BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

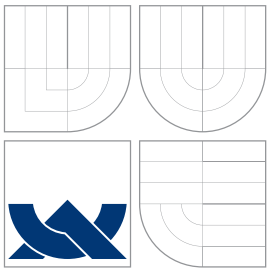
SHARING LOCAL INFORMATION FOR FASTER SCANNING-WINDOW OBJECT DETECTION

DOCTORAL THESIS
DISERTAČNÍ PRÁCE

AUTHOR
AUTOR PRÁCE

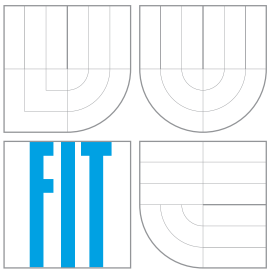
Ing. MICHAL HRADIŠ

BRNO 2014



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ



FACULTY OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

SHARING LOCAL INFORMATION FOR FASTER SCANNING-WINDOW OBJECT DETECTION

SDÍLENÍ LOKÁLNÍ INFORMACE PRO RYCHLEJŠÍ DETEKCI OBJEKTŮ

DOCTORAL THESIS

DISERTAČNÍ PRÁCE

AUTHOR — Ing. MICHAL HRADIŠ

AUTOR PRÁCE

SUPERVISOR — Prof. Dr. Ing. PAVEL ZEMČÍK

VEDOUCÍ PRÁCE

BRNO 2014

Abstrakt

Cílem této dizertační prace je vylepšit existující detektory objektů pomocí sdílení informace a výpočtů mezi blízkými pozicemi v obraze. Navrhuje dvě metody, které jsou založené na Waldově sekvenčním testu poměrem pravděpodobností a algoritmu WaldBoost. První z nich, *Early non-Maxima Suppression*, přesunuje rozhodování o potlačení nemaximálních pozic ze závěrečné fáze do fáze vyhodnocování detektoru, čímž zamezuje zbytečným výpočtům detektoru v nemaximálních pozicích. Metoda *neighborhood suppression* doplňuje existující detektory o schopnost zavrhnout okolní pozice v obraze. Navržené metody je možné aplikovat na širokou škálu detektorů. Vyhodnocení obou metod dokazují jejich výrazně vyšší efektivitu v porovnání s detektory, které vyhodnocují jednotlivé pozice obrazu zvlášť. Dizertace navíc prezentuje výsledky rozsáhlých experimentů, jejichž cílem bylo vyhodnotit vlastnosti běžných obrazových příznaků v několika detekčních úlohách a situacích.

Abstract

This thesis aims to improve existing scanning-window object detectors by exploiting information shared among neighboring image windows. This goal is realized by two novel methods which are build on the ideas of Wald's Sequential Probability Ratio Test and WaldBoost. *Early non-Maxima Suppression* moves non-maxima suppression decisions from a post-processing step to an early classification phase in order to make the decisions as soon as possible and thus avoid normally wasted computations. *Neighborhood suppression* enhances existing detectors with an ability to suppress evaluation at overlapping positions. The proposed methods are applicable to a wide range of detectors. Experiments show that both methods provide significantly better speed-precision trade-off compared to state-of-the-art WaldBoost detectors which process image windows independently. Additionally, the thesis presents results of extensive experiments which evaluate commonly used image features in several detection tasks and scenarios.

Klíčová slova

Detekce objektů, AdaBoost, WaldBoost, EnMS

Keywords

Object detection, AdaBoost, WaldBoost, EnMS, neighborhood suppression, scanning-window

Bibliographic citation

Michal Hradiš: *Sharing local information for faster scanning-window object detection*, doctoral thesis, Brno, Brno University of Technology, Faculty of Information Technology, 2014.

Sharing local information for faster scanning-window object detection

Declaration

I declare that this dissertation thesis is my original work and that I have written it under the guidance of Prof. Dr. Ing. Pavel Zemčík. All sources and literature that I have used during my work on the thesis are correctly cited with complete reference to the respective sources.

.....

Michal Hradiš

January 6, 2015

Acknowledgment

I would like to thank Pavel Zemčík, Adam Herout, and Roman Juránek for being excellent colleagues. The numerous discussions we had gave me important ideas and provided me with needed inspiration. I would like to thank all members of Department of Computer Graphics and Multimedia for creating a productive academic environment which I was delighted to be part of.
Very special thanks goes to my life partner Katka for being tolerant and for supporting me in my work, and to my family.

This work has been supported by the Technology Agency of the Czech Republic Centre of Competence „V3C – Visual Computing Competence Centre“, TE01020415. The infrastructure for the research was supported by the „IT4Innovations“ national supercomputing centre, project no. ED1.1.00/02.0070 under the Operational Programme Research and Development for Innovation funded by the EU.

Contents

List of Figures

List of Tables

Preface

Automatic detection of objects in images is an important task with applications ranging from face detection in hand-held cameras and cloud-based photo collections to general scene understanding and human-machine interaction. Development of practical detectors is a scientific and engineering challenge which combines fields of image processing, machine learning, and often hardware acceleration.

The range of methods for object detection is wide. One particular class of methods scans images with a small scanning-window and tries to determine for each of the windows separately if it contains an object of interest or if it contains background. These methods rely on fast classifiers to make the decisions and on efficient features to extract relevant information from the image windows.

Existing scanning-window detectors are fast and precise, able to detect even small objects in Full HD video in real-time. However, computational resources are still not sufficient in some situations and precision of detection has to be sacrificed for speed.

One drawback of many scanning-window detectors is that they process each image window independently even though they overlap and share lot of common information. In this thesis, I propose to make use of the shared information to improve existing detectors.

I explore the idea of sharing local information and I refine it into two novel practical detection methods. The first method augments existing detectors by an ability to suppress their evaluation at neighboring position in an image. This way, the detector is evaluated fewer times, saving significant computational effort.

The second method relies on the fact that objects cannot occupy the same space in an image. If two objects were too close, a detector would not be able to detect them anyway due to occlusion. This method lets neighboring image positions compete among themselves. It progressively evaluates small parts of a detector at the neighboring positions and gradually reject those positions which will not, with high probability, give the best detection score.

The proposed methods efficiently use the information shared among neighboring image positions, and thus push speed-precision envelope of a range of state-of-the-art detectors. Moreover, the two methods accelerate detection in different parts of an image. The *neighborhood suppression* is effective in background areas while the benefit of letting the detector locally compete improves speed mostly around objects. Because of that, the methods complement each other very well and should provide even greater benefits when combined.

CHAPTER 1

Introduction

This thesis focuses on *scanning-window* object detectors. Specifically, it extends existing detectors to efficiently utilize information shared among neighboring image windows. Theory of *optimal sequential decision making* [142, 124] is extended and combined with *boosting-based* detectors resulting in two practical methods which improve speed of pre-trained detectors by interlinking decisions at neighboring image windows. Both of the methods, *neighborhood suppression* and *Early non-Maxima Suppression* (EnMS), require only unlabeled images as they, in different ways, approximate the responses of the original detectors at the cost of a small and manageable precision reduction. The novel detectors were tested on practical problems demonstrating significantly improved speed-precision trade-off.

Over the years, many approaches to natural object detection [161] have been proposed ranging from simple template matching and hand-designed ad-hoc detectors [54, 73] to *appearance-based* [110, 119, 137, 124, 163, 21, 3] and *part-based* detectors [29, 16, 81, 77, 76, 28, 27, 164, 2]. The methods differ in their strengths and weaknesses; however, the best performing detectors of relatively rigid and visually distinct objects, especially at lower-resolution, are based on appearance-based approaches coupled with sliding-window image scanning. Two examples of such object classes are *faces* [158] and *pedestrians* [23].

Appearance-based detectors rely on discriminative and efficient feature extractors which provide useful information to a classifier deciding between a background class and one, or possibly more, object classes. In the simplest arrangement, the detection classifier considers image windows independently one by one [137] – computing the needed features, evaluating the decision function and outputting a per-window confidence score. The classifier needs to be evaluated very densely in order not to miss objects ($> 90\%$ overlap of adjacent regions at the same scale [18, 137] is

typical).

Such scanning-window methods are simple, but computationally expensive due to the large number of evaluated image windows. Most detectors improve speed by employing classifiers with an *attentional structure* [137, 152, 85, 7, 122, 8, 124, 12, 163] which decide very fast on background areas and spend more time at ambiguous positions.

Scanning-window detectors inevitably produce multiple positive responses per object due to combination of the dense scanning grid and their inherent robustness to small transformations. To get meaningful object positions, the confidence scores are aggregated by a simple *non-maxima suppression* [137, 22] or, possibly, by more elaborate methods, such as mean shift [17]. The *non-maxima suppression* assumes a minimum spacing between objects and, in some cases, smoothness of the detector responses.

The assumptions of the non-maxima suppression as well as the overlap of neighboring image windows can both be exploited to speed-up detection. For example, features [120, 18, 77] or their components [22] can be computed for the whole image in a pre-processing step and shared among all classified windows. Recent advances in convolutional neural networks [25, 128] show that even higher-level parts of classifiers can be shared. Chum and Zisserman [16] are able to locally optimize object bounding boxes thanks to the smoothness assumption. Similarly, Lampert et al. [77] find globally optimal bounding boxes using branch-and-bound search. Some coarse-to-fine detectors [106] exploit the minimum object distance assumption.

In general, object detectors often balance a speed-precision trade-off. A clever approximation of a slow detector may lead to a significant speed-up while retaining similar accuracy [140, 21, 3]. Moreover, the spared computational power can be utilized by additional features or more complex classifiers, in turn, improving detection accuracy.

The idea of approximating some aspects of detectors was taken a step further by Šochman and Matas [126] who proposed to approximate any binary detector as a whole by a generic WaldBoost detector which was originally proposed for face detection [124]. In their approach, an existing detector scans unannotated images and produces training examples for the WaldBoost algorithm which then creates a new detector the same way as when learning from hand-labeled data. As no annotation is needed, the approach can be applied even to hand-crafted detectors. Using suitable features, the authors report high speed-ups for Hessian-Laplace and Kadir-Brady interest region detectors without noticeable degradation of detection quality.

The methods proposed in this thesis are partly inspired by the detector emulation work of Šochman and Matas [126]. They are build on top of *Sequential Probability Ratio Test* [143, 143] as WaldBoost is, they emulate existing detectors in order to improve speed, and they only need unlabeled training images. However, the main

goal in this case is to make use of the fact that decisions at neighboring image windows are not independent due to the shared image content and the non-maxima suppression. Moreover, the original detectors are not discarded. Instead, they are preserved and extended only in a way necessary to handle interactions between the neighboring image positions.

1.1 Summary of Contributions

This thesis contributes to the state-of-the-art of appearance-based object detection methods. It explores an idea that existing *scanning-window detectors* [137, 124, 163, 126] could be improved by exploiting dependencies between neighboring image windows. The idea is refined into two novel, practical, and in certain aspects complementary methods which utilize the shared information to improve detectors. Both methods are demonstrated on specific detectors resulting in two practical detection algorithms.

The methods are general and are not limited to any specific type of detectors. The only requirement is that the detectors have to be decomposable into fragments which provide meaningful discriminative information. Exemplar applications presented in this thesis are based on *soft cascade* [124, 8, 12] detectors which satisfy the requirement very well; however, other detectors, such as *detection cascades* [137, 59, 152], *trees*, and multi-object detectors [30, 86, 65, 129, 58], could be considered as well.

Neighborhood suppression. A detection classifier computed at an image window extracts information relevant to other overlapping windows. The *neighborhood suppression* algorithm (Chapter 6) exploits this fact and trains new classifiers to reject neighboring image windows provided they contain background with high confidence. The new classifiers reuse features of an existing detector changing only the classification function. The *neighborhood suppression* can be realized with minimal computational overhead for *soft cascades* and *domain-partitioning weak classifiers* and it can be directly incorporated in existing detection engines requiring only minor modifications. *Neighborhood suppression* was originally published in [155].

Early non-maxima suppression (EnMS). Scanning-window object detection often includes some kind of *non-maxima suppression* which removes overlapping detections with non-maximal responses of the detection classifier. Such suppression decisions are made only after all the classifiers are fully evaluated. EnMS moves the decision to earlier stages of the classifier in order to stop evaluation of the classifiers which would, with high confidence, be rejected by the ordinary non-maxima suppression. Chapter 7 presents the general idea of EnMS together with a practical version of the algorithm which can be applied to *soft cascades*. EnMS is general and can be applied to a wide range of tasks even outside computer vision –

any task which searches for the highest response of a suitable classifier in a group of competing objects. Furthermore, EnMS could be modified to handle multiple classifiers evaluated on a single object. EnMS was originally published in [48].

Additionally, Chapter 4 presents novel evaluation of a number of existing features with WaldBoost [124] detector on several detection tasks. The evaluated features are considered in connection with the proposed algorithms in the corresponding chapters.

1.2 Authorship

Although most of the work presented in this thesis is my own, some parts resulted from a collaboration with my colleagues.

Pavel Zemčík contributed to my work by many of his ideas and consultations. He proposed the first basic principle of *neighborhood suppression* which I refined into a practical algorithm and tested in the experiments presented in this thesis. Also, Pavel Zemčík significantly influenced development of *Local Rank Patterns* and *Local Rank Differences* by our consultations.

Adam Herout proposed the initial idea of *Early non-Maxima Suppression*, and he helped me with some of the related face localization experiments. I formulated the *Conditioned Sequential Probability Ratio Test*, transformed it into a practical *Early non-Maxima Suppression algorithm*, implemented experimental tools, and performed large part of the experiments.

Roman Juránek implemented several parts of the application which I used to produce most of the experimental results [56]. The parts relevant to experiments in this thesis are my own work.

1.3 Text Structure

Chapter 2 shortly overviews existing *boosted scanning-window detectors* and *AdaBoost* learning algorithm [33, 34] which is one of the main components of such detectors. Also, AdaBoost is a building block of WaldBoost [124], *neighborhood suppression*, and EnMS. Chapter 3 introduces the idea of sequential decision making which leads to *Sequential Probability Ratio Test* [143, 143] and WaldBoost. Chapter 4 presents novel evaluation of a number of existing features used in sliding-window detectors. Chapter 5 discusses how existing detectors utilize dependencies between neighboring image windows, thus putting the methods proposed in the next two chapters into a wider perspective. *Neighborhood suppression* and EnMS together with the corresponding experiments and results are presented in Chapter 6 and Chapter 7, respectively. Chapter 8 discusses the experimental results, properties of the proposed methods, and possible applications. Finally, Chapter 9 summarizes the ideas and the findings of this thesis.

CHAPTER 2

Detection with boosted classifiers

The first practical object detector based on boosted classifiers was introduced by *Viola and Jones* [137] in 2001. This frontal face detector achieved an amazing *real-time performance* by combining computationally efficient image filters with a powerful learning algorithm, an attentional structure of the classifier, a good training dataset, and a large amount of training time. This tremendous success encouraged further research of similar approaches and resulted in great number of modifications [84, 90, 86, 152, 89, 91, 129, 7, 59, 122, 123, 37, 95, 139, 8, 124, 163, 39, 40, 79, 63, 55, 58, 97, 160, 12, 10, 68, 78, 22, 147, 126, 158, 21, 155, 144, 48, 23, 49, 66, 3, 83, 4].

The original detector of Viola and Jones is a standard *appearance-based sliding-window* detector which classifies overlapping constant aspect ratio image windows into *background* and *object* classes. A simple *non-maxima suppression* aggregates the raw detection scores into meaningful object positions.

The detector forms a *rejection cascade* (see Figure 2.1) where each *stage* rejects approximately half of background windows while retaining almost all faces. A very low *false positive rate can* be achieved by chaining multiple such rejection stages

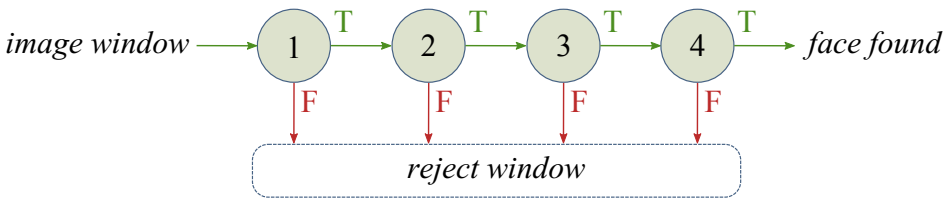


Figure 2.1: The detection cascade [137]. The cascade is composed of a series of increasingly more complex classifiers which either reject the classified sub-window as background or pass it to the subsequent stage. An object is detected only if the corresponding sub-window successfully passes through all of the stages.

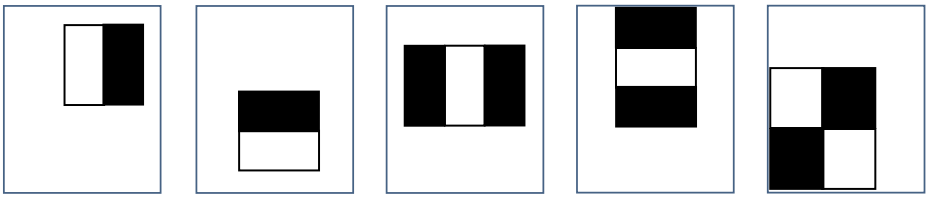


Figure 2.2: [137]Haar-like features used by Viola and Jones in their frontal face detector [137]. Sums of pixels which lie within the white rectangles are subtracted from the sums of pixels in the grey rectangles.

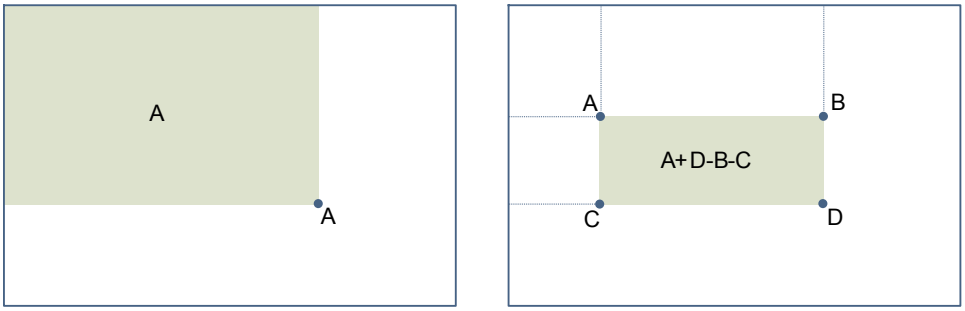


Figure 2.3: Integral image makes it possible to sum pixels within any axis-aligned rectangle by four memory accesses and three additions (subtractions).

without significant increase of computational cost. The computational cost remains low even for long cascades because only few first stages are evaluated on average – natural images contain mostly background which is rejected early in the cascade. Moreover, the classifiers in the early stages tend to be small and efficient as they are deciding very simple problems. On the other hand, later stages of the cascade can easily require hundreds of features to decide with the required confidence.

The stages of the Viola and Jones' detector are *weighted sums of weak classifiers* learned by *AdaBoost* [33, 34]. When the weak classifiers each use only a single feature, AdaBoost effectively performs *greedy forward feature selection* and it is able to create a very compact and fast classifier by picking small highly discriminant set of features from a large pool of features. Additionally, the greedy iterative nature of AdaBoost makes it possible to simply terminate the learning when a required detection rate and false positive rate is achieved.

Haar-like features, used by the detector, were first proposed by Papageoriou et al. [105] as a part of their general framework for object detection. The features are build on simple linear filters derived from *Haar wavelets* [44] which are energy-normalized to improve robustness to contrast changes. The linear filters are composed of several positive or negative axis-aligned adjacent rectangles. Viola and Jones used an immense set of Haar-like features created by shifting and scaling of five basic feature types from Figure 2.2 in 24-by-24 detection window (the total size of the feature pool was 180,000). Haar-like features can be computed very fast and in constant time regardless their size from an intermediate image representation called

the *integral image* (see Figure 2.3). The constant time computation of features makes it possible to scale the detector window instead of scaling the image, accelerating multi-scale detection.

There is no doubt that Viola and Jones created a functional frontal face detector for practical applications; however, the detector is more important due to the follow-up research it initiated then due to its direct use. Since, many authors proposed changes to the detector aiming to improve its performance in general, or in specific situations. All parts of the detector have been carefully considered and analyzed.

Several authors pointed out that the rejection cascade is far from optimal, mainly because it discards all the information accumulated by previous stages when learning a new stage [152, 7, 122, 124, 8, 12].

Large amount of work has been invested in improving the stage classifiers, including applications of advanced boosting algorithms [116, 117, 36, 84, 152, 19, 91, 123, 79] and weak learners [10, 106], improvements to the learning process [68], and even experiments with non-boosting classifiers [148, 95, 47].

Many alternative image features have been proposed [90, 86, 13, 153, 58, 18, 160, 82, 163, 130] varying in their strengths, extracted information, and computational speed. Some of the proposed features address other domains than gray-scale images – e.g. *motion data* [139], *depth data* [104], or *color images* [130, 144]. The type of information most features extract is generally well understood. However; it is usually not clear what information is suitable for a particular detection task, and selection of features mostly relies on empirical evidence from subjectively similar detection tasks. Although some features perform significantly better in some tasks and even enable detection of some objects, no single type of features is optimal for all types of objects and situations. Chapter 4 presents more closely several types of features together with their respective results in real-world detection tasks.

The detector of Viola and Jones is effective only for classes which are visually compact. If it was to be used to detect multiple object classes or multiple views of the same object, multiple detector would have to be used [7] and the detection would become inefficient. Several authors extended the detector to address this issue. Torralba et al. [129] proposed to learn multi-class detectors by *joint boosting* which finds common features that can be shared across the classes. Other proposed approaches include *scalar trees* by Fleuret and Geman [30], Li et al.'s *pyramid* [86], Jones and Viola's *decision tree* [65], and Huang et al.'s *Width-First-Search tree* [58].

Although several implementations of the Viola and Jones' detector have been created for *Field-programmable Gate Arrays* (FPGA) [75, 60, 74, 14] and *Graphics Processing Units* (GPU) [94, 62, 46], the design of the original detector was aimed solely at SISD[1] PC platform and it is not guaranteed to be optimal for other computing platforms with SIMD[2] architecture. Especially in the case of FPGAs,

[1] Single Instruction, Single Data
[2] Single Instruction, Multiple Data

the most efficient designs use different features [157]. Alternative features target GPUs [48] and SIMD CPUs [52] as well. Many state-of-the-art detectors are deployed on GPUs to achieve competitive speeds [3] at only minor development cost thanks to modern languages and tools, such as CUDA.

2.1 AdaBoost

In 1995, Freund and Schapire introduced a novel *boosting* algorithm which they named *AdaBoost* [33, 34]. The term *boosting* refers to a group of ensemble supervised learning algorithms. The basic idea of these algorithms is to iteratively combine relatively simple prediction rules (*weak classifiers or weak hypotheses*) into a very accurate prediction rule (*strong classifier*). In most boosting algorithms, the weak classifiers are linearly combined. For introduction to boosting look at [31, 114].

Boosting has its roots in the PAC (*Probably Approximately Correct*) machine learning model [131, 45]. In this framework, the learner's task is to find – with a high probability – a bounded approximation of a classification function using only training samples which are labelled by this particular function. The PAC model constrains the learning methods in terms of their effectiveness – learning time and size of training set have to be polynomial-bounded. The question, if a learning algorithm which performs just slightly better then random guessing in the PAC model can be boosted into arbitrarily accurate learning algorithm, was first suggested by Kearns and Valiant [70, 71]. The first polynomial-time boosting algorithms were introduced in 1990 by Schapire [113] and Freund [32, 33]. However, the early algorithms suffered from many drawbacks. For example, they needed some prior knowledge of the accuracies of the weak classifiers and the performance bound of the final classifier depended only on the accuracy of the least accurate weak classifier. AdaBoost solved most of these drawback. The significance of AdaBoost is pointed out by many authors. For example, Huang et al. in 2007 wrote :

> Boosting algorithm [34], which linearly combines a series of weak hypotheses to yield a superior classifier, has been regarded as one of the most significant developments in the pattern classification field during the past decade. [58]

AdaBoost. The AdaBoost algorithm is shown in Figure 1. It takes as an input a set of labelled examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ where $\mathbf{x}_i$ are the samples and y_i are the corresponding labels from a set of labels $\mathcal{Y}$. For the purpose of this text $\mathcal{Y} = \{-1, +1\}$, which is different from the originally published version of the AdaBoost algorithm where $\mathcal{Y} = \{0, 1\}$ [33]. However, the version which is presented here is functionally equivalent, more common, and it became a basis to derive many later boosting algorithms.

Algorithm 1 The AdaBoost algorithm as presented in [31].

Input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ where $\mathbf{x}_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Initialize $D_1(i) = \frac{1}{m}$.
For $t = 1, \ldots, T$:

1. Train weak learner using distribution D_t.
2. Get weak hypothesis $h_t : \mathcal{X} \to \{-1, +1\}$ with error
$$\epsilon_t = P_{i \sim D_t}(h_t(\mathbf{x}_i) \neq y_i).$$
3. Choose $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$.
4. Update:
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ if } h_t(\mathbf{x}_i) = y_i \\ e^{\alpha_t} \text{ if } h_t(\mathbf{x}_i) \neq y_i \end{cases} = \frac{D_t \exp(-\alpha_t y_i h_t(\mathbf{x}_i))}{Z_t}$$
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:
$$H(x) = \operatorname{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right).$$

AdaBoost calls a given *weak learning* algorithm repeatedly in a series of iterations $t = 1, \ldots, T$. In each iteration, the weak learning algorithm is supplied with different distribution D_t over the set of examples, and its task is to find a hypothesis $h_t :\to \mathcal{Y}$ minimizing a classification error with respect to the current distribution D_t

$$\epsilon_t = P_{i \sim D_t}(h_t(x_i) \neq y_i). \tag{2.1}$$

The *best weak classifier* is then added to the strong classifier with a coefficient α_t determined by the weighted error ϵ_t of the weak classifier:

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right). \tag{2.2}$$

After the weak classifier is selected and the α_t coefficient is computed, new distribution D_{t-1} is generated in such way that the weights of the samples which are correctly classified by h_t decrease and weights of the wrongly classified samples increase:

$$D_{t+1}(i) = \frac{D_t \exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}. \tag{2.3}$$

Here, Z_t is a normalization factor chosen such that D_{t+1} remains a distribution.

Maintaining the distribution D_t is one of the fundamental principles of AdaBoost.

The weight $D_t(i)$ of sample i reflects how well the sample is classified by all weak classifiers selected in previous rounds.

