Efficient pedestrian detection: a test case for SVM based categorization

Vincent Depoortere[1] Jeroen Cant[1] Bram Van den Bosch[1] Jan De Prins[1]
Rik Fransens[1] Luc Van Gool[1,2]
[1] ESAT/PSI, Kath. Univ. Leuven, Kasteelpark 12, B-3001 Leuven, firstname.lastname@esat.kuleuven.ac.be
[2] D-ITET/BIWI, ETH, Gloriastrasse 35, CH-8092 Zurich, lastname@vision.ee.ethz.ch

Abstract

We present a method for categorization that is based on the classification of wavelet coefficients by SVMs. In particular, pedestrian detection is elaborated as a case in point. Starting from an approach propounded by Papageorgiou et al.*, several extensions and modifications are presented that lead to an increase of both performance and speed. Pedestrian detection runs at 6Hz and is illustrated for realistic, cluttered imagery.*

1 Goal

Computer vision so far mainly focused on the recognition of specific objects. This already turned out to be a major problem, given the variability of viewing and lighting conditions that occur. Recently, there is a resurgence of activity in the even more demanding area of object categorization. There, members of object classes need to be detected, and the degree of variability is still substantially larger.

In this paper, we propose a system that can learn categories. In particular, we started with the problem of pedestrian detection as a case in point. It is a challenging problem, because people will appear in quite different poses and their clothing will add additional changes to their appearances. The pedestrians also have to be detected in cluttered scenes.

Section 2 succinctly discusses the basic system from which we started, but that we have accelerated with several orders of magnitude, without lowering its performance. These accelerations are the subject of section 3. Section 4 describes the architecture of the system, constructed on the basis of these results. Section 5 then discusses some experimental results. Section 6 concludes the paper.

2 Pedestrian detection by Papageorgiou *et al.*

Papageorgiou *et al.* [8] have proposed a system to build a detector for object classes. They have implemented their method for the practically interesting case of pedestrian detection. Their system will serve as a point of departure for our work and, hence, it is useful to concisely describe it here.

This method uses Haar wavelets as input features. Their sensitivity to intensity differences rather than absolute brightness is advantageous in settings with variable illumination conditions. Moreover, they take the maximum of the absolute values of the coefficients over the three color bands. In the sequel, if we refer to ‘coefficients’ this actually is shorthand for these maxima. This takes account of the fact that contrasts and colors in clothing can differ substantially. The coefficients are extracted from image windows of size 128×64 pixels. To increase the spatial resolution at which the patterns within the windows are investigated, the over-sampled,‘quadruple density Haar wavelet transform’is used [6]. Using wavelets of sizes 16×16 and 32×32 yields feature vectors $\mathbf{x} \in \mathbb{R}^p$, with $p = 1326$.

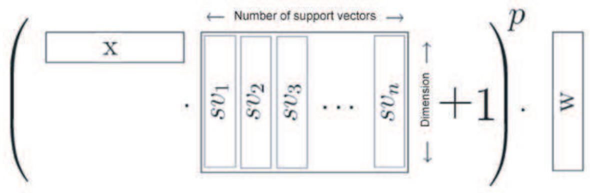
Based on a set of positive training examples (i.e. containing pedestrians) $\{\mathbf{x}_i^{(v)}\}_{i=1}^{i=n_v}$ and a set of negative examples $\{\mathbf{x}_i^{(nv)}\}_{i=1}^{i=n_{nv}}$, a Support Vector Machine [5, 2] is trained. SVMs are particularly suited for high-dimensional classification problems with only a small training set available. The kernel is polynomial and of degree 2. The decision surface is of the form:

$$f(\mathbf{x}) = \sum_{i=1}^{i=N_s} \lambda_{s_i} y_{s_i} (\mathbf{x}\mathbf{x}_{\mathbf{s_i}} + 1)^2 + b = 0. \qquad (1)$$

The feature vector, extracted from a window, is assigned to the pedestrian class if $f(\mathbf{x}) > 0$.

3 System acceleration

Our goal is to accelerate pedestrian detection, but without a loss of accuracy. An analysis of the classification function,



shows that the amount of computational work, is mainly determined by the dimensionality of the feature vector $\mathbf{x}$ and the number of support vectors. By reducing both, the system can be speeded up dramatically. In what follows we discuss three methods that accomplish this:

1. dimension reduction via feature selection

2. dimension reduction by projecting the feature vector in a low dimensional subspace

3. reduction of the number of support vectors.

