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

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

Total number of shared hyperplanes (features): 8

May scale sub-linearly with number of classes.
### Multi-class Boosting

We use the exponential multi-class cost function

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

where \( J \) is the cost function, \( z^c \) is the membership in class \( c \), \( H(v, c) \) is the 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:

\[ H(v_t, c) := H(v_t, c) + h_m(v_t, c) \]

### Specialize weak learners to decision stumps

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

\[ J_{\text{wce}} = \sum_{c=1}^{C} \sum_{i=1}^{N} w^c_i (z^c_i - h_m(v_i, c))^2. \]

### Feature sharing in additive models

Additive models for classification

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

+1/-1 classification classes feature responses

Multi-class Boosting

Feature output for class \( c \)

Weak learners are shared

Specialize weak learners to decision stumps
Find weak learner parameters analytically

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

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 \( J_{mse} \) for this boosting round.

Approximate best sharing

To avoid exploring all \( 2^c - 1 \) possible sharing patterns, use best-first search:

\[
S = [] \%
\text{Grow a list of candidate sharing patterns, } S.
\]
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
  \( S = [S, c_{\min \text{cost}}]\)
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)

Now, apply this to images.

Image features (weak learners)

\[
v_f(x) = \left( \sum_{l \in S} w_l(x) (g_l(x))^\alpha \right)^{1/\alpha} \%
\text{Mean} = 0
\]
\[
\text{Energy} = 1
\]
\[
\text{Binary mask}
\]
The candidate features

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

Example shared feature (weak classifier)

How the features were shared across objects
(features sorted left-to-right from generic to specific)

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

Performance evaluation

Area under ROC (shown is .9)

Performance improvement over training

Significant benefit to sharing features using joint boosting.
70 features, 20 training examples (left)

15 features, 20 training examples (middle)

15 features, 2 training examples (right)

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

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