The final strong classifier is a linear combination of the selected weak classifiers

$$H(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right). \tag{2.4}$$

AdaBoost is guaranteed to eventually reach perfect classification on training data if it is able to find informative weak classifiers ($\epsilon_t < 0.5$) [34]. Most weak learners used in practice always find informative weak hypotheses on finite training sets. The training error ϵ is *exponentially upper-bounded*:

$$\epsilon \leq \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)} \leq \exp\left(-2\sum_{t=1}^{T}\left(\frac{1}{2}-\epsilon_t\right)^2\right) \tag{2.5}$$

AdaBoost can be analyzed in terms of *margins* which are defined in this case as the distance of a sample from the decision boundary normalized by the size of the hypotheses space [31]:

$$\rho_i = \frac{y_i \sum_t \alpha_t h_t(\mathbf{x}_i)}{\sum_t \alpha_t} \tag{2.6}$$

It was proven that classifiers with larger margins on training data generalize better [108].

AdaBoost was first analysed in the context of *margin theory* by Schapire at al. [115]. The analysis provides a generalization bound which is independent of the number of combined weak hypotheses and which is more consistent with empirical results than the original generalization bound from [33]. The bound is linked to margins on training set and it is determined by the training error and an addition term based on VC-dimension [133] of the strong classifier.

Although AdaBoost was shown to create classifiers with large margins, it was proven that the margins are not optimal [112]. Boosting algorithms which maximize margins exist; however, these algorithms are not as practical as AdaBoost and are not used in object detectors.

Real AdaBoost. Schapire and Singer [116, 117] in 1998 generalized the AdaBoost algorithm in a way which removed the restriction of the *binary weak hypotheses*. The authors call this generalization *real AdaBoost* and the original version *discrete AdaBoost*. The authors show that when the weak hypotheses are allowed to take form $h_t :\to R$, it is still possible to find the optimal values α_t minimizing Z_t numerically – by binary search. More importantly, they show that for *domain partitioning weak hypotheses*[3] the α_t value can be incorporated directly into the weak hypotheses and

[3]Domain partitioning weak hypotheses assign each sample a value from a finite set of labels.

the optimal responses of the weak hypotheses are

$$c_j = \frac{1}{2} \ln \left(\frac{W_+^j}{W_-^j} \right), \tag{2.7}$$

where W_+^j and W_-^j are sums of weights of *positive* and *negative* samples assigned to the partition j, respectively. Schapire and Singer also proved that the weak hypotheses should be ideally created such that they minimize

$$Z_t = \sum_i D_t(i) \exp\left(-y_i h(\mathbf{x}_i)\right). \tag{2.8}$$

Further, the authors suggested to smooth the c_j values in case that either W_+^j or W_-^j is very small by

$$c_j = \frac{1}{2} \ln \left(\frac{W_+^j + \epsilon}{W_-^j + \epsilon} \right), \tag{2.9}$$

where ϵ is a small smoothing constant.

The real AdaBoost algorithm is significant, because it provides an efficient way how to use more complex weak hypotheses which partition the domain space into more than two partitions. Such domain partitioning weak hypotheses were shown to be superior to binary weak hypotheses [36, 150, 10].

CHAPTER 3

Sequential analysis in object detection

In object detection using the *sliding-window* technique, the decision at each image position can be regarded as a *statistical hypothesis test* where the *null hypothesis* states that the image patch does not contain an object of interest [124]. The *alternative hypothesis* is that the patch contains an object of interest.

The idea of defining the object detection task as a statistical hypothesis test may be counter-intuitive due to the fact that statistical tests are usually used to decide if an independent sample of a population can be explained by the null-hypothesis or if the sample provides enough evidence to reject the null-hypothesis in favor of some alternative hypothesis.

To make the definition of a statistical test more formal, consider $\mathcal{X}$ to be a random variable for which $p(x|\mathcal{H}_0)$ defines either probability distribution or probability density consistent with the null-hypothesis. Similarly, let the alternative hypothesis be that $\mathcal{X}$ follows distribution $p(x|\mathcal{H}_1)$. For N samples x_i drawn independently from $\mathcal{X}$, the most powerful statistical test [142] can be defined as

$$\frac{p(x_1,\ldots,x_N|\mathcal{H}_1)}{p(x_1,\ldots,x_N|\mathcal{H}_0)} \geq k \quad \equiv \quad \frac{\prod_{i=1}^{N} p(x_i|\mathcal{H}_1)}{\prod_{i=1}^{N} p(x_i|\mathcal{H}_0)} \geq k, \tag{3.1}$$

where k is a constant chosen such that the probability of falsely rejecting the null hypothesis is reasonably low.

In the sliding-window detection, the statistical test decides a single image region at a time from which dependent measurements are taken. Consequently, the test from the left side of 3.1 can be more appropriately written as

$$\frac{p(\mathbf{x}|\mathcal{H}_1)}{p(\mathbf{x}|\mathcal{H}_0)} \geq k, \tag{3.2}$$

were $\mathbf{x}$ is now a vector of features extracted from the single image position. In case the features were independent, the functions $p(\mathbf{x}|H_0)$ and $p(\mathbf{x}|H_1)$ could be factorized into products of marginal distributions of the individual features similarly to the right side of 3.1. Unfortunately, features describing the same object are generally not independent, and should be modeled jointly.

A fully joined model $p(\mathbf{x}|\mathcal{H})$ would be complex, hard to estimate, and computationally expensive. Practical detectors which utilize probabilistic models of background and foreground have to make compromises by omitting some of the dependencies [118, 119].

Sequential statistical test. Motivated by the need for efficient quality control of military supplies during the Second World War, A. Wald [142] defined a *sequential test of a statistical hypothesis* as a procedure which, at any stage of an experiment where samples are drawn *independently and identically distributed* from an unknown distribution, gives a specific rule, for making one of the three decisions: (1) to accept the null hypothesis, (2) to reject the null hypothesis, (3) to continue the experiment by making additional observation. A novel idea of the sequential test was that the number of observations needed to make a decision was not predetermined, rather, the number of observations was threated as a random variable. This made it possible to adjust the number of observations to each particular instance of an experiment, and thus reduce the average number of observations while maintaining the same expected error level. As is shown in the following text, the ideas of sequential statistical testing can be adapted in fast detection classifiers which compute and use only so many features at each image position such that a predetermined error rates are achieved.

3.1 Optimal Sequential Decision Strategy

In the following text, the sequential test is formalized in a way which is suited for a *two-class classification task* as opposed to the Wald's definition [142] for independent samples drawn from an unknown distribution. The formulation here follows formulations in [124, 141].

Sequential decision strategy. Let $\mathbf{x} \in \mathcal{X}$ be a vector of measurements $x_i \in \mathcal{X}_i$ representing an object. The task is to estimate an unknown class $y \in \{-1, +1\}$ associated with the object based on the values x_i. The sequential test can be formalized as a *sequential decision strategy* $S : \mathcal{X} \to \{-1, +1\}$ which is a sequence of *decision functions* $S = S_1, S_2, \ldots$ Each of the decision functions takes one measurement of the object, and makes its decision based on the previously obtained measurements including the new one – formally $S_t : \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_t \to \{-1, +1, \sharp\}$. The decision strategy terminates when a decision function outputs $+1$ or -1. The symbol '$\sharp$' defers the decision to the following function S_{t+1}.

Strength of a sequential decision strategy S is characterized by its *false negative rate* α_S and its *false positive rate* β_S

$$\alpha_S = P(S(\mathbf{x}) = -1|y = +1) \quad \text{and} \quad \beta_S = P(S(\mathbf{x}) = +1|y = -1). \tag{3.3}$$

Second important characteristic of a sequential decision strategy is its *speed* which is expressed as the *number of measurements needed to reach a decision*. This number is a random variable and it will be further denoted as N_S. The average number of measurements

$$\bar{T}_S = E\left[N_S\right], \tag{3.4}$$

depends on the object class. The average number of measurements for the two classes will be denoted as

$$\bar{T}_{S,-1} = E\left[N_S|y = -1\right] \quad \text{and} \quad \bar{T}_{S,+1} = E\left[N_S|y = +1\right]. \tag{3.5}$$

A sequential decision strategy S^* is considered to be best [142] or *evaluation-time-optimal* [141] if it provides the lowest $T_{S*,-1}$ and $T_{S*,+1}$ compared to any other decision strategy of equal strength – of those decision strategies that have equal *false negative rate* α_S and *false positive rate* β_S.

Sequential Probability Ratio Test. A. Wald [142] proposed a Sequential Probability Ratio Test (SPRT) which for practical purposes can be considered an *evaluation-time-optimal* sequential decision strategy. In his own words:

> ... for the so called Sequential Probability Ratio Test ... both $T_{S,-1}$ and $T_{S,+1}$[1] are very nearly minimized. Thus, for all practical purposes the Sequential Probability Ratio Test can be considered best. [142]

SPRT is defined as a sequential strategy S^* where

$$S_t^*(\mathbf{x}) = \begin{cases} +1, & \text{if } R_t(\mathbf{x}) \leq B \\ -1, & \text{if } R_t(\mathbf{x}) \geq A \\ \sharp, & \text{if } B < R_t(\mathbf{x}) < A \end{cases} \tag{3.6}$$

where $R_t(\mathbf{x})$ is a likelihood-ratio of the two competing hypotheses:

$$R_t(\mathbf{x}) = \frac{p\left(x_1, \ldots, x_t|y = -1\right)}{p\left(x_1, \ldots, x_t|y = +1\right)}. \tag{3.7}$$

The constraints A and B determine error rates α and β of the test. Fining A and B to give exactly the required α and β is rather tedious and not suitable for practical purposes. Instead, Wald [142] suggest A and B to be set to their upper

[1] The notation here was changed to match notation used in this work.

and lower bounds, respectively:

$$A = \frac{1-\beta}{\alpha}, \quad B = \frac{\beta}{1-\alpha}. \tag{3.8}$$

Setting A and B this way may increase at most one of the resulting error probabilities α' and β'. Wald [142] showed that the potential increase of one of the errors is extremely small.

Note that SPRT does not constrain the conditional class distributions in any way. In fact, the likelihood-ratios $R_t(\mathbf{x})$ could be even estimated directly without modeling the class-conditional distribution, e.g. by logistic regression.

3.2 WaldBoost

In order for SPRT to be efficient in a classification task where the measurements are *not independent and identically distributed* (non-i.i.d.), the decision functions (Equation 3.6) have to be evaluated very fast. Ideally, the decision functions should incorporate the new measurements in a computationally simple way which does not depend on the number of measurements taken so far. This would be very hard to achieve if the joined class-conditional densities or the likelihood ratios (Equation 3.7) would have to be actually estimated. Additionally, the *order of measurements* matters in the non-i.i.d. case. The first measurements taken should be those most informative, as those allow to accumulate enough evidence about the decision problem as early as possible, thus reducing average number of measurements needed.

Šochman and Matas proposed *WaldBoost* [124] which avoids computation of the likelihood ratios by projecting the classified objects to a scalar value using a *discriminatively trained classifier*, and by reformulating the decision functions accordingly in a way which directly thresholds output of the classifier.

The authors suggest to use *real AdaBoost* [117] as the classifier. AdaBoost is especially suitable for the task as it can naturally *choose and order the measurements* according to their discriminative power (when weak classifiers each use only single feature, see 2.1). Moreover, the resulting strong classifier is a sum of the weak classifiers which makes computational cost of incorporating additional measurement into the classifier's output constant and independent of the number of previous measurements.

Decision functions for classification. Let $H_t(\mathbf{x})$ be a real-valued output of a classifier incorporating features $1, \ldots, t$, the likelihood ratio R_t (3.7) is reformulated as

$$R_t(\mathbf{x}) = \frac{p\left(H_t(\mathbf{x})|y=-1\right)}{p\left(H_t(\mathbf{x})|y=+1\right)}. \tag{3.9}$$

Assuming[2] the likelihood ratio is a monotonic function of $H_t(\mathbf{x})$, the decision functions (3.6) can be equivalently redefined such that the decision conditions compare the classifier output instead of the likelihood ratio:

$$S_t^*(\mathbf{x}) = \begin{cases} +1, & \text{if } H_t(\mathbf{x}) \geq \theta_B^{(t)} \\ -1, & \text{if } H_t(\mathbf{x}) \leq \theta_A^{(t)} \\ \sharp, & \text{if } \theta_A^{(t)} < H_t(\mathbf{x}) < \theta_B^{(t)} \end{cases} . \tag{3.10}$$

The *thresholds* $\theta_A^{(t)}$ and $\theta_B^{(t)}$ have to be estimated on a suitable dataset such that the conditions are equivalet to the corresponding conditions using $R_t(\mathbf{x})$ (3.6). This could be achieved by estimating the class-conditional densities $p\left(H_t(\mathbf{x})|y=-1\right)$ and $p\left(H_t(\mathbf{x})|y=+1\right)$ by some standard procedure, e.g. histogram, Gaussian Mixture model, or kernel density estimation. In the original WaldBoost paper [124], the authors suggest using *Parzen window kernel density estimator* with the size of a Gaussian kernel set according to an oversmoothing rule [121]. However, such approach poses practical problems. Note that the required false negative rate α_S and false positive rate β_S are usually low values which makes A a large value and B close to zero (see Equation 3.8). If a decision function decides 10% of objects as the class -1, then at most 0.2% (assuming $\alpha_S = 0.02$) of the positive class distribution mass lies in the decided region. Only 0.2% of positive examples from a training set would be in the decided region, making density estimation problematic on such small number of samples.

Šochman [141] suggested to avoid the problems with estimation of $p\left(H_t(\mathbf{x})|y=-1\right)$ and $p\left(H_t(\mathbf{x})|y=+1\right)$ by treating $H_t(\mathbf{x})$ as a step function with discontinuities at $\theta_A^{(t)}$ and $\theta_B^{(t)}$. Such change transforms the continues density estimation into a discrete estimation with three bins. As a result, the thresholds should be set as strict as possible while satisfying [141]:

$$\sum_{\left\{\mathbf{x}:H_t(\mathbf{x})\leq\theta_A^{(t)}\right\}} p\left(H_t(\mathbf{x})|y=-1\right) \;\geq\; A \sum_{\left\{\mathbf{x}:H_t(\mathbf{x})\leq\theta_A^{(t)}\right\}} p\left(H_t(\mathbf{x})|y=+1\right) \tag{3.11}$$

respective

$$\sum_{\left\{\mathbf{x}:H_t(\mathbf{x})\geq\theta_B^{(t)}\right\}} p\left(H_t(\mathbf{x})|y=-1\right) \;\geq\; B \sum_{\left\{\mathbf{x}:H_t(\mathbf{x})\geq\theta_B^{(t)}\right\}} p\left(H_t(\mathbf{x})|y=+1\right). \tag{3.12}$$

[2]The assumption of monotonicity may be partially violated in some cases due to wrong assumptions about the types of class distributions or due to low representativeness of the training dataset; however, it generally holds and the deviations do not hamper practical applications of WaldBoost except for possible decrease in decision speed. In fact, WaldBoost does not require R_t to by monotonic function of $H_t(x)$ in order to work properly – it just may become less efficient.

The previous conditions can be rewritten in a more simple form as

$$p\left(H_t(\mathbf{x}) \leq \theta_A^{(t)} | y = -1\right) \;\geq\; Ap\left(H_t(\mathbf{x}) \leq \theta_A^{(t)} | y = +1\right) \tag{3.13}$$

and

$$p\left(H_t(\mathbf{x}) \geq \theta_B^{(t)} | y = -1\right) \;\geq\; Bp\left(H_t(\mathbf{x}) \geq \theta_B^{(t)} | y = +1\right). \tag{3.14}$$

These constraints are based on the probabilities that a sample of a certain class is from one of the decided regions. These probabilities are much easier to estimate.

WaldBoost classifier. The classification functions $H_t(\mathbf{x})$ in WaldBoost are sums of weak classifiers $h_t(\mathbf{x})$ as in real AdaBoost (see Section 2.1). A WaldBoost classifier is defined by an ordered set of T *weak classifiers* $h_t(\mathbf{x})$, by the *corresponding thresholds* $\theta_A^{(t)}$ and $\theta_B^{(t)}$, and by the *final threshold* γ which is applied to the full classifier response $H_T(\mathbf{x})$ if a decision is not reached earlier. The final threshold controls operating point of the WaldBoost classifier only to a small extent – most samples are usually decided before the final stage.

The classification algorithm is shown in Algorithm 2. It successively applies the decision functions. Each of the functions computes a response of its weak classifier $h_t(x)$ and adds it to the cumulative result of the previous decision function $H_{t-1}(\mathbf{x})$ to obtain $H_t(\mathbf{x})$. Subsequently, the decision conditions using $\theta_A^{(t)}$ and $\theta_B^{(t)}$ are evaluated. If the decision function does not reach a conclusion, the classification algorithm continues with the next decision function. Finally, the output of the classifier is thresholded by γ.

Algorithm 2 WaldBoost classification [141]

Given: h_t, $\theta_A^{(t)}$, $\theta_B^{(t)}$, and γ for $t \in \{1, \ldots, T\}$
Input: a classified object $\mathbf{x}$
For $t = 1, \ldots, T$:

1. If $H_t(\mathbf{x}) \geq \theta_B^{(t)}$, classify $\mathbf{x}$ to the class $+1$ and terminate.
2. If $H_t(\mathbf{x}) \leq \theta_A^{(t)}$, classify $\mathbf{x}$ to the class -1 and terminate.

end
If $H_t(\mathbf{x}) > \gamma$, classify $\mathbf{x}$ to the class $+1$, -1 otherwise.

WaldBoost learning for object detection. The complete WaldBoost learning algorithm is shown in Figure 3. It accepts as an input a large set of training examples P, desired error rates α and β, and a number of training iterations T. The output is a sequential decision strategy represented by an ordered set of weak classifiers $h_t(x)$, $t \in \{1, \ldots, T\}$ and the corresponding decision thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$. The algorithm extends *real AdaBoost* (see Section 2.1) by *bootstrapping* (or sampling of the training

set) and by the *decision thresholds*.

A weak classifier is learned in each iteration of WaldBoost as in real AdaBoost. It can be selected on a set of examples $\mathcal{T}$ sampled from P [68]. The sampled set $\mathcal{T}$ changes in each iteration and the weights have to be computed accordingly. The decision thresholds are then set such that they satisfy the constraints from Equation 3.13 and Equation 3.14 on the full training set P which is in turn pruned by the thresholds.

The bootstrapping is necessary as the training set is pruned very efficiently and only a small fraction of it remains in later iterations. In order to retain representative training set in the later iterations, the initial number of examples would have to be impractically large without the bootstrapping (it would considerably slow down the learning without measurable impact on the quality of weak hypotheses).

Algorithm 3 WaldBoost learning with bootstrapping. [141]

Input:

- sample pool $\mathcal{P} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}\,; \mathbf{x}_i \in \mathcal{X}, y_i \in \{-1, +1\}$
- desired final false negative rate α and false positive rate β
- the number of iterations T

Set $A = \frac{(1-\beta)}{\alpha}$ and $B = \frac{\beta}{1-\alpha}$
Initialize data weights $w_1(\mathbf{x}_i, y_i) = \frac{1}{N}$
For $t = 1, \ldots, T$:

1. Sample training set $\mathcal{T} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ from $\mathcal{P}$
2. Find $h_t(\mathbf{x})$ by real AdaBoost algorithm on training set $\mathcal{T}$ with weights w_t and compute new weights
3. Find decision thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$ such that eq. 3.13 and 3.14 hold
4. Throw away samples from $\mathcal{P}$ for which $H_t(\mathbf{x}) \geq \theta_B^{(t)}$ or $H_t(\mathbf{x}) \leq \theta_A^{(t)}$

end
Output: Weak classifiers $h_t(x)$ and decision thresholds $\theta_A^{(t)}$ and $\theta_B^{(t)}$

CHAPTER 4

Features and object detection

This chapter overviews basic *features* used in *appearance-based detectors* and presents experimental results of six representative feature types in real-world tasks. The experiments provide general insights into the behavior of these features in context of *boosted detectors* and put results presented throughout this thesis in the context of state-of-the-art methods. The features, training and evaluation methodology, datasets, and even some of the detectors from this chapter are further used in *EnMS* and *neighborhood suppression* experiments.

The purpose of features is to *extract useful information* from data in a *computationally efficient way*. Although it is possible to create a classifier directly on the raw image data, features potentially make the learning task much easier and they can be highly optimized for speed. Through features, the designer can express his *prior knowledge* of data, objects, and desirable invariances. For example, image features for object detection often reflect frequency properties of images, correlation of pixels, desire for shift and lighting invariance, or knowledge about distinguishing attributes of the objects. Although the recent development is shifting towards general learning methods which do not rely on features, such as *deep convolutional neural networks* [25, 128, 80], hand-designed features are still the core building block of majority of detectors. Many features for scanning-window object detection have been proposed varying in their strengths and weeknesses. In general, different features are suitable in different contexts and for different tasks. It is beneficial for anyone designing a new detector to have an idea what features he can choose from and what are their properties.

Existing features. Since the frontal face detector of Viola and Jones [137], *Haar-like feautres* have been extended to include 45° *rotated regions* by Lienhart and

Maydt [90], and similarly by Jones and Viola [65]. Li et al. [86] extended the features further by *relaxing the strict adjacency* of the rectangles composing the features. Viola et al. [136, 139] further extended Haar-like features to encode *motion information* for pedestrian detection.

The variants of Haar-like features are all linear filters which are normalized in order to improve robustness to illumination changes. Other linear filters used in object detection include *Gabor filters* [13], *Anisotropic Gaussian filters* [97], and various *wavelets* [119].

Although the fixed linear functions of predefined filters are designed to fit well the general frequency properties of images, they are not in any way adapted to the target detection task. In a response to that, many authors have tried to adapt linear features to particular tasks using *Principal Component Analysis* [118], *Fisher Discriminant Analysis* [145], *recursive nonparametric discriminant analysis* [145], *local non-negative matrix factorization* [87], *local receptive fields* [98, 38], *neural networks* [24], and other methods [58].

Fröba and Enst [37] aimed to improve *illumination invariance* with features based on *modified census transform*. Similarly, other authors used *Local Binary Patterns* [64, 160, 99, 67, 72], *Local Rank Patterns* [156, 57, 107, 52, 51, 48, 157], and *Locally Assembled Binary features* [154] which all discard illumination information.

Another group of successful features is based on regional statistics, such as histograms. These include *local edge orientation histograms* [82], *Histograms of Oriented Gradients* [18, 163, 39, 55, 78, 24, 127, 3, 4], *spectral histogram* features [149], and *spatial histograms* [159].

4.1 Selected features

The features selected for experiments are only a very small subset of features that have been proposed for object detection. They do not even represent all existing feature families. For example, all the selected features form a *finite set of functions* from which the boosting algorithm can select by *exhaustive search* – ignoring *adaptive features* which have to be optimized during or before learning to the detection task at hand, such as *local receptive fields* [98, 38], PCA, and ICA. Also, domain of all selected features is *gray-scale* images. Event though the selection is limited, it still represents a large fraction of features used in face, body part, and pedestrian detection. The selected features are summarized in Table 4.1.

The selected features are *Haar-like features*, *Local Binary Patterns* (LBP), *Histograms of Oriented Gradients* (HOG), *Local Rank Differences* (LRD), and *Local Rank Patterns* (LRP).

Haar-like features serve as a baseline and reference due to their wide adoption and long history in object detection. *Local Binary Patterns* are in many ways

	Haar	LBP	LRD	LRP	EHOG
positions	all	all	all	all	even
scales	all	all	all	all	even
types	6 basic types	3x3 grid	3x3 grid	3x3 grid	rectangular
# features	141,600	8464	304,704	304,704	70,227
# bins	10	256	80	17	10

Table 4.1: Details of features selected for experiments. The numbers of features are for base resolution of classifiers 24-by-24 pixels.

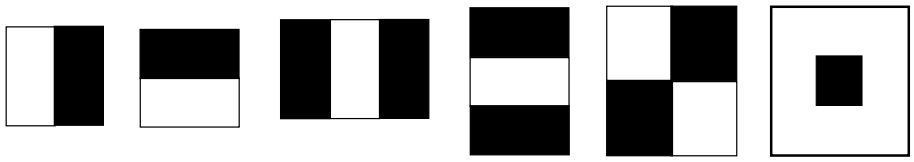


Figure 4.1: Haar-like features used in experiments. The first five feature shapes are the same as those used by Viola and Jones [137]. The last center-surround shape was first used by Lienhart and Maydt [90].

complementary to Haar-like features as they encode very different information. They are invariant to locally uniform illumination changes as they encode only shape of intensity surfaces and discard magnitude of changes.

Histograms of Oriented Gradients can not be omitted due to their success in *pedestrian detection* and their wide adoption outside the field of boosted detectors. HOG are locally normalized and invariant to translations of parts within the region of the feature. They are in between Haar-like features and LBP in terms of what information they extract – they describe local shape more weakly than LBP and they still, to an extent, reflect magnitude of local changes.

The previous features, which were all originally designed for serial processing on CPUs, are further complemented by *Local Rank Differences* and *Local Rank Patterns* which were originally designed specifically for parallel computation platforms, such as FPGA and GPU. Similarly to LBP, LRP and LRD are invariant to locally uniform illumination changes, but unlike LBPs, which capture local shape in a single complex descriptor, LRD and LRP can focus on various aspects of the local shape independently and describe them in a more compact way.

The sets of features as defined in this section are denoted as *Haar*, *LBP*, *LRD*, *LRP*, *EHOG*, and *EHOGS* in the further text.

Haar-like features. Haar-like features are simple linear filters derived from Haar wavelets [44] normalized to improve robustness to illumination changes. The features were first used by Papageorgiou et al. [105] and were made popular by the frontal face detector of Viola and Jones [137]. Since, the features were used in many detectors [152, 138, 95, 151, 59, 122, 123, 124, 8, 19, 98, 12, 10, 126, 155], various extensions were proposed [90, 84, 86, 65, 136, 139], and efficient detection engines

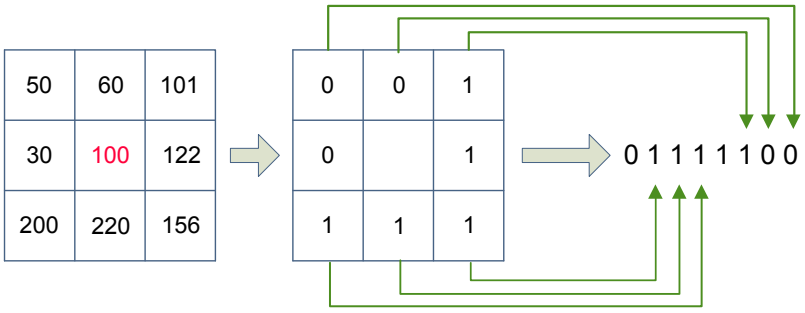


Figure 4.2: Local Binary Patterns as defined by Ojala and Pietikäinen [101]. In a 3-by-3 pixel area, the outer pixels are thresholded by the central value and the resulting ones and zeros are serialized into an 8-bit code.

using these features were implemented for a wide range of platforms, such as GPU [46, 62, 94] and FPGA [75, 14, 60, 74].

