Final Project: A Face Detector

Brendan Miller

March 12 2012

Abstract

A face detector implemented via a simplified Viola Jones algorithm.

1 Project Overview

For this project I implemented an binary image classifier that detects faces. This is based on the work of Paul Viola and Michael Jones[1].

I extracted features from images by first generating what Viola and Jones termed an integrated image. From the integrated image, I rapidly computer a large number of real valued features, described more fully in section 2.

Given a large set of real valued features, I constructed a weak learner by selecting a single feature and a threshhold value that partitioned example images into guessed face and non-face classes. This weak learner is conceptually similar to a decision tree stump, although the metric used to find the optimal feature and threshold is different (see section 4).

I combined weak learners using boosting (see section 5) in order to increase accuracy.

Finally, in the original Viola Jones paper, boosted classifiers were combined in a cascade. The cascade was designed to reduce the incidence of false positives, and to increase the performance of the final classifer and possibly of the training phase. However, as stated in my original project proposal, I have decided to descope the cascade from this project in order to fit it into the time allotted.

Even without the cascade, I've managed to achieve very high accuracy, upwards of 99%[1], using a boosted classifier over the features described in the original Viola Jones paper.

2 Feature Extraction

The underlying features used to train the weak classifier are illustrated in figure 1. These rectangular features are computed by summing values of the pixels under the greyed out section, and subtracting the sum of the pixels under the white section to produce a real valued feature. For each feature type, every possible scale and translation of the feature over the image is computed.

These features and their rapid computation is probably the most important innovation in Viola Jones. As discussed in the Viola Jones paper[1], other detectors provide similar levels of accuracy, but Viola Jones provides greatly improved performance and is suitable for real time processing of image data.

[1]Note this is on a validation set selected from the images provided with this project.

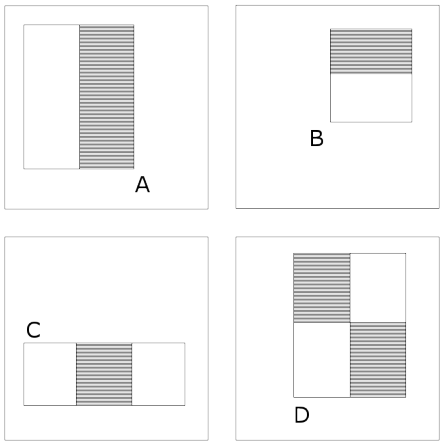

Figure 1: Image features used to train the weak learner. Image from wikipedia.

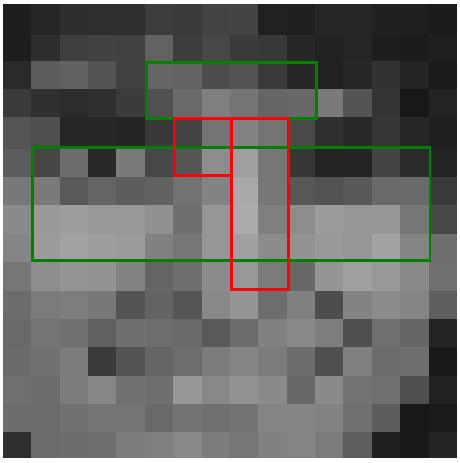

Figure 2: Plot of 4 most predictive features on an image. Red indicates feature A, Green indicates feature B.

To rapidly classify images, and also to provide more reasonable training speeds, the features discussed in this paper can be quickly computed by first creating an integral image. The integral image is of the same dimension of the source image, but every element of the integral image is the sum of all pixels above and to the left of that coordinate in the source image.

From this integral image, the sum of a rectangular area can be computer with only 4 array references, one to each corner of the rectangle. A little algebra will similarly allow features A, B, C, and D to be computed with a small number of references.

Even using the integrated image the performance of feature computation can be a drag on the algorithm. When training on large datasets, millions of features need to be computed. It is extremely important that feature computation is fast. For this reason, after finding my initial pure python implementation to be too slow, I rewrote the feature extraction code in C++. This optimization made training about 5 times faster.

Another optimization that benefits the speed of training, but not of classification, is to precompute all features and store them in main memory. Originally I implemented my training stage this way, but found that on large datasets the number of precomputed features was too large to fit in main memory on a 32 bit machine.

For a 16 by 16 image, in my implementation there are 26944 features. Given 6000 training examples and 4 bytes per feature, roughly 154 gigabytes of memory are necessary to precompute all the features. Currently this is not practical.

3 Feature Examples

The most predictive features learned by the weak learner (see figure 2) seem to focus around the eyes, which makes sense as the eyes are probably the most uniform feature from face to face. Comparatively, the mouth changes more, and may be covered by a beard.

4 Weak Learner

The weak learner used in the Viola Jones algorithm is very similar to a stump decision tree classifier. The primary difference is that while a stump classifier chooses a partition that maximizes weighted information gain, the weak learner minimizes weighted misclassification error.

$$w_i = \text{weight of example } i$$
$$h(x) = \text{weak learner classifier}$$
$$\sum_i w_i |h(x_i) - y_i|$$

This metric is important for performance reasons.