Furthermore, we will build a cascade of classifiers and we will apply integral images in the calculation of the Haar wavelet coefficients [10], to make the system real time.

Simply applying the aforementioned techniques would inevitably result in a decrease of accuracy. As an example, Papageorgiou *et al.* [8] have experimented with drastic reductions down to only 29 coefficients, and found that accuracy suffered substantially. A way to counteract such effects is to start from an extended feature set. This is all the more so as feature selection will be an integral part of the procedure. In this setting the initial number of features is restricted mainly by the number the off-line feature selection can cope with. In our first experiments, we have added wavelets of sizes 4×4 and 8×8, increasing the number of features to 4737. This already proved sufficient to achieve our goal of dramatically speeding up the system while keeping accuracy intact.

3.1 Feature selection

In [8] Papageorgiou *et al* introduce feature selection based on the mean of a wavelet coëfficient, relative to the overall mean of all coefficients of the same class. We compared this method with three other feature selection methods. The first method makes use of the Bhattacharyya measure. The second was described in [10] and is based on AdaBoost. The third one is based on an analysis of the SVM decision surface, and is called the method of *importance vectors* in [3]. Here, we briefly describe the different methods, the results of the comparison are presented in section 5.

3.1.1 Mean based

The method proposed by Papageorgiou *et al* [8] consists of two steps. First, for each class of wavelets (in our case, the 12 classes given by $\{vertical, horizontal, diagonal\} \times \{4 \times 4, 8 \times 8, 16 \times 16, 32 \times 32\}$) the mean is calculated, over all coefficients of that class and over all training images. The coefficients are then normalized by dividing with their respective class means. In the second step, for each coefficient, the mean is taken of the normalized values over all training data. The normalization has the property that the mean of the coefficients over random patterns will be 1. Mean values significantly larger than 1 are considered to correspond to a consistent intensity change. On the other hand, a value much smaller than 1 corresponds to a consistent, uniform region. Those most different from 1 indicate the interesting features.

3.1.2 Bhattacharyya based

Given the densities $p(x_i|\Omega)$ and $p(x_i|\overline{\Omega})$ for all features $i = 1 \cdots n$, and with Ω the pedestrian class and $\overline{\Omega}$ the non-pedestrian class, the most important features can be selected based on the Bhattacharyya measure:

$$b(p(x_i|\Omega), p(x_i|\overline{\Omega})) = \int \sqrt{p(x_i|\Omega)p(x_i|\overline{\Omega})}dx. \qquad (2)$$

The Bhattacharyya measure $b(p(x_i|\Omega), p(x_i|\overline{\Omega}))$ places an upper bound on the Bayesian error for the two-class classification problem, using only feature x_i [1]. Therefore, a small value for this measure indicates a low error-rate and thus a strong feature.

In our work the densities $p(x_i|\Omega)$ and $p(x_i|\overline{\Omega})$ were estimated with the Parzen-windows technique described in [9]. Briefly this method involves the convolution of the density histogram with a gaussian with variance:

$$\sigma^2 = \sigma^2_{data} \times h^2_{opt},$$

with

$$h_{opt} = \sqrt[5]{\frac{4}{3 \times N}}$$

and N the number of training examples. In this manner a smooth estimate is found based on the histogram.

3.1.3 AdaBoost based

In [10] Viola & Jones introduced a feature selection method based on the AdaBoost algorithm [11]. Originally AdaBoost is designed to combine so-called “weak” classifiers in such a way that, together, they form a “strong” classifier. In this procedure the best classifiers get the larger weights while the weaker ones receive smaller weights. By only considering classifiers that rely on just one feature, the algorithm can be considered to be a feature selection method. In this setting, classifiers that receive a large weight correspond to strong features.

Essential for the AdaBoost procedure is that all training data are assigned a weight that changes throughout the algorithm. Initially all positive and negative data are assigned the same weight respectively. Then, for each feature a classifier is constructed and its performance is assessed on the training data as the sum of the weights of the examples that are falsely classified. After selection of the classifier with best performance, the weights of the wrongly classified examples are increased while the weights of the correctly classified examples are decreased. In this manner the choice of the next classifier is more influenced by the examples that were missed by the previous one. This procedure is repeated until the wanted number of features is selected.