The basic Haar-like features are linear filters composed of several positive or negative axis-aligned adjacent rectangles. The filters have *zero sum* and no response to DC image component which makes them *invariant to shifts in gray scale*. However, responses of the filters are not invariant to multiplicative changes caused by varying illumination intensity. In order to make the features robust to such illumination changes, the output is typically scaled by an inverse of a local measure of energy, e.g. standard deviation of pixel values in the analyzed image window [137, 124]. The Haar-like features provide a normalized measure of a presence of simple shapes, such as edges, ridges, corners, and blobs.

The set of Haar-like features used in this thesis consists of the six basic types shown in Figure 4.1 – horizontal and vertical edge, horizontal and vertical line, diagonal line, and center-surround shape. The prototypes were *shifted* and *scaled* inside a detector area at its base scale to generate the whole feature set. The features were shifted by one pixel in horizontal and vertical direction, and the sizes of the rectangular regions were increased by single pixel at a time keeping size of all rectangles composing the feature the same.

Originally, the Haar-like features were used together with *threshold weak hypotheses* [137] (also decision stump weak hypotheses). However, later works show that better results can be achieved with slightly more complex weak hypotheses, such as *small decision trees* [9] and *piece-wise functions* [59]. The weak hypotheses in this thesis are *piece-wise functions* where boundaries of the left-most and right-most bins are set such that each contains 5% of training examples, and the interval in-between is separated into 8 more bins.

Local Binary Patterns. LBP were originally proposed as a texture analysis operator [101] which provides information of local image structure *invariant to monotonic changes in gray-scale*, and which can be optionally made partially *invariant to rota-*

tion [102]. LBP operator was used in many practical applications mostly connected to *static texture analysis* [101, 102, 100, 103] and *dynamic texture analysis* [162]. Other successful applications include *face recognition* [1, 53] and *authentication* [109], facial *expression recognition* [88], and *palm-print identification* [146]. In object detection, variants of LBP provide good results with *boosted classifiers* [160, 147, 155] and *random forests* [69]. Efficient detection engines using LBP were designed for GPU [99, 67], FPGA [72], and SIMD [67] architectures.

LBP create a binary code by thresholding a small *circular neighborhood* by the value of its centre (see Figure 4.2). In the original definition of LBP [101] the neighborhood was a 3-by-3 pixel area and values were taken from centers of the pixels. Mäenpää and Pietikäinen [93] extended the neighborhood to arbitrary circular shape with interpolation providing values from sub-pixel positions. The precise circular shape allows *rotational invariant* version of LBP which is important in may texture recognition tasks; however, it is not suitable for object detection as the detectors themselves are usually not rotational invariant.

Inspired by texture analysis applications of LBP, some object detectors rely on *histograms* of LBP responses which provide partial *translation invariance*. However, many objects are not defined by textures, but rather, by *distinct features* and *shapes*, and the translational invariance is not needed for rigid objects, such as faces. For use in boosted object detectors, Zhang et al. [160] proposed *Multi-Block LBP* (MB-LBP). The shape of the neighborhood of MB-LBP has the same 3-by-3 shape as the original LBP [101] (see Figure 4.2). The difference is that MB-LPB can scale independently in horizontal and vertical direction and the thresholded values are *sums of pixel values* inside corresponding grid cells. MB-LBP can be used as domain partitioning features for *real AdaBoost*.

The experiments in this thesis use MB-LPB features at all positions of the base scale of the detectors and at all possible scales. The produced 8-bit codes directly indicate one of 256 possible partitions in a weak hypothesis.

Histograms of Oriented Gradients. Various versions of *Histograms of Oriented Gradients* are widely used as a basis for description of local image patches (e.g. in local descriptors SIFT, SURF, GLOH, ...). Such descriptors provide state-of-the-art results in object class recognition [132], semantic class detection [26], wide-baseline stereo, content-based image retrieval [15], object detection [6, 83], and other tasks. HOG itself proved to be well suited for part-based object detectors [28, 27] and rigid appearance-based detectors, especially for pedestrian detection task [18, 17, 163, 55, 22, 144, 3, 4].

Histograms of Oriented Gradients, as the name suggests, locally compute *histograms* of *gradients* of image function. The gradients can be computed in any number of ways, e.g. from unsmoothed partial differences in x and y directions. The histograms are then created by accumulating *magnitudes* of the gradients in a

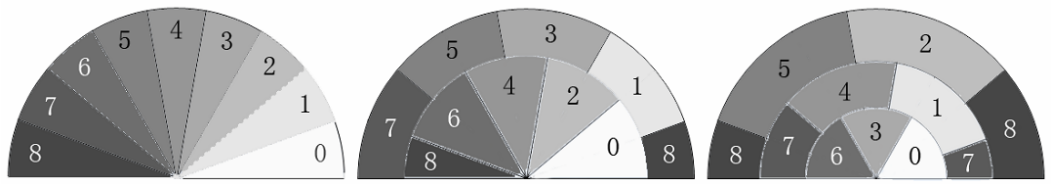


Figure 4.3: Illustration of dominant orientation of *Extended Histograms of Oriented Gradients* (EHOG) by Hou et al. [55]. Each fan represents encoding of dominant orientations with certain angular selectivity. From left: single orientation bin, merging two bins, and merging three bins. Taken from [55].

local region into bins corresponding to *orientation* of the respective gradient. The exact details of HOG features are not unified and can differ significantly, e.g. in the assignment of gradients to bins, normalization of histograms, shape of the HOG regions, arrangement of basic HOG cells into larger blocks.

The HOG descriptor is invariant to *translations* within its domain and expresses *dominant direction* of edges inside the region. As the histograms are usually *normalized* to unit length [18], HOG provides local shape information rather than information about magnitude or strength of the shape with respect to the rest of the image.

The specific type of HOG used in this thesis is closely inspired by the work of Hou et al. [55], specifically by the *Extended Histograms of Oriented Gradients* (EHOG). EHOG compute a histogram of gradients from an image region similarly to HOG, and normalize it to unit L1 norm. The outputs of EHOG are then the so-called *dominant orientations* which sum together up to three adjacent histogram bin values (see Figure 4.3 for illustration of dominant orientations). Because of the L1 normalization, the dominant orientations represent relative strength of gradients in a specific direction with higher or lower angular selectivity depending on how many bins are summed. The scalar dominant orientations can be used by simple weak learners the same way as, for example, Haar-like features. Additionally, Hou et al. propose a *heuristic search* strategy which is able to find discriminative EHOG with *non-rectangular shapes*.

The EHOG features used in this thesis compute gradients by differentiating neighboring pixels at the base scale of the respective detector. The gradients are accumulated to the nearest angular bin and the histograms are L1 normalized. The features are limited to *rectangular shapes* of any position, size, and aspect ratio, provided they completely fit inside the detector window, and *coordinates* and *width* and *hight* of the feature at the base scale of the classifier are *even* (2601 such rectangles fit into 24-by-24 detector window).

The scalar dominant orientations are discretized the same way as the responses of Haar-like features (described above).

The experiments in this thesis differentiate two versions of EHOG, denoted as

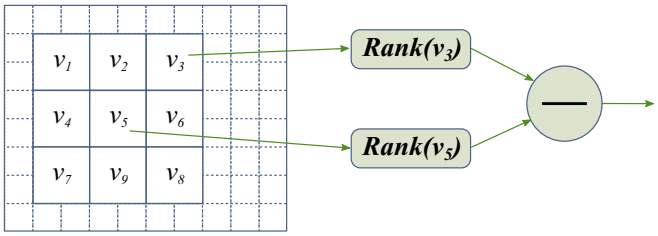


Figure 4.4: Local Rank Differences compute ranks of two local values and subtracts them.

EHOG and *EHOGS*, which differ only in the way they allocate gradients to angular bins. As in the original work of Hou et al. [55], both types use 9 bins; however, *EHOG* distinguish polarity of the gradients – the histogram bins cover whole 360° circle. On the other hand, the *EHOGS* features are invariant to intensity inversion, and the bins cover only 180° half-circle similarly to the original features.

Local Rank Differences and Local Rank Patterns. Both LRD [52] and LRP [57] were designed specifically for parallel computational platforms, such as FPGAs and GPUs, as an alternative to traditional CPU features. Since, efficient detection engines using these features were developed for GPU [107, 51, 48], FPGA [156, 157], and SIMD CPU [52].

LRD and LRP rely on a *rank transform* of several values extracted form a local image neighborhood. A proposed practical version, which is used in this thesis, sums pixels in cells of a 3-by-3 axis-aligned grid (see Figure 4.4. LDR subtract ranks of two grid cells, while LRP index a 2D lookup table by the two ranks. Considering the outputs of LRD and LRP are discrete values, they are used directly by weak hypotheses as in the case of LBP.

The experiments in this thesis use LRD and LRP features at all positions of the base scale of respective detectors and at all possible scales, resulting in a pool of 304,704 features (8464 unique positions).

4.2 Detectors

All detectors in this thesis were created by *WaldBoost* algorithm [124] presented in Section 3.2 and their length was 1000 *weak classifiers*. The particular version of WaldBoost that was used differs in several aspects from the original version published by Šochman [124]. It does not use *Parzen windows* to estimate the *probability ratio* of positive and negative class (Equation 3.9) on validation set to find the rejection thresholds, instead, the rejection thresholds were set on the full training set according to Equation 3.14.

Training set sampling. Training sets were *sampled* in each boosting iteration to increase speed of weak classifier selection. As Kalal et al. [68] showed, such sampling can significantly affect quality of detectors and sampling methods with low *bias* and *variance* are preferable. The type of sampling used here was *unique weighted sampling* which selects samples with *probability equal to their weights*, and produces a set of *unique samples* with weights proportional to the number of times each particular sample was selected. If not stated otherwise, weak classifiers were selected on 2500 *unique positive* and 2500 *unique negative* samples. The selected weak classifiers were refined on the whole active training set.

Small *random geometric transformations* were applied to annotated objects in order to generate 100,000 examples for each *positive training set*. Similarly, 100,000 windows were randomly selected for each *negative training set* from a large set of images not containing objects of interest. The negative training sets were replenished as need during WaldBoost training to compensate for rejected background examples. The collection of background images was large but finite resulting in a finite pool of examples. When a pool was exhausted due to increasingly smaller *false positive rate* of a trained detector, the size of negative training set shrank with each new rejection threshold. To prevent overfitting, no more early termination thresholds were set after reaching a minimum negative training set size of 40,000.

Image scanning. The created detectors were tested by scanning images with *position step* of 2 pixels at the base resolution of the classifier and with *scaling factor* 1.2. The position step was increased accordingly to current scanning-window size. The classifier responses were merged by a *non-maxima suppression* algorithm presented in [125]. This algorithm suppresses all non-maximal windows in neighborhoods defined by minimum window overlap which it approximates by mutual overlap o of circles inscribed in the rectangular windows:

$$o = \frac{r}{R}\left(1 - \frac{d_c}{r+R}\right). \tag{4.1}$$

In the equation R and r are radii of the larger and smaller window, respectively, and d_c is a distance of centers of the rectangles. The overlap approximation is illustrated in Figure 4.5. Minimum overlap of 0.4 was set in the experiments.

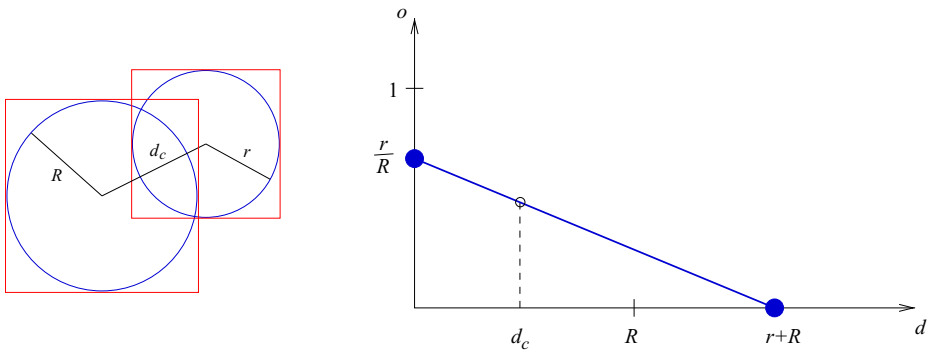


Figure 4.5: Scanning window overlap approximation for non-maxima suppression. The graph shows value of overlap o as function of mutual distance. Taken from [125].

4.3 Datasets

In the field of *appearance-based object detection*, several standard and commonly used *datasets* exist, notably, training and testing pedestrian datasets [98, 24, 23] and testing face datasets [119, 135]. However, these datasets do not cover the whole range of objects of interest and researchers often use their own proprietary data. This section describes several standard datasets, as well as several custom datasets, used in this thesis. Basic properties of all datasets are summarized in Table 4.2.

The *base resolution* of *face* and *traffic sign* detectors is 24-by-24 pixels. Resolution of *eye* and *pedestrian* detectors is 25-by-15 and 18-by-36, respectively, due to the natural aspect ratio of the objects.

Face detection. Several datasets for testing of face detectors exist. The most widely adopted is the *MIT+CMU* face dataset[1] [111, 118, 119, 137, 140, 122, 123, 124] which can be regarded as the main reference dataset in the field (exemplar images shown in Figure 4.7). The *MIT+CMU* dataset is relatively small, it consists of only 137 images and 511 faces, and the images are often of *poor quality* and *small resolution* with visible dithering. Additionally, some of the faces are line-drawings. Considering the small size of the *MIT+CMU* dataset and generally *high detection rates* achieved on this dataset [137], its usefulness is limited mostly to comparisons

[1] http://vasc.ri.cmu.edu/idb/html/face/frontal_images/

Figure 4.6: Random images from the traffic sign dataset. The top row show images from training part of the dataset and the bottom row show test images.

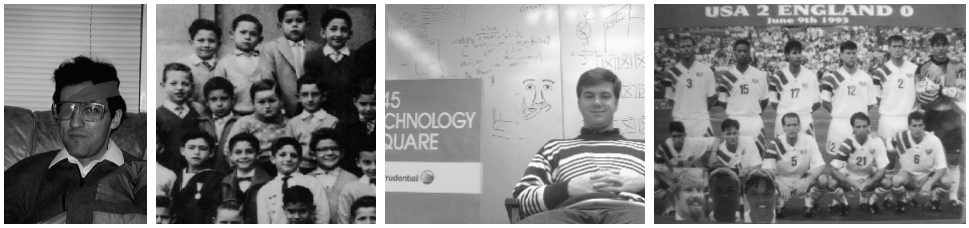


Figure 4.7: Random exemplar images from *MIT+CMU* dataset.

Figure 4.8: Random exemplar images from *GroupPhoto* dataset.

Figure 4.9: Random exemplar images from *Face training* dataset.

Figure 4.10: Random exemplar images from *Background* dataset.

Dataset	♯ imgages	♯ objects	♯ positions ($\times 10^6$)
Face training	3168	5398	645
MIT+CMU	137	511	17
GroupPhoto	111	2056	86
Background	x	x	x
Signs training	6000	655	645
Signs testing	735	483	31
XM2VTS	2365	4730	120
BioID	1530	3060	94
PAL	1045	2090	99
Daimler train	6754	15660	980
Daimler test	19338	2368	834

Table 4.2: Information about the datasets used to train and test the detection classifiers.

among different publications.

In addition the *MIT+CMU* dataset, face detectors were tested on *GroupPhoto* dataset which was gathered by searching for terms *group*, *gang*, and *team* on Google. 2056 faces were hand annotated in the 111 good quality images of this dataset (examples shown in Figure 4.8).

In contrast to the testing datasets, no standard *training dataset* of frontal face detectors has yet been established. Detectors in this thesis were trained on exemplar faces from a dataset previously used in [124, 57]. This dataset, which will be refered to as *Face training*, consists of images uploaded by regular users of the Internet to a face detector demo. The images were later hand-annotated. The dataset contains 3168 images and 5398 annotated faces.

Background training examples were sampled from 10,000 images which do not contain faces and which were downloaded from the Internet. This dataset was used in the same publications as *Face training* [124, 57] and will be refered to as *Background* set.

Eye detection. Eye detectors were trained on *XM2VTS*[2] [96] database and tested on *PAL*[3] and *BioID*[4] [61] databases. The *BioID* dataset contains low-resolution faces, cluttered background, and some variation in lighting. On the other hand, *PAL* dataset contains high-resolution images with constrained pose, simple background, and constant artificial illumination.

[2] http://www.ee.surrey.ac.uk/CVSSP/xm2vtsdb/
[3] https://pal.utdallas.edu/facedb/
[4] http://www.bioid.com/downloads/facedb/index.php

Pedestrian Detection. *Daimler Mono Pedestrian Detection Benchmark Dataset*[5] [24] was used for pedestrian detection task. This dataset defines separate training and testing sets which were used for the experiments without any modification. The testing part of the dataset is a sequence of 21,790 video frames captured from a vehicle during a 27 min drive through urban traffic and as such it contains fully visible pedestrian as well as partially occluded pedestrian or bicyclists. For this reason, the dataset distinguishes pedestrians whose detection is mandatory and pedestrians whose detection is optional. In the original study [24], this distinction was observed in the evaluation and we follow this practice.

Traffic sign detection. Training and testing data for the traffic sign detection task was collected by students with consumer digital cameras. The images were collected on streets of *Czech Republic*, *Belgium*, and *Greece* (examples shown in Figure 4.6). The set contains images of varying quality, view-point, lighting conditions, and some of the signs are damaged.

Czech Republic signs were used for training and Belgium and Greece signs were used for testing.

Additionally, images from *Daimler Pedestrian Classification Benchmark Dataset* were added to the testing set as *distractors* after removing all traffic signs from the images.

4.4 Detection experiments

This section presents and discusses results of detection experiments with features from Section 4.1. The experiments include evaluation of detectors of *faces*, *eyes*, *pedestrians*, and *traffic signs*. The experiments focused of performance of the features in three training scenarios: (1) full feature sets and datasets, (2) restricted datasets and all features, (3) restricted sizes of features and full datasets. Training datasets were restricted either by using only subset of the training sets, or by using fewer examples to choose the best weak hypotheses.

Speed-precision trade-off. When designing *real-time* detectors for practical applications, a trade-off between precision of detection and speed of the classifier should be considered. Detectors always operate under some *computational constrains* whether they run in hand-held digital cameras or in a data center as a part of an off-line experiment. Consequently, when comparing detectors, it is not sufficient to take into account their errors, e.g. in the form of ROC [6] and ignore their speed.

[5] http://www.gavrila.net/Datasets/Daimler_Pedestrian_Benchmark_D/Daimler_Mono_Ped__Detection_Be/daimler_mono_ped__detection_be.html

[6] ROC stands for *Receiver Operating Characteristic*. It is created by plotting true positive rate and false positive rate (or false positives) for various threshold settings.

Most of the focus of *attentional mechanisms* which are used in Viola and Jones' like detectors provide some way (although usually indirect by controlling the target error rate) to influence the speed of the detector. This makes it possible to explore the whole space of speed-precision trade-off, which in turn makes it possible to truly compare different detectors.

For experiments in this thesis, WaldBoost detectors were trained with five different target *false negative rates* (see Section 3.2). Higher values of target *false negative rate* result in faster detectors while lower values result in slower detectors. Detectors created this way cover large range of speeds and allow to compare detectors without focusing on a particular application which may have strict requirements for speed or, conversely, strict requirements for detection quality.

Presentation of results. Accuracy of the classifiers is reported as an *area above Receiver Operating Characteristic* (ROC) curve which represents the *miss rate* averaged over a certain range of *false alarm rate*. This measure will be referred to as AMR in the further text. Similarly to other integral performance measures (e.g. *average precision*), the AMR enables comparison of classifiers when the target application, and thus the desired operating point, is not known. In the experiments, the miss rate was averaged over the range 0 to 200 (20,000 for the pedestrian detection task) false alarms. This range represents useful operating points of most detection applications for the respective test datasets.

AMR is very similar to detection quality measures used by other authors. For example, Dollár et al. [20] used *log-average miss rate* which is exactly the same as AMR except it averages *miss rate* over *logarithmic false positives per image* between 10^{-2} and 10^{0} [7].

Many graphs in this thesis (e.g. Figure 4.11) show relation between AMR and *detection speed* which is expressed as *average number of weak classifiers* (#WC) computed per image position. The graphs allow to analyze behavior of classifiers across the whole range of classification speeds. This could be useful, for example, when selecting features for an application with specific speed or accuracy requirements. On the other hand, the plots of AMR and #WC are not too convenient for fast comparisons over the whole speed range. For that purpose Table 4.3 and Table 4.4 summarize the results as ranks of the individual detector types. The ranks were determined according to subjective assessment of which detectors dominate which in the corresponding AMR/#WC plots. Average ranks were assigned when the order was not clear.

[7] The range of *false positive rate* which AMR averages is similar

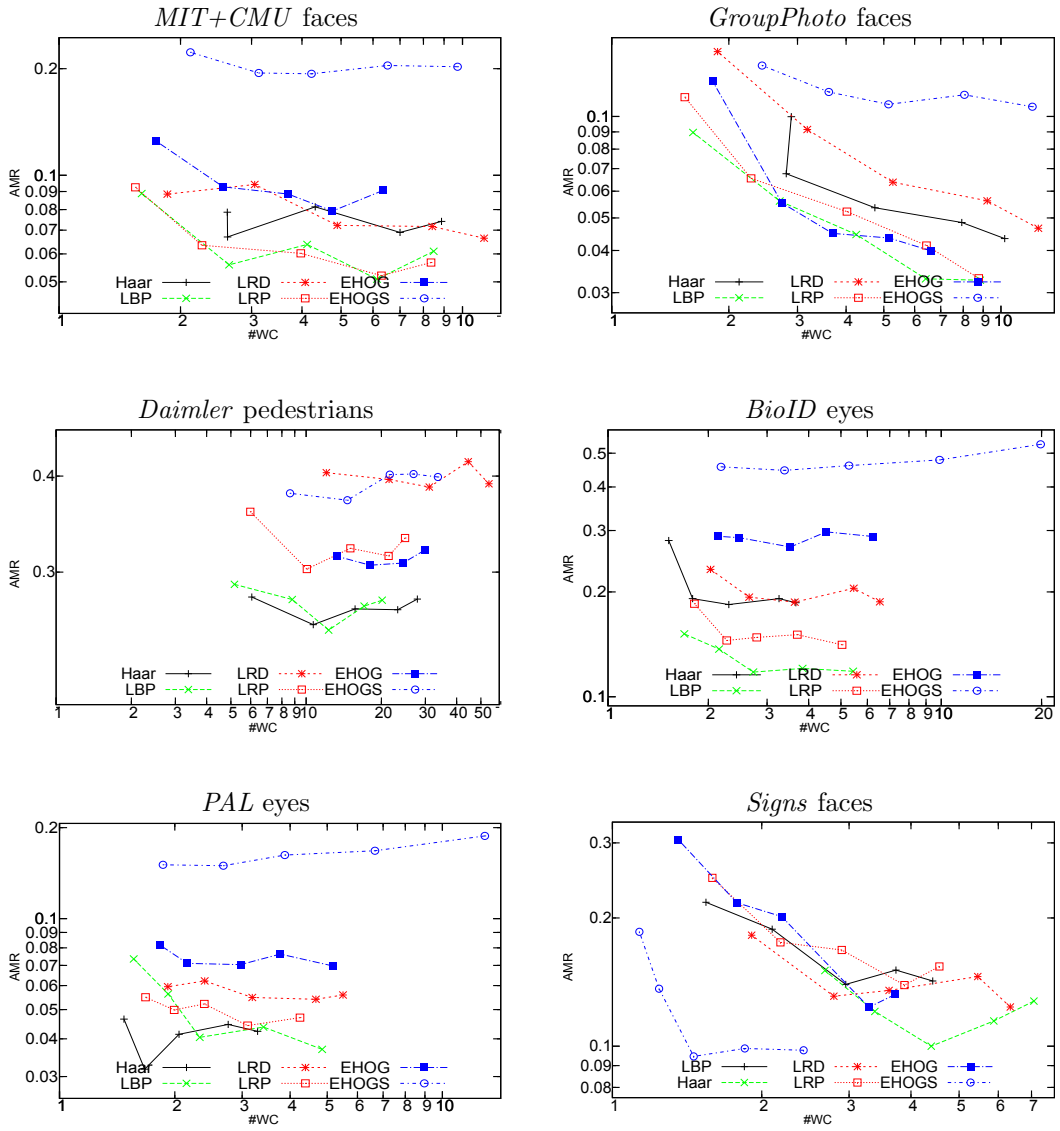


Figure 4.11: Object detection results. Y-axis: *average miss rate* (lower is better); X-axis: average number of weak classifiers evaluated per window (left is faster).

	Haar	LBP	LRD	LRP	EHOG	EHOGS
MIT+CMU faces	4	**1.5**	4	**1.5**	4	6
GrpuPhoto faces	4	**2**	5	**2**	**2**	6
Daimler pedestrians	**1.5**	**1.5**	5.5	3.5	3.5	5.5
BioID eyes	3	**1**	4	2	5	6
PAL eyes	**1**	2.5	4	2.5	5	6
Signs	4	4	4	4	4	**1**
average rank	2.92	**2.08**	4.42	2.58	3.92	5.08

Table 4.3: Features ranked according to their performance in detection tasks shown in Figure 4.11.

4.4.1 Object Detection

The first series of tests was performed to evaluate the selected feature types (see Section 4.1) in different detection tasks. The results of this experiment are shown in Figure 4.11, and Table 4.3 summarizes the results as ranks of the individual features in the detection tasks.

LBP provide the best results overall. They perform consistently well across the six detection task except on *Signs* dataset where *EHOGS* are significantly better than all the other features. Second best overall are *LRP*, then *Haar*, *EHOG*, *LRD*, and finaly *EHOGS*.

The most significant perturbation is the best result of *EHOGS* in the *traffic sign* detection task. The most probable explanation of this behavior is that the *invariance to color inversion* makes these features very suitable to *model silhouette edges* which are the predominant features of the traffic signs when color is not considered. However, the same effect is not present on the *pedestrian* detection task where the distinguishing feature is also the silhouette.

Surprisingly, *Haar* is better in pedestrian detection than *EHOG* which is in contrast to other published results. This could be possibly explained by relatively low resolution of the detector (18-by-36 pixels). Similarly, *Haar* gives best results on the PAL eye detection test set. In this case, a reasonable explanation is that this is due to clutter-free background in the test images. Further, the Haar-like features achieve good performance in the pedestrian detection task where they match *LBP*.

LRP consistently outperform *LRD* except on traffic sign detection task where *LRD* give slightly better results.

