Sharing visual features for multiclass and multiview object detection

Antonio Torralba, Kevin P. Murphy, William T. Freeman
In the previous papers we learned how to detect and classify images with 1 class.

Our Recognition Problem is to recognize many object category with few images per category.

How to detect many categories?
Find each category

[Images of bottles, people, cars, and a street scene]
Why not?

- Require a lot of training data since each classifier require many different images.

Training data → Causes → Increase computational complexity and sample complexity
Find common features that distinguish a subset of classes against the rest.
Template-like features
100% accuracy for a single object
But too specific.

Wavelet-like features,
weaker discriminative power
but shared in many classes.
| screen | poster | car frontal | chair | keyboard | bottle | car side | mouse | mouse pad | can | trashcan | head | person | mug | speaker | traffic light | one way Sign | do not enter | stop Sign | light | cpu |
|--------|--------|-------------|-------|----------|--------|----------|-------|-----------|-----|----------|------|--------|-----|---------|---------------|-------------|-------------|-----------|-------|------|-----|

**Left part:** common features which are being shared in most classes

**Right part:** specific features which are being shared by only a few classes
What we gain from sharing features?

- Essentially more positive samples
- Reuse the data
Example of the strong classifier

- It is a sequential procedure:

Each data point has a class label:

\[ y_i = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases} \]

and a weight:

\[ w_t = 1 \]
Weak learners from the family of lines

Each data point has a class label:

\[ y_t = \begin{cases} 
+1 \ (\bigcirc) \\ 
-1 \ (\bigotimes) 
\end{cases} \]

and a weight:

\[ w_t = 1 \]

\[ h \Rightarrow p(\text{error}) = 0.5 \text{ it is at chance} \]
This one seems to be the best

Each data point has a class label:

\[ y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases} \]

and a weight:

\[ w_t = 1 \]

This is a ‘weak classifier’: It performs slightly better than chance.
Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]

We set a new problem for which the previous weak classifier performs at chance again.
We set a new problem for which the previous weak classifier performs at chance again.
We set a new problem for which the previous weak classifier performs at chance again...

Each data point has a class label:

\[ y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases} \]

We update the weights:

\[ w_t \leftarrow w_t \exp\{-y_t H_t\} \]
The strong (non-linear) classifier is built as the combination of all the weak (linear) classifiers.
We can minimize a similar function using one-vs-all strategy

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

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

This doesn’t work very well, since it is separable in \( c \).

Put constraints. -> **shared features!**
Join Boost

- In each iteration, choose
  - One common feature
  - A subset of classes that use this feature
- So that the objective decreases most
# Iteration

<table>
<thead>
<tr>
<th>Class</th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Features</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

Red squares indicate active features during iterations.
Each class may have its own favorite feature. A common feature may not be any of them, however it **simultaneously** decreases errors of many classes.
Choose the best subset is prohibitive

Use greedy approach

Choose one class and one feature so that the objective decreases the most

Iteratively add more classes until the objective increases again

Note the common feature may change
Joint boosting Vs. Independent
Total number of features for all the classes

- **Shared features**
- **Class-specific features**

Number of features for a single class

**Graphs:**
- **Left graph:** Shows the total number of features for all classes as a function of the number of object classes.
- **Right graph:** Shows the number of features for a single class as a function of the number of object classes.
\#FEATURES = O(\log \#CLASS)
Dictionary

- 2000 random sampled patches
  - Of size from 4x4 to 14x14
  - no clustering
- Each patch is associated with a spatial mask
Features

- Building feature vectors
  - Normalized correlation with each patch to get response
  - Raise the response to some power
    - Large value gets even larger and dominate the response (max operation)
  - Use spatial mask to align the response to the object center (voting)
  - Extract response vector at object center
The Data

- Multiclass object recognition
  - Dataset: LabelMe
  - 21 objects, 50 samples per object
  - 500 rounds

- Multiview car recognition
  - Train on LabelMe, test on PASCAL
  - 12 views, 50 samples per view
  - 300 rounds
PASCAL Cars 2 RPC Curves

12 views
50 samples per class
300 features

France Telecom: pascal_develtest
specific-1-800
TSI-pLSA (Fergus)
shared-1-100
Constellation (Fergus)
specific-50-300
shared-50-300
Darmstadt: ISMbig4
Darmstadt: ISMSVMbig4
shared-50-300
INRIA: dalal: ndalal_competition_number_6

Precision vs Recall

0 0.1 0.2 0.3 0.4 0.5
0 0.1 0.2 0.3 0.4 0.5
Conclusion

- Joint Boosting indeed works
  - Especially when the number of images per class is not too small (otherwise NN)
- Better performance in the presence of
  - Many classes, each class has only a new samples
  - Introduce regularization that reduce over fitting
- Disadvantages
  - Train slowly, \(O(C^2)\).
The End