2018-2019
Coursework submission
Cover sheet

Submit to Graduate Education
Date of submission

Student name…………………………………………

College…………………………………………………

CRSID…………………………………………………..

Module name

Module Code No.

Assignment No. **Assignment mark/grade**

Assessor comments

GE Records……………..

GE stamp date and initials

CRSID…………………………………………

Module No………………………………………….

Assignment No…………………………………….

Research Report: Rapid Object Detection using a Boosted Cascade of Simple Features

Zhuoni Jie
Dept. of Computer Science and Technology
Wolfson College
University of Cambridge
zj245@cam.ac.uk

***Abstract*—Face detection has been one of the most studied and important topics in computer vision field. Detecting faces is a crucial step in many identification applications. For example, most face recognition algorithms assume that the face location is known, and face tracking algorithms often assume the initial face location is known. Face detection is a very challenging task for computers, due to variations in scale, location, view point, illumination, occlusions, etc. Viola and Jones face detection algorithm stands out as a very important face detection method based on AdaBoost, which is capable of processing images extremely rapidly and achieving high detection rates. The core idea of Viola-Jones algorithm is to select and combine several weak learners in order to find a strong learner, which will be a linear combination of the weak learners. This project is going to study and understand the Viola-Jones algorithm by analysing the original paper, implementing the whole detection framework, conducting experiment, comparing with other approaches and hopefully further improving the performance.**

Word Count: 2350

I. Introduction

This research report is a study of the paper, ”Rapid Object Detection using a Boosted Cascade of Simple Features”, published by Paul Viola and Michael Jones in CVPR 2001 [1]. This famous paper describes a new machine learning approach based on AdaBoost (Adaptive Boosting), and presents an application of it in face detection. This robust and extremely rapid object detection algorithm is known as Viola-Jones detection algorithm.

There are three main contributions of this paper:

- Introduces a new image representation called integral image that allows for very fast feature evaluation.
- Introduces an effective learning algorithm by selecting a small number of important features using AdaBoost.
- Proposes a method combining classifiers in a cascade structure which dramatically increases the detection speed.

The remainder of this report contains four parts. It first discusses key details of concepts, technical details and implementation of the Viola-Jones algorithm. Then, related work and other detection approaches are introduced, with discussion of limitations and comparisons of these approaches. Third, experimental implementation and evaluation of the Viola-Jones with modifying parameters are described and analysed. Another two approaches (Histogram of Gradient Points, HOG and Local Binary Patterns, LBP) are implemented on our experimental dataset for comparison and discussion. Finally, conclusion about this research study and future work are summarised.

II. Technical Content

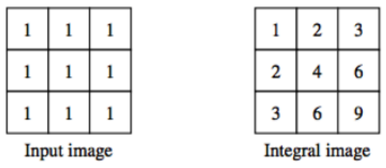
The Viola-Jones detection algorithm uses classification methods based on the value of simple rectangle features, implements AdaBoost both to select a small set of features and train the classifier, and constructs a cascade of classifiers which achieve increased detection performance while radically reducing computation time. In the domain of face detection, the proposed system yields results comparable to the best previous systems, and can be used in real-time applications.

A. Integral Image and Feature Extraction

The Viola-Jones detection uses rectangle features that can be computed very rapidly using and intermediate representation for the image called the *integral image*. Instead of using pixels directly, this method using features operates faster and easily encodes ad-hoc domain knowledge. The integral image at location x, y contains the sum of the pixels above and to the left of x, y, inclusive:

$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y') \tag{1}$$

Here $ii(x, y)$ is the integral image and $i(x, y)$ is the original image.



Input image

1	1	1
1	1	1
1	1	1

Integral image

1	2	3
2	4	6
3	6	9

Fig. 1. Example of an integral image.

The sum of pixels in rectangle $ABCD$ can be calculated from integral image:

The features calculated are three kinds of Haar-like features (two-rectangle features, three-rectangle feature and four-rectangle feature), as shown in Fig. 3.

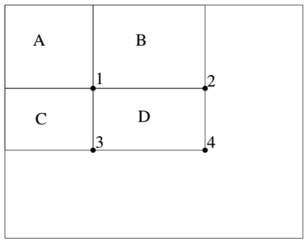


Fig. 2. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is $A + B$, at location 3 is $A + C$, and at location 4 is $A + B + C + D$. The sum of the pixels within rectangle D can be computed as $4 + 1 - (2 + 3)$.