Detector	♯WC	False Positives													
		6	9	10	26	31	39	41	46	50	57	65	77	78	95
Viola and Jones [138]	8	-	-	76.1	-	88.4	-	-	-	91.4	-	92.0	-	92.1	92.9
Li and Zhang [85]	18.9	-	-	83.6	-	90.2	-	-	-	-	-	-	-	-	-
Schneiderman [120]	-	89.7	-	-	-	-	-	-	**95.7**	-	-	-	-	-	-
Wu et al. [7]	-	-	-	90.1	-	-	-	-	-	-	94.5	-	-	-	-
Luo [92]	-	86.6	-	87.4	-	90.3	-	-	-	91.1	-	-	-	-	-
Bourdev [8]	37	**90.9**	91.9	-	93.5	-	-	94.3	-	-	-	-	-	-	-
Bourdev [8]	25	-	-	-	-	91.7	92.1	-	-	92.7	-	-	92.9	-	-
Brubaker et al. [10]	8	81.7	-	85.8	-	88.8	-	-	90.1	90.1	-	90.3	-	90.5	90.9
Brubaker et al. [10]	-	89.1	-	89.5	-	91.3	-	-	91.9	91.9	-	92.1	-	92.1	92.3
Sochman [141]	**3.32**	87.4	87.5	88.2	90.3	90.5	90.7	90.7	91.1	91.1	91.3	91.9	92.1	92.1	92.5
Zhang and Viola [12]	14.6	88.8	-	91.7	-	93.2	-	-	-	94.6	-	-	-	-	95.2
Our LBP	6.84	90.6	92.9	**93.3**	**93.9**	93.9	94.1	94.1	94.1	94.3	94.5	94.9	95.3	95.3	**95.7**
Our LRP	6.23	89.0	90.2	91.0	**93.9**	**94.9**	**94.9**	**94.9**	94.9	95.1	95.1	95.1	95.5	95.5	**95.5**
Our Haar	6.98	81.7	84.9	85.1	90.0	90.6	91.0	92.9	93.7	93.7	94.3	94.5	94.7	94.7	95.1

Table 4.4: Results of selected classifiers on the *CMU+MIT* face dataset. The table shows the *detection rates* as a function of the number of *false positives*. Note the differences in the *average number of weak hypotheses* computed per scanned position. The results should be interpreted with caution as different training sets and slightly different evaluation methodologies were used by different authors. The table extends similar table by Šochman [141].

	Haar	LBP	LRD	LRP
MIT+CMU faces	4	**1**	3	2
GrpuPhoto faces	4	**1**	3	2
Daimler pedestrians	2	**1**	4	3
BioID eyes	4	**1**	3	2
PAL eyes	3.5	2	3.5	**1**
Signs	2.5	2.5	2.5	2.5
average rank	3.33	**1.42**	3.17	2.08

Table 4.5: Features restricted to sizes 1x1, 1x2, 2x1, and 2x2 ranked according to their performance in detection tasks shown in Figure 4.11.

Comparison to state-of-the-art. In order to put detectors from this thesis into the context of other state-of-the-art methods, Table 4.4 shows results of the WaldBoost face detectors used in this thesis together with results of other methods on the *MIT+CMU* dataset. The WaldBoost detectors compare favorably to other methods in terms of both *detection rate* and *speed*. The detectors use on average less than 7 features per image position which is less than the other detectors except the WaldBoost detector by Šochman [141] which, however, provides much worse *detection rates*. Also, the detectors provide best or very good *detection rates* throughout the whole range of *false positives*.

The best results achieved by a cascade of Haar-like features on the *Daimler Mono Pedestrian Detection Benchmark Dataset* reported in [24] are approximately 57% detection rate for 0.1 false alarms per frame and 81% detection rate for 1 false alarm per frame. Our WaldBoost detector which uses on average 11 Haar-like features per classified position achieves detection rates 61% and 84% for the same false alarm rates (even thought the scanning is sparser in this case).

Considering the face and pedestrian detection results, it is reasonable to conclude that detectors used in this thesis are comparable to other state-of-the-art detectors.

4.4.2 Restricted feature sizes

This experiment evaluates performance of *Haar*, *LBP*, *LRD*, and *LRP* feature sets when the size of building blocks of the features is limited to 1-by-1, 1-by-2, 2-by-1, and 2-by-2 pixels. The building blocks are grid cells in the case of *LBP*, *LRP*, and *LRD*. In the case of *Haar*, the building blocks are the rectangular areas which the features are composed of. This type of size restriction can lead to alternative simple and efficient ways to compute the features on highly parallel platforms [50] (e.g. GPU and FPGA). *EHOGS* and *EHOG* were not considered in this experiment as they are not composed of building blocks.

Figure 4.12 shows results of this experiment, and Table 4.3 summarizes the results as ranks of the individual features in the detection tasks.

The overall ordering of the three best features is *LBP*, *LRP*, and *LRD*. The mutual

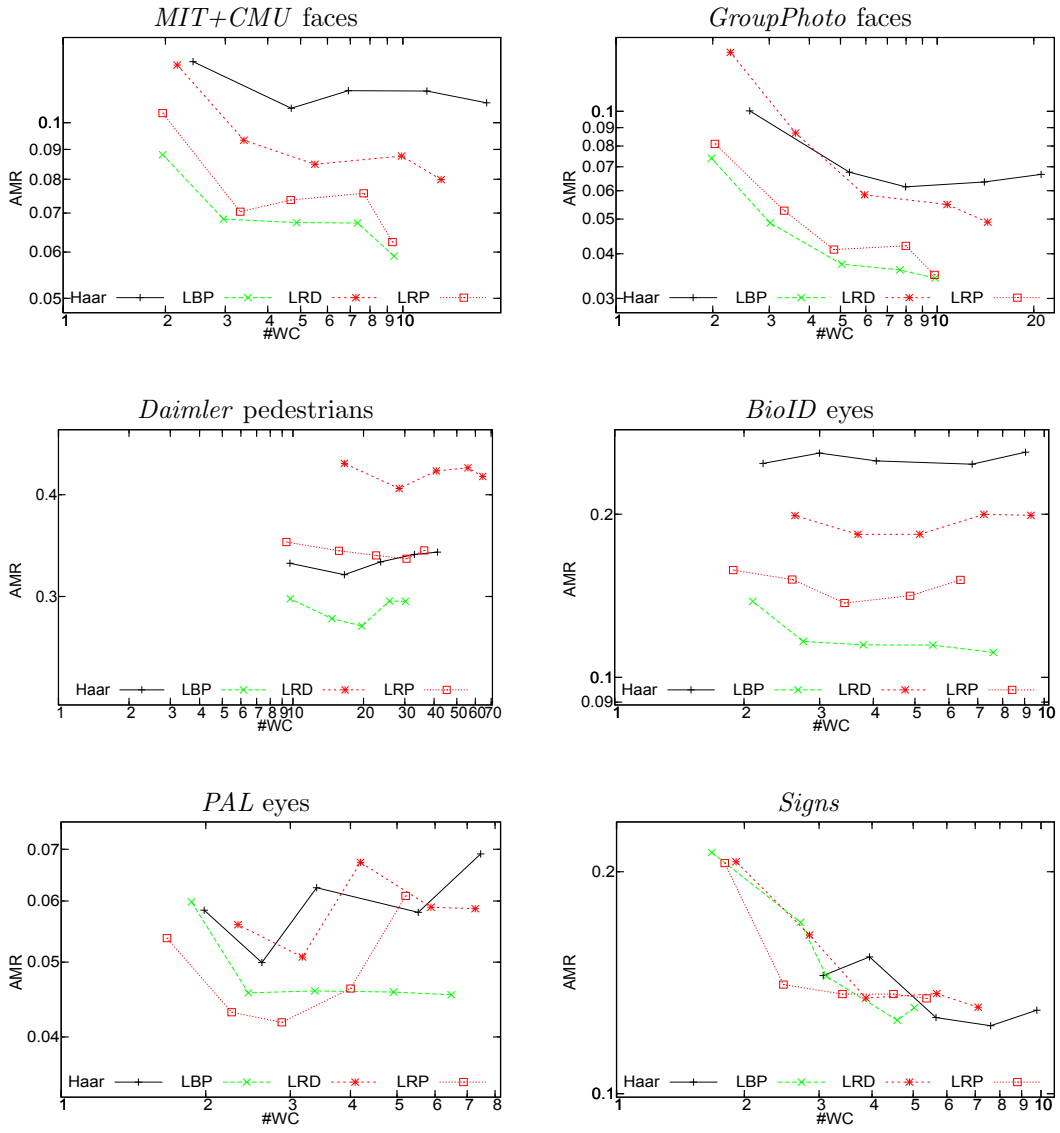


Figure 4.12: Object detection with restricted size of features. The basic building blocks of features were restricted to sizes 1x1, 1x2, 2x1 and 2x2. Y-axis: *average miss rate* (lower is better); X-axis: average number of weak classifiers evaluated per window (left is faster).

performance differences of these features are similar to their differences without the size restriction.

Compared to the other features, the size restriction has much more adverse effect on *Haar*, which are now the worst. Although unrestricted *Haar* shared the first place with unrestricted *LBP* in the pedestrian detection task (see Figure 4.11), now it is significantly worse than *LBP*. On the PAL dataset, where unrestricted *Haar* achieved the best results, the restricted *Haar* give the worst results together with *LRD*.

4.4.3 Effect of Training Set Size

Performance of object detectors strongly depends on the *quality of training sets*. One of the important characteristics of a training set is its *size*. Intuitively, one would expect features that discard more irrelevant information to cope better with smaller training set sizes as they should be able to generalize better.

To evaluate the ability of features to cope with *smaller training sets*, I trained *face detectors* on progressively fever exemplar faces (from 5000 down to 19 faces). All of the face detectors were created for target *false negative rate* 5%.

Results of the experiment are shown in Figure 4.13. As expected, the smaller sizes of the positive training set result in higher AMR. Also, the detectors get faster with fewer positive examples. The reason for faster speed is that the classification task WaldBoost has to solve gets easier with fewer examples (the detector becomes less general). The detectors would not get faster if *validation set* was used to select rejection thresholds (see Section 3.2), however, the detection quality would still degrade.

The results show that AMR degrades at different rates for different features. Namely, *EHOG* features cope with very small sizes of training set much better than the other features. For the smallest training set size, the *EHOG* features perform best on both dataset. On the *GroupPhoto* dataset *EHOG* gives similar AMR as *Haar*, *LBP*, *LRP*, and *LRD* with *four times larger training set*. The *EHOGS* features also seem to cope with the small set sizes better, but their accuracy is significantly lower compared to the other features for large training set sizes.

4.4.4 Training Set Sampling

As stated in Section 4.2, *unique weight sampling* selects a subset of training samples to be used to choose the best image feature in each iteration of WaldBoost. The sampling significantly reduces training time while retaining performance very similar to classifiers created using the full training set. The question is, how this sampling affects different types of features.

To evaluate the effect the training set sampling, face detectors were trained while sampling 500, 2500, and 5000 examples of both types in each boosting iteration. Note that, although selection of weak classifiers on 500 examples is by an order of

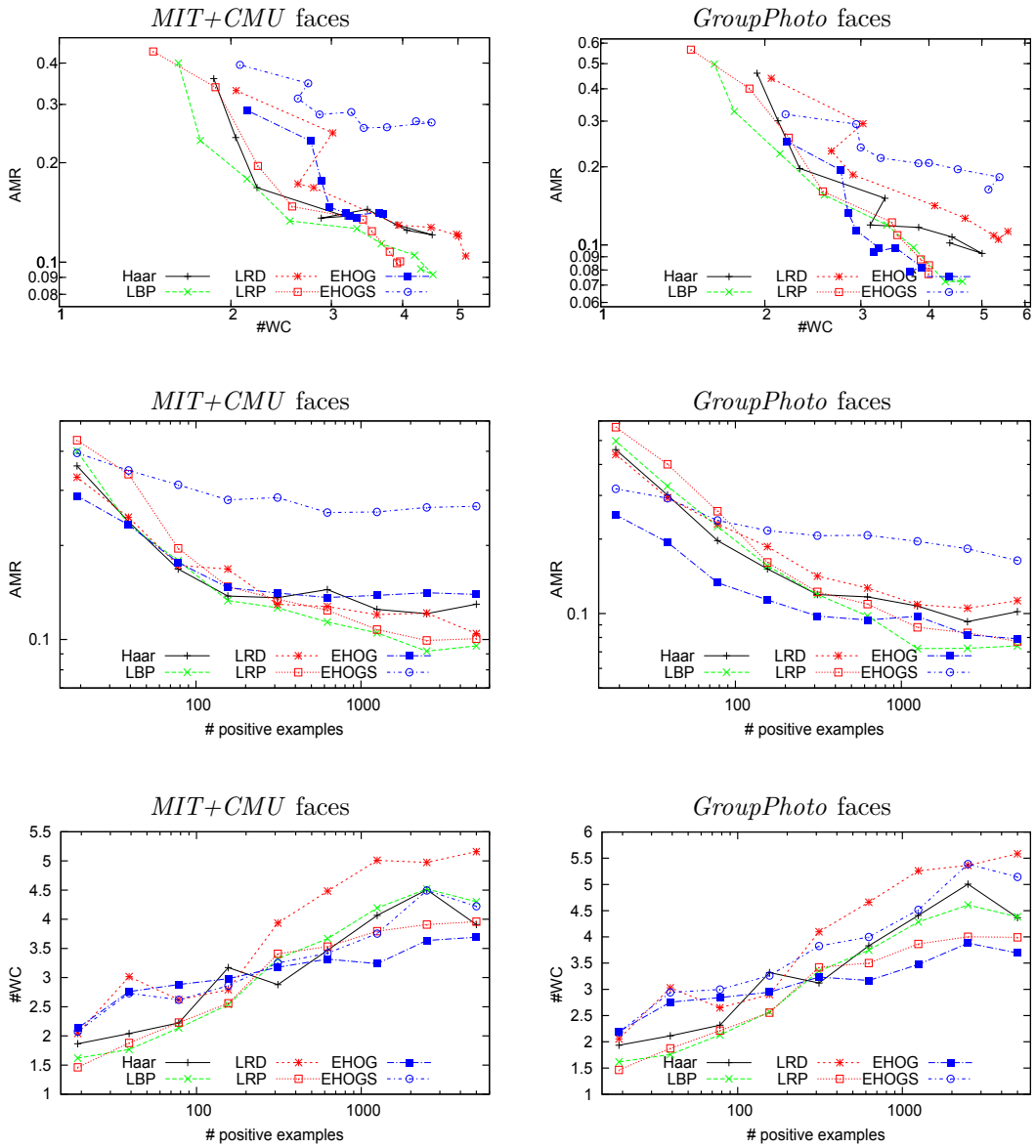


Figure 4.13: Effect of the size of training set. Each curve corresponds to single type of features and points on the line correspond to classifiers trained on different number of annotated object resulting in different speed and error rates. The numbers of annotated faces for training are 5000, 2500, 1250, 625, 312, 156, 78, 39, and 19. Second row shows *mean miss rate* as a function of training set size. Third row shows average number of weak classifiers evaluated per window as a function of training set size.

magnitude faster than when 5000 examples are used, speed-up of the whole training process is much lower as a large portion of the training time is spend by *bootstrapping* the background examples.

Two different scenarios were considered in this experiment. In one scenario, features were selected on sampled subsets and predictions of the corresponding weak classifiers were refined on the whole training set. This is the way previous detectors were created. In the second scenario, the prediction refinement step was skipped. Skipping the refinement step should result in more profound performance degradation with smaller examples.

The results in Figure 4.14 show that the effect of training set sampling in the considered range is negligible when the weak classifier refinement step is employed. In case the refinement step is skipped, the performance is significantly degraded when sampling only 500 examples. Compared to the other features the *LRP* exhibit the highest sensitivity to the sub-sampling. On the other hand, *Haar* and *EHOG* cope with the sub-sampling relatively well.

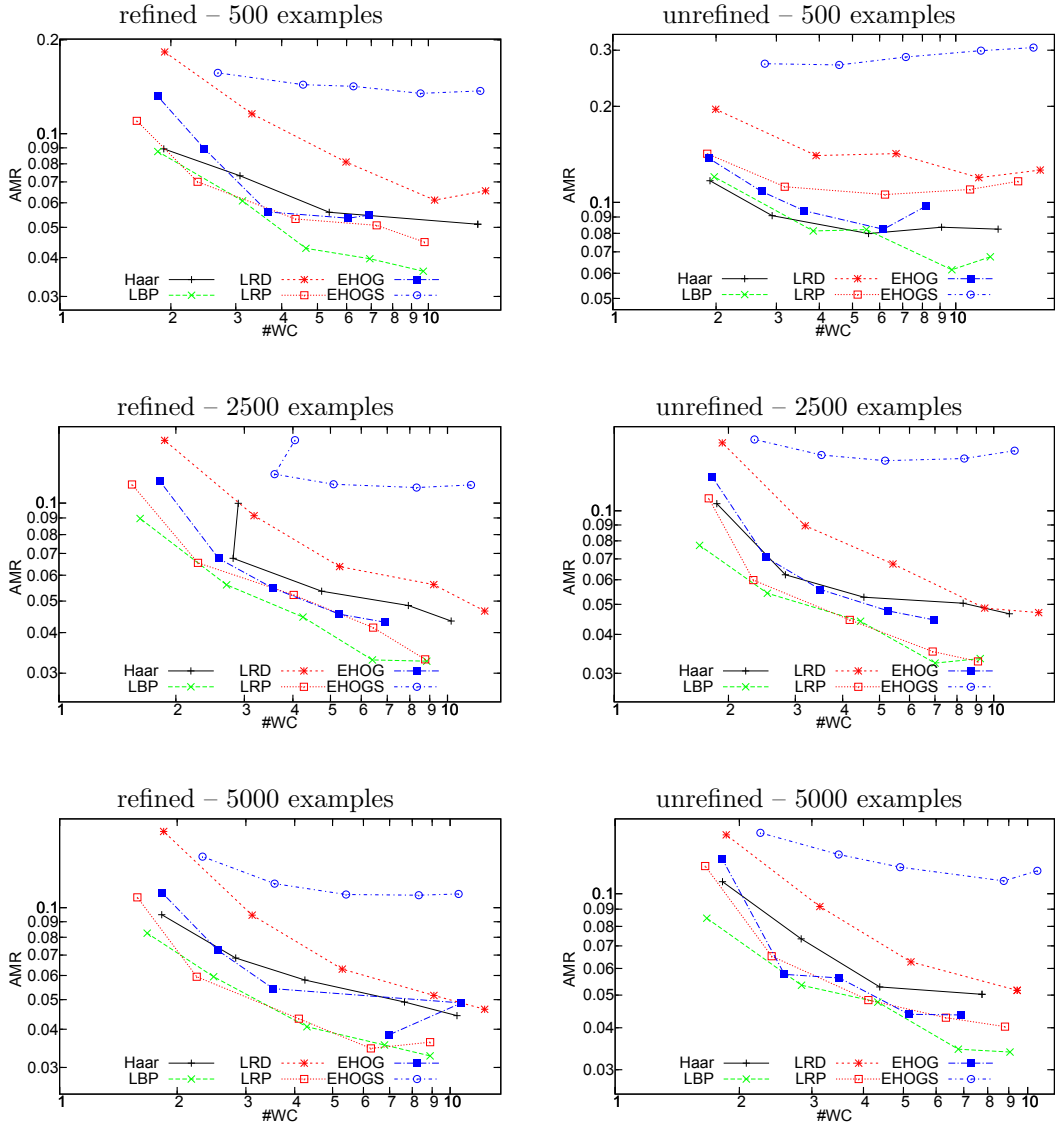


Figure 4.14: Effect of sub-sampling the training set in each boosting iteration. The first column corresponds to refinement of weak hypotheses on full training set (not only its sampled subset), and second column is without this refinement. The rows are for 500, 2500 and 5000 training samples. The results are shown for face dataset *GroupPhoto*. Y-axis: *average miss rate* (lower is better); X-axis: average number of weak classifiers evaluated per window (left is faster).

CHAPTER 5

Information sharing in scanning-window detection

In their basic form, *scanning-window detectors* process image regions independently one by one. An advantage of such design is its simplicity which makes it possible to define the detection task as a *standard binary classification* that can be solved by general learning algorithms without any modifications. However, the independent processing is *sub-optimal in terms of computational cost.*

A detection task can be regarded as accumulation of evidence and inference of probable object positions in an image. The range of possible evidence the detection can rely on is large, e.g. local shape, color, texture, coocurance of local shapes, self-similarities, position, surrounding objects and surfaces, scene type, author, and acquisition time and place. Practical detectors are necessarily limited in what information they are able to work with due to computational constraints, limitations of available data, and limitations of current human knowledge. Addressing the computational constraints, detectors should extract the information they are limited to as efficiently as possible. Unfortunately, such optimality would be very hard to achieve even for very simple detectors.

Lets consider a WaldBoost detector with LRP features form Chapter 4. The detector is a majority vote of the LBP features which is dynamically terminated depending on the progress of the voting. The detector can be regarded as optimal when considering single image position, but it completely ignores overlapping neighboring positions. Ideally, the features of the detector should be selected according to how much information they contribute to all surrounding positions, they should update positions for which it is efficient to do so, and the early terminations should depend on results of neighboring positions. Even for such limited detector, the resulting learning algorithm would be complex and, possibly, computationally expensive. Also, it is often hard to estimate compuational cost of different parts of the detector in a

target hardware platform.

The ways in which existing scanning-window detectors are optimized with respect to detection window overlap can be divided in two basic groups. Many detectors share some computations across image windows in the form of image preprocessing [137, 22, 21, 3, 4] and in the form of common features [120, 18, 16, 77, 134, 2, 83] or parts [27]. Even higher level information can be shared as in convolutional networks [25, 128].

The second group includes methods which *make local decisions interdependent* in various ways. These methods include detectors which try to minimize the number of processed image windows by exploiting *smoothness* of a particular detector responses [16, 77, 20], and some detectors improve speed by assuming *minimum distance between objects* [106, 20] in the same way as non-maxima suppression does.

The rest of this chapter overviews existing detectors which locally share information and discusses how the detectors relate to *neighborhood suppression* and EnMS.

Computation sharing. Most scanning-window detectors do not process image windows completely independently. Even the original detector of Viola and Jones [137] computes an *integral image* which is shared among all windows and which significantly improves speed of Haar-like features. Other detectors take the preprocessing idea further. Notably, Dollár et al. [22] extend the idea of integral images to other types of information with their *integral channel features* which compute local sums, histograms, Haar-features, and their various generalizations using a range of integral channels. The approach was later extended [21] to approximate feature responses at nearby scales, and further improved by Benenson et al. [3, 4].

Sharing of features interlinks neighboring positions even further. Such approach was advocated by Schneiderman [120] as *feature-centric computation* which computes several first features densely across a whole image. Similarly, the pedestrian detector by Dalal and Triggs [18] computes HOG features on a dense grid and uses them as an input for a linear classifier.

Similarly, most *part-based detectors* share *visual words* or *parts*. Detectors based on visual words [16, 77, 134, 2, 83] compute the words from independently of the detection task as a first step similarly to the *feature-centric computation*. A visual word represents a local area deemed important by a *key-point detector* (e.g. SIFT, SURF, MSER) by few most similar prototypes from a *codebook*. In its simple form, the projection to prototypes is a standard *vector quatization* of real-valued high-dimensional descriptors of the patches.

Some part-based detectors detect the parts first and infer positions of objects from responses of the part detectors. For example, Felzenszwalb et al. [27] detect objects from *response maps* of discriminatively trained part detectors.

Smoothness of detector responses. Responses of many detectors are smooth due to their *robustness to small shifts* and *other transformations*. Such smoothness can be used to infer responses in local neighborhoods or to reason about a whole group of regions as about a single homogeneous set. The goal of methods which use the smoothness assumption is usually to minimize the number of windows on which the detector is evaluated.

Chum and Zisserman [16] use discriminative features to locate likely object positions which serve as seeds for *discrete gradient ascent search* for a maximal responses of a window classifier. Related is also the *efficient subwindow search* by Lampert et al. [77] which searches the space of all windows in an image guided by an *upper bound* on the classifier response over a set of rectangles. However, the search can be efficient only if the bound is reasonably tight and computationally efficient, which is possible only for relatively simple classifiers which have high invariance to geometrical transformations.

A successful way how to apply the smoothness assumption to fast detectors with attentional structure is to first scan an image relatively sparsely and then re-scan the promising regions more densely. Examples of such approaches are by Butko and Movellan [11] and Gualdi et al. [42, 43].

A promising method was proposed by Dollár et al. [20]. Their *excitatory cascades* realize the sparse scanning idea with soft cascades. The authors suggest an algorithm which sets excitatory thresholds for stages of an existing soft cascade on an unlabeled set of images such that regions containing positive responses of the original cascade are missed during the sparse scanning phase only with some small and defined probability. However; the authors do not claim that the thresholds are set in optimal way and, in fact, they are clearly sub-optimal.

Non-maxima suppression assumptions. Non-maxima suppression, which is part of most scanning-window detectors [137, 22, 20], is based on the assumption that two objects can overlap only to a limited extent. This assumption is valid for most detectors as they are usually not able to handle severe occlusions anyway. The assumption allows detectors to merge overlapping responses into a single object position, which is usually the window with the highest detector response.

The assumption of non-maxima suppression can be used to accelerate detection. If the final object position is determined only by the window with the highest responses, responses at neighboring positions are not needed and the detector only has to determine that they are to be suppressed. This idea was utilized, for example, by Pedersoli et al. [106] in their *coarse-to-fine detector* which splits an image into a set of neighborhoods that can contain only one object and searches the neighborhoods in greedy recursive coarse-to-fine fashion. First, the object is localized at a coarse resolution, and the position is further refined at higher resolutions.

An interesting application of the non-maxima suppression assumption is the *inhibitory cascade* by Dollár et al. [20]. The inhibitory cascades evaluate neighboring image positions in parallel and terminate computation of those windows which will likely give non-maximal results. The decisions are based on *ratios of partial cascade responses*. The authors proposed an algorithm which sets thresholds on the response ratios for an existing *soft cascade* using unlabeled images. Although the thresholds are set such that the inhibitory cascade introduces a small and defined error, the thresholds are not optimal in terms of decision speed (why *inhibitory cascades* are not optimal and how they relate to EnMS is discussed in Chapter 8).

Relations to EnMS and neighborhood suppression. All methods which accelerate detectors by *sharing computations of features* or by *image pre-processing* are orthogonal to *neighborhood suppression* and EnMS, and could be combined with the proposed methods for even faster detection.

Many of the methods which strongly rely on *smoothness of detector responses* are not applicable to fast detectors with *attentional structures*, which produce discontinuous responses due to the early terminations. The local search methods [16] and the branch-and-bound search by Lampert et al. [77] target relatively slow detectors which are not the primary focus of *neighborhood suppression* and EnMS.

The *excitatory cascades* by Dollár et al. [20] focus on the same detectors as *neighborhood suppression* and their underlining idea is similar as well. However, the *excitatory cascades* try to select image positions which should be evaluated and *neighborhood suppression*, in contrast, selects image positions which should be skipped.

The coarse-to-fine detector of Pedersoli et al. [106] is in many aspects related to EnMS, which could, in fact, be applied to a multi-stage coarse-to-fine detector in order to create a detector with similar behavior. An advantage of EnMS is that it produces optimal time-to-decision detector for a target localization error.