In [10] the weak classifier for feature x_i consists of a single threshold δ_i on the value of x_i. This corresponds to the fact that they use a lineair classifier. The decision surface is a plane that crosses the respective axes in only one point. In our case however, where we use polynomial SVMs of degree 2, the decision surfaces may cross an axis in more than 1 point. Therefore, features that need more than one boundary to discriminate between the two classes must be considered too. To achieve this we use a Bayesian classifier and impose a threshold on the ratio $\frac{p(x_i|\Omega)}{p(x_i|\overline{\Omega})}$, rather than on the value x_i. The densities are estimated as described above.

3.1.4 Importance Vectors

The previously described feature selection methods are applied before the SVM is trained. The method of "importance vectors" [3], however, makes use of the decision surface of a SVM that was trained with the full set of features. The decision surface is then analyzed to identify those features that influence the decision most. These are the features for which the average magnitude of the derivative of $f(\mathbf{x})$ with respect to that feature, in a volume around the decision surface, are maximal. In practice this analysis is done by evaluating the gradient of $f(\mathbf{x})$ in the support vectors $\mathbf{x}_{s_i}, i = 1, \ldots, N_s$ only (see figure 1). Since the support vectors are the data points that lie closest to the decision boundary, we can expect these gradients to approximate the normals on the decision boundary.

3.2 Projecting onto a low dimensional subspace

The dimension of the feature vector can be further reduced by working with the projections onto a lower dimensional subspace. Again, in order to preserve accuracy, it is important that this subspace is spanned by axes with maximal discriminative power. Intuitively, these will be oriented maximally perpendicular to the decision surface [4]. These

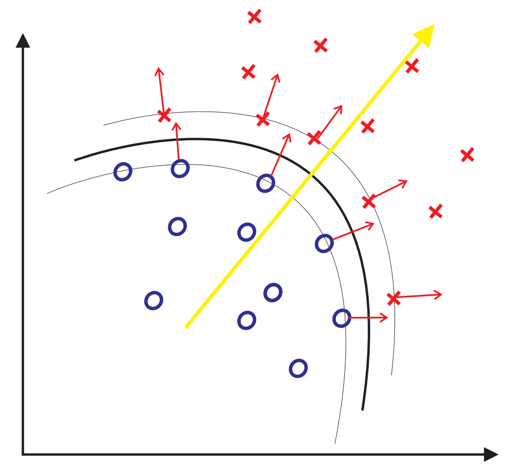


Figure 1. *The normals to the decision surface are approximated well by the gradients in the support vectors. Dominant orientations determine the axes of a subspace in which classification can proceed more efficiently.*

discriminative directions are found as the dominant directions of normals to the decision surface. The gradients in the support vectors are a good approximation of these normals. The intended directions are found as the dominant eigenvectors of

$$\boldsymbol{N}\boldsymbol{N}^T, \tag{3}$$

with the gradients in the support vectors as columns of $\boldsymbol{N}$ and ^{T} the transpose.

During the classification stage the extracted feature vectors $\mathbf{x}$ are first projected onto this lower dimensional subspace. Remark that the extra cost of the projection is negligible compared to the gain in the evaluation of eq.(1).

3.3 Reducing the number of support vectors

The evaluation of eq. (1) can be speeded up further through a reduction in the number of support vectors. To accomplish this we used a method proposed by Osuna *et al.*[7], that makes use of 'Support Vector Regression Machines' (SVRMs). SVRM is an extension of SVM to the domain of function approximation.

3.4 Cascade of classifiers

The methods described above try to minimize the computational work, needed to evaluate the classification function (1). In our quest for real-time performance, we finally add a divide-and-conquer strategy based on a cascade of

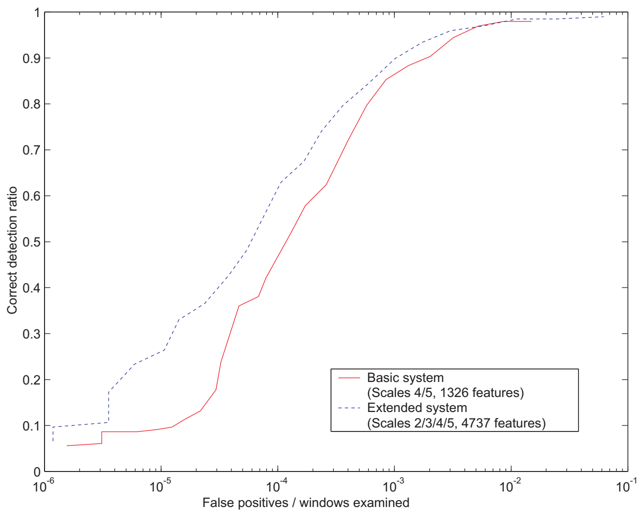


Figure 2. *By increasing the number of features, the system was made more accurate.*

