'Histograms of Oriented Gradients for Human Detection' versus 'Fast Human Detection Using a Cascade of Histograms of Oriented Gradients'

Heber Stefan

November 20, 2007

Contents

Abstract

Dalal and Triggs [1] studied the question of feature sets for robust visual object recognition. They first considered existing edge and gradient based descriptors and then they showed experimentally that grids of Histograms of Oriented Gradients (HoG) descriptors significantly outperform existing feature sets for human detection. After this they studied the influence of each stage of the computation with reference to performance. They finally came to the conclusion that fine scale gradients, fine orientation binning, relatively coarse spatial binning, and high quality local contrast normalization in overlapping descriptor blocks were important to get good results. Their detector reached nearly perfect results on the original MIT pedestrian database, so they produced a more challenging dataset called INRIA which contained extremely complicated backgrounds and dramatic illumination changes.

Qiang Zhu, Shai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng [2] then showed that the combination of the cascade of rejectors approach and the HoG features led to a fast and accurate human detection system. The features were HoGs with variable block-size. To select the best blocks for the detection, from a larger set of possible blocks, AdaBoost (short for Adaptive Boosting) was used. Furthermore, the integral image representation was used and the rejection cascade significantly speeded up the computation.

1 Introduction

After the development of successful face detection algorithms, human detection was the logical next step. But owing to the wide range of poses and the variable appearance because of clothing, articulation and illumination, human beings were a much more difficult object to detect. So first of all they needed a robust feature set that allowed an accurate detection of the human form.

Dalal and Triggs [1] showed that locally normalized HoG descriptors provide excellent performance compared with existing feature sets (including wavelets). Their descriptors were computed on a dense grid of uniformly spaced cells and they used overlapping local contrast normalization for improved performance. They developed a human detection algorithm with excellent detection results. They were using a dense grid of HoG, computed over blocks of size 16x16 pixels, which corresponded to a detection window. Combined with a linear SVM this representation was powerful enough to classify human beings. However, this method could only process 320x240 images at 1 FPS. And they only used a very sparse scanning, which evaluated about 800 detection windows per image.

Qiang Zhu, Shai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng [2] improved the computation speed and increased the number of detection windows from 800 to 12800. To do so they combined the cascade of rejector

approach with the HoG features. Furthermore, they discovered that the use of fixed size blocks was not informative enough to allow fast rejection in the early stage of the cascade. Hence they used a much larger set of blocks that varied in size, location and aspect ratio. They used AdaBoost to select the appropriate set of blocks and constructed the rejector based cascade. The result was a nearly real time human detection system. Compared with existing methods it matched them in terms of accuracy and significantly outperformed them in terms of speed.

2 The Dalal Triggs Algorithm

The algorithm is based on evaluating well normalized local histograms of image gradient orientations in a dense grid. The main idea is that local object appearance and shape can often be characterized rather well by the distribution of local intensity gradients or edge directions. In practice this is achieved by dividing the image into cells and for each cell a 1-D histogram of gradient directions or edge orientations over the pixels of the cell is calculated. A contrast-normalization is used on the local responses to get better invariance regarding illumination, shading, etc.. This is done by accumulating a measure of local histogram energy over the blocks (2x2 cells) and the result is then used to normalize the cells in the block.
So each detection window is divided into cells of size 8x8 pixels and each group of 2x2 cells is integrated into a block. Hence blocks overlap with each other.
Sub-dividing the detection window into a grid of overlapping normalized blocks and using the combined feature vector to train a linear SVM gives the human detection chain used by Dalal and Triggs (see Fig.1).

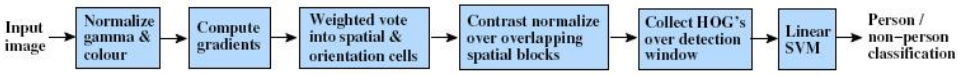


Figure 1: Dalal and Triggs's feature extraction and object detection chain. The detection window is sub-divided into overlapping blocks in which HoG feature vetors are extracted. The combined feature vectors are then used to feed a linear SVM for human/non-human classification. The detection window is scanned across the image at all positions and scales.