The *inhibitory cascades* by Dollár et al. [20] are build exactly on the same idea as EnMS and the way they process images is very similar. The methods differ only in the exact form of the conditions which decide when non-maximal windows are to be rejected, and EnMS, unlike inhibitory cascades, finds thresholds for the decisions which optimize detection speed.

CHAPTER 6

Neighborhood suppression

The algorithm proposed in this chapter extends existing appearance-based detectors with an ability to suppress image positions in the neighborhood of the position being currently classified. The proposed method is effective and, at the same time, simple and computationally inexpensive. It learns a new *suppression classifier* which predicts the responses of the original detector at neighboring positions. When the predictions are negative and confident enough, computation of the detector is suppressed at the respective positions.

The idea of *neighborhood suppression* is demonstrated in Figure 6.1. While a detector is deciding an image position, it is, at the same time, trying to reject neighboring positions. Evaluation of the detector is suppressed at the positions which get rejected.

The suppression is possible because the neighboring positions share information due to overlap of the image windows caused by small horizontal and vertical scanning steps. In order for the *neighborhood suppression* to be efficient, the detector and the suppression classifier have to *share computation*. These reused parts can be image features in the case of Viola & Jones' [137] and similar detectors or possibly other partial computations. The reuse of computation is crucial and, in fact, it is the only reason why faster detection can be achieved this way. Although the *neighborhood suppression* algorithm proposed here considers only sharing of features, the general idea could be applied to wider range of detectors and in other ways.

The effectiveness of *neighborhood suppression* relies on the amount of information shared between neighboring scanned windows – which is clearly high if the windows overlap closely. However, it is not immediately clear how suitable are the features of the original detector, as those features were selected specifically for detection of well aligned centered objects and they are not necessarily suitable for other tasks.

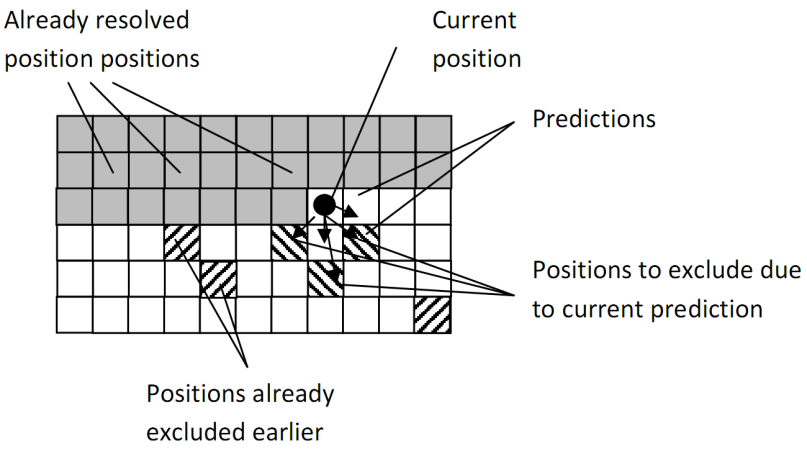


Figure 6.1: Scanning an image in ordinary line-by-line fashion while using *neighborhood suppression*.

The amount of information provided by the reused features, and consequently the possible effectiveness of *neighborhood suppression*, surely varies with different types of features and objects. Also, the amount of shared information decreases with distance of the windows.

Efficiency of *neighborhood suppression* is affected by the fact that detectors with attentional structure *compute on average only few features per window* (see Table 4.4), and the suppression classifiers should use only these features if they are to improve speed of detection.

Although this chapter considers *neighborhood suppresson* only for *soft cascades* [8, 124] (see WaldBoost detector in Algorithm 2), the proposed approach is not limited to this type of detectors. *Neighborhood suppression* could be easily extended to detectors with different attentional structures in a straightforward and trivial way.

The *neighborhood suppression* creates new suppression classifiers for an existing *soft cascade*. The new classifiers are trained by WaldBoost [124] and they reuse features of the original soft cascade.

The task of learning the suppression classifiers is similar to emulation of existing detectors by WaldBoost as proposed by Šochman and Matas [125, 126]. Formulating the *neighborhood suppression* task as detector emulation makes it possible to use unlabeled data for training, and it allows the approach to support existing detectors without any modifications.

6.1 Learning Neighborhood Suppression

This section formally defines a learning algorithm for *neighborhood suppression* [155]. It first summarizes necessary notation and facts about *sequential decision strategies* and *WaldBoost* previously presented in Section 2.1 and Chapter 3, and then it presents the novel algorithm which was inspired by the WaldBoost algorithm [124].

Soft cascade. A soft cascade is a *sequential decision strategy* with decision functions S_t based on a *majority vote* of weak hypotheses $h_t : \mathcal{X} \to \mathbb{R}$:

$$H_T(\mathbf{x}) = \sum_{t=1}^{T} h_t(\mathbf{x}) \tag{6.1}$$

with corresponding decision thresholds (as discussed in Chapter 3).

For *neighborhood suppression*, the three-way decision functions from Equation 3.10) are simplified to *two-way decision functions* which terminate only for negative decisions:

$$S_t(\mathbf{x}) = \begin{cases} -1, & \text{if } H_t(\mathbf{x}) \leq \theta^{(t)} \\ \sharp, & \text{if } \theta^{(t)} < H_t(\mathbf{x}) \end{cases} . \tag{6.2}$$

Although it would be possible to suppress computation of a detector at neighboring positions which contain an object with high probability as well, most object detectors include some variant of non-maxima suppression which requires the detector to be fully computed at the most promising positions in order to obtain optimal location of the objects (usually a position with the highest response of the detector).

Weak hypotheses used in practical detectors [124, 141] are in vast majority of cases *space partitioning weak hypotheses* [117] which internally operate with disjoint partitions of the object space $\mathcal{X}$. The functions partitioning the object space $f : \mathcal{X} \to \mathbb{N}$ will be reffered to in the following text simply as *features*. The space partitioning weak hypotheses are combinations of such features and a *look-up table function* $l : \mathbb{N} \to \mathbb{R}$

$$h_t(\mathbf{x}) = l_t(f_t(\mathbf{x})). \tag{6.3}$$

In the further text, $c_t^{(j)}$ specifies the real value assigned by l_t to the output j of f_t. The $c_t^{(j)}$ values may be set in many different ways depending on the learning algorithm used to build the detector. In the case of WaldBoost, $c_t^{(j)}$ values are set according to Equation 2.9.

Neighborhood suppression learning algorithm. The task of learning a suppression classifier can be formalized as learning a new soft cascade with a decision strategy S' consisting of hypotheses $h'_t = l'_t(f_t(\mathbf{x}))$, which reuse features f_t of the original detector S, and which only differs in the look-up table functions l'_t and in the rejection thresholds $\theta'^{(t)}$. The goal of the new decision strategy S' is to emulate the original detector at neighboring locations. The whole algorithm for learning suppression classifiers is summarized in Algorithm 4. The learning algorithm is closely related to WaldBoost (see Algorithm 3).

The inputs of the algorithm are target *false negative rate*, existing soft cascade S and a set of unlabeled images.

The target *false negative rate* applies to the binary decision of the suppression

classifier. Total change of *false negative rate* of the whole final detector will be lower. This discrepancy is natural and it has two reasons. *Neighborhood suppression* can be performed only within a small neighborhood and, as a consequence, a detector has to be evaluated at many image positions even if all the suppression decisions are successful. Also, the target false negative rate in Algorithm 4 would be reached only if the suppression classifier managed to reject all background positions, which it is not able to do in practice (see Table 6.1) as its decision evidence is limited only to the features computed by the original detector.

The *training set* consists of image windows extracted from unlabeled images which should be close to the target domain of the detector. The image windows represent positions at which the detector is evaluated. As the task is to predict response of the original detector S at some other position in neighborhood, corresponding labels for the learning task are obtained by evaluating the original soft cascade S at an image position with a particular displacement.

The algorithm proceeds in iterations in which it consecutively creates new weak hypotheses for the suppression classifier – it sets values of the look-up table l'_t and of the early termination threshold $\theta'^{(t)}$ for feature f_t of the original detector S. The look-up table values are set according to *real AdaBoost* (Equation 2.9). The termination threshold $\theta'^{(t)}$ is set as in WaldBoost (Equation 3.13). As the algorithm does not have to select an optimal weak hypothesis from a large pool of available features (which is generally the most time consuming step in WaldBoost), the learning of the suppression classifiers is very fast.

The training set is *pruned* twice in each iteration. First, examples rejected by the new suppression classifier must be removed from the training set. In addition, examples rejected by the original detector S must be removed as well. This corresponds to the behavior during image scanning when only those features which are needed by the original detector to make decision are computed.

Multiple suppressions. Suppression classifiers learned by Algorithm 4 aim to suppress only a *single image position*. This limitation is not inherent to this approach, in fact, multiple neighboring position can be suppressed by single classifier, and Algorithm 4 can be easily extended to learn such classifiers. This behavior can be achieved by setting labels of the training samples to -1 only when the original detector rejects all of the considered positions.

In addition, multiple suppression classifiers focusing on different parts of a neighborhood can be combined.

Algorithm 4 *Neighborhood suppression* learning algorithm based on WaldBoost as published in [155].

Input:

- original soft cascade S defined by features f_t, corresponding weak hypotheses $h_t(\mathbf{x})$, and rejection thresholds $\theta^{(t)}$
- training set $P = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_m, y_m)\}, \mathbf{x}_i \in \mathcal{X}, y_i \in \{-1, +1\}$, where the labels y_i are obtained by evaluating the original soft cascade S at an image position with particular displacement with respect to the position of corresponding $\mathbf{x}_i$ in an respective image
- desired miss rate α

Output:

- look-up table functions l'_t and early termination thresholds $\theta'^{(t)}$ of the new suppression classifier

Initialize sample weight distribution $D_1(i) = \frac{1}{m}$
for $t = 1, \ldots, T$

1. estimate new l'_t using f_t such that

$$c_t^{(j)} = -\frac{1}{2} \ln \left(\frac{P_{i \sim D}(f_t(\mathbf{x}_i) = j | y_i = +1)}{P_{i \sim D}(f_t(\mathbf{x}_i) = j | y_i = -1)} \right)$$

2. add l'_t to the suppression classifier

$$H'_t(\mathbf{x}) = \sum_{r=1}^{t} l'_r(f_r(\mathbf{x}))$$

3. find optimal threshold $\theta'^{(t)}$ satisfying Equation 3.13
4. remove training set samples for which $H_t(\mathbf{x}) \leq \theta^{(t)}$
5. remove training set samples for which $H'_t(\mathbf{x}) \leq \theta'^{(t)}$
6. update the training set weight distribution

$$D_{t+1}(i) \propto \exp(-y_i H'_t(\mathbf{x}_i))$$

6.2 Neighborhood suppression in real-time detection

Adding the ability to suppress neighbors to existing detector engines requires only slight modifications which may, however, introduce some computational and storage overhead. Although the computational overhead is small and may not affect detection speed on some architectures at all (e.g. on SIMD architectures, and on wide-register architectures), it should be considered.

Starting from an existing implementation of a *soft cascade* detector, one has to expand it to be able to perform the new *table lookups* l'_t, update *accumulators* of the suppression classifiers, perform *threshold tests* on the accumulators, and maintain a *list of the suppressed positions*.

The prediction values of a suppression classifier have to be loaded from memory in addition to the prediction values of the original detector. Fortunately, the lookup tables l_t and l'_t are always indexed by the same value corresponding to an output of the same feature $f_t(\mathbf{x})$. This coordinated access pattern allows the lookup tables to be merged into a single table with double size of entries. Assuming suitable memory architecture, the two values can be retrieved at the same cost in a single memory access. On a standard PC, the memory access cost will remain the same for up to 16 bytes long entries when no cache misses are considered (assuming proper memory alignment). In standard situation, the 16 bytes can accommodate 4 classifiers (four 32-bit floating point values). Additionally, previous work [156] indicates that the look-up table values can be quantized down to 8-bit values without significant performance degradation. Such quantization would increase the number of classifiers which can fit into a 128-bit register to 16.

The prediction values have to be accumulated and the accumulated values compared to thresholds. This can be done in parallel with no additional cost on SIMD architectures, such as MMX/SSE/AVX instruction set extensions of contemporary PC processors. Using the AVX instruction set, which supports 256-bit registers, eight 32-bit accumulators can be handled in parallel.

On systems with wide enough data words but no SIMD support, the implementation can be similar as on a SIMD architecture. All the accumulators may be packed into a single long integer accumulator manually as long as the accumulators do not overflow. The non-overflow condition can be easily fulfilled as the maximum possible value of each portion of the register can be calculated in advance from values contained in the look-up tables.

The suppression itself can be handled by a binary mask covering positions to be scanned. Some positions in such mask would be marked as suppressed and would be excluded from further processing. The image scanning pattern can remain the same as in ordinary scanning-window approach, even though it restricts the positions which can be suppressed to those which are to the right and bottom of the currently

classified position[1] (see Figure 6.1). Possibly, more efficient scanning strategies may be developed.

6.3 Neighborhood suppression experiments

I tested the *neighborhood suppression* on *frontal face* detection and *eye* detection tasks. In both tasks, two separate test image sets were used - one with less constrained poses and lower quality images and one with easier poses and good quality images. All the datasets are described in more detail in Section 4.3.

Face detection experiments were performed on *MIT+CMU* frontal face dataset and on *GroupPhoto* dataset. From these two, *MIT+CMU* contains lower quality images. *GroupPhoto* contains good quality group shots with close to frontal faces. Eye detection experiments were performed on *XM2VTS* database and on *BioID* database. *XM2VTS* is much easier compared to *BioID* as it contains clutter-free backgrounds. The datasets are described in Section 4.3. Suppression classifiers were trained on a large set of unannotated images containing faces.

The tests were performed with four types of image features: *Haar*, *LBP*, *LRD*, and *LRP* (see Section 4.1 for definitions of these feature sets). The base WaldBoost detectors were created and evaluated as described in Section 4.2.

Effect of neighborhood suppression. The first experiment focuses on the general effect of *neighborhood suppression* using a single classifier to suppress single positions and using twelve such classifiers to suppress twelve different relative positions in the neighborhood. The resulting effects were measured in terms of relative speed-up of detection and relative change in *average detection rate* [2]. The tests were performed with moderately fast base detectors (4.5 - 6 features per position) and moderate target *false negative rate* of the suppression classifiers ($\alpha = 0.05$).

Results of the experiment are shown in Table 6.1 and Figure 6.2. The results indicate large differences between individual feature types. While the average number of weak hypotheses computed per position was reduced with twelve suppressed positions down to 30% for *LBP* and 40% for *LRP*, only 55% suppression was achieved for *LRD* and 65% for *Haar*. This can be explained by generally higher descriptive power of LBP and LRP features – it is reasonable to expect that they capture lot of information which is not directly relevant to their primary detection task. In general, the *average detection rate* degraded only slightly – by no more than 1% in all cases except for twelve suppressed positions with *LBP* on *MIT+CMU* and *BioID* and with *LRP* on *BioID*.

[1] Assuming standard scanning order from left to right and from top to bottom.
[2] Average detection rate equals to 1 - AMR. AMR is defined in Section 4.4.

dataset	value	Haar single	Haar 12	LBP single	LBP 12	LRD single	LRD 12	LRP single	LRP 12
BioID	ROCA(%)	-0.02	0.07	-0.48	-3.44	-0.16	-1.08	-0.24	-2.04
	Time	0.96	0.68	0.78	0.33	0.92	0.54	0.82	0.37
PAL	ROCA(%)	-0.00	-0.39	-0.08	-0.21	-0.09	-0.85	-0.05	-0.44
	Time	0.96	0.71	0.77	0.31	0.91	0.51	0.82	0.36
CMU	ROCA(%)	-0.03	-0.36	-0.27	-1.92	-0.02	-0.49	-0.08	0.01
	Time	0.93	0.62	0.74	0.31	0.93	0.62	0.87	0.47
Group	ROCA(%)	-0.04	-0.54	-0.21	-1.02	-0.02	-0.27	-0.06	-0.65
	Time	0.93	0.60	0.73	0.29	0.93	0.60	0.87	0.45

Table 6.1: The effect of *neighborhood suppression* for different features and datasets. ROCA(%) is the percentage difference between *average detection rate* without and with *neighborhood suppression*. Time represents an average number of features computed per position relative to the original detector without *neighborhood suppression*. „single“ stands for suppressing single position. „12“ stands for suppressing twelve positions with twelve suppression classifiers. Target miss rate of the suppression classifiers was 5 % and speed of the original detectors 4.5 - 6 features per position.

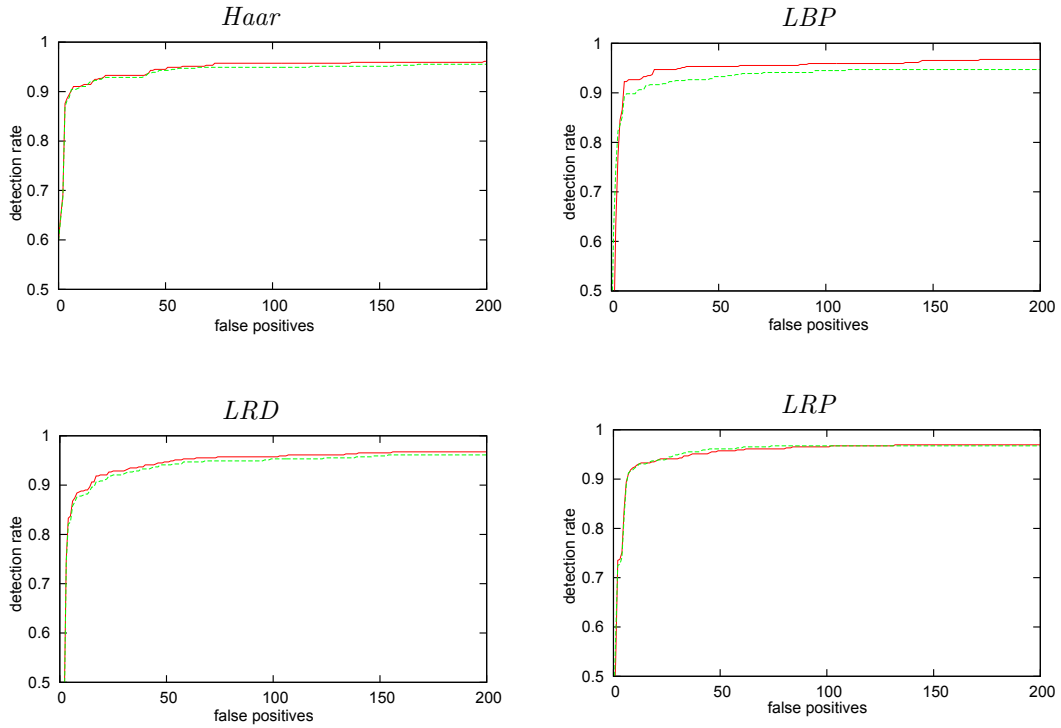


Figure 6.2: The ROC curves on *MIT+CMU* dataset without suppression (full line) and with 12 suppression classifiers (dashed line). Target miss rate α of the suppression classifiers is 5 %.

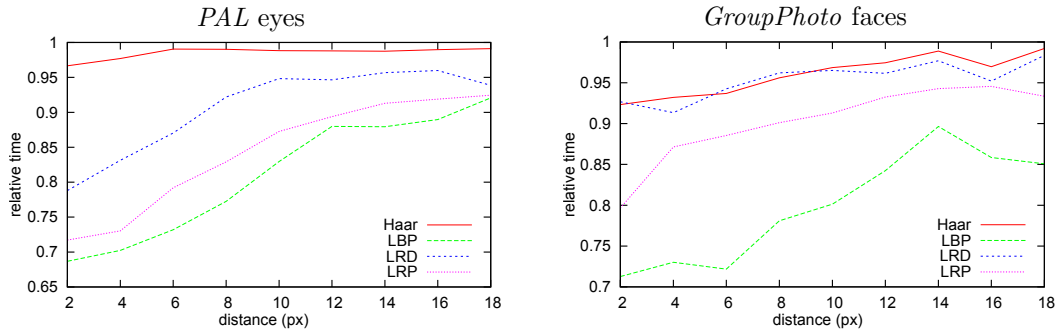


Figure 6.3: Reduction of detection time represented as average number of features computed per position relative to the original detector (y-axis) when suppressing single positions in different horizontal distance from the classified position (x-axis). Target error of the suppression classifiers is 5 %.

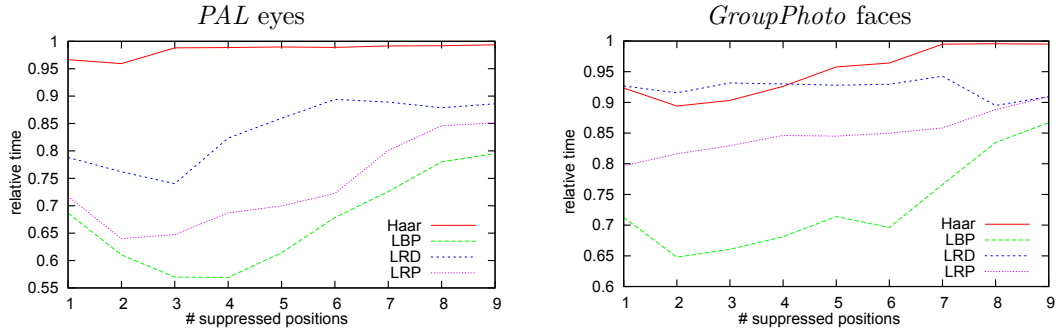


Figure 6.4: Reduction of detection time represented as average number of features computed per position relative to the original detector (y-axis) when suppressing multiple positions on single image line by single classifier. x-axis is the number of suppressed positions. Target error of the suppression classifiers is 5 %.

Suppression distance. This experiment evaluates changes in suppression ability with distance form the evaluated position. Figure 6.3 shows that suppression ability decreases relatively slowly with distance and large neighborhood of radius at least 10 pixels can be suppressed for the tested LBP and LRP classifiers.

Suppressing multiple positions. As mentioned before, single suppression classifier can suppress larger area than just a single position. Relation between speed-up and size of the area suppressed by a single classifier is shown in Figure 6.4. The results show that larger area increases speed compared to suppressing single positions. However, the speedup is not directly proportional to the area size as the suppression task becomes harder with higher number of suppressed positions. Multiple suppression classifiers would always achieve higher speed-up than a single classifier suppressing the same positions. In practical application, the optimal number of suppression classifiers would be determined by the induced computational overhead on the respective platform.

Speed-precision trade-off. If *neighborhood suppression* is to be useful, it has to provide higher speed than the simple detector for the same precision of detection. To validate this, I have trained number of WaldBoost detectors with different speeds (in terms of average number of features computed per position) for each feature type. Then, I learned three suppression classifiers with α set to 0.01, 0.05, and 0.2 for each of the WaldBoost detectors. The corresponding speeds and detection rates of the detectors are shown in Figure 6.5. Even thought only a single suppression classifier of a single position is used in this case for each of the detectors, the results clearly show that higher speed for the same detection rate can be reached by using *neighborhood suppression*.

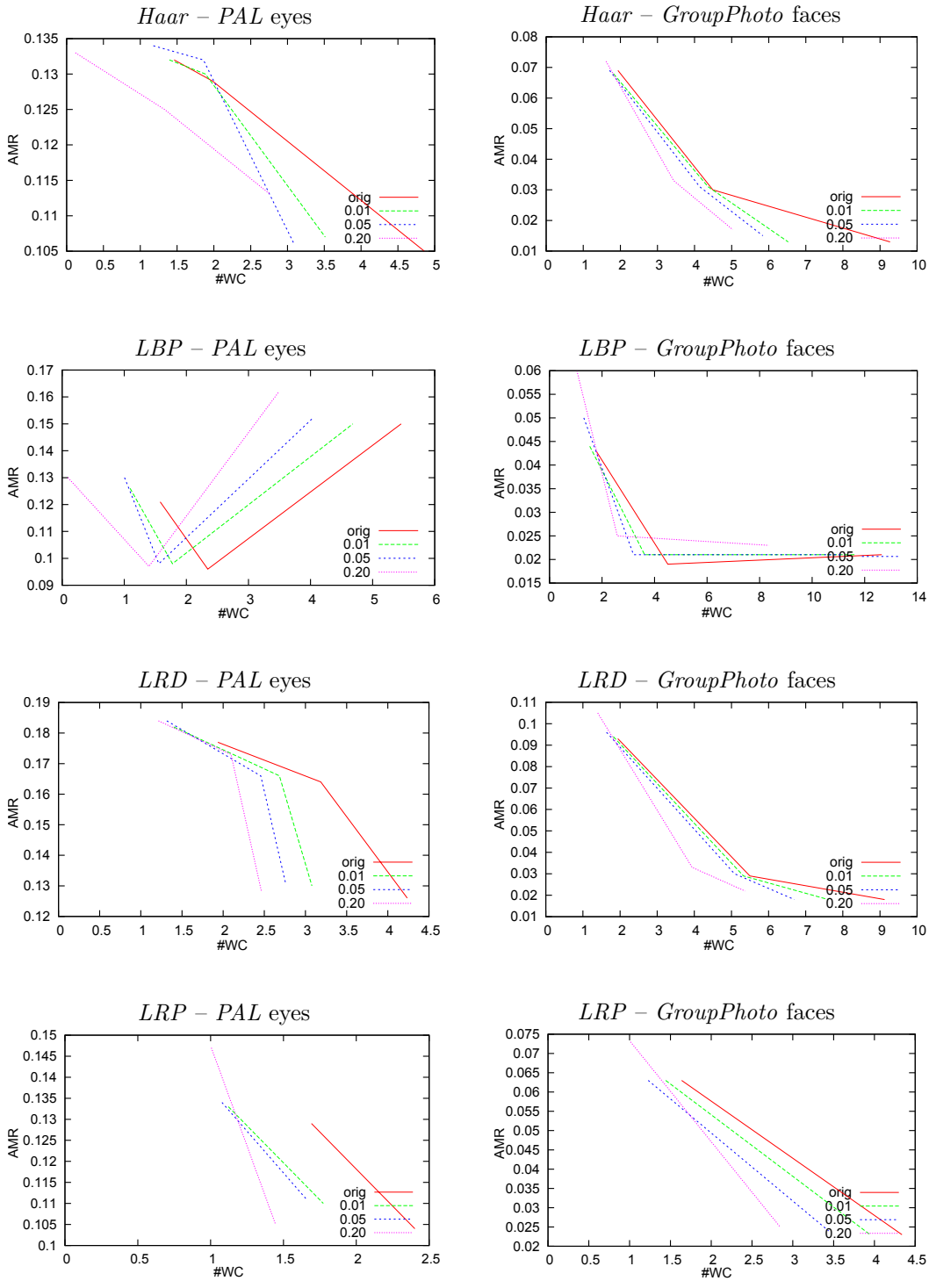


Figure 6.5: Speed-up achieved by suppressing single position for different speeds of the original detector and different target false negative rates α. *Neighborhood suppression* detectors achieve better speed-precision trade-off. Each line represents results for different α for three original detectors of different speed. X-axis is the speed of classifier in number of weak hypotheses evaluated on average per single scanned position (left is faster). Y-axis is average miss rate (lower is more accurate). Better detectors are closer to the left-bottom corner. On the left are results of eye detection on *PAL* dataset and on the right are results of frontal face detection on *GroupPhoto* dataset.