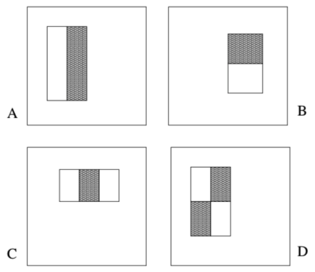


Fig. 3. Example rectangle features shown relative to the enclosing detection window. (A) and (B): two-rectangle features. (C): a three-rectangle feature. (D): a four-rectangle feature.

Haar-like feature value is calculated by the pixel sum of the darker rectangle minus the pixel sum of the lighter rectangle, each Haar-like feature can then be computed as the sum of 6 values from I:

$$\sum_{(x,y)\in ABCD} i(x,y) = ii(D) + ii(A) - ii(B) - ii(C) \quad (2)$$

The base resolution of a sub-window in our implementation is 24 by 24 pixels. Therefore, the total number of features I can extract from one image input is:

$$(24 + 22 + \ldots + 2) * (24 + 23 + \ldots + 1) * 4 = 43200 \quad (3)$$

By setting both the stride and increment to 2, I got 1665 features for analysis input.

B. AdaBoost Learning

Given a feature set and a training set of positive and negative images, any number of machine learning approaches could be used to learn a classification function. Different from the original form of AdaBoost that is used to boost the classification performance of a simple learning algorithm, the AdaBoost in Viola-Jones is used both to select features and train the classifier.

A single AdaBoost classifier consists of a weighted sum of many weak classifiers, where each weak classifier is a threshold on a single Haarlike rectangular feature. For each feature, the optimal threshold classification function is determined by each weak learner such that the minimum number of examples are misclassified. A single weak classifier is defined as:

$$h(x, f, p, \theta) = \begin{cases} 1 & pf(x) < p\theta \\ 0 & otherwise \end{cases} \quad (4)$$

Here x is a 24×24 pixel sub-window of an image. f denotes the feature value, θ is the threshold and p is the parity indicating the direction of the inequality.

The AdaBoost algorithm implemented in this classifier learning is illustrated as follows. It guarantees a very low training error of the strong classifier in implementation.

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
     $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
     so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \, |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
     $$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$$
     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
  $$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
  where $\alpha_t = \log \frac{1}{\beta_t}$

Fig. 4. The AdaBoost algorithm for classifier learning. Each round of boosting selects one feature from the 180,000 potential features.

C. Cascade Classifiers

The cascaded classifier is implemented to increase detection performance while radically reducing The cascaded classifier is implemented to increase detection performance while radically reducing computation time. It is composed of stages each containing a strong classifier from AdaBoost, and the threshold is adjusted to minimise false positives. The job of each stage is to determine whether a given sub-window is definitely not a face or maybe a face. When a sub-window is classified to be a non-face by a given stage, it is immediately discarded. Conversely, a sub-window classified as a maybe-face is passed on to the next stage in the cascade. In this way, subsequent classifiers are trained using examples, which pass through all the previous stages. As a result, the more stages a given sub-window passes, the higher the chance the sub-window contains a face.

In most cases, classifiers with more features will achieve higher detection rates and lower false positive rates, but require more time to compute. In principle, one could define an optimisation framework in which: i) the number of classifier stages, ii) the number of features in each stage, and iii) the threshold of each stage, are traded off in order to minimise the expected number of evaluated features. Unfortunately, finding this optimum is very difficult. In practice, a very simple framework is used to produce an effective classifier which is highly efficient. Each stage in the cascade reduces the false positive rate and decreases the detection rate. A target is selected for the minimum reduction in false positives and the maximum decrease in detection. Each stage is trained by adding features until the target detection and false positives rates are met. Stages are added until the overall target for false positive and detection rate is met.

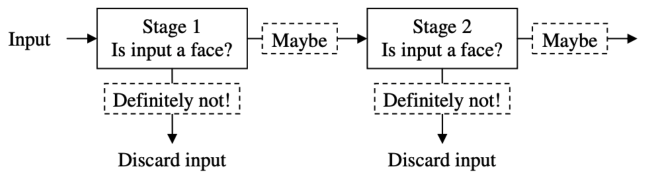


Fig. 5. The cascade classifier. [2]

III. Related Work and Alternate Approaches

There is an extensive literature on object detection. See [3] for a survey. Detecting faces is a crucial step in many identification applications. For example, most face recognition algorithms assume that the face location is known, and face tracking algorithms often assume the initial face location is known [4]. Prior to Viola-Jones, various approaches had been implemented to face detection. Feraud et al. (2001) used neural networks extracting motion, colour and texture features to detect frontal and profile head poses [5]. Other approaches including machine learning [6], information theory [7], geometrical modelling [8], deformable template matching [9], Hough transform [10]. Motion extraction [11] and colour analysis [12] had been implemented.