2.1 Data Sets

Dalal and Triggs tested their detector on two data sets. The first was the MIT pedestrian database, which contained 509 training and 200 test images of pedestrians in city scenes. There were only front and back views with a limited range of pose. They received nearly perfect results on this data set,

so they produced a new and significant more challenging data set, called INRIA. This data set contained 1805 64x128 images of humans. The people appeared in any orientation and against a wide varied of background image.

Figure 2: Some samples from the database INRIA.

3 Fast Human Detection

Dalal and Triggss algorithm makes use of three key components:
1)HoG as a basic building block
2)A dense grid of HoGs across the entire detection window
3)A normalization step within each block
Qiang Zhu and collaborators criticized the missing of multiple scale blocks. Dalal and Triggs only used a small block size (16x16 pixels) which maybe miss the global features of the whole detection window. Actually they mentioned that the use of different scales for the blocks or cells would improve the results but simultaneously it would significantly increase the computation cost.
To speed up the detection process Qiang Zhu and collaborators now used a cascade of rejectors and AdaBoost to select the features which need to be evaluated in each stage (each feature corresponds to one block). But the problem was that none of the small blocks (used in Dalal and Triggss algorithm) was informative enough to reject enough patterns. As a consequence they increased their feature space to include blocks of different sizes, locations and aspect ratios. In addition they detected that the first stages of the cascade, which rejected the majority of the detection window, actually used large blocks and the small blocks were used much later in the cascade. Owing to the fast evaluation of specific blocks by the AdaBoost-based feature selection algorithm Qiang Zhu and collaborators used the 'integral image' representation to compute efficiently the HoG of each block.

3.1 Integral Image/Histogram

The integral image [3] is an intermediate representation for images which allows a rapid computing of rectangular features. The integral image at location x, y contains the sum of the pixels above and to the left of the pixel (x,y), including (x,y).

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

(where ii(x,y) is the integral image and i(x,y) is the original image) Using the integral image any rectangular sum can be computed with the four edge values.

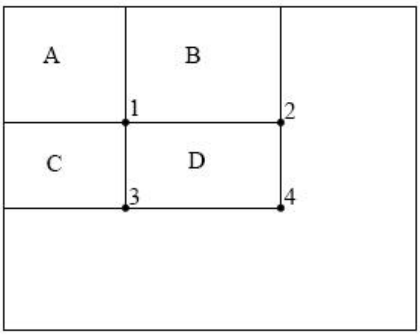


Figure 3: The sum of the pixels within the rectangular D can be computed with the four edge values. The value of the integral image at location 1 is the sum of the pixels in the rectangular A. The value at location 2 is A+B, at location 3 it is A+C and at location 4 it is A+B+C+D. Hence the sum within D can be computed as followed: D=4-3-2+1.

Porikli [4] suggested a similar method to efficiently compute histograms over arbitrary rectangular regions, called 'Integral Histogram' (see Fig.4).

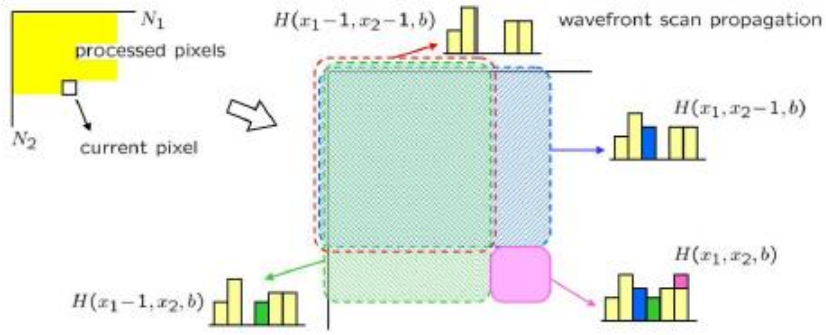


Figure 4: Integral Histogram

3.2 Integral HoG

Based on the last methods Qiang Zhu and collaborators developed the following fast way to calculate the HoG feature: First of all they discretized