CHAPTER 7

Early non-Maxima Suppression

Non-maxima suppression is an important part of most scanning-window detectors [141, 137, 158]. It aggregates *per-window responses* of a detector into probable object positions, and it suppresses multiple detections of the same object. Non-maxima suppression usually operates locally in a small neighborhood defined by a range of positions, scales, rotations, aspect ratios, and possibly other transformations. In such neighborhood, only the highest response of the classifier which is above a specific threshold is kept and all lower responses are suppressed. As the suppressed responses have no influence on the final detections, there is no need to compute them, and it should be possible to terminate computation of the detector at such positions as soon as it is certain they will, in fact, be suppressed. Such *early terminations* would improve speed without any changes to detection results.

The main idea of *Early non-Maxima Suppression* (EnMS) is to merge existing focus-of-attention approaches with non-maxima suppression, and take the non-maxima suppression decisions from the post-processing step to the classification phase itself. Such shift of the non-maxima suppression decisions could reduce unnecessary computations with only low overhead and could significantly increase detection speed.

The EnMS algorithm proposed in this chapter is formalized as a *sequential decision strategy* and it builds upon the *Sequential Probability Ratio Test* [142] and *WaldBoost* [124] which optimize time-to-decision for a certain target error level. It creates a new sequential decision strategy based on an existing *soft cascade* detector by replacing all its rejection thresholds with variable thresholds which depend on tentative results of the detector in neighboring positions. The proposed algorithm does not require labeled training data, it only needs an existing detector and a set of images similar to the target domain of the detector.

The benefit of EnMS in the context of object detection may be only moderate since the sizes of non-maxima suppression neighborhoods are relatively small in such applications. Also, individual windows inside the non-maxima suppression neighborhood and the corresponding detector responses tend to be correlated [12, 39, 41, 20]. In other applications, such as object *localization* and *tracking* by detection, where the size of the neighborhood is larger, the benefit of EnMS should be much greater (as is demonstrated by the results in Section 7.4).

Although EnMS was primarily motivated by object detection, it is applicable in various other pattern classification tasks where the magnitude of classifier response is significant and the classifier can be divided into separate steps.

7.1 Dynamics of boosted classifiers

Potential benefit of an EnMS strategy strongly depends on dynamics of the detector for which it should be created. In essence, EnMS should work better when tentative results of the classifier are more correlated with its final decision – strong correlations allow to better anticipate the final decision.

In detectors created by WaldBoost and similar algorithms based on boosting (see Section 2.1 and Section 3.2), the correlations should indeed be strong. Such claim can be rationalized by considering that boosting algorithms produce viable classifiers in any iteration and that the weak hypotheses of boosted classifiers should be in general sorted by their discriminative power [117]. Also, existing publications support such claim [12, 20].

In order to asses the dynamics of boosted detectors, the following text presents analysis of a *real AdaBoost* [117] face detector composed of 1000 weak hypothesis based on *LRD* features (see Section 4.1). The classifier was learned on *Face training* dataset (se Section 4.3) and on 400,000 background examples. The large set of negative examples allowed the classifier to achieve very low *false positive rate* similar to classifiers with attentional structure, such as soft cascades.

The detector was run on a large set of images while collecting probability distributions $p\left(H(\mathbf{x})\,|H_t(\mathbf{x})\right)$ of the final classifier response $H(\mathbf{x})$ conditioned by intermediate value $H_t(\mathbf{x})$ at stages $t = 200$ and $t = 500$. The distributions are show in Figure 7.1. They are close to normal distributions and well separated even for $t = 200$.

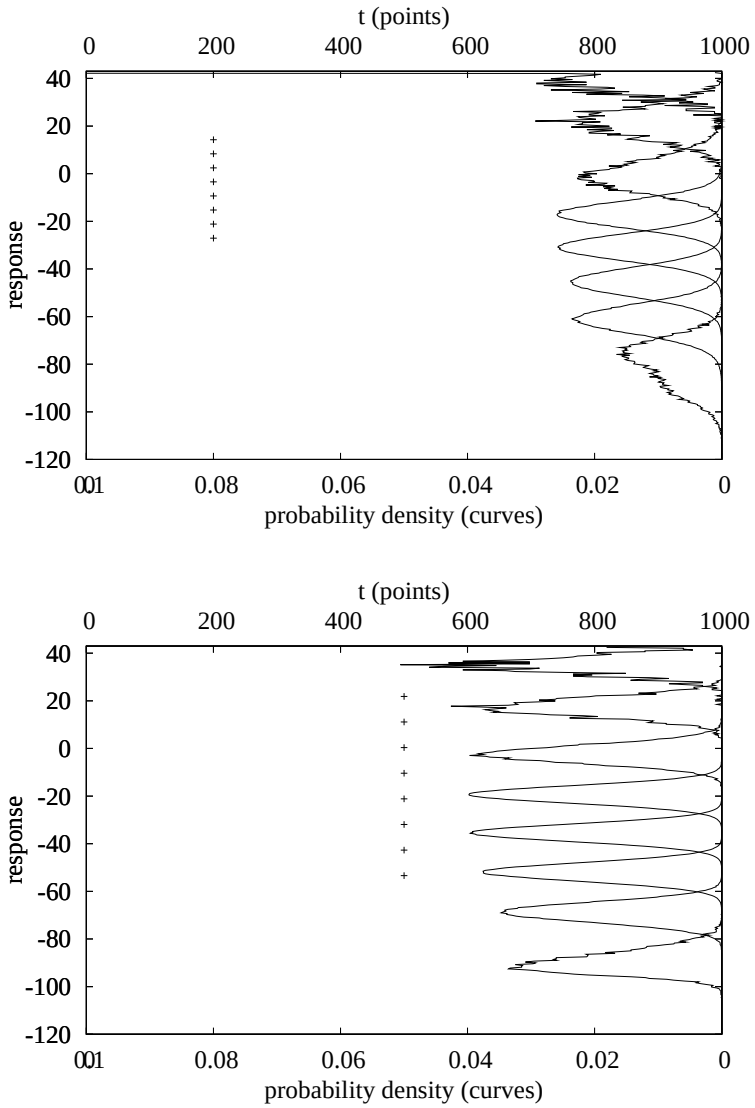


Figure 7.1: Distributions $p\left(H(\mathbf{x})\,|H_t(\mathbf{x})\right)$ of classification results $H(x)$ conditioned on eight different values of intermediate tentative results $H_t(\mathbf{x})$ (shown as small crosses). It can be clearly seen that the relative order of final classifier responses can be estimated with high probability even in early stages of the classifier. Taken from [48].

7.2 Coming to Early non-Maxima Suppression

The goal of EnMS is to find, with minimal computational effort, a sample $\mathbf{x}_{\text{best}}$ from a set of samples $\mathcal{X}_0 = \{\mathbf{x}_i\}_{i=1}^{N}$ for which the response of a classifier $H(\mathbf{x})$ is the highest – $\forall \mathbf{x}_i \in \mathcal{X} : H(\mathbf{x}_{\text{best}}) \geq H(\mathbf{x}_i)$.

The search can be speeded up by evaluating the classifier simultaneously for all samples $\mathbf{x}_i$ and by terminating some of them based on comparisons of their tentative results $H_t(\mathbf{x}_i)$.

In theory, it is possible get exact results even with the early suppression by making the early termination decisions only when they can not cause errors. Although such EnMS would give exactly the same quality of detection as the original classifier, the speed-up would be low. In order for EnMS to be effective, it has to be allowed to make occasional errors. The errors have to be low and controlled, otherwise detection quality could degrade.

In a case when only two samples are considered, the error caused by rejecting sample $\mathbf{x}_2$ in favor of $\mathbf{x}_1$ can be expressed as an integral of probabilities of all cases when the decision is wrong:

$$\begin{aligned} \varepsilon_t(\mathbf{x}_1, \mathbf{x}_2) &= p\left(H(\mathbf{x}_1) < H(\mathbf{x}_2) \,|H_t(\mathbf{x}_1), H_t(\mathbf{x}_2)\right) \\ &= \int_{\xi=-\infty}^{\infty} p\left(H(\mathbf{x}_1) = \xi \,|H_t(\mathbf{x}_1)\right) \\ &\qquad \left(\int_{\zeta=\xi}^{\infty} p\left(H(\mathbf{x}_2) = \zeta \,|H_t(\mathbf{x}_2)\right) d\zeta\right) d\xi \end{aligned} \tag{7.1}$$

This error is computed using the conditional probabilities $p\left(H(\mathbf{x}) \,|H_t(\mathbf{x})\right)$ shown in Figure 7.1.

The *two sample error* can be extended to *larger sets of samples*. When a set of samples $\mathcal{L}$ is rejected in favor of another set of samples $\mathcal{C}$, the expected error becomes:

$$\bar{\varepsilon}(\mathcal{C}, \mathcal{L}) = 1 - \prod_{\forall \mathbf{x}_\mathcal{L} \in \mathcal{L}} \left(1 - \prod_{\forall \mathbf{x}_\mathcal{C} \in \mathcal{C}} \varepsilon(\mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{L})\right). \tag{7.2}$$

Although the error caused by an EnMS decision can be computed, a question remains how to select the two sets of samples $\mathcal{L}$ and $\mathcal{C}$. Considering that the probability that a sample is $\mathbf{x}_{\text{best}}$ increases with the value of $H_t(\mathbf{x})$, it is reasonable to expect that a threshold on $H_t(\mathbf{x})$ would provide a good selection criteria for the sets $\mathcal{L}$ and $\mathcal{C}$. However, the threshold can not be static, instead it should depend on the competing samples.

The following text describes how to choose the pruning thresholds for each stage of a classifier such that the combined error of the pruning would not exceed some specified value and the highest possible speedup would be achieved at the same time.

7.3 Conditioned SPRT and EnMS

The previous text showed how to estimate *error of a single EnMS decision*. Using such error estimate, it would be possible to create an EnMS decision rule for a single classfier stage $H_t(x)$ with a required degree of confidence based on statistics of the classifier on an unlabelled image set. However, the EnMS strategy should make multiple decisions at multiple stages of the classifier in order to be effective. This section presents a method which automatically constructs such EnMS strategy for a given classifier and a target error rate.

EnMS can be formalized as a *two-class sequential decision problem* where the first class contains samples $\mathbf{x}_{\text{best}}$ which get the highest response of the whole classifier, and the second class contains all the other samples. When formalized in this way, the task is to create an *optimal strategy* which would decide at each stage of the sequential classifier for each sample from a competing set: (1) whether to reject it, (2) whether to accept it as the best sample, (3) or if this problem cannot be decided yet with high enough confidence and further information is needed. Such strategy would compute one stage of the classifier at a time and make the decision simultaneously for each of the competing samples.

The following EnMS algorithm is an extension of SPRT (see Chapter 3) and it utilizes the WaldBoost's projection trick for dependent measurements.

Conditioned SPRT. The classification task in the case of EnMS is specific in that the goal is to use information from a set of competing samples to guide the decisions about any of the individual samples. Unfortunately, the original SPRT cannot accommodate such sharing of information and has to be extended. The resulting *Conditioned Sequential Probability Ratio Test* (CSPRT) allows the decision to be conditioned by an arbitrary function over additional data. In CSPRT, the decision functions when combined with the projection trick of WaldBoost (see Equation 3.10) become:

$$S_t^*(x, z_t) = \begin{cases} +1, & \text{if } H_t(x) > \theta_B^{(t)}(z_t) \\ -1, & \text{if } H_t(x) < \theta_A^{(t)}(z_t) \\ \sharp, & \text{if } \theta_A^{(t)}(z_t) \leq H_t(x) \leq \theta_B^{(t)}(z_t) \end{cases} \tag{7.3}$$

where $z_t \in \mathcal{Z}$ is some additional conditioning data and the thresholds on the classifier response $\theta_B^{(t)} : \mathcal{Z} \to \mathbb{R}$ and $\theta_A^{(t)} : \mathcal{Z} \to \mathbb{R}$ are now functions of this additional data. The likelihood-ratio from Equation 3.9, which is used to estimate optimal $\theta_B^{(t)}(z_t)$ and $\theta_A^{(t)}(z_t)$, becomes

$$R_t = \frac{p\left(H_t(x)|z_t, y=-1\right)}{p\left(H_t(x)|z_t, y=+1\right)}. \tag{7.4}$$

The conditioning parameter z_t is not constrained to any particular form. It could be scalar or a vector, continuous or discrete. The main considerations which should guide the decision about the form of z_t and the functional forms of $\theta_B^{(t)}(z_t)$ and

$\theta_A^{(t)}(z_t)$ should be what information can be exploited to create the strategy faster and if the two threshold functions can be reliably estimated on limited data given the particular form of z_t. Generally, the more information z_t encodes, the less reliably the likelihood-ratio R_t is estimated with the same amount of data. For example, the amount of needed data increases exponentially with the dimensionality of z_t according to the curse of dimensionality [5] if no additional structural constraints are imposed.

CSPRT for EnMS. As stated above, the goal of EnMS is to find the sample $\mathbf{x}_{\text{best}}$ with maximal response of classifier $H(\mathbf{x})$ (the champion) among a set of competing samples $\mathcal{X}$ based on the intermediate result of the classifier $H_t(\mathbf{x})$. Whether a classifier response for a sample is maximal or not depends highly on the other competing samples. Considering this, it is reasonable to make z_t a function of $\mathcal{X}$.

As shown in Section 7.1, $H_t(\mathbf{x})$ becomes very good (and very likely the best) indicator of the final value of $H(\mathbf{x})$ with increasing t. With this in mind, z_t should obviously be a function of $H_t(\mathbf{x})$ of all samples from $\mathcal{X}$ and should indicate the probable highest value of $H(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$. When the strategy is evaluated synchronously in parallel for all samples from $\mathcal{X}$, such information can be extracted by:

$$z_t = \max_{\mathbf{x} \in \mathcal{X}_{t-1}} (H_t(\mathbf{x})), \tag{7.5}$$

where $\mathcal{X}_{t-1}$ is a set of samples still not decided by the previous decision function S_{t-1}.

Further, the two threshold functions have to be chosen appropriately. Similarly to WaldBoost, it is not practical for EnMS to make positive decisions – it should only reject samples. Such behavior is satisfied by setting the corresponding thresholds

$$\theta_B^{(t)}(z_t) = +\infty. \tag{7.6}$$

A reasonable form of the negative threshold $\theta_A^{(t)}(z_t)$ is

$$\theta_A^{(t)}(z_t) = z_t - \lambda_t, \tag{7.7}$$

where λ_t can be interpreted as a *handicap* of the leading sample. With this choice of $\theta_A^{(t)}(z_t)$, the condition for rejecting samples as losers from Equation 7.3 becomes

$$H_t(\mathbf{x}) < z_t - \lambda_t. \tag{7.8}$$

The term handicap for λ_t is appropriate as samples are "disqualified" from the rest of the competition if their value of $H_t(x)$ is lower than that of the leading sample with this handicap.

Although this choice of the threshold function is very simple, it performs well

Algorithm 5 Learn algorithm for EnMS strategy as published in [48].

Input: classifier $H(\mathbf{x})$ consisting of T stages $H_t(\mathbf{x})$; training sets of samples $\left\{\mathcal{X}_0^{(k)}\right\}_{k=1}^{N}$; target false negative rate α

Output: EnMS handicaps $\{\lambda_t\}_{t=1}^{t_{\max}}$

1: find champions $\mathbf{x}_{\text{best}}^{(k)} = \underset{\mathbf{x}\in\mathcal{X}_0^{(k)}}{\arg\max}\left(H(\mathbf{x})\right)$

2: count losers $L_{\text{all}} = \sum_k \left\|\mathcal{X}_0^{(k)} \setminus \left\{\mathbf{x}_{\text{best}}^{(k)}\right\}\right\|$

3: **for** each stage $t = 1$ to T **do**

4: find all $z_t^{(k)} = \underset{\mathbf{x}\in\mathcal{X}_{t-1}^{(k)}}{\max}\left(H_t(\mathbf{x})\right)$

5: $\lambda_t = \min\tilde{\lambda}_t$, such that $\alpha\frac{L_{\text{killed}}(\tilde{\lambda}_t)}{L_{\text{all}}} > \frac{C_{\text{killed}}(\tilde{\lambda}_t)}{N}$,
where the number of killed losers $L_{\text{killed}}(\tilde{\lambda}_t) =$
$L_{\text{all}} - \sum_{k=1}^{N}\left\|\left\{\mathbf{x}\,\middle|\,H_t(\mathbf{x}) > z_t - \tilde{\lambda}_t, \mathbf{x}\in\mathcal{X}_{t-1}^{(k)} \setminus \left\{\mathbf{x}_{\text{best}}^{(k)}\right\}\right\}\right\|$
and where the number of killed champions $C_{\text{killed}}(\tilde{\lambda}_t) =$
$N - \sum_{k=1}^{N}\left\|\left\{\mathbf{x}\,\middle|\,H_t(\mathbf{x}) > z_t - \tilde{\lambda}_t, \mathbf{x}\in\mathcal{X}_{t-1}^{(k)} \cap \left\{\mathbf{x}_{\text{best}}^{(k)}\right\}\right\}\right\|$

6: prune sample sets
$\mathcal{X}_t^{(k)} = \mathcal{X}_{t-1}^{(k)} \setminus \left\{\mathbf{x}\,\middle|\,H_t(\mathbf{x}) < z_t^{(k)} - \lambda_t, \mathbf{x}\in\mathcal{X}_{t-1}^{(k)}\right\}$

7: **end for**

and the optimal handicap can be reliably estimated using relatively small amount of data. Also, preliminary experiments have shown that more complex functions follow this simple form very closely when their parameters are estimated.

Learning Early non-Maxima Suppression. The process of learning an EnMS strategy is depicted in Algorithm 5. The algorithm uses the specific choices of z_t and $\theta_A^{(t)}(z_t)$ from the previous text, but it could be easily modified for other choices.

The inputs of the algorithm are the *training sets* of samples $\left\{\mathcal{X}_0^{(k)}\right\}_{k=1}^{N}$, the *target false negative rate* α of the strategy, and a classifier $H(x)$ for which the EnMS strategy should be created. The classifier must provide real-valued responses and must be evaluated in stages $H_t(x)$ where each subsequent stage gives better estimate of the final decision. These conditions are satisfied by most real-time scanning-windown object detectors [137, 120, 124, 141, 39, 41, 8, 9, 12, 58]. The individual training sets $\mathcal{X}_0^{(k)}$ each represent one competing set of samples (e.g. local image neighborhood in object detection).

The algorithm outputs the EnMS parameters, in this case the handicaps λ_t for each stage (Equation (7.7)).

In the first step of the algorithm, the champions $\mathbf{x}_{\text{best}}^{(k)}$ (there is one champion in each set of competing samples, all the rest of the samples are losers) are found in each set of competing samples $\mathcal{X}_0^{(k)}$. This requires the whole classifier $H(x)$ to be evaluated on all samples and thus presents the most computationally expensive part

Algorithm 6 Execute EnMS

Input: classifier $H(\mathbf{x})$ consisting of t_T stages $H_t(\mathbf{x})$ with corresponding handicaps λ_t, and a set of competing samples $\mathcal{X}_0$

Output: $\mathcal{X}_T$

1: **for** each stage $t = 1$ to T **do**

2: $z_t^{(k)} = \max_{\mathbf{x} \in \mathcal{X}_{t-1}} (H_t(\mathbf{x}))$

3: prune sample sets

$\mathcal{X}_t = \mathcal{X}_{t-1} \setminus \left\{ \mathbf{x} \middle| H_t(\mathbf{x}) < z_t^{(k)} - \lambda_t, \mathbf{x} \in \mathcal{X}_{t-1} \right\}$

4: **end for**

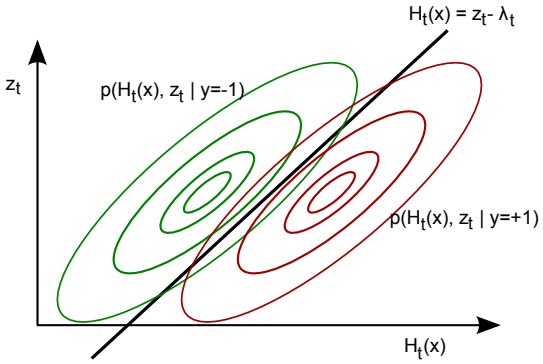


Figure 7.2: This figure shows the decision line of EnMS for single stage t together with illustrative distributions of tentative classifier responses of champions ($y = +1$) and losers ($y = -1$). The slope of the decision line is fixed to 45° and only its position, which is controlled by α_t, is adjusted during learning. Note that z_t is not a property of a specific sample, but it depends on a whole set of competing samples.

of the algorithm. However, this step can be executed only once for a classifier and stored for multiple uses. It is also possible that the classifier already contains some form of focus-of-attention, in which case the search for overall champions should not require too much computation time. Additionally, all losers should be counted during the scanning for champions. The total number of losers L_{all} is later needed when estimating the parameters of the EnMS strategy.

After the initial steps, the algorithm proceeds in iterations $t = 1 \ldots T$. In each of the iterations, a single decision function is estimated starting from the first stage of the classifier.

The iterations consist of three steps. In Step 4, the conditioning parameters $z_t^{(k)}$ (see Eq. (7.5)), which are the "best responses so far", are found for all the training sets of competing samples. Then, the only parameter of the stage decision function λ_t (from Equation 7.7) is estimated and, finally, the individual sets of competing samples are pruned by the newly estimated EnMS decision function (from Equation 7.8). Note that the gradual pruning of the sets of samples significantly reduces computational time of later iterations.

The parameter λ_t should be set such that the condition imposed by the threshold

(Equation 7.8) rejects only samples for which the likelihood-ratio R_t (Equation 7.4) satisfies

$$\begin{aligned} &R_t(\mathbf{x}, z_t) \geq \frac{1}{\alpha}, \text{i.e.} \\ &\alpha p\left(H_t(\mathbf{x}) \,|z_t, y=-1\right) \;>\; p\left(H_t(\mathbf{x}) \,|z_t, y=+1\right), \end{aligned} \tag{7.9}$$

which comes Equation 3.8 when $\beta = 0$. The condition is equivalent to

$$\alpha p\left(H_t(\mathbf{x}), z_t \,|y=-1\right) \;>\; p\left(H_t(\mathbf{x}), z_t \,|y=+1\right). \tag{7.10}$$

The two probability distributions from previous equations can be visualized as two-dimensional functions of $H_t(\mathbf{x})$ and z_t, and the condition imposed by the threshold λ_t is a separating line in this plane with 45° slope (see Figure 7.2). The condition divides the samples into two disjoint sets, which can be used to reformulate the constraint from Equation 7.10 in terms of these two sets as is done in WaldBoost (see Equation 3.13):

$$\alpha p\left(H_t(x) < z_t - \lambda_t \,|y=-1\right) \;>\; p\left(H_t(\mathbf{x}) < z_t - \lambda_t \,|y=+1\right), \tag{7.11}$$

which is already expressed in terms of λ_t. The handicap λ_t should be set as low as possible while still satisfying this constraint. In experiments presented in this thesis, the optimal values of λ_t were found by an exhaustive search in a discretized space of possible values $\tilde{\lambda}_t$.

In the algorithm, the constraint on $\tilde{\lambda}_t$ becomes

$$\alpha \frac{L_{\text{killed}}(\tilde{\lambda}_t)}{L_{\text{all}}} > \frac{C_{\text{killed}}(\tilde{\lambda}_t)}{N}, \tag{7.12}$$

where $C_{\text{killed}}(\tilde{\lambda}_t)$ gives the total number of killed champions $\mathbf{x}_{\text{best}}$ from all training sets of competing samples for a given handicap $\tilde{\lambda}_t$, and $L_{\text{killed}}(\tilde{\lambda})$ gives the number of killed losers. These sets include the champions and losers killed in all previous stages (1 to $t-1$) and those that would be rejected at stage t if the handicap was $\tilde{\lambda}_t$. L_{all} is the total number of losers in the original input sets and N is the number of champions.

The two functions $C_{\text{killed}}(\tilde{\lambda}_t)$ and $L_{\text{killed}}(\tilde{\lambda}_t)$ can be both implemented as accumulators for discrete values of $\tilde{\lambda}_t$. Such accumulators can be filled sequentially in a single pass over the individual sets of competing samples which can be processed separately and in parallel. This "stream" processing *reduces memory requirements* to an insignificant amount.

In the case that the stream processing is not an option (e.g. the amount of data per sample is too large), it is possible to estimate λ_t in the early iterations on a smaller part of the available training data. Such approach requires that L_{all} and N are updated correctly.

EnMS decision algorithm. The algorithm of applying EnMS strategy on a set of samples is described in Algorithm 6. It is essentially the *pruning mechanism* presented already in the EnMS learning algorithm. An important feature of EnMS is that it diverges very little from the standard classifier runtime: only the "*best so far*" response must be found after each stage of the classifier and then each instance's response is compared to a calculated threshold (as in the case of most other focus-of-attention strategies). Although this kind of synchronization could be undesirable on some parallel architectures, it requires only minimal additional computation and modern parallel architectures (e.g. CUDA) support constant-time voting operations, such as finding the maximal value among concurrent threads.

7.4 EnMS in face localization

The following text presents EnMS experiments on a *face localization* task. The experiments aim to asses how effective EnMS is compared to attentional detectors which process image windows independently.

The input classifier used in the experiments was a *monolithic real AdaBoost* face detector composed of 1000 weak classifiers based on *LRD* features. It is the same classifier as was used to gather the conditional distributions in Section 7.1.

EnMS strategies were learned on a separate training set of unlabelled images (described further) and the strategies were then applied to a separate testing set. Several error rates were measured and are reported in the tables of results:

- "=X" – rate of images where the EnMS strategy rejected the ultimate champion $\mathbf{x}_{\text{best}}$, i.e. $\mathbf{x}_{\text{best}} \notin \mathcal{X}_T$,
- ">X–2" – rate of images where the found best sample's score was different from $H(\mathbf{x}_{\text{best}})$ by more than 2, i.e. $\max_{\mathbf{x} \in \mathcal{X}_T} H(\mathbf{x}) < H(\mathbf{x}_{\text{best}}) - 2$,
- ">X–6" – similarly, $\max_{\mathbf{x} \in \mathcal{X}_T} H(\mathbf{x}) < H(\mathbf{x}_{\text{best}}) - 6$,
- ">2" – rate of images where the reported best sample's score was below 2, i.e. the reported maximum was not a face.