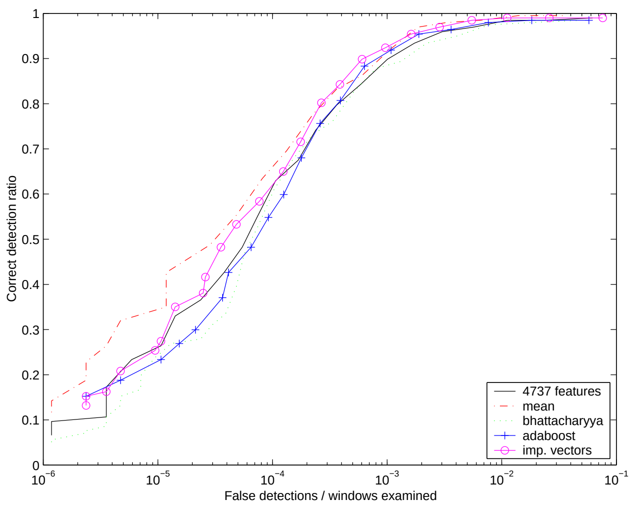


Figure 3. *ROC curves for 1326 features selected with the different feature selection methods. 'Importance vectors' scores best in the region of 80-95% correct detections.*

SVM classifiers. Each classifier is more sophisticated – and hence slower – than the previous one, but is also better able to deal with the increasingly difficult cases left over by the previous classifier. As smaller and smaller sets of windows need to be passed on to the subsequent, more complex classifiers, the result is a substantial gain in speed. It is crucial that a minimum of pedestrians are being lost in the process.

4 System architecture

This section gives an overview of the overall organization of our approach, i.e. of the different steps and the selected method for each of them. A series of detectors are proposed, to be operated as a cascade. All tests in this work were carried out on the basis of 98 cluttered city scenes, containing 197 pedestrians.

4.1 Feature selection

As mentioned before, we first increased the number of features to 4737, in order to compensate for the loss in accuracy expected from the speed up process. Figure 2 shows the receiver operator curves (ROC) for the basic system'' (Papageorgiou *et al.*) and the extended system'' using the additional features. The results for the extended system are clearly superior. In a first step, the number of features was reduced back from 4737 to 1326, the original number of features used by Papageorgiou *et al.*, using the 'importance vectors' method. As fig. 3 shows, this method yields the best accuracy of the four aforementioned feature selection techniques in the relevant part of the ROC graph. Notice that the performance has not even decreased with respect to the 4736 feature system. This detector based on 1326 features will serve as our most sophisticated one, at the end of the cascade.

Encouraged by this success, a further feature reduction was undertaken and their number was lowered to 29. Again, all four feature selection techniques were compared. As fig. 4 demonstrates, for such drastic reduction Adaboost, which is an aggressive and greedy scheme, performed best. The accuracy is significantly lower than the extended system (4737 features) and slightly worse than basic system (original 1326 features). As will be explained later on, this system is still very useful as one stage in the cascade of classifiers.

4.2 Projection onto a low dimensional subspace

As it stands, the 1326 feature based detector at the end of the cascade would still be slow, even if it gets only a small minority of windows to handle. In this section, its speed is improved, by projection onto a lower dimensional subspace.

An analysis of the eigenvalues of the matrix (3) shows that the 13 largest ones make up for 99% of the total sum (fig. 5). As can be expected, the 13 corresponding eigenvectors form an effective subspace. This is corroborated by figure 6. It shows the performance for a SVM that was trained on the projections of the original training data. Also the classification was carried out on the projections of the feature vectors. The accuracy is only slightly inferior to that of the system with 1326 selected features (see section 4.1).

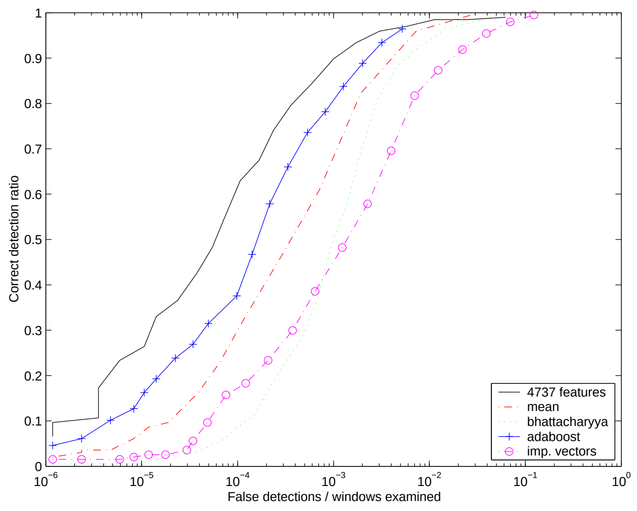


Figure 4. *ROC curves for 29 features selected with the different feature selection methods. 'AdaBoost' scores best.*