each pixel's orientation (including its magnitude) into 9 histogram bins. Next they computed and stored an integral image for each bin of the HoG. And finally they used them to compute efficiently the HoG for any rectangular image region.
On the one hand this method was fast to compute but on the other hand it was inferior to Dalal and Triggs approach because of the following two reaons:
First, they could not use a Gaussian mask and tri-linear interpolation in constructing the HoG for each block because this would not fit with the integral image approach.
Second, they used L1 normalization instead of L2 normalization because it allowed a faster computation when using the integral images.

3.3 Variable Size Blocks

Dalal and Triggs only used a fixed block size of 16x16. They mentioned that using blocks and cells with different scales improved the results but increased the computational cost.
Qiang Zhu and collaborators used feature selection to handle this problem. For a 64x128 detection window they considered blocks with a size between 12x12 and 64x128. The ratio between block width and height could be (1:1), (1:2) or (2:1). The step size could be 4, 6 or 8 pixels and depended on the block size, since they wanted to obtain a dense grid of overlapping blocks. So each 64x128 detection window defined 5031 blocks. And each block contained a 36-D histogram vector of concatenated 9 orientation bins from 2x2 sub-regions (cells).
There are two advantages of using different block sizes. First, the useful patterns usually spread over different scales, so the original fixed size blocks will only encode very limited information. Second, some of the blocks in the larger set of blocks might correspond to a special part of the human body (e.g. leg). For a small number of blocks with fixed sizes, such a mapping is less likely to establish.

3.4 The Cascade

The cascade can be seen as an object specific focus-of-attention mechanism. One or more classifiers are ranged in a cascade structure which dramatically increases the speed of the detector by focusing attention on specific regions. The key measure of such an approach is the false negative rate.

This detection process works like a degenerated decision tree. A positive result on the first classifier triggers the second, and so on. A negative result in any classifier stage leads to an immediate rejection of the sub-window.
The classifiers used by Qiang Zhu and collaborators were the 'separated hyperplane', computed using linear SVM.

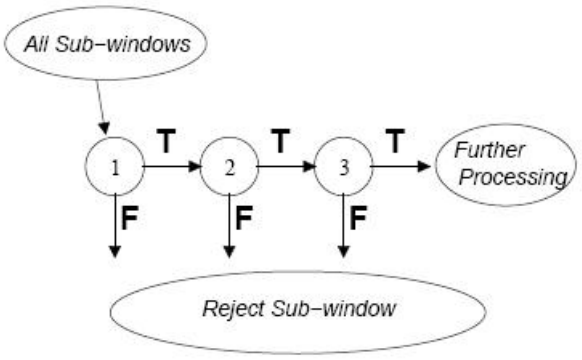


Figure 5: Schematic representation of a detection cascade.

Because of the fact that evaluating 5031 blocks in each stage would be very time consuming, they made use of a sampling method suggested by Scholkopf and Smola [5]. This method says that one can find, with a high probability, the maximum of n random values, in a small number of trials. So in practice Qiang Zhu and collaborators only evaluated randomly 250 blocks in each stage.

3.5 Training the Cascade

Qiang Zhu and collaborators constructed for each level of the cascade a strong classifier consisting of weaker classifiers (linear SVMs). In each level of the cascade weak classifiers were added until they reached the predefined quality requirements.