If for each feature we consider all thresholds mid way between the unique values of the feature in sorted order, then for K features and N examples there are $K(N-1)$ feature threshold combinations.

For each feature assuming the examples are already sorted, the threshold that minimizes weighted misclassification can be found in linear time with respect to N using a simple algorithm described in Robust Real Time Face Detection[2]. Finding the threshold for each of K features with minimal misclassification is thus an $O(NK)$ operation[2]

Comparatively, calculating the information gain of an individual partition is a linear time operation with respect to the number of examples N. Since that operation needs to be performed $N-1$ times for each of the K features, training a weak classifier using information gain as a metric would take $O(KN^2)$ time.

5 Boosted Classifier

The weak learners are combined in a fairly standard boosting algorithm as described by Schapire[3]. Since we covered boosting in detail in the last assignment and the interesting aspects of the Viola Jones technique are in the weak learner and the cascade, I will omit a detailed discussion of boosting.

Taking M rounds of boosting into account and the total complexity of the training phase is $O(NKM)$.

6 Experimental Results

My training data[4] is provided with my project and consists of 3000 images of faces and 10000 images of non-faces. All are 16 X 16 grayscale images.

Using 50 rounds of boosting and 5000 randomly selected training images, I produced a classifier[3] that accurately classified faces over a held out validation set of 1000 images with 0.9% error. It produced 5 false positives and 4 false negatives.

This classifier is relatively rapid. In one test once the integrated images had been generated, classification of 6000 images took place in 1180 milliseconds using the C++ backend[4]. Thus it takes about 0.2 milliseconds to classify an individual image.

[2]My current implementation technically runs in $O(KN\log(N))$ as I am sorting the examples over again for each round of boosting; however, profiling indicates that in practice this is not that big of a problem.

[3]This classifier is serialized as JSON in the file large_classifier.json.

[4]By comparison the python backend took 9615 milliseconds.

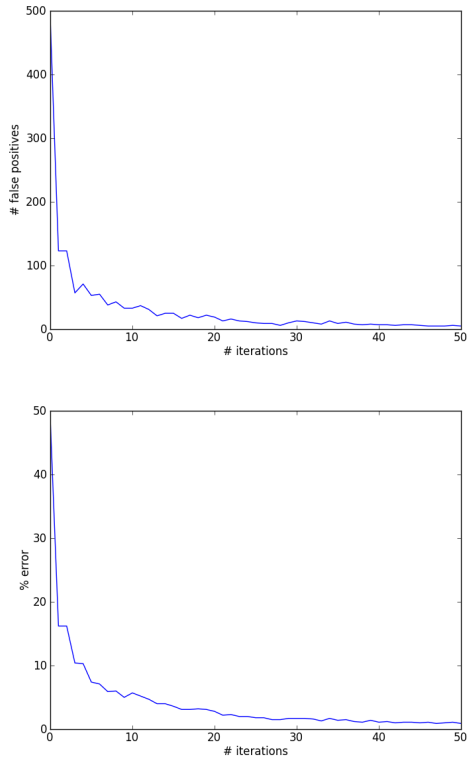

Figure 3: Percent error over 1000 held out validation images given rounds of boosting. Classifier trained from 5000 images.

Figure 4: False positives within a larger image.

Thus even without the cascade described in the Viola Jones paper, it's possible to build a very good face classifier of whole images. However, in order to detect faces within a larger image, it's necessary to run the classifier many thousands of times at different locations and scales. In this scenerio, 4 false positives out of 1000 is actually too high. See figure 4.

Adding the cascade in the original Viola Jones paper would probably resolve this issue, but that feature was scoped out of this project at proposal time due to the time constraints of the project.

7 Future Directions

I'd like to experiment with measures to further reduce false positives. The obvious choice would be to add in the cascade. However, my suspicion is that other perhaps simpler mechanisms could reduce false positives. My suspicion is that the Viola Jones cascade in practice mainly serves as an elaborate biasing mechanism, encoding the knowledge that non-faces are far more common than faces into the classifier.

I think it should be possible to either introduce a simpler biasing mechanism into adaboost, or to encode the knowledge into the data by adjusting the ratio of faces to non-faces in the training

data.

Another future direction would be to speed up the training process by either introducing multiple threads, or spreading the work across a map reduce cluster.

References

[1] Paul Viola, Michael Jones
Rapid Object Detection using a Boosted Cascade of Simple Features
http://research.microsoft.com/en-us/um/people/viola/Pubs/Detect/violaJones_CVPR2001.pdf

[2] Paul Viola, Michael Jones
Robust Real Time Face Detection
http://www.vision.caltech.edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf

[3] Robert E. Schapire
The Boosting Approach to Machine Learning: An Overview
http://www.cs.princeton.edu/~schapire/uncompress-papers.cgi/msri.ps

[4] Training data used for my project
http://www.stat.ucla.edu/~yuille/courses/Stat231-Fall08/