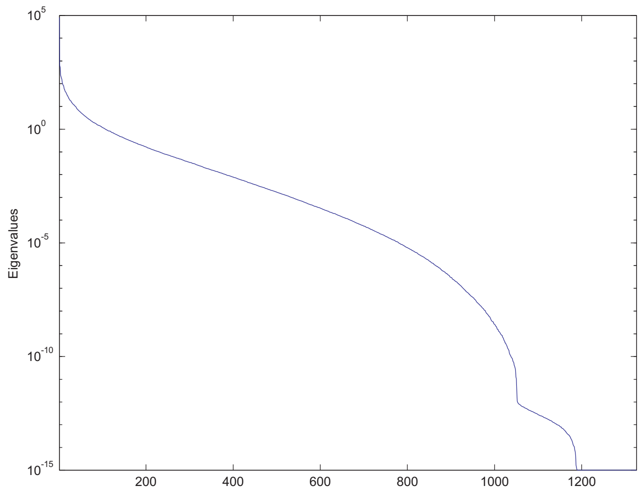


Figure 5. *Ordered eigenvalues for the system with 1326 features, selected by 'Importance Vectors'.*

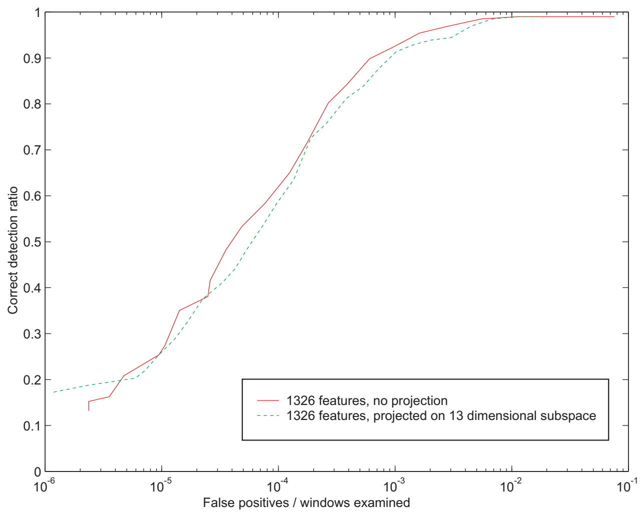


Figure 6. *The performance before and after projection onto a* 13 *dimensional subspace. The performance dropped slightly.*

4.3 Reducing the number of support vectors

Given the high speed that we want to achieve, both detectors – the one based on 1326 as well as that based on 29 features – need further acceleration. This was achieved by reducing the number of support vectors, using SVRMs as described in 3.3. For the 1326 features system, after projection in the 13 dimensional subspace, the number of support vectors could be reduced from 1594 to 106 without a loss of accuracy (fig. 7). The number of support vectors of the 29 feature classifier was reduced from 2118 to 299, again without loss of accuracy (fig. 8.

This procedure makes sense here, as there is quite a proportion of so-called 'upper bounded support vectors'. The 29 feature system, for example, had 2006 upper bounded support vectors on a total of 2118 support vectors.

4.4 Cascade of classifiers

The cascade of classifiers is then configured as follows:

1. a SVM on the basis of 29 features, but with a linear kernel
2. the SVM on the basis of 29 features, with a kernel or degree 2
3. the SVM on the basis of 1326 features, with a kernel of degree 2.