According to [13] and [14], the original Haar-like feature set has limitation for multi-view face detection and lack robustness in handling faces under extreme lighting conditions, despite that the Haar features are usually normalised by the test window's intensity covariance.

Histogram of Orientated Gradient (HOG) becomes a great alternative feature option as it is largely invariant to global illumination changes and is capable of capturing geometric properties of faces that are difficult to capture with linear edge filters such as Haar-like features [15]. HOGs, which are the distribution of intensity gradients or edge directions, describe the local object appearance and shape within an image. Unlike the Haar-like features, the HOG feature space is relatively small for this case with our 24×24 pixel sub-window of images. But still, we can implement it for comparison and analysis. HOG has been successfully used in many detection tasks, including pedestrian detection [16] and facial landmark detection [17].

Local Binary Patterns (LBP) is a texture descriptor powerful for texture matching and image classification [18]. To calculate the LBP value for a pixel in the grayscale image, after dividing the examined window into cells, we compare the central pixel value with the neighbouring pixel values. Then after computing the histogram over the cell and optionally normalising the histogram, concatenate histograms of cells. This description captures very fine grained details in images, and is simple and fast to implement. It works well to gray-scale and rotation invariant texture classification based on local binary patterns and nonparametric discrimination of sample and prototype distributions. Comparing LBP with Haar features, the advantages of Haar include its high detection accuracy and low false positive rate. For LBP, it has advantages including shorter training time, robustness to local illumination changes and to occlusion, and it is also more computationally simple and fast [19].

Apart from HOG and LBP methods, Principal Component Analysis (PCA) [20] and Artificial Neural Network (ANN) [21] [22] are also comparably good face detection methods. PCA is a statistical procedure aiming to analyse data to identify patterns and reduce data with minimal loss of information. In face detection cases. ANN examines each window to determine whether there is a face, and it reduces the computational task as it does not require to train the non-face images [23]. However, it also require a large number of training examples and are designed primarily to locate frontal faces in gray-scale images [24].

IV. Experiment Setup and Results

The data set used for training and testing comes from MIT CBCL Face Database [25]. The training set face were generated for [26]. The training set non-faces were generated for [27]. The test set is a subset of the CMU Test Set 1 [28], information about how the subset was chosen can be found in [29]. All samples are 19×19 pixels Grayscale PGM format images. The training set has 2,429 faces and 4,528 non-faces. The test has 472 faces and 23,573 non-faces. Some sample images from the dataset are shown as in Figure 6. All the source code is written in python and can be found here.

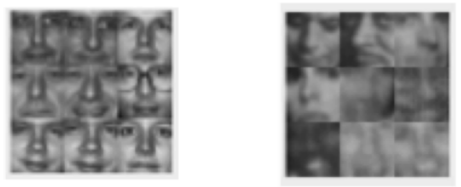


Fig. 6. The sample images. Left images are from the training set, and right images are from the test set.

First, the images are opened and transformed to greyscale if needed (here we do not need to). Then, the images are scaled to 24×24 pixels. The images are saved as a variance normalised matrix.

The final Cascade classifier is trained on 2000 faces and 2000 non-faces from the dataset, and tested on 1000 faces and 1000 non-faces from the dataset. All of the images are randomly selected. I experiment the algorithm by choosing different pairs of stages of cascade classifiers and rounds of AdaBoost training. The best final Cascade has 6 stages and 50 rounds. The best detector successfully detects 999 faces and mis-classifies 243 non-faces. The false positive rate is 12.15% and the false negative rate is 0.35%. The result is shown as in Table 1 and Figure 7 below.

