner to minimize cost.

Conceptually,

for all features:
    for all class sharing patterns:
        find the optimal decision stump, $h_m(v,c)$
    end
end

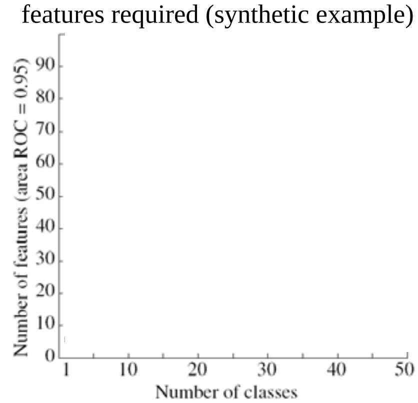
select the $h_m(v,c)$ and sharing pattern that minimizes the weighted squared error J_{wse} for this boosting round.

Approximate best sharing

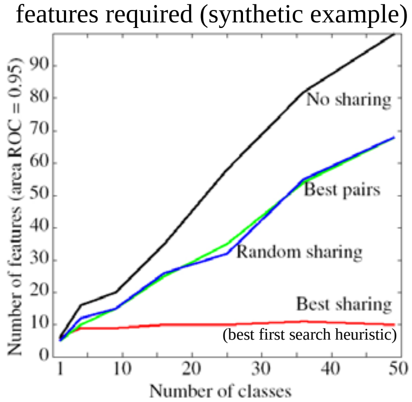
To avoid exploring all $2^C - 1$ possible sharing patterns, use best-first search:

S = []
% Grow a list of candidate sharing patterns, S.
while length S < N_c
    for each object class, c_i, not in S
        % consider adding c_i to the list of shared classes, S
        for all features, h_m
            evaluate the cost J of h_m shared over [S, c_i]
        end
    end
    S = [S, c_{min_cost}]
end
Pick the sharing pattern S and feature h_m which gave the minimum multi-class cost J.

Effect of pattern of feature sharing on number of features required (synthetic example)



Effect of pattern of feature sharing on number of features required (synthetic example)

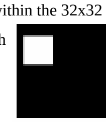


Now, apply this to images.
Image features (weak learners)

32x32 training image of an object

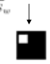
12x12 patch

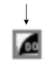
Location of that patch within the 32x32 object



Feature output

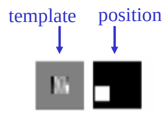
$$v^f(x, \sigma) = \left(\sum_{x \in S_w} w_f(x) |I \otimes g_f|^p \right)^{1/p}$$



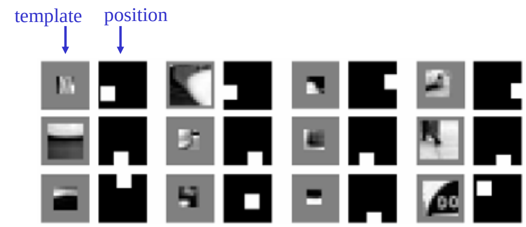


$g_f(x)$
Mean = 0
Energy = 1
$w_f(x)$
Binary mask

The candidate features



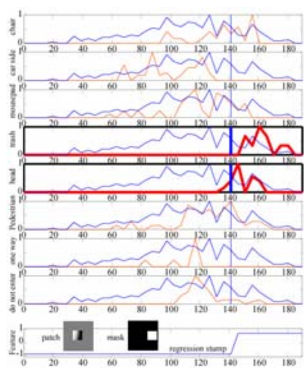
The candidate features



Dictionary of 2000 candidate patches and position masks, randomly sampled from the training images

Example shared feature (weak classifier)

Response histograms for background (blue) and class members (red)



At each round of running joint boosting on training set we get a feature and a sharing pattern.

How the features were shared across objects

(features sorted left-to-right from generic to specific)

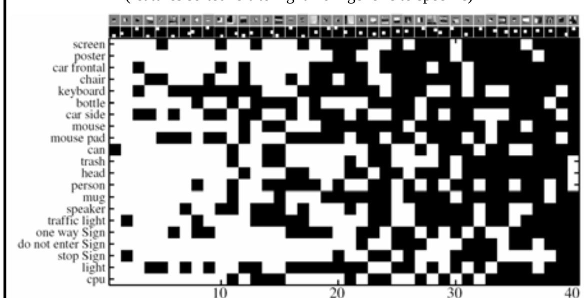
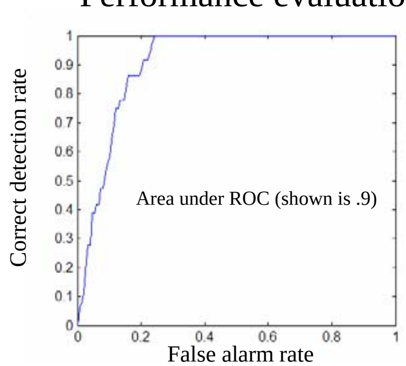
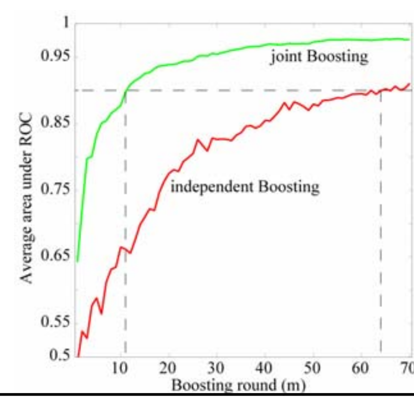


Fig. 13. Matrix that relates features to classifiers, which shows which features are shared among the different object classes. The features are sorted from left to right from more generic (shared across many objects) to more specific. Each feature is defined by one filter, one spatial mask and the parameters of the regression stump (not shown). These features were chosen from a pool of 2000 features in the first 40 rounds of boosting.