The latter two classifiers have been described before. The first one is based on the same 29 features as the second one,

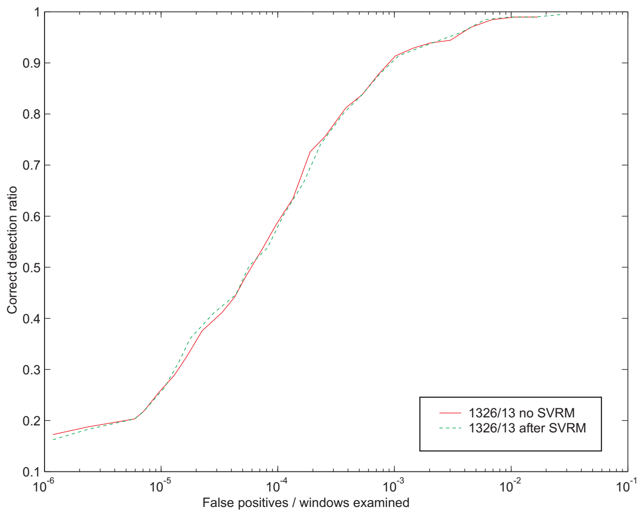


Figure 7. *ROC curves before and after SVRM, for the system with 1326 features, projected onto a 13 dimensional subspace*

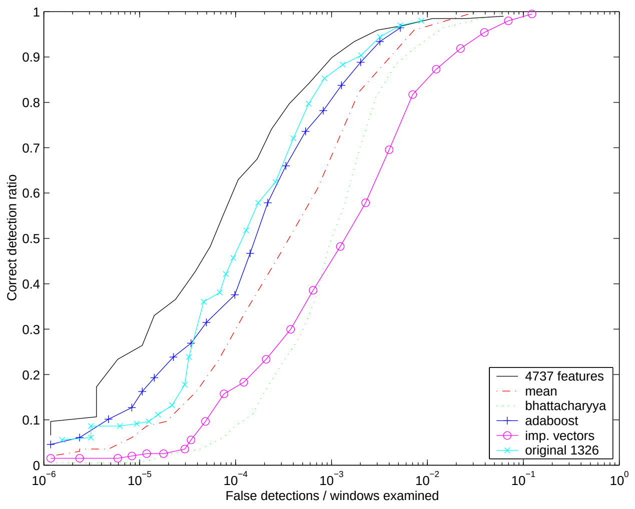


Figure 8. *ROC curves before and after SVRM, for the system with 29 features*

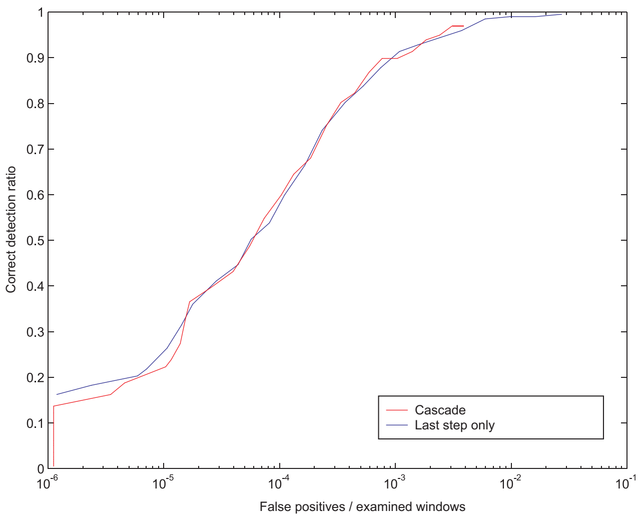


Figure 9. *The performance for the cascade with 3 classifiers, compared to the performance of the last and most accurate step only. Adding fast and simple classifiers increases the speed, without loosing performance.*

but uses a linear kernel. By choosing an appropriate threshold on the output of this classifier, this first, very fast step, is able to discard 97% of windows from further consideration, while loosing fewer than 1% of pedestrians. Indeed, as fig.9 shows, performance doesn't suffer from putting the first two classifiers in front of the last one.

At this point, the computation time needed for the calculation of wavelet coefficients, exceeded that of classification. Therefore, the efficiency of extracting Haar wavelet coefficients was improved by using integral images [10]. As a further measure of acceleration, we dropped the color information and turned to gray scale data. Pedestrians wear all kinds of colors, so color *per se* was expected not to be a very distinctive feature. As a result of these accelerations, the focus returned to the classification part, now the slowest link again. In keeping with the cascade philosophy, adding a fourth classifier proved effective. For this extra step, we used only 4 features, selected from the extended set of 4737 features, by the AdaBoost procedure. A linear SVM was trained with these 4 features, while giving a higher cost to false negatives. Half of the windows could be discarded from further processing after this classifier.

