Modification of the AdaBoost-based Detector for Partially Occluded Faces

Jie Chen, Shiguang Shan, Shengye Yan, Xilin Chen, Wen Gao
Harbin Institute of Technology, Harbin, China
ICT-ISVISION FRJDL, Beijing, China
{chenjie, sgshan, syyan, xlchen, wgao}@jdl.ac.cn
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Motivation

- Most state-of-the-art face detection systems degrade **rapidly** when faces are partially **occluded** by other objects.
  - Are there any faces in the following images?
  - If they have, How can we locate them?
Motivation

- One of the challenges associated with face detection is occlusion.
- Occlusions include:
  - occluded by other objects (accessories or other faces),
  - self-occlusion (a profile face),
  - local highlight or underexposure
  - poor image quality (blurred).
Contribution

- Detect *partially occluded* faces;
  - Reasonably modifying the AdaBoost-based face detector.
Background

- **AdaBoost & Haar Feature**
  - Viola and Jones, *CVPR* 2001
  - Some example rectangle features
  - The feature value is the **difference** between the **sum** of the pixels within the **white** rectangles and the sum of pixels in the **grey** rectangles;
  - The final classifier is a cascade model.

- **Haar feature & Weak classifier (same)**
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Problem

- Occluded faces
- Weak classifiers are disabled;
Solution

- Sample division and weak classifier mapping

(a) A typical Haar-like feature and a face sample  
(b) Divide this sample into patches without overlapping  
(c) Map a weak classifier to patches
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Previous work

- Leung, Burl and Perona, ICCV1995
  - Using random labeled graph matching to couple a set of local feature detectors
- Loutas, Pitas, and Nikou, CSVT2004
  - Probabilistic Multiple Face Detection and Tracking Using Entropy Measures
- Hotta, ICIP2004
  - SVM with local kernels
- Lin, Liu and Fuh, ECCV 2004
  - motivated by the work of Viola and Jones,
  - a robust boosting scheme
  - reinforcement training
  - cascading with evidence
- Ichikawa, Mita and Hori, FG2006
  - Component-based using AdaBoost and decision tree
    - The whole face classifier scans over an input image
    - The classifier for right eye scans over the input image, etc
    - The results of whole face and individual parts classifiers are combined
- Winn and Shotton, CVPR2006
  - The Layout Consistent Random Random Field
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Flow chart

1. Train an AdaBoost based classifier
2. Combine all the weak classifiers
3. Map each weak classifier to patches
4. Construct a face detector for occlusions
5. Determine the threshold for a face candidate
6. Determine the threshold for each patch

[Diagram showing a flow chart with the steps described above]
Train and Combine

- Train an AdaBoost based face detector

- Combine all of the weak classifiers

1st layer

2nd layer

3rd layer

... ...

20th layer
Sample division and weak classifier mapping

- Evenly divide the window into patches
  
  \[ S = \{ patch_0, patch_1, \ldots, patch_{mn-1} \} \]

- Find the patch \( patch_{max} \) with the largest overlapped area

- For \( patch_i \),
  - If the overlapped area between the patch and a feature is no less than \( \xi \times patch_{max} \),
  - then the feature is high correlative to the patch,
  - otherwise, it is weak correlative.
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

\[ \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i\text{)}}{\text{area(patch}_i\text{)}} \]
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

- \[ \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i)}{\text{area(patch}_i \text{)}} \]
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

- \( \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i\text{)}}{\text{area(patch}_i\text{)}} \)

- \( \text{patch}_{\text{max}} \) is 0.67
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

  \[
  \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i)}{\text{area(patch}_i)}
  \]

- \text{patch}_{max} is 0.67
- \( \zeta \times \text{patch}_{max} = 0.8 \times 0.67 = 0.54 \)
An example

- The overlapped area *ratios* between the rectangle feature and 16 patches are shown as left:

- \[ \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i)}{\text{area(patch}_i)} \]

- \( \text{patch}_{\text{max}} \) is 0.67
- \( \xi \times \text{patch}_{\text{max}} = 0.8 \times 0.67 = 0.54 \)

- The high correlative patches for this feature is the four 5, 6, 9, 10. i.e. the given feature is mapped to these four patches.
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

- \[\text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i)}{\text{area(patch}_i)}\]

- \(\text{patch}_{\text{max}}\) is 0.67
- \(\xi \times \text{patch}_{\text{max}} = 0.8 \times 0.67 = 0.54\)

- The high correlative patches for this feature is the four 5, 6, 9, 10. i.e. the given feature is mapped to these four patches.
An example

- The overlapped area ratios between the rectangle feature and 16 patches are shown as left:

- \( \text{ratio}_i = \frac{\text{area(overlapped between the feature and patch}_i\text{)}}{\text{area(patch}_i\text{)}} \)

- \( \text{patch}_{\text{max}} \) is 0.67

- \( \xi \times \text{patch}_{\text{max}} = 0.8 \times 0.67 = 0.54 \)

- The high correlative patches for this feature are the four 5, 6, 9, 10. i.e. the given feature is mapped to these four patches.
Threshold for each patch

- A histogram example for patch 5 to determine the threshold $\theta_5^p$ according to Minimum Error Rate Criterion:
  - $x$-coordinate denotes the outputs of the all weak classifiers associated to Patch 5;
  - $y$-coordinate denotes the number of the activated features for each bin.
Threshold for a face candidate

- $\theta_{fc}$ is the threshold for a face candidate.

- Determined during the experiments to adjust the detection rates and false alarms.
Construct the final detector

- Train an AdaBoost based classifier
- Combine all the weak classifiers
- Map each weak classifier to patches
- Determine the threshold for a face candidate
- Determine the threshold for each patch
- Construct a face detector for occlusions
Classification of an input window

- **Input:** An sub-window from an image
- **Output:** The label of the window

  - **Step 1:** Divide the input into patches
  - **Step 2:** Compute the outputs for all weak classifiers associated to each patch;
  - **Step 3:** Count the high relative weak classifiers for each patch
  - **Step 4:** Determine whether each patch is valid
  - **Step 5:** Label the input window

\[ \Theta_{fc} = 10 \]

11 > 10

Label = 1
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Experiments

- **training set**
  - 15,000 faces
  - 15,000 non-faces
  - bootstrapped from 16,536 images containing no faces
Experiments

- **testing set**
  - A collected test set from web
    - 174 images showing 328 faces.

- MIT+CMU frontal face test set
  - 130 images showing 507 upright faces.
Experimental results comparison

The collected test set from web

Comparison of Different Detectors

MIT+CMU frontal face test set

RCC Curves on Comparison with Others

Lin, Liu, and Fuh, ECCV 2004
For occlusion
Outline

- Motivation
- Problem
- Previous work
- Method
- Experiment
- Summary
Summary

• Conclusion:
  – Present a solution to detect partially occluded faces by reasonably modifying the AdaBoost-based face detector.

  • Reasonable to divide the whole face region into multiple patches and then map those weak classifiers to the patches.

  • implement easily and work well.

  – The only inheritance from the trained cascade by AdaBoost is the weak classifiers and their corresponding thresholds.

• Future work:
  • Speed
  – Cascade
Some outputs of our detector
Thank you very much!