Performance evaluation

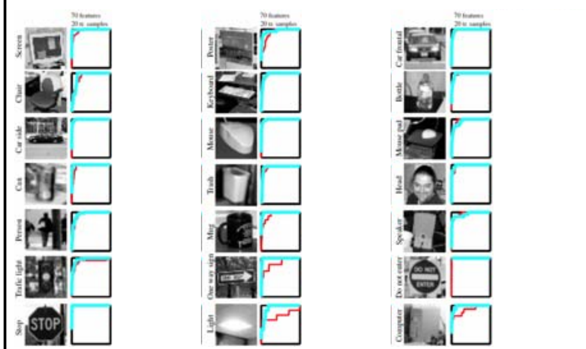


Performance improvement over training



Significant benefit to sharing features using joint boosting.

70 features, 20 training examples (left)

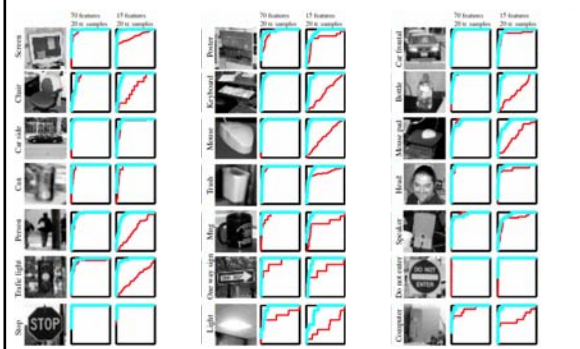


Shared features ——— Non-shared features ———

---

70 features, 20 training examples (left)

15 features, 20 training examples (mid)



Shared features ——— Non-shared features ———

---

70 features, 20 training examples (left)

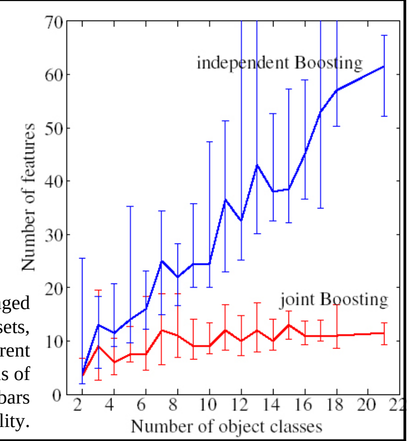
15 features, 20 training examples (middle)

15 features, 2 training examples (right)

Shared features ——— Non-shared features ———

Scaling

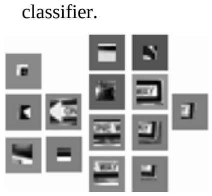
Joint Boosting shows sub-linear scaling of features with objects (for area under ROC = 0.9).



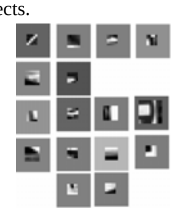
Results averaged over 8 training sets, and different combinations of objects. Error bars show variability.

Generic vs. specific features

Parts derived from training a binary classifier.



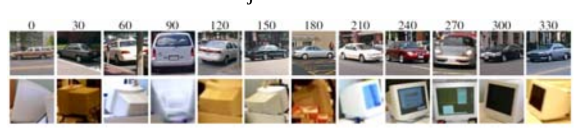
Parts derived from training a joint classifier with 20 more objects.



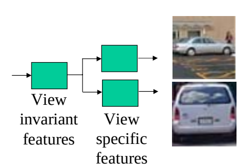
In both cases ~100% detection rate with 0 false alarms

Multi-view object detection

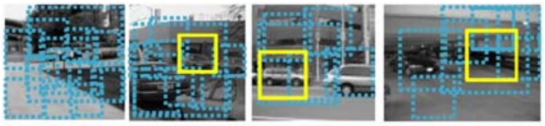
train for object and orientation



Sharing features is a natural approach to view-invariant object detection.



Multi-view object detection



a) Multiview car detection with independent boosting for each view.

b) Multiview car detection with joint boosting.

Fig. 18. View invariant car detection (dashed boxes are false alarms, and solid boxes are correct detections). The figure shows a comparison of car detections with a battery of binary classifiers for each view trained individually (a), and jointly (b). The joint training provides more robust classifiers with the same complexity. In both cases, the classifiers were trained using 20 samples per view (12 views), and use 70 stumps in total. Both classifiers are set in order to provide 80% detection rate. The independent training of each view provides poor results with over 8 false alarms per image. When training the classifiers using joint boosting, the detector has 1 false alarm per image on average. Images are about 128x128 and contain more than 17000 patches to be classified.

Summary

- Feature sharing essential for scaling up object detection to many objects and viewpoints.
- Joint boosting generalizes boosting.
- The shared features
  - generalize better,
  - allow learning from fewer examples,
  - with fewer features.
- A novel class will lead to re-training of previous classes