The net effect of these efforts, is an overall acceleration by a factor of 905 compared to the original 'basic system with 1326 features [8], and this with improved accuracy (see fig. 10). On an AMD Athlon 1.4GHz, the system runs at 6 frames/sec for a matlab/C implementation. Alternative accelerations of this system have been proposed by Papageorgiou *et al.* in the same reference, but with a substantial

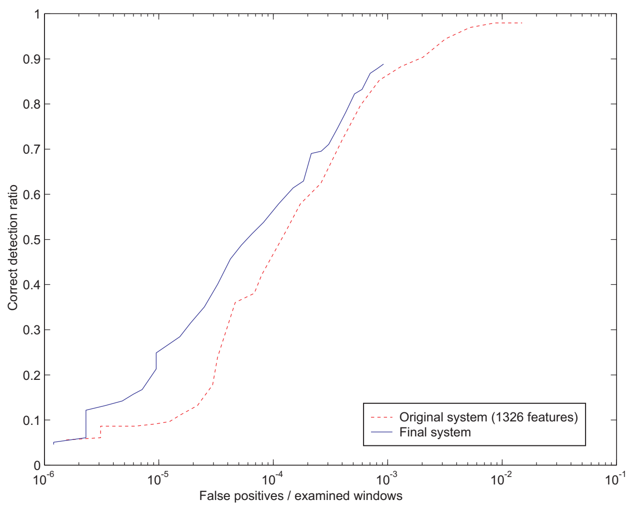


Figure 10. *The ROC curve of the final system, running at 6Hz, compared to the original system with* 1326 *features.*

loss of precision as a result. Our improved results are based on several modifications to ther system:

1. an extended feature set to start from,
2. efficient wavelet calculations through integral images,
3. a gamut of feature selection methods, with the best one used in each step,
4. projection onto a low dimensional subspace,
5. a cascade rather than one shot classification.

5 Experimental results

Fig. 11 shows a number of correct pedestrian detections. Notice the cluttered backgrounds and the different poses of the pedestrians. The system is robust for the presence of special attributes, such as bags and purses. In figure 12 some examples of false detections are depicted. The system strongly responds to some symmetric, vertical structures that are not pedestrians. The classification values for the pedestrians are in most cases higher though. Sometimes the system wasn't able to find the pedestrians. In the case of figure 13, for instance, the pedestrians didn't contrast strongly enough with the background.

6 Conclusions and future work

We have presented a pedestrian detector that operates at $6Hz$ on data of video resolution. The performance is better than that of the reimplemented approach of Papageorgiou *et al.* with 1326 features [8], and our detector is about 3 orders of magnitude faster.

Several extensions are worth pursuing, e.g. to combine the detector – which processes single frames – with a tracker, so that the pedestrian detector can exploit temporal consistency in video data. Also, other features should be considered to serve as additional inputs. Last but not least, the system should be made flexible and self-adaptive enough for it to learn other classes when given the appropriate training data. Car detection work has just started.

References

[1] F. J. Aherne, N. Thacker, and P. Rockett. The bhattacharyya metric as an absolute similarity measure for frequency coded data. *KYBERNETIKA*, 32(4):001–007, 1997.

[2] Burges,C.J.C. A tutorial on support vector machines for pattern recognition data mining and knowledge discovery. 1998.

[3] T. Evgeniou, M. Pontil, C. Papageorgiou, and T. Poggio. Image representations for object detection using kernel classifiers, 2000.

[4] R. Fransens, V. Depoortere, and J. D. Prins. Boundary based feature selection. Technical report, KU. Leuven, 2002.

[5] O. . Freund,R. and Girosi,F. Training support vector machines: an application to face detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[6] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. *CVPR '97*, June 1997.

[7] E. Osuna and F. Girosi. Reducing the run-time complexity of support vector machines, 1998.

[8] C. Papageorgiou, T. Evgeniou, and T. Poggio. A trainable pedestrian detection system. *Procs IEEE Intelligent Vehicles Symposium '98*, pages 241–246, Oktober 1998.

[9] B. Silverman. Density estimation for statistics and data analysis. *Chapman and Hall*, 1986.

[10] P. Viola and M. Jones. Robust real time object detection. *Cambridge Research Laboratory*, February 2001.

[11] F. Yoav and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory: Eurocolt 95. Springer-Verlag*, 1995.