"=X" is the true error of the sequential decision strategy and it should ideally correspond to the target *false negative rate* α. For the classifier used as input, image windows well aligned on objects give $H(\mathbf{x})$ around 40–60, so decision errors which comply the ">X–2" or ">X–6" condition are still well usable for most applications.

WaldBoost detector as a baseline reference. The main question concerning the proposed EnMS approach is what is the real benefit of the additional information shared by the competing samples compared to traditional focus-of-attention mechanisms which do not share such information. To estimate this, we compared the

EnMS to WaldBoost [124] face detector with the same properties as the monolithic classifier.

Although the WaldBoost classifier does not directly aim to emulate the monolithic classifier, its task is the same. Also the experiments show that for small target *false negative rate* α, the WaldBoost classifier achieves minimal error rate with respect to the monolithic classifier. Moreover, this or similar approach would probably be used today when optimizing object localization for speed. All of the WaldBoost detectors created for different target false negative error rate rejected on average 99.99% of image windows.

In detail, the baseline WaldBoost face localization works as follows:

1. run WaldBoost detector on images and record non-rejected regions,
2. in the set of the non-rejected regions, find the one with the highest response of the *monolithic classifier*,
3. the error of this solution is the rate of images when the sample with the highest response (by the monolithic classifier) was rejected by the WaldBoost detector; the other error classes ("=X", ">X–2", ">X–6" and ">2") are evaluated similarly as in the case of EnMS from the responses of the monolithic classifier on the set of the detections reported by the fast detector.

Responses of the monolithic classifier and of the WaldBoost detector need not to be comparable. Therefore, the magnitude of response of the WaldBoost detector is not regarded at all in the comparisons. The experiments measure only the rate of cases when the sample with the highest response of the monolithic classifier is early rejected by the WaldBoost detector.

Dataset A The *Dataset A* consists of images with a dominant face in them which were collected from several sources: BioID dataset [61], XM2VTS dataset [96], and Internet search for images with "dominant face" in them. The images were re-scaled such that their width and high would not exceed 150 pixels. The training set contains 3780 images, the test set 1975 images.

EnMS results on the *Dataset A* are presented in Table 7.1 and graphically in Figure 7.3. Table 7.2 shows results of the WaldBoost reference. Figure 7.4 compares EnMS to the baseline.

In practice, the error rate "=X" is not necessarily the most important. Error rates ">X–2" and ">X–6" are much lower and for some applications can be acceptable. It is up to the application which one of them should be considered as the error of EnMS.

The benefit of the additional information used by EnMS is clear from the results. The speed-up thus gained is about 2× higher compared to the approach of first detecting the objects and then suppressing the non-maxima values (see Figure 7.4) –

target % error α	average speed-up	% error "=X"	">X–2"	">X–6"	">2"
1.00	205.4	2.48	1.27	0.35	0.00
2.00	305.0	3.70	2.08	0.66	0.00
3.00	333.6	4.71	2.58	1.01	0.00
4.00	367.3	6.08	3.85	1.42	0.00
5.00	439.8	7.24	4.91	2.23	0.00
6.00	454.2	8.96	6.08	2.89	0.05
7.00	474.7	10.23	7.14	3.44	0.05
8.00	488.5	11.70	8.41	4.35	0.10
9.00	500.5	13.01	9.27	5.16	0.25
10.00	519.9	15.39	11.34	6.28	0.51

Table 7.1: EnMS results on *Dataset A*. **Target error** is the target false negative rate (in %) set for the training process, **speed-up** is averaged over the testing dataset, **4 error rates** (in %) are actually measured on the testing set: "=X" is the fraction of images when the maxima reported by the EnMS was different from the real maxima, ">X-2" and ">X-6" are the fractions of images when the maxima reported was different from the real maxima by more than 2 and 6 respectively, ">2" is the fraction of images where the reported maximal value was below 2 (practically a non-face).

WB α	average speed-up	% error "=X"	">X–2"	">X–6"	">2"
0.02	58.4	1.37	1.01	0.20	0.00
0.05	105.6	3.04	2.08	0.76	0.00
0.10	183.3	7.14	5.06	2.43	0.10
0.20	310.1	16.41	12.46	7.24	0.30

Table 7.2: Baseline WaldBoost results on Dataset A. The table is structured identically as Table 7.1.

which can be regarded as the state-of-the-art solution. This is true for all the error types "=X", ">X–2", ">X–6" and ">2".

Dataset B The *Dataset B* was created from images from group "*portraits*" (training set) and "*just_faces*" (test set) from server flicker.com. In the images, near-frontal faces were semi-automatically annotated. A total of 84,251 faces were annotated in the training set and 6704 in the test set. The images were then rescaled so that the size of faces was 50-by-50 pixels. Further, the images were cut to a defined size with the faces centered in the middle. The sizes were 70-by-70, 85-by-85, 100-by-100, 120-by-120, and 150-by-150 pixels resulting in five versions of the testing set. The size of training images was 100-by-100 pixels.

Results of EnMS on *Dataset B* with 100-by-100 testing images are given in Table 7.3 and graphically in Figure 7.5 – the figure contains results of the WaldBoost baseline as well. Note that performance on *Dataset B* is notably worse than on

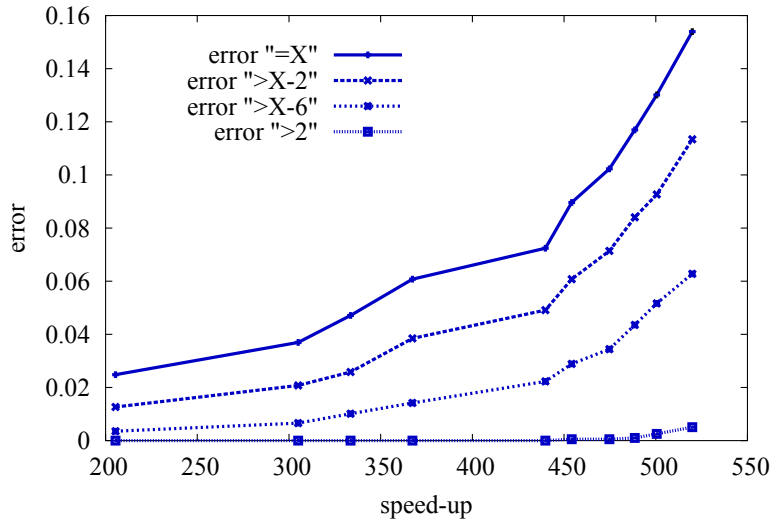


Figure 7.3: EnMS results on *Dataset A* – see table 7.1 for numerical values.

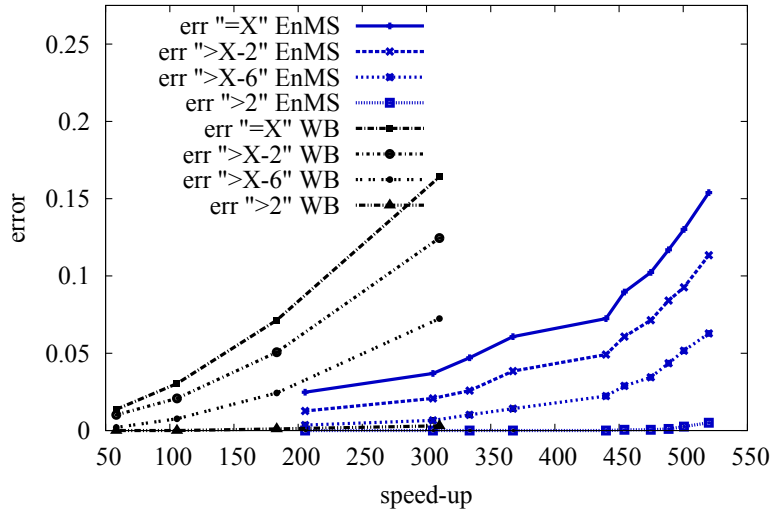


Figure 7.4: Comparison of EnMS and WaldBoost baseline on *Dataset A*.

Dataset A. The extent of the performance drop is similar for the EnMS and for the WaldBoost baseline, and it can easily be explained by higher difficulty of images in this dataset – it contains more cluttered background, non-frontal faces, partially occluded faces, etc. Also, the performance of EnMS is negatively influenced by the smaller size of testing images in this experiment compared to *Dataset A* (150-by-150 pixels). Still, the performance of EnMS is approximately twice as good as the WaldBoost baseline.

Effect of neighborhood size. As stated in the previous text, the testing images of *Dataset B* were cropped to five different sizes 70-by-70, 85-by-85, 100-by-100, 120-by-120, and 150-by-150 pixels. The size of faces is the same in all the versions, so they only differ in the amount of background clutter they contain. EnMS should be more effective on larger images as the larger images contain more competing

target % error	average speed-up	% error "=X"	">X–2"	">X–6"	">2"
1.00	103.3	3.07	1.48	0.31	0.03
2.00	119.4	4.77	2.31	0.60	0.06
3.00	136.5	6.01	3.18	0.84	0.06
4.00	152.7	6.80	3.76	1.24	0.06
5.00	165.0	8.86	5.06	1.67	0.09
6.00	175.4	10.10	5.95	2.09	0.16
7.00	198.8	12.86	7.97	3.06	0.19
8.00	215.5	15.32	9.83	3.65	0.31
9.00	222.4	15.16	9.80	3.85	0.31
10.00	236.3	17.99	11.99	4.79	0.39
12.00	257.8	16.38	11.22	4.95	0.39
14.00	264.6	16.77	12.28	5.85	0.51
16.00	330.6	29.52	22.24	10.99	1.18

Table 7.3: EnMS results on *Dataset B* with image size 100-by-100 pixels. The table is structured identically as Table 7.1.

windows. To asses this relation, EnMS strategy learned on the training set (all samples 100-by-100) was executed on the five different test sets (see Figure 7.6 for results). Note that the average speed-up of EnMS increases with the number of competing samples. The speed-up is roughly 2× higher on 150-by-150 images than on 70-by-70 images.

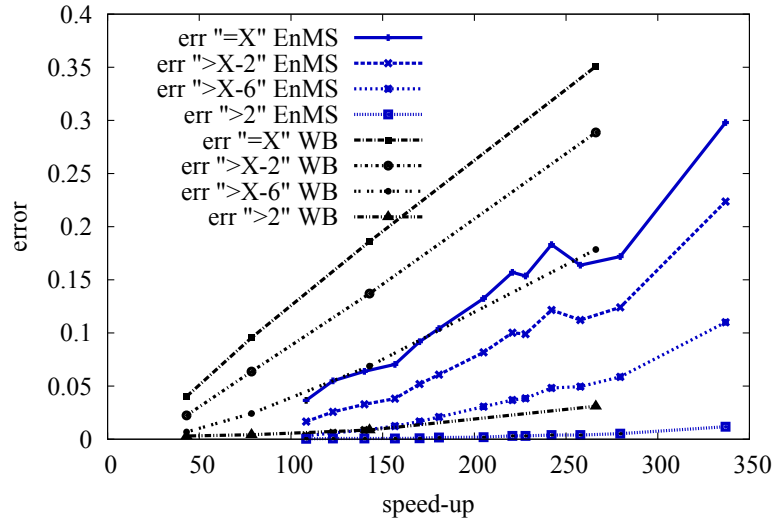


Figure 7.5: Comparison of EnMS and WaldBoost baseline on *Dataset B* with image size 100-by-100 pixels.

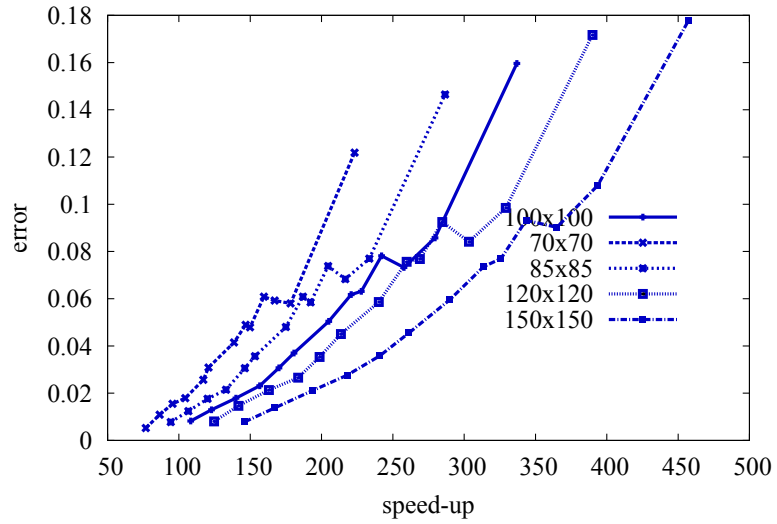


Figure 7.6: Performance of EnMS on test datasets with samples of different dimensions.

CHAPTER 8

Discussion

The experimental results of *neighborhood suppression* (Section 6.3) and of EnMS (Section 7.4) indicate that both methods effectively share information between neighboring image positions, and that they are both able to improve detectors which process image windows independently. EnMS achieved roughly 2× speed-up at same error level compared to WaldBoost in *face localization* on small images. *Neighborhood suppression* improved speed of face detectors up to 3× at the expense of only minor reduction of detection rates (average detection rate was reduced in most cases no more than by 2%). The results show that both *neighborhood suppression* and EnMS provides better *speed-precision trade-off* compaterd to WaldBoost baseline.

The improvements in speed are impressive considering that the baseline WaldBoost detectors are already very fast – the fastest ones compute as few as two features per image window. *Neighborhood suppression* was able to reduce the average number of computed features per window down to single feature in some of the experiments.

From the nature of EnMS, its performance should not depend on specific properties of the detector it is based on, such as which features it uses, as long as the detector conforms to the basic requirements of the method. On the contrary, behavior of *neighborhood suppression* depends strongly on the type of features(see Table 6.1, Figure 6.3, and Figure 6.4). The suppression is much more effective with features which encode rich information (LBP) and it provides only mediocre benefit for simple features, such as Haar-like features. A reasonable explanation is that the simple features provide information specific to the original detection task they were selected for and not much else. The richer features, on the other hand, encode much more information beyond that for which they were selected.

The experiments in this thesis measured speed as a *number of features computed per image window*. Such measure ignores computational overhead of *neighborhood*

suppression and EnMS. The overhead would lower speed improvements in real applications. However, the proposed algorithms are simple, and it is reasonable to expect they would be computationally efficient on existing platforms and the induced overhead would be small. Computational overhead of EnMS could be reduced to almost zero by making the suppression decisions only for small number of stages (as is done by Dollár et al. [20]). Efficient implementation of *neighborhood suppression* is discussed in Section 6.2.

8.1 Neighborhood suppression.

Although *neighborhood suppression* aims to improve speed of an existing detector by sacrificing precision in a controlled way, it, in fact, provides better speed-precision trade-off as does EnMS (as shown in Figure 6.5). Combination of *neighborhood suppression* with a slow and more precise detector achieves on average better detection accuracy compared to WaldBoost detector with the same speed.

A downside of the *neighborhood suppression* as described in this thesis is that it does not provide a mechanism how to create a detector from scratch with specific error or speed. The two stage process which improves existing detectors has its benefits, but the only way an optimal *neighborhood suppression* detector with a specific error rate can be created this way is to try to learn multiple *neighborhood suppressions* for multiple detectors and select the best combination. Such approach would be tedious and time-consuming. In order to be able to create optimal detectors with specific error, the *neighborhood suppression* should be integrated with the detector learning algorithm. While such integration is certainly possible, it may require significant changes of existing learning tools to be able to work with dependencies between training image windows. Alternatively, *neighborhood suppression* could update rejection thresholds of the original detector while learning the suppression classifiers. Such approach would be similar to the combination of a *soft cascade* and *excitatory cascade* of Dollár et al. [20] and may provide good compromise with respect to complexity of training.

Although the *neighborhood suppression* effectively utilizes information shared in neighborhoods, it is not optimal. The evidence gathered at neighboring locations is only used to potentially reject image windows. Such behavior is similar to *multi-stage detectors*, such as the *rejection cascade* of Viola and Jones [137], which discard evidence between its stages. As shown by Sochman [122] and others [152, 7], it is possible to connect stages of such detectors. Similarly, it should be possible to use the evidence extracted from neighboring locations as a starting point for decision at current image location. However, the interconnection of neighboring detectors presents specific challenges as the amount of evidence gathered in the neighborhood depends on the image content and on possible local interactions in such detection

process.

8.2 Early non-Maxima Suppression

The task that EnMS solves is the same as the one addressed by *efficient subwindow search* by Lampert et al. [77] and by the *inhibitory cascades* of Dollár et al. [20]. It is also similar to the taks of *recursive coarse-to-fine localization* by Pedersoli et al. [106]. The problem which these methods solve is to find an image window with the highest response of a detector in a set of windows.

Unlike EnMS, the *efficient subwindow search* is guaranteed to always find the optimal window, but it can only be applied to simple detectors for which an efficient upper bound on detector response exists. Although EnMS is, in a way, constrained with respect to what classifiers it can be applied to as well, it can support classifiers of arbitrary complexity and strength.

The recursive coarse-to-fine localization is an ad-hoc process which, unlike EnMS, does not provide any indications of what is the error caused by the coarse-to-fine structure of the detector. In fact, EnMS could be applied to a multi-stage coarse-to-fine detector which would result in a detector with similar behavior, but with controlled error and optimal computational complexity for the target error and detector structure.

The closest competitors of EnMS are the *inhibitory cascades* of Dollár et al. [20] which let positions with strong tentative detector responses suppress other positions in a local neighborhood in almost exactly the same way as EnMS. Both methods enhance existing detectors, have similar requirements on the detectors, and require only unlabeled images as training data. *Inhibitory cascades* and EnMS differ in two aspects: (1) exact functional form of suppression conditions, (2) method for choosing suppression thresholds. As the following text argues, the choices made by Dollár et al. in *inhibitory cascades* are not optimal in contrast to EnMS. Considering that both methods have the same computational overhed, EnMS should be considered superior.

Unlike EnMS, *inhibitory cascades* base their decisions on the *ratio* of tentative results – a window $\mathbf{x}$ with competing neighbors $\mathcal{X}$ gets suppressed if

$$\frac{H_t(\mathbf{x})}{H_t(\mathbf{x}_{\max})} \;<\; \theta_t, \tag{8.1}$$

where $\mathbf{x}_{\max} = \arg\max_{\mathbf{x}\in\mathcal{X}} H_t(\mathbf{x})$. Although this condition makes certain intuitive sense at the first sight, it becomes less reasonable when the underlying meaning of $H_t(\mathbf{x})$ is considered.

The value of $H_t(\mathbf{x})$ can be directly linked to a *logarithm of a-posterior probability*

ratio of the two corresponding classes [35]:

$$\lim_{T\to\infty} H_T(\mathbf{x}) = \frac{1}{2}\log\frac{p(y=+1|\mathbf{x})}{p(y=-1|\mathbf{x})}. \tag{8.2}$$

The limit can be further rewritten using Bayes formula [124] to:

$$\begin{aligned}\lim_{T\to\infty} H_T(\mathbf{x}) &= -\frac{1}{2}\log R_T(\mathbf{x}) + \frac{1}{2}\log\frac{P(+1)}{P(-1)} \\ &= -\frac{1}{2}\log\frac{p(\mathbf{x}|y=-1)}{p(\mathbf{x}|y=+1)} + \frac{1}{2}\log\frac{P(+1)}{P(-1)}\end{aligned} \tag{8.3}$$

Even though the limit is defined for infinitely long detectors, it can be safely used for certain reasoning about shorter detectors as well.

The limit can be substituted into the condition used by *inhibitory cascades* (Equation 8.1), resulting in:

$$\sqrt[\frac{p(\mathbf{x}_{\max}|y=-1)}{p(\mathbf{x}_{\max}|y=+1)}]{\frac{p(\mathbf{x}|y=-1)}{p(\mathbf{x}|y=+1)}} > e^{\theta_t}. \tag{8.4}$$

Seeing the condition in this form makes it clear that it does not have any clear or meaningful interpretation.

On the other hand, the condition used by EnMS (Equation 7.8), which can be rewritten as

$$H_t(\mathbf{x}) - H_t(\mathbf{x}_{\max}) < \lambda_t, \tag{8.5}$$

can be similarly expressed by substituting the limit from Equation 8.3 as

$$\log\frac{p(\mathbf{x}|y=+1)}{p(\mathbf{x}|y=-1)} - \log\frac{p(\mathbf{x}_{\max}|y=+1)}{p(\mathbf{x}_{\max}|y=-1)} < \frac{1}{2}\lambda_t. \tag{8.6}$$

As the *log-likelihood ratios* can be understood as *confidence levels*, the EnMS condition can be interpreted as: Reject $\mathbf{x}$ if it is at least by $\frac{1}{2}\lambda_t$ less likely to contain an object than $\mathbf{x}_{\max}$. The condition does not depend on the classifier's confidence for the two individual windows, only on the difference of confidence. This is a necessary property for the condition to work the same way in regions which certainly contain an object as well as in regions which are ambiguous.

The second difference between EnMS and *inhibitory cascades* is that Dollár at al. set the thresholds such that the decisions in all stages induce the same constant error. Such approach does not take into account that the errors are traded off by different speed-up in the different stages. Generally, the computational savings by rejections in early stages are much greater per window compared to rejections in late stages as larger part of the classifier remains to be computed. EnMS takes these differences into account and produces optimal time-to-decision detector for the target error.

The benefits of EnMS, and any other similar strategy, depend on the number of

competing image windows (see Figure 7.6). The larger the set is, the higher speed-ups can be achieved. This is not a problem in *face localization* or *tracking by detection*, but it may limit the benefits in purely detection tasks. The sizes of *non-maxima suppression neighborhoods* in practical object detection applications are similar to the face localization on images 70-by-70 (considering maximum object overlap of 50%). While EnMS still outperforms WaldBoost detectors on such neighborhoods, it should be extended to include fixed rejection thresholds in order to guaranteed to outperform basic sliding-window detectors regardless the neighborhood size. Such extension would be similar to the combination of a *soft cascade* and an *inhibitory cascade* in [20].

EnMS as presented in this thesis is learned to find the optimal image window in local neighborhood with certain probability. This requirement fits the standard non-maxima suppression approaches. In some applications, the requirement could be extended to find n-best image windows with certain probability. EnMS could by adapted for such tasks by changing definition of training sets.

8.3 Comparison of EnMS and neighborhood suppression

Although both EnMS and *neighborhood suppression* were demonstrated on simple boosted (or WaldBoost) detectors, the approaches can be directly applied to other detectors with similar structure which are composed of stages. These include all detectors with attentional structure [110, 137, 90, 136, 152, 59, 95, 37, 122, 7, 124, 12]. Monolithic detectors [18] would have to be split into meaningful parts first.

Neighborhood suppression and EnMS are, in a certain sense, complementary and most powerful in different situations. *Neighborhood suppression* does not assume anything beyond what is required for existing scanning-window detectors, it behaves as a standard scanning-window detector and it can process image positions sequentially. In essence, it just extends existing attentional structures by an early rejection stage which extracts information from neighboring positions and which is very cheap as it relies on features which would be computed anyway by the neighboring classifiers. The fact that a detector with *neighborhood suppression* behaves as a standard detector with attentional structure implies that the suppression can not help at regions which are likely to contain objects of interest. Although the suppression stage can rely on richer information in ambiguous regions, as the original detector tends to compute more features there, it is unreasonable to expect that the *neighborhood suppression* would be able to reject windows which the original detector itself is unable to reject. A reasonable conclusion is that *neighborhood suppression* may have lower effect in images which contain many objects of interest.

EnMS, on the other hand, diverges from the standard scanning window procedure

and assumes that only the locally maximal position are of interest. It is inherently parallel and requires all image positions to be evaluated concurrently. EnMS should remain effective even in regions which are likely to contain an object of interest as it adapts to the image content. The only requirement for EnMS to be effective is that the competing regions differ in how likely they contain an object. In fact, it is reasonable to expect that EnMS would improve a detector with fixed rejection thresholds mostly in ambiguous regions where the fixed thresholds are not effective.

Neighborhood suppression is closely linked with object detection as it explicitly relies on topological relations. On the other hand, EnMS can be directly applied on any task, even outside computer vision, which uses classifiers and which is interested in finding the highest response in a set of candidates.

CHAPTER 9

Conclusions

This thesis studied scanning-window detectors and, especially, how such detectors can be improved by sharing local information and by interlinking decisions at neighboring positions. This general idea resulted in two novel methods, *neighborhood suppression* and *Early non-Maxima Suppression*, which improve existing scanning-window detectors by utilizing the information shared between neighboring image positions. The methods provide higher speed (up to 2× faster in experiments) at the same detection rates or conversely better detection rates at the same speed compared to detectors which process image windows independently.

Both methods were developed into practical algorithms which can be used in real world applications with minimum changes to existing detection engines on various platforms including highly parallel environments, such as FPGA and GPU. Especially, EnMS matches the nature of highly parallel platforms well, as it requires a high number of competing hypotheses to be computed concurrently in parallel. The novel methods have potential to improve object detectors in a wide range of applications from embedded devices and smart cameras to high-throughput GPU clusters in cloud-based photo galleries and surveillance systems.

The novel algorithms are build upon *Sequential Probability Ratio Test* [142] and *WaldBoost* [124] which optimize time-to-decision for a certain target error level. These ideas were directly used in *neighborhood suppression* and extended into *Conditioned Sequential Probability Ratio Test* for EnMS.

Although both *neighborhood suppression* and EnMS were tested on boosted detectors with simple image features and *soft-cascade* attentional structure, they are not in any way limited to these detectors. *Neighborhood suppression* can be directly applied to any detector which can be decomposed into smaller predictive functions (such as features in boosted classifiers). EnMS requires the original detector to be

composed of stages which give progressively more confident predictions of the final decision. Also, EnMS, being inspired by non-maxima suppression, finds only the region with the highest response of the detector in a local group of competing regions. Although the requirements of EnMS are stricter, it can be applied to wider range of tasks even outside computer vision – any task which searches for the highest response of a suitable classifier in a group of competing objects.

Experimental results show that *neighborhood suppression* is able to use information from neighboring positions effectively to suppress evaluation of a detector; however, the same information could be potentially used even more effectively as initial evidence by the detector. Such tight integration should be further explored as it could lead to significant speed-up without any degradation of detection quality.

EnMS as presented in this thesis becomes less effective on small neighborhoods, such as those used by non-maxima suppression in face detection. To ensure competitiveness of EnMS in such situations, it should be extended by adding WaldBoost-style fixed rejection thresholds. Adding such thresholds does not presents any difficulties; however, an algorithm which sets both types of thresholds in a unified way such that the speed is optimized for specific target error rate should be developed.

Ideally, EnMS should be combined with *neighborhood suppression* or with a method similar to the *excitatory cascade* of Dollár et al. [20]. Such combination would benefit from the complementary strengths of the methods and it could result in very fast detectors.

In addition to the novel algorithms, the thesis presented comprehensive experiments which compared common types of features in several detection tasks. These experimental results show that WaldBoost detectors with *Local Binary Patterns* give consistently good result across a wide range of detection tasks.