Algorithm: Training the cascade

Input: F_{target}: target overall false positve rate
f_{max}: maximum acceptable false positive rate per cascade level
d_{min}: minimum acceptable detection per cascade level
POS: set of positive samples
NEG: set of negative samples

initialize: i=0, d_i=1.0, F_i=1.0
loop $F_i > F_{target}$
i=i+1
f_i=1.0
loop $f_i > f_{max}$
1) train 250 (5% at random) linear SVMs using POS and NEG samples
2) add the best SVM into the strong classifier,
update the weight in AdaBoost manner

3) evaluate POS and NEG by current strong classifier
4) decrease threshold until d_{min} holds
5) compute f_i under this threshold
loop end
$F_{i+1} = F_i \times f_i$
$D_{i+1} = D_i \times d_{min}$
Empty set NEG
if $F_i > F_{target}$
then evaluate the current cascaded detector on the negative,
i.e. non-himan, images and add misclassified samples into det NEG.
loop end

Output: A i-levels cascade
each level has a boosted classifier of SVMs
Final training accuracy: F_i and D_i

3.6 Experiments

Figure 6: HoG vs. Rectangular filters. (a) The best HoG filter. (b) The best rectangular filter.

Qiang Zhu and collaborators compared a rectangular filter with the HoG filter. In their first experiment they showed the large difference in stability between these two filters. For the stability analysis they took the best HoG feature and the best rectangular feature (see Fig.6). They proceeded as followed:
They computed the HoG features on each of the training images, and took the mean. Then they measured the correlation between the HoG features and the mean HoG feature. After that they did the same with the rectan-

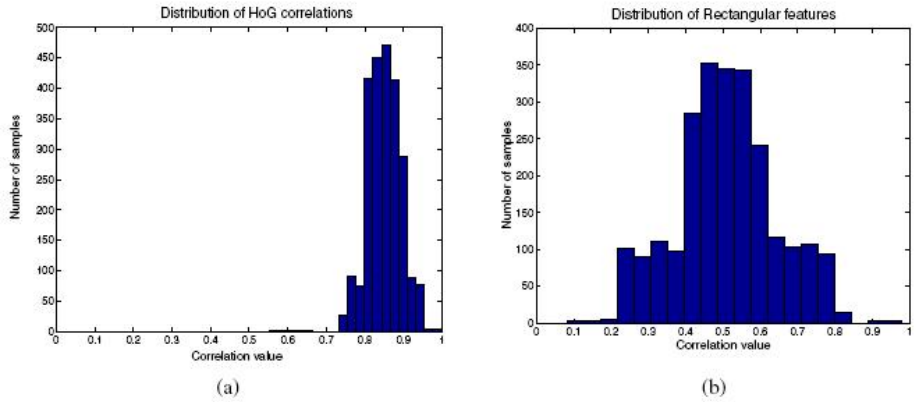


Figure 7: Stability of HoG vs. Rectangular filter. (a) Stability of the HoG feature. (b) Stability of the Rectangular feature

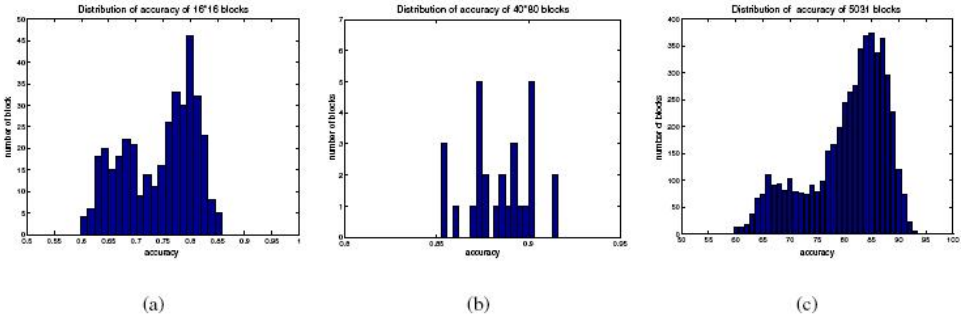


Figure 8: Classification of blocks of different sizes. (a) small blocks (16x16), (b) large blocks (40x80) and (c) all 5031 blocks