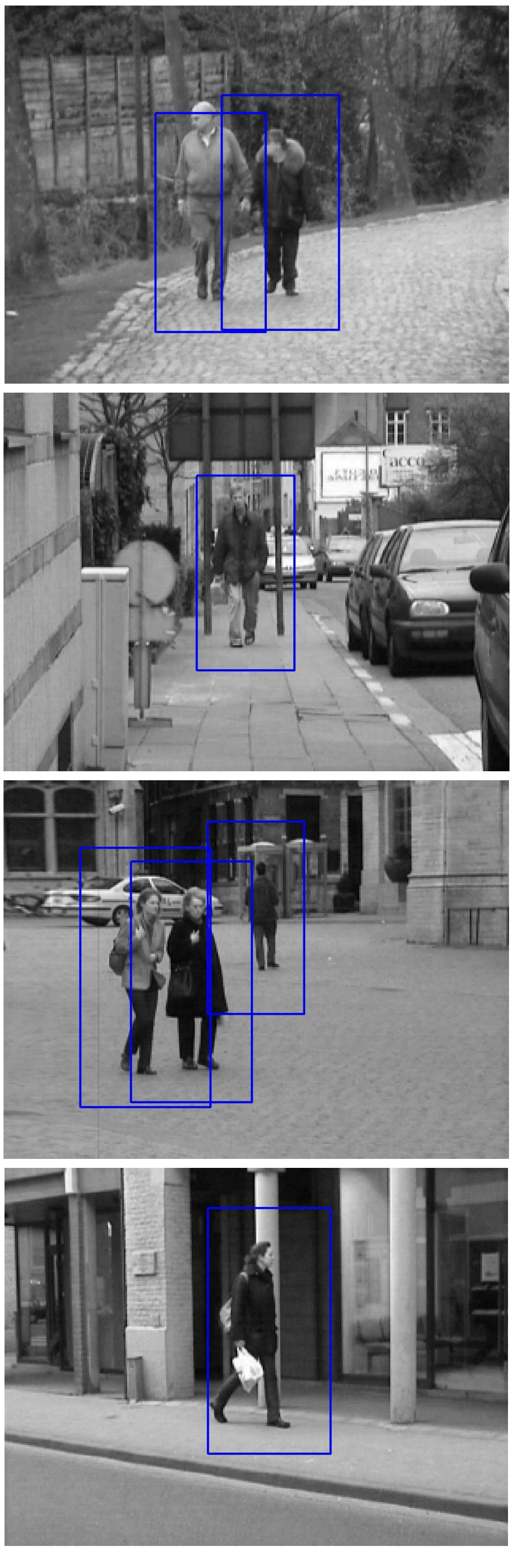


Figure 11. *Some examples of correct detections. Notice the cluttered city scenes, and the different poses of the pedestrians.*

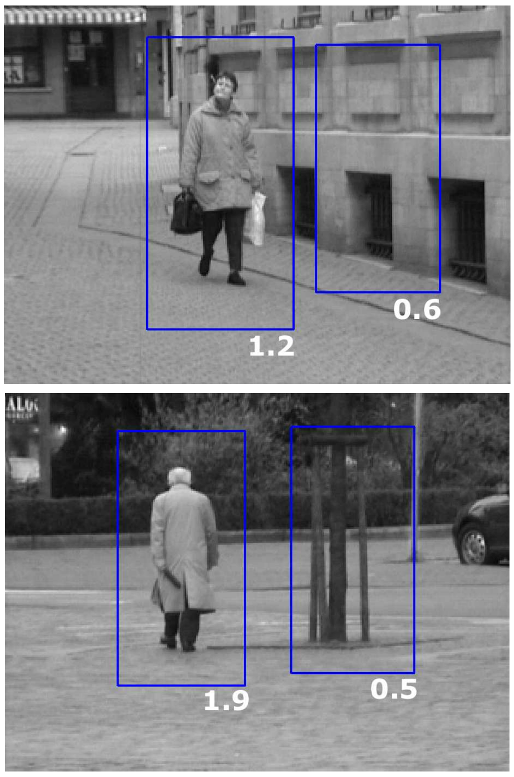


Figure 12. *Some examples of false detections. It seems that the system can be misled by symmetric vertical structures. The values, returned by the last classifier in the stage are depicted. Notice the higher values for the pedestrians.*

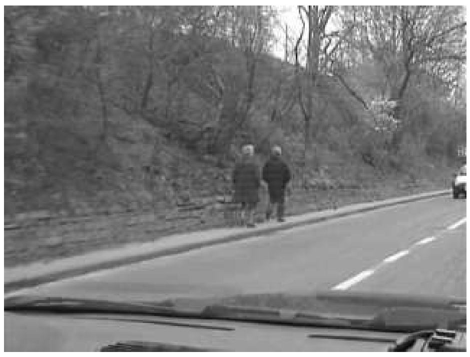


Figure 13. *An example of two pedestrians, the system wasn't able to find. Notice the poor contrast against the background.*