Bibliography

[1] Timo Ahonen, Abdenour Hadid, and Matti Pietik?inen. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[2] Bogdan Alexe, Nicolas Heess, Yee Whye Teh, and Vittorio Ferrari. Searching for objects driven by context. In *NIPS*, pages 890–898, 2012.

[3] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Pedestrian detection at 100 frames per second. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2903–2910. IEEE, June 2012.

[4] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool. Seeking the strongest rigid detector. In *CVPR*, 2013.

[5] Christopher M Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[6] M B Blaschko and C H Lampert. Object Localization with Global and Local Context Kernels. In *British Machine Vision Conference*, 2009.

[7] Bo Wu, Haizhou Ai, Chang Huang, and Shihong Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pages 79–84. IEEE, 2004.

[8] Lubomir Bourdev and Jonathan Brandt. Robust Object Detection Via Soft Cascade. In *CVPR*, 2005.

[9] S Charles Brubaker, Matthew D Mullin, and James M Rehg. Towards Optimal Training of Cascaded Detectors. In *In ECCV06*, pages 325–337, 2006.

[10] S Charles Brubaker, Jianxin Wu, Jie Sun, Matthew D Mullin, and James M Rehg. On the Design of Cascades of Boosted Ensembles for Face Detection. *Int. J. Comput. Vision*, 77(1-3):65–86, 2008.

[11] N.J. Butko and J.R. Movellan. Optimal scanning for faster object detection. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2751–2758. IEEE, June 2009.

[12] Zhang Cha and Paul Viola. Multiple-Instance Pruning For Learning Efficient Cascade Detectors. In *NIPS*, 2007.

[13] Jie Chen, Shiguang Shan, Peng Yang, Shengye Yan, Xilin Chen, and Wen Gao. Novel Face Detection Method Based on Gabor Features. In *Sinobiometrics 2004*, Lecture Notes in Computer Science, pages 90–99. Springer Berlin / Heidelberg, 2004.

[14] Junguk Cho, Shahnam Mirzaei, Jason Oberg, and Ryan Kastner. Fpga-based face detection system using Haar classifiers. In *Proceeding of the ACM/SIGDA international symposium on Field programmable gate arrays - FPGA '09*, page 103, New York, New York, USA, February 2009. ACM Press.

[15] O Chum, M Perdoch, and J Matas. Geometric min-Hashing: Finding a (thick) needle in a haystack. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:17–24, 2009.

[16] O. Chum and A. Zisserman. An Exemplar Model for Learning Object Classes. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[17] Navneet Dalal. *Finding people in images and videos.* PhD thesis, Institut National Polytechnique de Grenoble, 2006.

[18] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.

[19] C Demirkir and B Sankur. Face Detection Using Look-Up Table Based Gentle AdaBoost. In *Audio- and Video-Based Biometric Person Authentication 2005*, page 339, 2005.

[20] Piotr Dollár, Ron Appel, and Wolf Kienzle. Crosstalk cascades for frame-rate pedestrian detection. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *ECCV'12 Proceedings of the 12th European conference on Computer Vision -Volume Part II*, Lecture Notes in Computer Science, pages 645–659, Berlin, Heidelberg, October 2012. Springer-Verlag Berlin.

[21] Piotr Dollar, Serge Belongie, and Pietro Perona. The Fastest Pedestrian Detector in the West. In *Procedings of the British Machine Vision Conference 2010*, pages 68.1–68.11. British Machine Vision Association, 2010.

[22] Piotr Dollár, Zhuowen Tu, Pietro Perona, and Serge Belongie. Integral Channel Features. In *BMVC*, 2009.

[23] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: an evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–61, April 2012.

[24] Markus Enzweiler and Dariu M Gavrila. Monocular Pedestrian Detection: Survey and Experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:2179–2195, 2009.

[25] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable Object Detection using Deep Neural Networks. In *To appear in CVPR 2014*, December 2014.

[26] C G M Snoek Et al. The MediaMill TRECVID 2009 Semantic Video Search Engine. In *TRECVID 2009: Participant Notebook Papers and Slides*, Gaithersburg, MD, US, 2009. National Institute of Standards and Technology.

[27] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–45, September 2010.

[28] P.F. Felzenszwalb, R.B. Girshick, and D. McAllester. Cascade object detection with deformable part models. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010.

[29] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–264–II–271. IEEE Comput. Soc, 2003.

[30] Francois Fleuret and Donald Geman. Coarse-to-Fine Face Detection. *International Journal of Computer Vision*, 41(1-2):85–107, January 2001.

[31] Y Freund and R Schapire. A short introduction to boosting. *Japonese Society for Artificial Intelligence*, 14(5):771–780, 1999.

[32] Yoav Freund. Boosting a weak learning algorithm by majority. In *COLT '90: Proceedings of the third annual workshop on Computational learning theory*,

pages 202–216, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[33] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, London, UK, 1995. Springer-Verlag.

[34] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[35] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. *Annals of Statistics*, 28, 1998.

[36] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. *Annals of Statistics*, 28(2):337–655, 2000.

[37] Bernhard Fröba and Andreas Ernst. Face detection with the modified census transform. In *FGR' 04 Proceedings of the Sixth IEEE international conference on Automatic face and gesture recognition*, pages 91–96, Washington, DC, USA, May 2004. IEEE Computer Society.

[38] Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):826–834, September 1983.

[39] H. Grabner and H. Bischof. On-line Boosting and Vision. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)*, volume 1, pages 260–267. IEEE.

[40] H. Grabner, M Grabner, and H Bischof. Real-Time Tracking via On-line Boosting. In *Procedings of the British Machine Vision Conference 2006*, pages 6.1–6.10. British Machine Vision Association, 2006.

[41] Helmut Grabner, Jan Sochman, Horst Bischof, and Jiri Matas. Training sequential on-line boosting classifier for visual tracking. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, December 2008.

[42] Giovanni Gualdi, Andrea Prati, and Rita Cucchiara. Multi-stage sampling with boosting cascades for pedestrian detection in images and videos. In *ECCV'10 Proceedings of the 11th European conference on Computer vision: Part VI*, pages 196–209. Springer-Verlag, September 2010.

[43] Giovanni Gualdi, Andrea Prati, and Rita Cucchiara. A multi-stage pedestrian detection using monolithic classifiers. In *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 267–272. IEEE, August 2011.

[44] Alfred Haar. Zur Theorie der orthogonalen Funktionensysteme. *Mathematische Annalen*, 69:331–371, 1910.

[45] D Haussler. Over view of the Probably Approximately Correct (PAC) Learning Framework. 1995.

[46] Daniel Hefenbrock, Jason Oberg, Nhat Tan Nguyen Thanh, Ryan Kastner, and Scott B. Baden. Accelerating Viola-Jones Face Detection to FPGA-Level Using GPUs. In *2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines*, pages 11–18. IEEE, 2010.

[47] Bernd Heisele, Thomas Serre, Sam Prentice, and Tomaso Poggio. Hierarchical classification and feature reduction for fast face detection with support vector machines. *Pattern Recognition*, 36(9):2007–2017, September 2003.

[48] Adam Herout, Michal Hradis, and Pavel Zemik. EnMS: Early non-Maxima Suppression. *Pattern Analysis and Applications*, 2011(1111):10, 2011.

[49] Adam Herout, Michal Hradiš, and Pavel Zemčík. EnMS: Early non-Maxima Suppression. *Pattern Analysis and Applications*, 2012(2):121–132, 2012.

[50] Adam Herout, Radovan Jošth, Roman Juránek, Jiří Havel, Michal Hradiš, and Pavel Zemčík. Real-time object detection on CUDA. *Journal of Real-Time Image Processing*, 2010(1111):12, 2010.

[51] Adam Herout, Radovan Jošth, Pavel Zemčík, and Michal Hradiš. GP-GPU Implementation of the „Local Rank Differences“ Image Feature. In *Proceedings of International Conference on Computer Vision and Graphics 2008*, Lecture Notes in Computer Science, pages 1–11. Springer Verlag, 2008.

[52] Adam Herout, Pavel Zemčík, Roman Juránek, and Michal Hradiš. Implementation of the „Local Rank Differences“ Image Feature Using SIMD Instructions of CPU. In *Proceedings of Sixth Indian Conference on Computer Vision, Graphics and Image Processing*, page 9. IEEE Computer Society, 2008.

[53] Guillaume Heusch, Yann Rodriguez, and Sebastien Marcel. Local Binary Patterns as an Image Preprocessing for Face Authentication. In *FGR '06: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, pages 9–14, Washington, DC, USA, 2006. IEEE Computer Society.

[54] Erik Hjelmå s and Boon Kee Low. Face Detection: A Survey. *Computer Vision and Image Understanding*, 83(3):236–274, September 2001.

[55] C Hou, H Z Ai, and S H Lao. Multiview Pedestrian Detection Based on Vector Boosting. In *ACCV07*, Lecture Notes in Computer Science, pages I: 210–219, Tokyo, 2007.

[56] Michal Hradiš. Framework for Research on Detection Classifiers. In *Proceedings of Spring Conference on Computer Graphics*, pages 171–177, 2008.

[57] Michal Hradiš, Adam Herout, and Pavel Zemčík. Local Rank Patterns - Novel Features for Rapid Object Detection. In *Proceedings of International Conference on Computer Vision and Graphics 2008*, Lecture Notes in Computer Science, pages 1–12, 2008.

[58] C Huang, H Z Ai, Y Li, and S H Lao. High-Performance Rotation Invariant Multiview Face Detection. *PAMI*, 29(4):671–686, 2007.

[59] Chang Huang, Haizhou Ai, Bo Wu, and Shihong Lao. Boosting Nested Cascade Detector for Multi-View Face Detection. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2*, pages 415–418, Washington, DC, USA, 2004. IEEE Computer Society.

[60] Chen Huang and Frank Vahid. Scalable object detection accelerators on FPGAs using custom design space exploration. In *2011 IEEE 9th Symposium on Application Specific Processors (SASP)*, pages 115–121. IEEE, June 2011.

[61] Oliver Jesorsky, Klaus J Kirchberg, and Robert Frischholz. Robust Face Detection Using the Hausdorff Distance. In *AVBPA '01: Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95, London, UK, 2001. Springer-Verlag.

[62] Haipeng Jia, Yunquan Zhang, Weiyan Wang, and Jianliang Xu. Accelerating Viola-Jones Facce Detection Algorithm on GPUs. In *2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems*, pages 396–403. IEEE, June 2012.

[63] Hui-Xing Jia and Yu-Jin Zhang. Fast Human Detection by Boosting Histograms of Oriented Gradients. In *Fourth International Conference on Image and Graphics (ICIG 2007)*, pages 683–688. IEEE, August 2007.

[64] Hongliang Jin, Qingshan Liu, Hanqing Lu, and Xiaofeng Tong. Face Detection Using Improved LBP under Bayesian Framework. In *Third International Conference on Image and Graphics (ICIG'04)*, pages 306–309. IEEE, December 2004.

[65] Michael Jones and Paul Viola. Fast multi-view face Detection. Technical report, Mitsubishi Electric Research Laboratories, 2003.

[66] Roman Juránek, Michal Hradiš, and Pavel Zemčík. *Real-time Algorithms of Object Detection using Classifiers*, pages 1–22. InTech - Open Access Publisher, 2012.

[67] Roman Juránek, Pavel Zemcík, and Adam Herout. Implementing the Local Binary Patterns with SIMD Instructions of CPU. In *Proceedings of WSCG 2010*, pages 39–42. University of West Bohemia in Pilsen, 2010.

[68] Z Kalal, J G Matas, and K Mikolajczyk. Weighted Sampling for Large-Scale Boosting. In *BMVC08*, pages xx–yy, 2008.

[69] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk. P-N learning: Bootstrapping binary classifiers by structural constraints. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 49–56. IEEE, June 2010.

[70] Michael Kearns and Leslie G Vahant. Learning Boolean Formulae or Finite Automata is as Hard as Factoring. Technical Report TR 14-88, Harvard University Aiken Computation Laboratory, 1988.

[71] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994.

[72] Dong-Kyun Kim, Jun-Hee Jung, Thuy Tuong Nguyen, Dai-Jin Kim, Mun-Sang Kim, Key-Ho Kwon, and Jae-Wook Jeon. An FPGA-based Parallel Hardware Architecture for Real-time Eye Detection. *JSTS:Journal of Semiconductor Technology and Science*, 12(2):150–161, June 2012.

[73] D.J. Kriegman and N. Ahuja. Detecting faces in images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[74] Christos Kyrkou and Theocharis Theocharides. A Flexible Parallel Hardware Architecture for AdaBoost-Based Real-Time Object Detection. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19(6):1034–1047, June 2011.

[75] Hung-Chih Lai, Marios Savvides, and Tsuhan Chen. Proposed FPGA Hardware Architecture for High Frame Rate (¿100 fps) Face Detection Using Feature Cascade Classifiers. In *2007 First IEEE International Conference on Biometrics: Theory, Applications, and Systems*, pages 1–6. IEEE, September 2007.

[76] Christoph H. Lampert. An efficient divide-and-conquer cascade for nonlinear object detection. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1022–1029. IEEE, June 2010.

[77] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, June 2008.

[78] Ivan Laptev. Improving object detection with boosted histograms. *Image and Vision Computing*, 27(5):535–544, April 2009.

[79] Duy-Dinh Le and Shinichi Satoh. Ent-Boost: Boosting Using Entropy Measure for Robust Object Detection. *Pattern Recognition, International Conference on*, 2:602–605, 2006.

[80] Honglak Lee, Peter Pham, and Andrew Y Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Science*, 22:1–9, 2009.

[81] Bastian Leibe, Aleš Leonardis, and Bernt Schiele. Robust Object Detection with Interleaved Categorization and Segmentation. *International Journal of Computer Vision*, 77(1-3):259–289, November 2007.

[82] Kobi Levi and Yair Weiss. Learning Object Detection from a Small Number of Examples: The Importance of Good Features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:53–60, 2004.

[83] Jianguo Li and Yimin Zhang. Learning SURF Cascade for Fast and Accurate Object Detection. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3468–3475. IEEE, June 2013.

[84] S Li, Z Zhang, H Shum, and H Zhang. FloatBoost learning for classification. In *The Conference on Advances in Neural Information Processing Systems (NIPS)*, 2002.

[85] S Z Li and Z Zhang. FloatBoost learning and statistical face detection. *PAMI*, 26(9):1112–1123, 2004.

[86] Stan Z Li, Long Zhu, ZhenQiu Zhang, Andrew Blake, HongJiang Zhang, and Harry Shum. Statistical Learning of Multi-view Face Detection. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*, pages 67–81, London, UK, 2002. Springer-Verlag.

[87] S.Z. Li. Learning representative local features for face detection. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–1126–I–1131, Li2001, 2001. IEEE Comput. Soc.

[88] S Liao, W Fan, A C S Chung, and D Y Yeung. Facial Expression Recognition using Advanced Local Binary Patterns, Tsallis Entropies and Global Appearance Features. pages 665–668, 2006.

[89] Rainer Lienhart, Er Kuranov, and Vadim Pisarevsky. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. In *In DAGM 25th Pattern Recognition Symposium*, pages 297–304, 2003.

[90] Rainer Lienhart and Jochen Maydt. An Extended Set of Haar-Like Features for Rapid Object Detection. In *IEEE ICIP 2002*, pages 900–903, 2002.

[91] Ce Liu and Hueng-Yeung Shum. Kullback-Leibler boosting. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages 587–594, 2003.

[92] Huitao Luo. Optimization Design of Cascaded Classifiers. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:480–485, 2005.

[93] T Maenpaa and M Pietikainen. Multi-scale Binary Patterns for Texture Analysis. pages 885–892, 2003.

[94] Jan Masek, Radim Burget, Vaclav Uher, and Selda Guney. Speeding up Viola-Jones algorithm using multi-Core GPU implementation. In *2013 36th International Conference on Telecommunications and Signal Processing (TSP)*, pages 808–812. IEEE, 2013.

[95] Thomas Vetter Matthias Rätsch, Sami Romdhani. Efficient face detection by a cascaded support vector machine using haar-like features. In *26th DAGM Symposium*, pages 62–70, Tübingen, Germany, 2004. Springer Berlin Heidelberg.

[96] K Messer, J Matas, J Kittler, J Lottin, and G Maitre. XM2VTSDB: The Extended M2VTS Database. In *In Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, 1999.

[97] Julien Meynet, Vlad Popovici, and Jean-Philippe Thiran. Face detection with boosted Gaussian features. *Pattern Recognition*, 40(8):2283–2291, August 2007.

[98] S Munder and D M Gavrila. An Experimental Study on Pedestrian Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1863–1868, 2006.

[99] Narmada Naik and Rathna G.N. Real Time Face Detection on GPU Using OPENCL. In *Computer Science & Information Technology (CS & IT)*, pages 441–448. Academy & Industry Research Collaboration Center (AIRCC), February 2014.

[100] T Ojala, M Pietikäinen, and D Harwood. A Comparative Study of Texture Measures with Classification Based on Feature Distributions. 29(1):51–59, 1996.

[101] Timo Ojala and Matti Pietikäinen. Unsupervised Texture Segmentation Using Feature Distributions. In *ICIAP '97: Proceedings of the 9th International Conference on Image Analysis and Processing-Volume I*, pages 311–318, London, UK, 1997. Springer-Verlag.

[102] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Gray Scale and Rotation Invariant Texture Classification with Local Binary Patterns. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, pages 404–420, London, UK, 2000. Springer-Verlag.

[103] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[104] Hong Pan, Yaping Zhu, Siyu Xia, and Kai Qin. Improved generic categorical object detection fusing depth cue with 2D appearance and shape features. In *ICPR*, pages 1467–1470. IEEE, 2012.

[105] Constantine P Papageorgiou, Michael Oren, and Tomaso Poggio. A General Framework for Object Detection. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 555, Washington, DC, USA, 1998. IEEE Computer Society.

[106] Marco Pedersoli, Jordi Gonzàlez, Andrew D. Bagdanov, and Juan J. Villanueva. Recursive coarse-to-fine localization for fast object detection. In *ECCV 2010*, pages 280–293. Springer-Verlag, September 2010.

[107] Lukas Polok, Adam Herout, Michal Hradiš, Radovan Jošth, Roman Juránek, and Pavel Zemčík. „Local Rank Differences“ Image Faeture Implemented on GPU. In *Lecture Notes in Computer Science*, Lecture Notes in Computer Science, pages 170–181, 2008.

[108] Gunnar Ratsch. *Robust Boosting via Convex Optimization: Theory and Applications*. PhD thesis, Mathematisch-Naturwissenschaftichen Fakultat der Universitat Potsdam, 2001.

[109] Yann Rodriguez and Sebastien Marcel. Face Authentication Using Adapted Local Binary Pattern Histograms. In *9th European Conference on Computer Vision (ECCV)*, 2006.

[110] H.A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 203–208. IEEE Comput. Soc. Press, 1996.

[111] Henry A Rowley, Shumeet Baluja, and Takeo Kanade. Neural Network-Based Face Detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20:23–38, 1998.

[112] Cynthia Rudin, Ingrid Daubechies, and Robert E Schapire. The Dynamics of AdaBoost: Cyclic Behavior and Convergence of Margins. *J. Mach. Learn. Res.*, 5:1557–1595, 2004.

[113] Robert E Schapire. The Strength of Weak Learnability. *Mach. Learn.*, 5(2):197–227, 1990.

[114] Robert E Schapire. The Boosting Approach to Machine Learning: An Overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[115] Robert E Schapire, Yoav Freund, Peter Bartlett, and Wee S Lee. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

[116] Robert E Schapire and Yoram Singer. Improved Boosting Algorithms using Confidence-Rated Predictions. In *COLT*, pages 80–91, 1998.

[117] Robert E Schapire and Yoram Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. *Mach. Learn.*, 37(3):297–336, 1999.

[118] H. Schneiderman and T. Kanade. Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition. In *CVPR '98*, page 45, Washington, DC, USA, June 1998. IEEE Computer Society.

[119] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 1, pages 746–751. IEEE Comput. Soc, 2000.

[120] Henry Schneiderman. Feature-Centric Evaluation for Efficient Cascaded Object Detection. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:29–36, 2004.

[121] David W Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization.* Probability and Statistics. Wiley-Interscience, 1992.

[122] J Sochman and J Matas. Adaboost with totally corrective updates for fast face detection. In *AFGR04*, pages 445–450, 2004.

[123] Jan Sochman and Jiri Matas. Inter-Stage Feature Propagation in Cascade Building with AdaBoost. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 1*, pages 236–239, Washington, DC, USA, 2004. IEEE Computer Society.

[124] Jan Sochman and Jiri Matas. WaldBoost - Learning for Time Constrained Sequential Detection. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 150–156, Washington, DC, USA, 2005. IEEE Computer Society.

[125] Jan Sochman and Jiri Matas. Learning a Fast Emulator of a Binary Decision Process. In Yasushi Yagi, Sing Bing Kang, In So Kweon, and Hongbin Zha, editors, *ACCV*, volume II of *LNSC*, pages 236–245, Berlin Heidelberg, 2007. Springer.

[126] Jan Sochman and Jiri Matas. Learning Fast Emulators of Binary Decision Processes. *International Journal of Computer Vision*, 83(2):149–163, 2009.

[127] Patrick Sudowe and Bastian Leibe. Efficient use of geometric constraints for sliding-window object detection in video. In *ICVS 2011*, pages 11–20, Sophia Antipolis, September 2011. Springer-Verlag.

[128] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep Neural Networks for Object Detection. In C J C Burges, L Bottou, M Welling, Z Ghahramani, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2553–2561. 2013.

[129] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, pages 762–769. IEEE, 2004.

[130] Oncel Tuzel, Fatih Porikli, and Peter Meer. Region covariance: a fast descriptor for detection and classification. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *ECCV 2006*, volume 3952 of *Lecture Notes in Computer Science*, pages 589–600, Berlin, Heidelberg, May 2006. Springer Berlin Heidelberg.

[131] L G Valiant. A theory of the Learnable. *Communications of the ACM*, pages 1134–1142, 1984.

[132] Koen E A Van De Sande, Theo Gevers, and Cees G M Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[133] Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data*. Information Science and Statistics. Apringer, 1982.

[134] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 606–613. IEEE, September 2009.

[135] Erik Learned-miller Vidit Jain. FDDB: A benchmark for face detection in unconstrained settings. Technical report, University of Massachusetts, Amherst, 2010.

[136] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 734–741 vol.2. IEEE, 2003.

[137] Paul Viola and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:511, 2001.

[138] Paul Viola and Michael J Jones. Robust Real-Time Face Detection. *Int. J. Comput. Vision*, 57(2):137–154, 2004.

[139] Paul Viola, Michael J. Jones, and Daniel Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. *International Journal of Computer Vision*, 63(2):153–161, February 2005.

[140] Paul Viola and Mike Jones. Robust Real Time Object Detection. In *Second International Workshop On Statistical and Computational Theories of Vision*, 2001.

[141] Jan Šochman. *Learning for Sequential Classification*. PhD thesis, Czech Technical University in Prague, 2009.

[142] A. Wald. Sequential Tests of Statistical Hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, June 1945.

[143] A Wald and J Wolfowitz. Optimum character of the sequential probability ratio test. *The Annals of Mathematical Statistics*, 19(3):326–339, 1948.

[144] Stefan Walk, Nikodem Majer, Konrad Schindler, and Bernt Schiele. New features and insights for pedestrian detection. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1030–1037. IEEE, June 2010.

[145] Peng Wang and Qiang Ji. Learn Discriminant Features for Multi-View Face and Eye Detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 373–379. IEEE.

[146] Xianji Wang, Haifeng Gong, Hao Zhang, Bin Li, and Zhenquan Zhuang. Palmprint Identification using Boosting Local Binary Pattern. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 503–506, Washington, DC, USA, 2006. IEEE Computer Society.

[147] Xiaoyu Wang, Tony X. Han, and Shuicheng Yan. An HOG-LBP human detector with partial occlusion handling. In *2009 IEEE 12th International Conference on Computer Vision*, pages 32–39. IEEE, September 2009.

[148] Yubo Wang, Haizhou Ai, Bo Wu, and Chang Huang. Real Time Facial Expression Recognition with Adaboost. *Pattern Recognition, International Conference on*, 3:926–929, 2004.

[149] C.A. Waring and X. Liu. Face Detection Using Spectral Histograms and SVMs. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 35(3):467–476, June 2005.

[150] B Wu, H Z Ai, and C Huang. LUT-Based AdaBoost for Gender Classification. pages 104–110, 2003.

[151] Bo Wu, Haizhou Ai, Chang Huang, and Shihong Lao. Fast rotation invariant multi-view face detection based on real Adaboost. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 79–84, 2004.

[152] Rong Xiao, Long Zhu, and Hong-Jiang Zhang. Boosting Chain Learning for Object Detection. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 709, Washington, DC, USA, 2003. IEEE Computer Society.

[153] Y Abramson, B Steux, and H Ghorayeb. Yef real-time object detection. pages 5–13, 2005.

[154] Shengye Yan, Shiguang Shan, Xilin Chen, and Wen Gao. Locally Assembled Binary (LAB) feature with feature-centric cascade for fast and accurate face detection. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, June 2008.

[155] Pavel Zemcik, Michal Hradis, and Adam Herout. Exploiting neighbors for faster scanning window detection in images. In *ACIVS 2010*, LNCS 6475, page 12. Springer Verlag, 2010.

[156] Pavel Zemcik and Martin Zadnik. Adaboost Engine. In *Proceedings of FPL 2007*, page 5. IEEE Computer Society, 2007.

[157] Pavel Zemčík, Roman Juránek, Martin Musil, Petr Musil, and Michal Hradiš. High Performance Architecture for Object Detection in Streamed Videos. In *Proceedings of FPL 2013*, pages 1–4. IEEE Circuits and Systems Society, 2013.

[158] Cha Zhang and Zhengyou Zhang. A Survey of Recent Advances in Face detection. Technical report, Microsoft Research, Redmond, 2010.

[159] Hongming Zhang, Wen Gao, Xilin Chen, and Debin Zhao. Object detection using spatial histogram features. *Image and Vision Computing*, 24(4):327–341, April 2006.

[160] Lun Zhang, Rufeng Chu, Shiming Xiang, ShengCai Liao, and Stan Z Li. Face Detection Based on Multi-Block LBP Representation. In *ICB*, pages 11–18, 2007.

[161] Xin Zhang, Yee-Hong Yang, Zhiguang Han, Hui Wang, and Chao Gao. Object class detection: A survey. *ACM Computing Surveys*, 46(1):1–53, October 2013.

[162] G Y Zhao and M Pietikainen. Dynamic Texture Recognition Using Volume Local Binary Patterns. pages 165–177, 2006.

[163] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1491–1498, Washington, DC, USA, 2006. IEEE Computer Society.

[164] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2879–2886. IEEE, June 2012.