TABLE I
CLASSIFICATION FALSE-POSITIVE RESULT USING VIOLA-JONES

stage\round	20	50	80
1	0.2745	0.3285	0.348
2	0.26	0.2565	0.282
3	0.231	0.1985	0.262
4	0.224	0.194	0.245
5	0.224	0.1225	0.235
6	0.224	0.1215	0.233
7	0.224	0.1215	0.225
8	0.224	0.1215	0.1935
9	0.224	0.1215	0.1925
10	0.224	0.1215	0.1875

TABLE II
CLASSIFICATION FALSE-NEGATIVE RESULT USING VIOLA-JONES

stage\round	20	50	80
1	0.0005	0.0005	0.0005
2	0.0005	0.0005	0.0005
3	0.0005	0.0005	0.0005
4	0.0005	0.0005	0.0005
5	0.0005	0.0035	0.0005
6	0.0005	0.0035	0.0005
7	0.0005	0.0035	0.0005
8	0.0005	0.0035	0.003
9	0.0005	0.0035	0.003
10	0.0005	0.0035	0.003

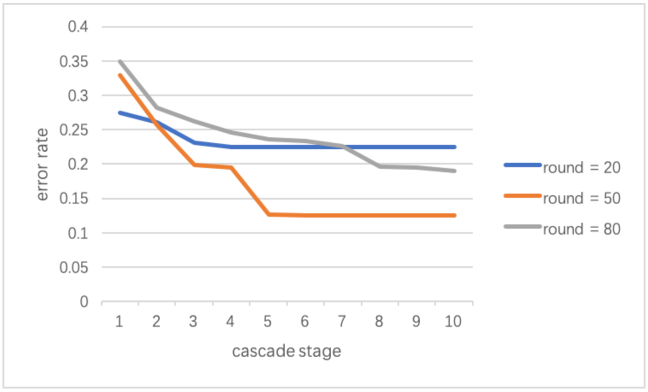


Fig. 7. The classification error (false-positive + false-negative) result using Viola-Jones.

We can see from the result that generally, more AdaBoost rounds and cascade stages bring better classification result. However, giving a limited range of stages, more rounds do not neccessarily mean better classification performance. When AdaBoost rounds are fixed, as cascade stages increase, the false positive rates fall down and tend to stabilise, while false negative may increase. This is probably because of trade-off of the parameters. As stated in section 2.3, finding this optimum is very difficult, and a trail-and-error practical framework is to select for the target that results minimum reduction in false positives and the maximum decrease in detection, and this is how I select my best parameters here.

For comparison, I also used HOG and LBP classifiers imported from OpenCV library [30] for analysis. For the HOG + SVM classifier, after tuning HOG hyper parameters, the best performance on the same size of training and test subsets, which are extracted from the same dataset, is a 1.80% misclassification rate, with $cellsize = 6$, $blocksize = 2$ (12×12 pixels) and $binnumber = 9$. For the LBP + SVM classifier, the best performance on the same size of training and test subsets is a 12.83% misclassification rate, with $radius = 3$ and number of points to be considered as neighbours being 8.

We can see both of the methods get very good classification results, and they run very quickly. The HOG + SVM detector uses a simpler architecture with a single detection window, but appears to give significantly higher classification performance. This may be because of its robust features set which allows faces to be discriminated cleanly, even under rather difficult illumination conditions. The LBP + SVM is also proved to be a very computationally simple, yet efficient classification approach.

V. CONCLUSION

The purpose of this project is to systematically study this research paper, implement the Viola-Jones face detection algorithm, obtain reasonable performance, and compare it with other related detection algorithms. The experimental results successfully explain some of Viola-Jones algorithm's features and meet my expectations. By experimenting on different parameters, I effectively improved the performance of the implementation, getting a best classification model. The Haar feature computation is efficient given the relatively small scale of images, and the learning time of the classifier is longer than I expected. It is likely that my code for cascade classification can be improved in a more time-efficient manner.

In addition, there remain many other aspects that can be improved. First, data preparation seems to be a major source of performance limitation. Regarding the dataset, larger-scaled images containing more head positions can be used for robustness. With subwindow larger than 24×24 pixels, a lot more feature values can be extracted for cascade classifier, HOG + SVM classifier and LBP + SVM classifier, thus more details of the face can be captured. Secondly, in terms of feature extraction, our implementation only extracts 4 types of Haar-like features. To improve performance, maybe more types of Haar-like features can be applied. Additionally, as discussed in section 3, Haar-like features have limitation for texture extraction and multi-view face detection, and lack robustness in handling faces under extreme lighting conditions. We may also consider concatenate Haar features together with other more descriptive and robust features such as HOG and LBP. With increasing data size, the PCA and SVM classifier may also be incorporated into the feature extraction and cascade structure to speed up.

References

[1] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on (Vol. 1, pp. I-I). IEEE.

[2] Jensen, O. H. (2008). Implementing the Viola-Jones face detection algorithm (Master's thesis, Technical University of Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark).

[3] Gavrila, D. M. (1999). The visual analysis of human movement: A survey. Computer vision and image understanding, 73(1), 82-98.

[4] Hsu, R. L., Abdel-Mottaleb, M., & Jain, A. K. (2002). Face detection in color images. IEEE transactions on pattern analysis and machine intelligence, 24(5), 696-706.

[5] Féraud, R., Bernier, O. J., Viallet, J. E., & Collobert, M. (2001). A fast and accurate face detector based on neural networks. IEEE Transactions on pattern analysis and machine intelligence, 23(1), 42-53.

[6] Viola, P., & Jones, M. J. (2004). Robust real-time face detection. International journal of computer vision, 57(2), 137-154.

[7] Turk, M. A., & Pentland, A. P. (1991, June). Face recognition using eigenfaces. In Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on (pp. 586-591). IEEE.

[8] Heisele, B., Ho, P., & Poggio, T. (2001). Face recognition with support vector machines: Global versus component-based approach. In Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on (Vol. 2, pp. 688-694). IEEE.

[9] Yuille, A. L., Hallinan, P. W., & Cohen, D. S. (1992). Feature extraction from faces using deformable templates. International journal of computer vision, 8(2), 99-111.

[10] Xu, L., Oja, E., & Kultanen, P. (1990). A new curve detection method: randomized Hough transform (RHT). Pattern recognition letters, 11(5), 331-338.

[11] Viola, P., Jones, M. J., & Snow, D. (2003, October). Detecting pedestrians using patterns of motion and appearance. In null (p. 734). IEEE.

[12] Hsu, R. L., Abdel-Mottaleb, M., & Jain, A. K. (2002). Face detection in color images. IEEE transactions on pattern analysis and machine intelligence, 24(5), 696-706.

[13] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on (Vol. 1, pp. I-I). IEEE.

[14] Zhang, C., & Zhang, Z. (2010). A survey of recent advances in face detection.

[15] Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on (Vol. 1, pp. 886-893). IEEE.

[16] Bertozzi, M., Broggi, A., Del Rose, M., Felisa, M., Rakotomamonjy, A., & Suard, F. (2007, September). A pedestrian detector using histograms of oriented gradients and a support vector machine classifier. In IEEE intelligent transportation systems conference (Vol. 27).

[17] Zhu, X., & Ramanan, D. (2012, June). Face detection, pose estimation, and landmark localization in the wild. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 2879-2886). IEEE.

[18] Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on pattern analysis and machine intelligence, 24(7), 971-987.

[19] https://github.com/informramiz/Face-Detection-OpenCV

[20] Kim, K. I., Jung, K., & Kim, H. J. (2002). Face recognition using kernel principal component analysis. IEEE signal processing letters, 9(2), 40-42.

[21] Rowley, H. A., Baluja, S., & Kanade, T. (1998). Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence, 20(1), 23-38.

[22] Rowley, H., Baluja, S., & Kanade, T. (1998, June). Rotation invariant neural network-based face detection. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (p. 38). sn.

[23] Soni, L. N., Datar, A., & Datar, S. (2017). Implementation of Viola-Jones Algorithm Based Approach for Human Face Detection. International Journal of Current Engineering and Technology, 7(5), 1819-1823.

[24] Hsu, R. L., Abdel-Mottaleb, M., & Jain, A. K. (2002). Face detection in color images. IEEE transactions on pattern analysis and machine intelligence, 24(5), 696-706.

[25] CBCL Face Database, MIT Center For Biological and Computation Learning. Available online, see http://cbcl.mit.edu/software-datasets/FaceData2.html.

[26] Sung, K. K. (1996). Learning and example selection for object and pattern recognition. PhD Thesis, MIT Al Lab.

[27] Heisele, B., & Pontil, M. (2000). Face detection in still gray images. MASSACHUSETTS INST OF TECH CAMBRIDGE MA CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING.

[28] Rowley, H. A., Baluja, S., & Kanade, T. (1998). Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence, 20(1), 23-38.

[29] Heisele, B., & Pontil, M. (2000). Face detection in still gray images. MASSACHUSETTS INST OF TECH CAMBRIDGE MA CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING.

[30] Bradski, G., & Kaehler, A. (2000). OpenCV. Dr. Dobb's journal of software tools, 3.