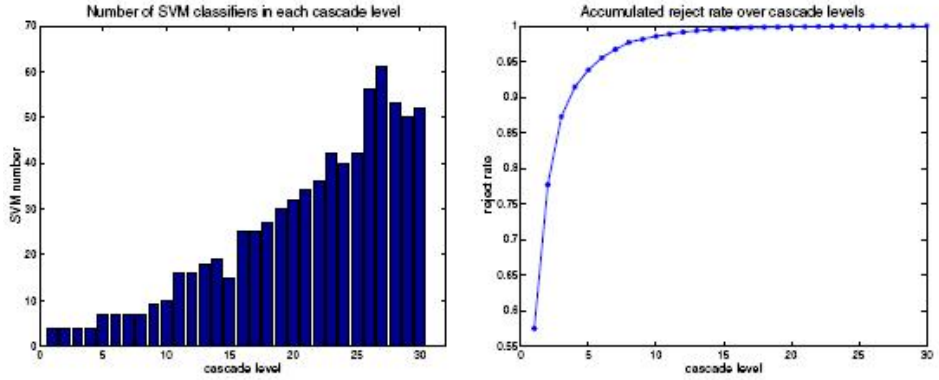
gular feature (see Fig.7). The result showed that the HoG feature was much more stable (peak at 0.85; variance about 0.1) than the rectangular feature (peak around 0.5; variance about 0.3).

With their second experiment they showed the importance of different block-sizes.

Figure 8 shows the distribution of the classification accuracy of blocks with different sizes. The first histogram (in Fig.8) is of all blocks of size 16x16 pixels, the second histigram is of all blocks of size 40x80 and the third histogram is of all blocks (5031). The first result is that 16x16 blocks are least informative and the second result is that increasing the number of blocks contributes significantly to the performance of their method.

The next experiment (see Fig.9) gives some details concerning the cascade. The cascade consists of 30 levels, where the weak classifieres are linear SVM. Figure 9 (a) shows the number of SVM clasifiers in each level. Figure

9 (b) shows the rejection rate as cumulative sum over all cascade levels. The first four levels contain four SVM classifiers each and reject about 90% of the detection windows. Hence, on average, 4.6 block-evaluations are needed to classify a detection window.



(a) (b)

Figure 9: Cascade details. (a) The number of SVM classifiers in each level. (b) The rejection rate as cumulative sum over cascade levels

In the following experiment they took a look at the most informative blocks.
Figure 10 (a) shows the top five blocks with reference to the minimum error rate (block-size is about 36x80). The blocks selected at cascade level 1, 2 and 8 are visualized in Figure 10 (b) to (d). During the cascade training randomly 5% of all possible blocks are evaluated. Hence, the blocks choosen in the cascade may not be the globally best ones. But they observed that the blocks loacted in some specific regions (e.g. body, legs or hand) achieve a much higher accuracy and therfore these blocks are mor likely to be selected.
The Dalal and Triggs algorithm, with fixed block-size (16x16), has a comparatively much higher error rate. Blocks of size 16x16 appear only after the 8th stage of the cascade. The early stages use much larger block-sizes.
Another notable thing is that the most selected blocks are located in the middle of the area. This demonstrates the efficency of AdaBoost selecting the most informative blocks.

At the end they compared four approaches: Dalal and Triggs, a cascade of rectangular features and their approach using L1 and L2 norm. Figure 11 shows that the method is comparable to Dalal and Triggs' method. In low FPPW area both L1 and L2 norm work better than Dalal and Triggs, but by reducing the miss rate the Qiang Zhu and collaborators-approach incures a higher FPPW.

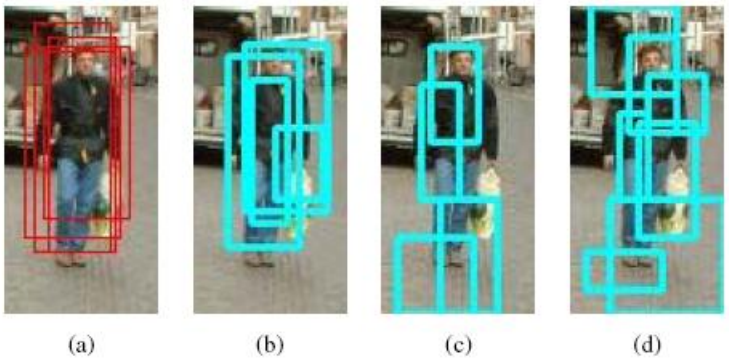


Figure 10: Visualizing the selected blocks. (a) Top 5 blocks with minimum error rate. (b) Blocks in level 1, (c) in level 2 and (d) in level 8

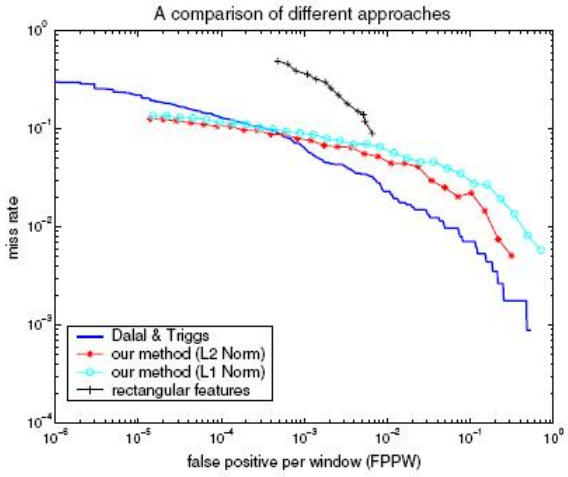


Figure 11: Comparing Dalal and Triggs' algorithm, a Rectangular filter detector and the cascade of HoG mehtod (using L1 or L2 norm)

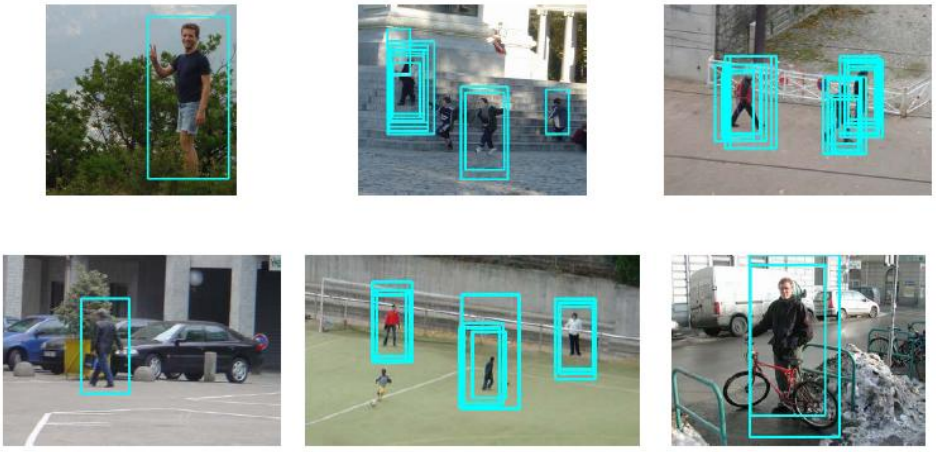


Figure 12: Some typical results from the cascade of HoG method

4 Conclusion

Dalal and Triggs showed that the use of locally normalized HoG features in a dense overlapping grid led to a very good human detection system. They studied the influence of different descriptor parameters and came to the conclusion that fine-scale gradients, fine orientation binning, relatively coarse spatial binning, and high-quality local contrast normalization in overlapping descriptor blocks were important for good results.
Qiang Zhu, Shai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng then picked up Dalal and Triggs' idea of using HoG features and developed a near real time human detection system. Compared with other methods it reached a speedup of up to 70x. The most important point of their approach was that features were selected from a large set of blocks of multiple sizes, locations and aspect ratios.

5 References

[1] N. Dalal and B. Triggs: Histograms of Oriented Gradients for Human Detection

[2] PQiang Zhu, Sai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng: Fast Human Detection Using a Cascade of Histograms of Oriented Gradients

[3] Paul Viola and Michael Jones:
Rapid Object Detection using a Boosted Cascade of Simple Features

[4] Fatih Porikli: Integral Histogram: A FastWay to Extract Histograms in Cartesian Spaces

[5] B. Scholkopf and A. Smola:
Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond