Available online at www.sciencedirect.com
SCIENCE DIRECT
Image and Vision Computing 24 (2006) 987–1000
image AND vision COMPUTING
www.elsevier.com/locate/imavis

Context-based object detection in still images

N.H. Bergboer *, E.O. Postma, H.J. van den Herik

Department of Computer Science, Maastricht University, Minderbroedersberg 6a, P.O. Box 616, 6200 MD Maastricht, The Netherlands

Received 5 January 2005; received in revised form 30 September 2005; accepted 21 February 2006

Abstract

We present a novel dual-stage object-detection method. In the first stage, an object detector based on appropriate visual features is used to find object candidates. In the second stage, the object candidates are assigned a confidence value based on local-contextual information. Our context-based method is called COBA, for COntext BAsed object detection. At a given detection rate COBA is able to lower the false-detection rate. Experiments in which frontal human faces are to be detected show that the number of false positives is lowered by a factor 8.7 at a detection rate of 80% when compared to the current high-performance object detectors. Moreover, COBA is capable of flexibly using other new object-detection algorithms as 'plug-ins' in the second stage. Hence, object detection can be straightforwardly improved by our method a soon as new insights emerge and are available in algorithmic form.

Keywords: 42.30.T

Keywords: Computer vision; Machine learning; Object recognition

1. Introduction

In this paper, we present a novel context-based object-detection method named COBA. Object detection by a computer is the automatic determination of the locations and sizes of objects in an image, where the objects belong to a predefined class. Object detection is an active research area that has yielded many methods (see, e.g. [1–4]). Currently, the field focusses on multiview and real-time approaches. However, so far pure accuracy, in terms of the number of false detections at a given detection rate, has received less attention.

Two concepts from behavioural sciences, spatial context and visual attention [5,6], inspired us in mitigating this drawback. The first concept, spatial context, is important because observers use spatial context when locating and detecting objects. Objects are found faster and more reliably if they are embedded in familiar contexts [7], since such familiarity may prevent false detections of object-like patterns in unlikely contexts. Fig. 1 shows two examples of false detections. In the figure, the small square images (left) are enlarged versions of the square regions indicated by boxes in the large images. Considered in isolation, both small images are highly similar to faces. When considered in their natural context, a potential interpretation as faces is quickly suppressed. It is important to note that the spatial context of the object contains both the object itself, and its direct spatial surroundings.

The second concept, visual attention, enables primates to select rather efficiently part of a visual scene [8]. This selection allows the primates to reduce the amount of information to be processed, and confine their attention to appropriate regions. Visual attention can be shifted overtly through eye movements (saccades) [9], or covertly by a mental process [10]. Overt attention focusses the appropriate region on the fovea, the high-resolution part of the retina [11]. Covert attention is a mental process that can be likened to an internal 'spotlight' that is focussed on parts of the retina to select future locations for overt attention [5,12–14].

According to theories of human vision [15], the concepts of spatial context and visual attention are closely related. Recent results in human perception studies [8] suggest that spatial attention in humans involves local contextual information as well as object features to direct the focus of attention to the object of interest. Inspired by these behavioural findings we propose our new method COBA. It combines the object features and the local contextual information to guide attention for object detection. In the present study, our conception (definition) of local context reads: 'the visual features within a

* Corresponding author. Tel.: +31 43 3883901; fax: +31 43 3884897.
E-mail address: n.bergboer@cs.unimaas.nl (N.H. Bergboer).

0262-8856/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2006.02.024

Fig. 1. Examples of patterns that are similar to faces, but that are clearly not faces when viewed in their context.

confined spatial region around a centred object'. Our definition of local context includes features that are both internal and external to the object (see Section 4.3; an example is given in Fig. 3). As such, our conception of context corresponds to that of Kruppa et al. [16], who define context as a slightly larger region centred around objects. Our definition differs from that of Torralba and Oliva [17], who define context as the entire image region.

COBA is a two-stage method that uses (1) an object detector to find object candidates, and (2) the spatial context in order to validate the object candidates. We claim that the use of context will lower the false-detection rate. Below we mention two different approaches that are related to our approach: Torralba and Oliva [17] and Kruppa et al. [16].

First, Torralba and Oliva [17] show that the features of an object's spatial context are statistically related to the features of the object itself. They claim that the spatial context could be used to reduce the search space in object detection. However, they do not actually combine their method with object detection. Second, Kruppa et al. [16] use an object's spatial context to infer the location of the object; they detect human heads in order to find faces. However, their approach solely focusses on the detection of the context. The object is than located by assuming that it is in the centre of the detected context. We will discuss the relation between our work and the work by Torralba et al. [17] and Kruppa et al. [16] in more detail in Section 7.

We stress the idea of COBA, viz. by (1) using contextual information of actual object examples and (2) learning how to use this information to direct visual attention.

The paper is organised as follows. Section 2 presents our context-based method COBA. Section 3 outlines the algorithmic approach and provides implementation details. Section 4 describes the experimental set-up for the application of COBA to detect frontal human faces. Section 5 describes the evaluation procedure. In Section 6 we present results, which are discussed in Section 7. In Section 8, we conclude upon the performance of COBA and provide three recommendations.

2. Description of the context-based method

As stated in the introduction, our context-based object-detection method COBA proceeds in two stages; an object-detection stage and a contextual-validation stage. In the following we discuss both stages in detail.

First, in the object-detection stage the image is scanned using a window-sliding technique [1–3,18]. This technique extracts information from a square window of a given size from a grid of locations and scales. At each position and scale, the contents of a window are classified as either an object candidate, or a background pattern. This stage is illustrated in the left panel of Fig. 2. The solid box with an arrow denotes the sliding window. The stage results in a set of object candidates, denoted by the other solid boxes in the left panel of Fig. 2.

Second, in the context-validation stage, confidence is assigned to each of the first-stage object candidates. A sliding-window technique is used in locations close to the object candidates only. The contents of a window is used to obtain a probability density function (PDF) that describes the most probable *relative* location for the object with respect to the current location. For a given location, the PDF is illustrated in the centre panel of Fig. 2. The solid square defines the region from which information is extracted and the arrow represents the expectation value of the relative location of the object with respect to the current location. The circles are contour lines for the probability density function. The probability density value at locations corresponding to first-stage object candidates (denoted by the dashed squares) is stored for each of these candidates.

The algorithm that generates the PDFs is trained on example patterns taken from the spatial context of the object class of interest. As noted in the introduction, we define the object as part of its spatial context; we can thus expect that features from the object itself and features from its surroundings will both be used to guide the visual attention process. The main difference

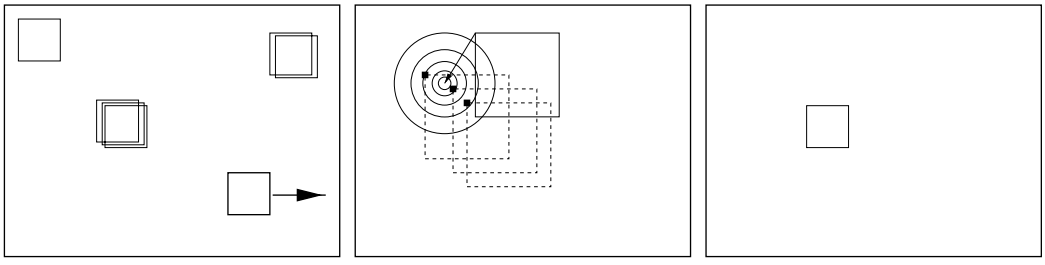

Fig. 2. Graphical representation of our context-based method.

between COBA and other methods based on object detection [16,19], is that COBA detects objects by combining local contextual cues into an estimate the location of the object with respect to the current location.

In order to obtain a confidence value for each object candidate, PDFs for all locations that overlap an object detection are added. The result of the second stage is a confidence value for each of the first-stage object candidates.

The final result of COBA is obtained by thresholding the confidence value, and regarding only those object candidates with a sufficient confidence value as *final detections*. These final detections are shown in the right panel of Fig. 2.

The dual-stage nature of COBA renders it independent of the type of object classifier. That is, arbitrary object classifiers can be used as 'plug-ins' in the first stage. To emphasise the flexibility of the method, we shall employ two different object classifiers.

In both detection stages, the contents of a window is first transformed into visual features that are meaningful in terms of human vision [11]. The choice of features is essential for obtaining good results. Appropriate visual features for detecting the object candidates, and determining the spatial context, are discussed in the Sections 2.1 and 2.2.

2.1. Visual features for the object-detection stage

To investigate the 'plug-in' property for object classifiers in COBA, we implemented two different object detectors. The first detector is called the *monolithic detector* and is based on the work by Papageorgiou and Poggio [1]. The monolithic detector employs a support-vector classifier to distinguish between objects and non-objects. The second detector is called the *boosted-cascade detector* and is based on the work by Viola and Jones [4] and Lienhart et al. [20]. They use a boosted cascade of classifiers to distinguish between objects and non-objects. Each of the classifiers in the cascade detects almost all the objects while allowing a very high (in the order of tens of per cents) false-detection rate. A cascade of many of these classifiers then yields a relatively high true-detection rate and a low false-detection rate. Viola and Jones use AdaBoost [21] to select which of the edge features and line features are most effective in each of the classifiers.

Both object detectors are based on existing work, and will be used as 'plugins' in the first stage of COBA. Below, we briefly describe the features used.

2.1.1. Visual features for the monolithic detector

Papageorgiou and Poggio employ features derived from an overcomplete wavelet basis at two scales to detect various object classes, i.e. human faces, pedestrians, and automobiles. They use either gray-scale or pseudo-colour wavelets; the choice for gray-scale or pseudo-colour depends on the object class of interest. This holds true for the appropriate spatial scale, too.

2.1.2. Visual features for the boosted-cascade detector

Viola and Jones employ edge features and line features that are very similar to Haar wavelets. The set of simple edge features and line features has been extended by Lienhart et al. [20] to include centre-surround features and oriented edge features. The extraction of these features, together with an efficient classification and appropriate training software, is now part of the Intel Open Computer Vision library[1].

2.2. Visual features for the context-validation stage

The visual features for the context-validation stage have to fulfil the following two requirements: (1) provide sufficient information about the spatial context of an object, and (2) they should give rise to a low intra-class variance for the context of the object. This means that the features should encode information that does not vary too much over the class of local spatial contexts of objects.

One of the earliest stages in human vision is the detection of edges at several orientations in the retinal image [11]. Consequently, many existing algorithms in the area of computer vision use features that are derived from edge information, such as wavelet features [1] and scale-space features [22].

We use overcomplete Haar wavelet features [1] for our context-validation stage. The reason for choosing Haar wavelet features is that it is one of the most elementary features that encodes edge information at different orientations and that simultaneously can be calculated efficiently. The reason for choosing an overcomplete representation is twofold. First, it is better in capturing constraints between neighbouring regions and complex patterns than a complete representation [1]. Second, despite being overcomplete, it is sufficiently small in the number of features as compared to the Viola–Jones feature set. Since the number of training samples for contextual validation is small, a limited number of features is a prerequisite to prevent overfitting (see. e.g. [23]).

The scales that contain the largest possible amount of the information unique to the object's context are included. These scales are specific to the object class of interest.

3. Algorithmic approach

Below we describe the approach used in the context-validation stage, i.e. the second stage of COBA. The main focus of the context-validation stage is to *learn* the relation between patterns and their relative positioning within the direct spatial context of an object. Section 3.1 describes the relative localisation in more detail. We employ two different techniques to estimate PDFs: cluster-weighted models (Section 3.2), and decision-trees (Section 3.3).

3.1. Relative localisation

Our objective is to estimate the location of the object *relative* to the current location, i.e. to estimate the vector $\vec{x}_r = (x_r, y_r)$ by means of the window features v, where x_r

[1] http://www.sourceforge.net.

represents the horizontal relative location and y_r represents the vertical relative location. The function that minimises the mean square error between the estimated $\widehat{\vec{x}_r}$ and the real relative location $\vec{x}_r$ is the conditional expected value [24, p. 247]:

$$\widehat{\vec{x}_r} = \int \vec{x}_r f(\vec{x}_r, v) \mathrm{d}\vec{x}_r, \tag{1}$$

where the joint PDF $f(\vec{x}_r, v)$ describes the relation between the two random variables $\vec{x}_r$ and v. It is given by:

$$f(\vec{x}_r|v) = \frac{f(\vec{x}_r, v)}{f(v)}. \tag{2}$$

As will be discussed in the next section, determining the relative object location $\vec{x}_r$ transforms into finding the joint PDF $f(\vec{x}_r, v)$.

3.2. Estimating the PDF using cluster-weighted models

For building the relative location estimator, we need to estimate the joint PDF $f(\vec{x}_r, v)$. In the framework of regression algorithms, several approaches have been proposed for the estimation of joint PDFs. We will use cluster-weighted modelling ([25], p. 178), as it provides a simple algorithm for the learning stage [26,27]. For completeness, we reproduce the main expressions of the estimation algorithm here.

In the cluster-weighted modelling algorithm, a joint PDF is expanded as a sum of m elliptical Gaussian clusters that each model the local relationship between the n-dimensional input distributions and ℓ-dimensional output distributions. In our specific application an output dimensionality of $\ell = 2$ is used.

$$f(\vec{x}_r, v) = \sum_{i=1}^{m} g(\vec{x}_r|v, c_i) g(v|c_i) p(c_i), \tag{3}$$

in which c_i refers to the ith cluster. Moreover, $p(c_i)$ is a cluster weight, and $g(v|c_i)$ is a multivariate elliptical Gaussian with mean μ_i and covariance matrix X_i defining the domain of influence in the input space of the cluster:

$$g(v|c_i) = \frac{\exp\left[-\frac{1}{2}(v-\mu_i)^{\mathrm{T}} X_i^{-1} (v-\mu_i)\right]}{(2\pi)^{(n/2)}\sqrt{\det(X_i)}}. \tag{4}$$

The output distribution of the cluster is modelled by a similar expression:

$$g(\vec{x}_r|v, c_i) = \frac{\exp\left[-\frac{1}{2}(\vec{x}_r - a_i - b_i^{\mathrm{T}} v_t)^{\mathrm{T}} S_i^{-1} (\vec{x}_r - a_i - b_i^{\mathrm{T}} v_t)\right]}{(2\pi)^{(\ell/2)}\sqrt{\det(S_i)}}. \tag{5}$$

The centres of the output clusters are dependent on the input samples v_t by a linear transformation (described by the scalars a_i and vectors b_i); their covariance matrices are S_i. Given the model parameters, the joint PDF for the output is given by:

$$f(\vec{x}_r|v) = \frac{\sum_{i=1}^{m} g(\vec{x}_r|v, c_i) g(v|c_i) p(c_i)}{\sum_{i=1}^{m} g(v|c_i) p(c_i)}. \tag{6}$$

When estimating $\vec{x}_r$ given v, the expectation value $\widehat{\vec{x}_r}$ is calculated as

$$\widehat{\vec{x}_r} = \frac{\sum_{i=1}^{m} (a_i + b_i^{\mathrm{T}} v_t) g(v|c_i) p(c_i)}{\sum_{i=1}^{m} g(v|c_i) p(c_i)}, \tag{7}$$

and a covariance-matrix estimate, that functions as an inverse confidence measure, is calculated as

$$\hat{S}_{\vec{x}_r} = E[(\widehat{\vec{x}_r} - \vec{x}_r)(\widehat{\vec{x}_r} - \vec{x}_r)^{\mathrm{T}}|v] = \frac{\sum_{i=1}^{m} S_i g(v|c_i) p(c_i)}{\sum_{i=1}^{m} g(v|c_i) p(c_i)}. \tag{8}$$

The estimated relative location $\widehat{\vec{x}_r}$ and the estimated covariance matrix $\hat{S}_{\vec{x}_r}$ are used to calculate a one-cluster PDF. The image-wide probability measure is formed by adding all the one-cluster PDFs.

3.3. Estimating the PDF using decision trees

A C4.5 decision tree [28] maps its input attributes v onto m output classes. For each output class c_i, a conditional probability $p(c_i|v)$ is determined such that $\sum_{i=1}^{m} p(c_i|v) = 1$.

In COBA, we employ a set-up in which the area covered by possible relative locations is discretised into a grid of n rows and n columns. In this way, using $m = n^2$, we can regard the conditional class probabilities yielded by the decision tree as a discretised relative location PDF.

4. Experimental set-up: human face detection

In order to assess the performance of COBA we apply it to the detection of a single class of objects: frontal human faces. The class of faces is an appropriate challenge for COBA because it is well documented and data for comparison is readily available [29]. For the task of frontal face detection, we expect that the context consists mainly of the head. In the experiments, we examined our claim that using context in COBA lowers the false-detection rate.

In order to obtain training data, a total of 3383 faces were manually labelled in 995 colour images obtained from the Internet. An additional set of 1900 positive instances representing frontal faces is obtained by labelling unoccluded faces from the AR faces database [30].

Below, we describe the experimental procedure in three steps: the selection of face candidates, the contextual validation, and the training.

4.1. Detection of face candidates

For the detection of face candidates we use the monolithic and boosted-cascade detectors introduced in Section 2.1.

The monolithic detector is based on the method of Papageorgiou and Poggio with gray-scale wavelets, as these yield the most compact representation of the object class [1].

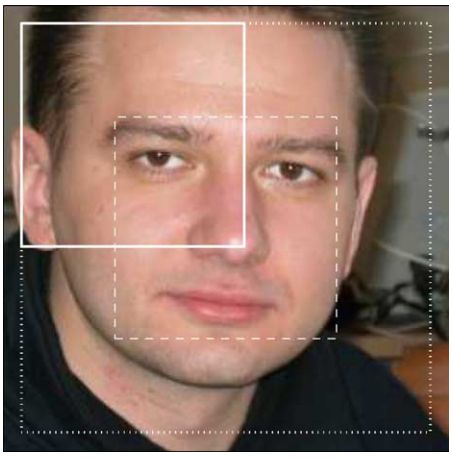

Fig. 3. The region used for obtaining training samples, overlaid on a face.

A sliding window of size 19×19 pixels is used and the feature set consists of wavelet coefficients for square-support Haar wavelets of size 2×2 and 4×4. We use an overcomplete representation of 17×17 coefficients for a given scale and orientation. For both scales, the low-pass filtered data is discarded and the absolute value of only the horizontal, vertical, and diagonal detail coefficients are retained. Each of the resulting six components is normalised by its mean, thus a feature vector $\theta_O \in \mathbb{R}^{1734}$ is obtained by vectorising and concatenating the six components.

In the boosted-cascade detector, the choice of line features and edge features used for classification is determined by the training algorithm.

4.2. Contextual validation

For the contextual validation we use a window of size 19×19 pixels. Context-validation models are trained on patterns from a region almost twice as large as the face itself. Each training face is down-scaled such that its dimensions 19×19 pixels. The patterns used have a relative displacement from −8 to +8 pixels in both the horizontal and vertical direction.

Fig. 3 illustrates the extent of the training region; the region is overlaid on a high-resolution face. The large dotted square denotes the region from which the training samples are taken, the dashed square in the centre denotes the actual face location, and the solid square in the upper left corner of the training region denotes the window for relative displacement of −8 pixels in both the horizontal and vertical direction in the down-sampled image[2].

We use one of three different groups of overcomplete wavelet feature sets [1] in order to determine which feature set best describes appropriate contexts of faces while incorporating the least amount of uninformative information.

[2] We show the training region overlaid on a *high-resolution* face in Fig. 3 to illustrate clearly what portion of the head is used as context. When extracting features from a facial context, the high-resolution image is first down-scaled in such a way that the face has a size of 19×19 pixels.

(1) Coarse representation:
   - Coarse scale-2: wavelets at decomposition level 1
   - Coarse scale-4: double-resolution wavelets at decomposition level 2

(2) Fine representation:
   - Fine scale-2: double-resolution wavelets at decomposition level 1
   - Fine scale-4: quad-resolution wavelets at decomposition level 2

(3) Multiscale representation: double-resolution wavelets at decomposition level 1 (wavelet scale 2) and quad-resolution wavelets at decomposition level 2 (wavelet scale 4)

Fig. 4 illustrates what contextual features are common to instances of the class of facial contexts; we show averages of the multiscale features over the entire training region. The top row shows double-res scale-2 wavelets while the lower row shows quad-res scale-4 wavelets. The three columns show horizontal details, vertical details, and diagonal details, respectively. The features actually used in the context-validation stage are windows of size 19×19 cropped from the training region, as indicated in Fig. 4. The figure clearly shows that the multiscale representation captures dominant large features such as the eyes, the mouth, and the edge of the head. In particular, the internal context (eyes, nose, mouth) appears to be more important than the external context (edge of the head and background). In addition, it should be remarked that the differences between scale-2 and scale-4 wavelets is rather small, which suggests that the useful information resides in actual physical parts of the object or its surroundings. This conforms to contemporary studies, such as described in [31].

Below, we motivate the choice of our three different feature sets. For brevity, we refer to double-res 'scale-2' or 'scale-4' features whenever we mean Haar wavelets that are at the double of the resolution that would be required for a complete basis [1]; the same holds for quad-res 'scale-4' features.

In all cases, cluster-weighted models as described in Section 3.2 require a feature space with a reasonably small dimensionality. Therefore, principal component analysis (PCA) is used to reduce the dimensionality of the feature

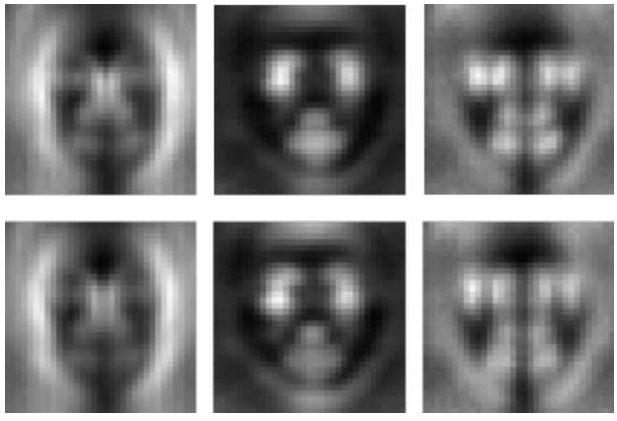

Fig. 4. Average wavelet features over the training region.

space to project the raw feature vectors θ_O to a reduced context-validation vector $\theta_O \in \mathbb{R}^{16}$. A separate PCA is performed for each of the three feature types.

The decision-tree models are capable of using the original raw feature-data. Therefore, no feature reduction is performed for these models.

4.2.1. Coarse representations

Our first group of representations contain the course representations. It consists of 9×9 wavelets for a given orientation. As it is not clear in advance which wavelet scale yields the best performance, we use both single-res scale-2 wavelets (*Coarse scale 2*) and double-res scale-4 wavelets (*Coarse scale 4*).

These representations enable us to capture details at a spatial scale larger than the face and its internal components. For instance, it captures the edges of the head and the cheeks that are descriptive for localising the face. We expect the representation to yield a low variance within the class of the context of faces, while still being sufficiently descriptive.

The coarse representations yield 9×9 wavelets for a given orientation for a window of size 19×19. As we use three orientations (horizontal, vertical, and diagonal), this yields a raw feature vector $\theta_O \in \mathbb{R}^{243}$.

4.2.2. Fine representations

Our second group of representations, the fine representations, incorporates more information on the spatial layout by using double-res scale-2 wavelets (*Fine scale 2*), or quad-res scale-4 wavelets (*Fine scale 4*). In this way, we incorporate relations between large-scale features on a finer spatial resolution. On the one hand, the additional spatial information provides relational information that facilitates the localisation of the face. On the other hand, one might expect that incorporating the additional information increases the variance within the class of contexts of faces such that localisation becomes worse.

The fine representations yield 17×17 wavelets for a given orientation for a window of size 19×19. As we use three orientations (horizontal, vertical, and diagonal), this yields a raw feature vector $\theta_O \in \mathbb{R}^{867}$.

4.2.3. Multiscale representation

Our third group of representations consists of the multiscale representation. It is the representation used by Papageorgiou and Poggio [1]. This representation uses double-res scale-2 wavelets and quad-res scale-4 wavelets. The representation incorporates a large amount of information on two different scales. Again, this representation might either improve detection by adding descriptive information, or deteriorate detection by adding too much variance within the class of facial contexts.

Both double-res scale-2 wavelets and quad-res scale-4 wavelets yield 17×17 wavelets for a given orientation and scale for a window of size 19×19. As we use three orientations (horizontal, vertical, and diagonal) at two scales, this yields a raw feature vector $\theta_O \in \mathbb{R}^{1734}$.

4.3. Training

In COBA, training proceeds in two stages. In the first stage, the two types of relative face locators are trained, viz. with the help of cluster-weighted models (Section 4.3.1) and with the help of decision trees (Section 4.3.2). In the second stage, object classifiers are trained for the two object detectors (Section 4.3.3).

4.3.1. Training the relative face locator based on cluster-weighted models

The cluster-weighted model is fully described by the cluster weights $p(c_i)$, the clusters' input-space means μ_i, the clusters' input-space covariance matrices X_i, the clusters' linear transformations (scalars a_i and vectors b_i), and the clusters' output covariance matrices S_i. These parameters are optimised by an iterative EM algorithm [25,26], based on training data $\{\vec{x}_{r,t}\}_{t=1,\ldots,N}$ and $\{v_t\}_{t=1,\ldots,N}$. Each iteration is composed of two steps: an E-step and an M-step. When denoting the current iteration number k, these steps are defined as follows.

- E-step: computes the posterior probabilities of the clusters given the observed data:

$$P^k(c_i|\vec{x}_{r,t}, v_t) = \frac{g^k(\vec{x}_{r,t}|v_t, c_i)g^k(v_t|c_i)p^k(c_i)}{\sum_{i=1}^{m} g^k(\vec{x}_{r,t}|v_t, c_i)g^k(v_t|c_i)p^k(c_i)} \tag{9}$$

- M-step: computes the most likely cluster parameters given the posterior probabilities as

$$p^{k+1}(c_i) = \frac{\sum_{t=1}^{N} P^k(c_i|\vec{x}_{r,t}, v_t)}{\sum_{i=1}^{m}\sum_{t=1}^{N} P^k(c_i|\vec{x}_{r,t}, v_t)}, \tag{10}$$

$$\mu_i^{k+1} = \langle v \rangle_i \equiv \frac{\sum_{t=1}^{N} P^k(c_i, v_t, v_t)v_t}{\sum_{t=1}^{N} P^k(c_i, v_t, v_t)}, \tag{11}$$

$$X_i^{k+1} = \left\langle (v-\mu_i^{k+1})(v-\mu_i^{k+1})^{\mathrm{T}} \right\rangle_i, \tag{12}$$

$$b_i^{k+1} = (X_i^{k+1})^{-1}\left\langle (v-\mu_i^{k+1})\vec{x}_r^{\mathrm{T}} \right\rangle_i, \tag{13}$$

$$a_i^{k+1} = \left\langle \vec{x}_r - (b_i^{k+1})^{\mathrm{T}}\vec{x}_r \right\rangle_i, \tag{14}$$

$$S_i^{k+1} = \left\langle \left(\vec{x}_r - a_i^{k+1} - (b_i^{k+1})^{\mathrm{T}}v\right)\left(\vec{x}_r - a_i^{k+1} - (b_i^{k+1})^{\mathrm{T}}v\right)^{\mathrm{T}} \right\rangle_i, \tag{15}$$

where $\langle\cdot\rangle_i$ denotes the weighted average as defined in (11).

An iterative procedure as outlined above requires an initial estimate for the first iteration. The initial estimate is obtained using a k-means clustering model [26].

The cluster-weighted modelling algorithm used for the context-validation stage is trained on a dataset of 1885 faces.

Within the context shown in Fig. 3, training samples are obtained at relative displacements $-8, -6, -4, -2, 0, +2, +4, +6, +8$ in both the horizontal and vertical directions. Thus, 81 relative displacements for each face are used, which yields a total of 152,685 samples in the training set.

A model is trained using $m=8$ clusters, as preliminary results have shown that further increasing the number of clusters does not improve performance. A separate model is trained for each of the three feature sets.

Fig. 5 shows the 8-cluster centres in the wavelet space for each of the three feature set groups. The left panel of Fig. 5 shows the cluster centres for the coarse scale-2 feature set. Each row corresponds to one cluster. The first three columns show the horizontal, vertical, and diagonal details for each cluster. The last column shows the cluster centre and confidence interval in the output space; the cross in the centre marks the actual location of the object, and the ellipse shows a one standard-deviation confidence interval around the cluster centre. The cluster centres for the coarse scale-4 feature set are similar.

The centre panel of Fig. 5 shows the cluster centres for the fine scale-2 feature set. The columns have the same interpretation as in the left panel, with the exception that all details are sampled at a 17×17 resolution. The cluster centres for the fine scale-4 feature set are similar.

The right panel of Fig. 5 shows the cluster centres for the multiscale feature set. The first three columns show horizontal, vertical, and diagonal details at scale 2, and the fourth to sixth columns show horizontal, vertical, and diagonal details at scale 4. The last column again shows the relative position for each cluster.

The three figures clearly show that spatially large features both internal to the face (such as the eyes, the nose, and the mouth), and external to the face (such as the edges of the head), are used for localising the face. For instance, the second row in the right panel of Fig. 5 shows that the presence of the eyes forms a strong indication that the current position is slightly above the position of the face. That is, the patterns shown in the second row are an indication that the upper-left corner of the current window is likely to lie in a small area above the actual upper-left corner of the face.

It is apparent that the cluster centres in the output space are similar for the three different feature sets. In addition, the corresponding cluster centres in the input space correspond to similar physical components in the three feature sets.

4.3.2. Training the relative face locator based on decision trees

The decision-tree algorithm used for the context-validation stage is trained on a dataset of 1885 faces. Within the context shown in Fig. 3, the relative locations are discretised to a 5×5 grid, yielding 25 output classes. Samples are obtained for the 25 relative displacements for each of the 1885 faces. This yields a total of 47,125 samples in the training set.

Finally, a standard C4.5 learning scheme [28] is applied, where attribute selection and branching points are based on the information maximisation criterion.

4.3.3. Training the object classifier

We employ the two object classifiers introduced in Section 2.1: the monolithic and the boosted-cascade detectors.

A support vector machine (SVM) is used in the monolithic detector for the classification of objects, as SVMs perform well and can be efficiently trained on large datasets [23]. In addition, efficient software is readily available [32,33]. A quadratic kernel is employed, as results by Papageorgiou and Poggio [1] indicate that such a kernel yields good object-detection performance.

For the monolithic detector we use a modification of the 'boot-strap training' strategy proposed by Sung and Poggio [2]. First, an initial training set is constructed that consists of 1900 positive instances from the AR faces database and 19,000 negative instances. These instances are obtained from random regions in a part of the set of images obtained from the Internet that is disjoint from the set that will be used for validation. A classifier is then trained and run on another 60 images from this set, which yields 31,613 false positives that are added to the set of negative training instances. Subsequently, a classifier is trained on this set, and it uses 2036 support vectors.

For the boosted-cascade detector, a prebuilt classifier optimised by Lienhart et al. [20] has been employed. This classifier uses 25 stages, a window of 24, and was trained using 5000 positive and 3000 negative examples per boosted stage. The output threshold on the final stage has been increased by five in order to obtain a reasonable false-detection rate.

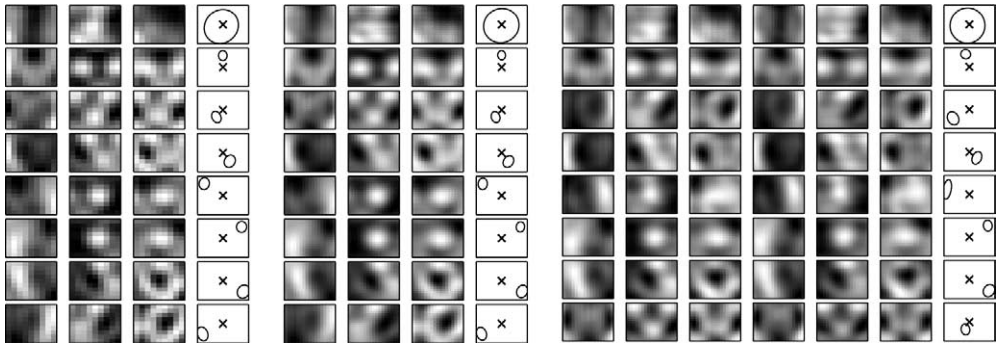

Fig. 5. The cluster centres for the Gaussian mixture models (in wavelet space) together with their relative position and confidence intervals for the three feature sets used.

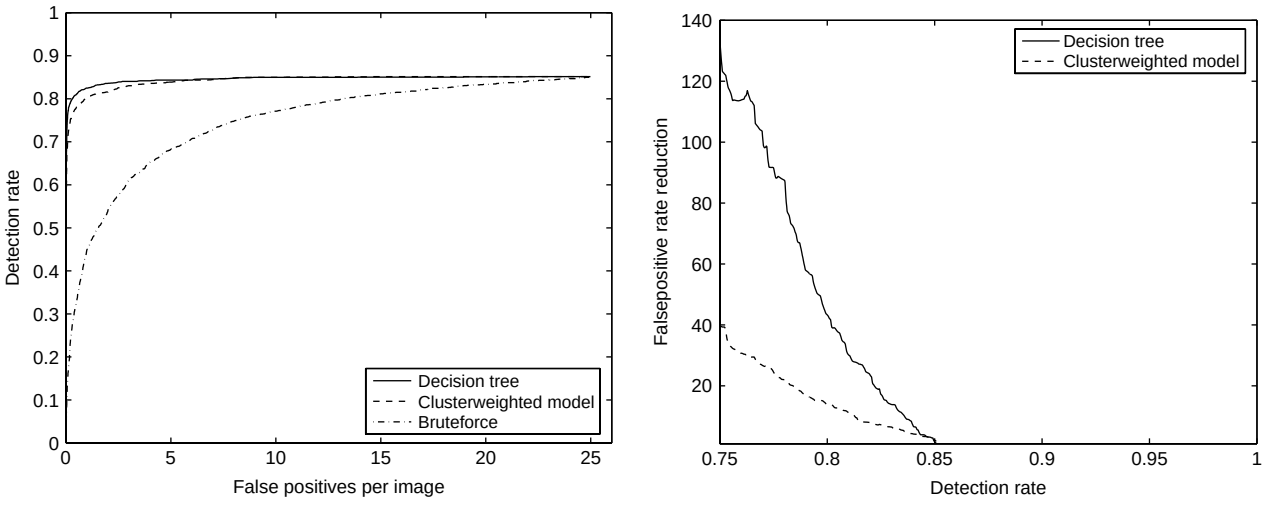

Fig. 6. Detection results for context-based face detection using the monolithic detector.

5. Evaluation procedure

The performances of the context-validation method is assessed on 775 images from the Internet that together contain 1885 labelled faces. This set is a subset from the larger set referred to in Section 4. In the remainder of this article, we refer to this 775-image set as the 'web-set'. The images contain labelled faces that are at least 30×30 pixels in size. To ensure that all faces are found, the images are classified for face sizes of 24.5×24.5 pixels and up. To this end, a scale-space pyramid of the image is calculated in which subsequent scales differ by a factor 1.1.

In order to obtain statistically valid results, we perform a 10-fold cross-validation procedure. We split the dataset into 10 parts: in the model used for the localisation of faces in images belonging to set i, we leave out part i from the training set. In addition, the model is used at different 'step sizes'; at a given 'step size' s, a PDF is calculated at only every s pixels in both the horizontal and vertical direction. To determine the benefits of using our context-validation method, the images are also scanned for faces in a brute-force manner by both detectors introduced in Section 2.1.

To assess the object-detection quality, one needs a consistent definition of true positives, false negatives, and false positives [34]. As there is no consensus over the exact definition of these terms, we specify the definition we use below.

In general, a face gives rise to multiple overlapping detections in close proximity. For the purpose of evaluation we employ a grouping procedure similar to [35]. The grouping procedure groups all regions classified as Object Region with an area overlap of at least 90% into a single detection. Each group is then considered a single detection, of which the coordinates are the means of the coordinates of the grouped regions. From all detections, at least one should overlap sufficiently with a labelled face in order for that face to be a *true positive*; the size of the detection should be within a factor 1.5 of the size of the labelled face, and the detection's centre should be within a distance to the labelled face's centre no larger than 30% of that face's size. If none of the detected

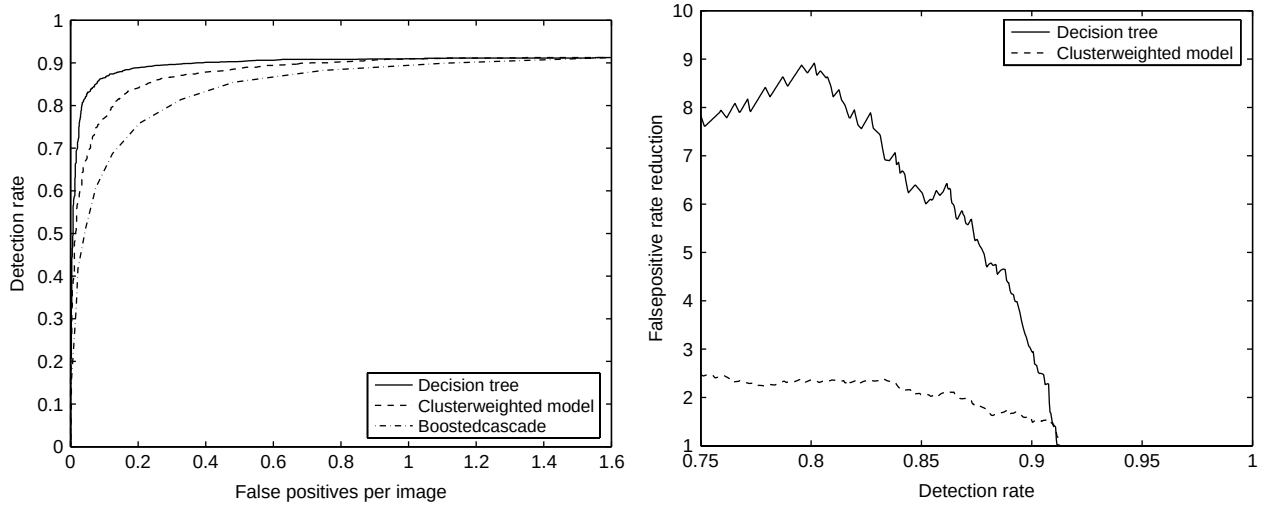

Fig. 7. Detection results for context-based face detection using the boosted-cascade detector.

Table 1
Detection results for the brute-force monolithic and boosted-cascade detectors

Object detector	FP rate at det.rate		
	80%	85%	90%
Monolithic	13.21	24.87	–
Boosted-cascade	0.296	0.465	1.135

windows satisfies these criteria for a given labelled face, that labelled face is a *false negative*. Detections that do not satisfy the overlap criteria for any of the labelled faces are regarded as *false positives*.

6. Results

Fig. 6 shows detection results for both COBA methods, viz. the cluster-weighted models and the decision trees, using the monolithic detector. Fig. 7 shows detection results for both COBA methods obtained with the boosted-cascade detector. In both figures, the region selection is based on the multiscale feature set described in Section 4.2.3. Both figures are the result of varying the threshold on the confidence value obtained in the second stage of COBA. Table 1 shows numerical detection results for the brute-force detectors, i.e. the first-stage detection results. No false-positive results are shown for a detection rate of 90% for the monolithic detector, as that number is so high ($>10^3$) that analysis using COBA proved impractical.

In both figures, the graph on the left is a ROC curve, which shows the trade-off between detection rate and false-positive rate. In order to arrive at the right graph, both COBA and the object detector are tuned to obtain a certain detection rate (as specified on the horizontal axis). The right graph shows COBA's false-positive reduction rate as a function of the detection rate. Tuning of COBA is performed by varying the threshold on the confidence level—for each first-stage detection—obtained in the second stage of COBA, where the object detector itself is not tuned. Tuning of the object detector is performed by varying the output threshold of the detector [1,4]. In each figure, results are shown for a step size of 1. The graphs show that a stricter context selection causes a lower false-positive rate at the cost of a lower detection rate.

These results lead us to four observations. The first observation is that the false-positive rate can be reduced considerably while still retaining a high detection rate. Although one could in principle choose any point on the graphs to compare the results, we choose to compare detection results under conditions at which the method is practically applicable. The criteria we use for practical applicability is that the method should yield an 80, 85, or 90% detection rate. At those rates, the number of false detections is still relatively low. Table 2 lists the number of false positives per image for detection rates of 80, 85, and 90% for the five feature sets defined in Section 4.2. The best results at one false positive per image are obtained when using the boosted-cascade detector combined with our decision-tree model with step size 1 using the *fine scale-2* feature set: the false-positive rate is reduced by a factor 8.7 at a detection-rate of 80% when compared to the results obtained using a tuned object detector.

The second observation is that when using the decision-tree model in COBA, the false-positive rate is reduced by a considerably larger factor than by using the cluster-weighted model in COBA. We will elaborate on this difference in Section 7.

A third observation is that when using the boosted-cascade detector in the second stage, the detection results are better than those obtained by using the monolithic detector in the second stage, in terms of the false-positive rate and detection rate. This is caused mainly by the differences in the two detectors; on our training set, the monolithic detector yields a lower intrinsic detection rate and a higher intrinsic false-positive rate. The fact that results can be improved by using a well-performing object detector taken from current research as a 'plug-in' in the second stage of COBA clearly is beneficial.

Our fourth observation concerns results for different step sizes. Tables 3 and 4 show detection results for different step sizes, using the monolithic and boosted-cascade detector, respectively. In both tables, the best-performing feature set (the *fine scale-2* set) has been used. The results obtained are relatively insensitive to the step size for step sizes ranging from 1 to 4 for the cluster-weighted model, and step sizes 1 and 2 for the decision-tree model. A step size of, e.g. 4 implies that our region-selection model has to be evaluated at only (1/16)th of

Table 2
Detection results with a threshold chosen for various detection rates

Object detector	Feature set	Cluster-weighted model			Decision-tree model		
		FP rate at det.rate			FP rate at det.rate		
		0.80	0.85	0.90	0.80	0.85	0.90
Monolithic	Multiscale	1.343	12.546	–	0.385	14.223	–
	Fine, scale 2	0.945	11.045	–	0.305	15.132	–
	Fine, scale 4	1.646	14.317	–	0.541	17.502	–
	Coarse, scale 2	0.741	12.861	–	0.192	18.701	–
	Coarse, scale 4	23.483	24.951	–	0.434	19.007	–
Boosted-cascade	Multiscale	0.094	0.215	0.803	0.048	0.092	0.408
	Fine, scale 2	0.125	0.226	0.742	0.034	0.074	0.384
	Fine, scale 4	0.117	0.278	0.850	0.061	0.130	0.641
	Coarse, scale 2	0.076	0.165	0.730	0.043	0.088	0.437
	Coarse, scale 4	0.118	0.269	0.756	0.061	0.154	0.592

Table 3
Detection results for various step sizes, using the *fine scale-2* feature set and the monolithic detector

Context model	Step size	FP rate at det.rate	
		0.80	0.85
Cluster weighted	1	0.945	11.045
	4	1.312	13.201
	8	2.606	17.130
	12	3.554	20.833
Decision tree	1	0.305	15.132
	2	0.645	19.038
	3	1.259	19.991
	4	2.596	20.895

Table 4
Detection results for various step sizes, using the *fine scale-2* feature set and the boosted-cascade detector

Context model	Step size	FP rate at det.rate		
		0.80	0.85	0.90
Cluster weighted	1	0.125	0.226	0.742
	4	0.067	0.161	0.685
	8	0.103	0.215	0.864
	12	0.142	0.295	0.875
Decision tree	1	0.034	0.074	0.384
	2	0.054	0.105	0.474
	3	0.050	0.116	0.646
	4	0.102	0.194	0.841

the locations compared to a step size of 1, thereby yielding a speed increase of roughly a factor 15. Larger step sizes −8 and 12 for the cluster-weighted model, and 3 and 4 for the decision-tree model—reduce the number of region-selection model evaluations even further, but deteriorates the detection results. Cluster-weighted models yield a good performance at larger step sizes than the decision-tree models. This fact will be elaborated upon in Section 7.

Fig. 8 shows detection results at a step size of 1 for COBA in conjunction with the boosted-cascade detector. The differences between the results for the five feature sets is small; the *fine scale-2* representation performs best, followed by the *coarse scale-2* and multiscale representations. Our two scale-4 representations do not perform as well, although their reduction of the false-positive rate is still considerable.

Fig. 9 shows typical examples of the results obtained on images from our dataset. In each image, there are both solid white detection boxes, and detection boxes having a black and white striped edge. All these boxes denote detections obtained using the boosted-cascade object detector. The solid white boxes are final detections after using COBA, whereas the striped boxes are detections marked by COBA as too unlikely given their local context. The bottom row of images in Fig. 9 shows a number of failure modes of COBA.

7. Discussion

In this section, we discuss the performance improvement realized by COBA, and the reasons for this improvement. In Section 7.1, we discuss the performance differences between cluster-weighted models and decision trees. In Section 7.2, we discuss how the model type and features types affect COBA's performance. In Section 7.3, we elaborate the aforementioned points to explain the performance improvements. Finally, in Section 7.4, we comment on the relation between COBA and other existing work.

7.1. Performance differences between cluster-weighted models and decision trees

In our experiments, we observed that (1) cluster-weighted models perform well using larger step sizes than decision-tree models, and (2) decision-tree models perform better than cluster-weighted models in reducing the false-positive rate. Below, we discuss possible causes for these observations.

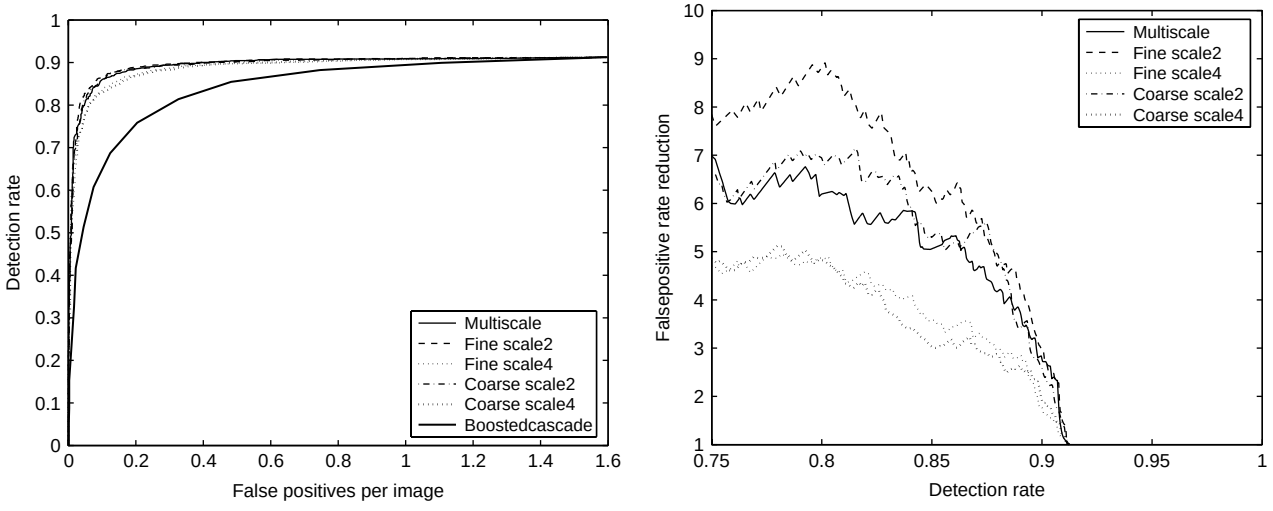

Fig. 8. Detection results for context-based face detection using the boosted-cascade detector.

Fig. 9. An overview of COBA performance on the dataset used in this work.

7.1.1. Superior performance of cluster-weighted models using large step sizes

The fact that cluster-weighted models are successful for larger step sizes than the decision-tree models can be attributed to the difference in output representation of both models. On the one hand, the cluster-weighted model outputs a PDF in terms of an expected relative location, and a covariance matrix. This PDF has a rather large spatial extent. Thus, even when

using larger step sizes, the PDFs still overlap to a large degree, and the actual object location will be assigned a large confidence. On the other hand, the decision-tree models output a conditional class probability for each of the 25 spatial classes. In the majority of cases, one class will have a probability of 1, and the rest will have a probability of 0. This means that the 'PDF' has a smaller spatial coverage than the PDF generated by a cluster-weighted model. Hence, using a larger step size will reduce the overlap of individual PDFs. As a consequence, actual object location are assigned a low confidence. Presumably, cluster-weighted models are more resilient to increasing step sizes than decision trees, because of the large extent of their PDFs.

7.1.2. Superior performance of decision-tree models

The superior performance of decision-tree models as compared to cluster-weighted models may be due to the sampling density in output space and/or the homogeneity of samples. Below, we discuss both possibilities.

First, the sampling density in the decision tree's output space (25 classes) is larger than that of the cluster-weighted model (eight clusters). In addition, preliminary experiments using a denser output sampling for the cluster-weighted model (12 clusters) show no further improvement the 8-cluster version. This suggests that the superior performance of decision-tree models over cluster-weighted models is not due to the sampling density in output space.

Second, the decision tree has a uniform sampling over all 25 output classes. In contrast, the cluster-weighted models tend to have their cluster centres closer to the actual location of the face. Effectively, the number of local samples that contribute to the probability measure of a detection, is lower in cluster-weighted models than in decision trees. Raising the number of clusters to 12, as alluded to above, does not change this result. The tendency of cluster-weighted models to place the cluster centres close to the actual object location may be either because of non-Gaussian properties of the data, or because of the training algorithm ending up in a local optimum. Decision trees are non-parametric learning methods, and are therefore better capable of handling non-Gaussian data. This suggests that the performance difference observed is due to the uniform sampling and the non-parametric nature of decision trees.

7.2. Feature types

In our selection of features for the region-selection stage in Section 4.2, we claimed that the coarse representation contains sufficient contextual information, without too much intra-class variance. We also claimed that the other two, more complex representations would deteriorate results because of the incorporation of too much intra-class variance.

Our results of Fig. 8 shows that the methods using the multiscale, or the scale-2, representations outperform the scale-4 representations. In particular, both *scale-2* representations—including the *coarse scale-2* representation—outperform the multiscale representation. The results suggest that scale-2 representations are more informative than scale-4 representations. This explains the superior performance of the *fine scale-2* representation. The fact that the multiscale representation does not outperform the *coarse scale-2* representation is probably due to the high dimensionality ($\mathbb{R}^{1734}$) of the feature space, which together with the limited number of available samples, hampers the generalisation performance of the classifier.

7.3. Explaining COBA's performance

Although our experiments show that COBA succeeds in differentiating between object detections within likely and unlikely contexts, the question that remains is: what is causing the success? We discuss three possible explanations: (1) the use of local context, (2) the relative-localisation model, and (3) the training using relative locations.

7.3.1. The use of local context

If the use of the local context of an object causes the successful performance of COBA, a similar improvement would be achieved when using a one-stage binary object detector trained on the local contextual region from which COBA takes its training samples. We trained such a detector, henceforth referred to as the *context detector*, on the 35×35 pixels training region also used for training COBA. We downsampled the region to 20×20 pixels because the number of features scales as s^4 with the linear size s of the region. The learning algorithm employed for the context detector is taken from Lienhart et al. [20]. The classifier was trained using the same image set as the brute-force classifier. Fig. 10 shows the detection results for the context detector, compared to the boosted-cascade detector, and the best-performing COBA method.

It is clear from the results that the performance of the context detector is inferior to that of both the boosted-cascade detector and the results obtained using the COBA methods. We therefore conclude that purely using a local context does not explain COBA's improvement.

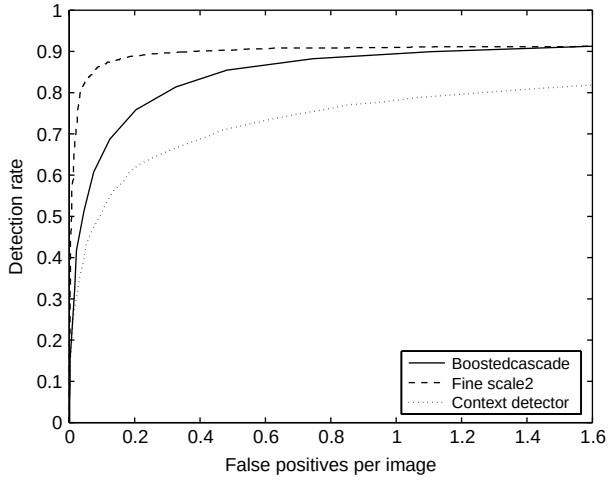

Fig. 10. Detection results for the *context detector* compared to the boosted-cascade classifier and the COBA method.

The features used for context validation may contain information that complements the information in the context detector discussed in explanation above While it is true that our context-validation method uses features outside the central face region, the features we use are a subset of those used by the context detector. This implies that the features used for context validation are included in that context detector, and that the features do not add additional information. It is important to remark, that COBA employs a subset of the features used by the context detector. Apparently, the use of this subset of features leads to an improved detection performance.

7.3.2. The relative-localisation model

The type of learning algorithm employed for inducing the contextual-validation model—cluster-weighted models or C4.5 decision trees—may explain COBA's successful performance. However, the types of algorithms employed in our context-validation stage have a rather different structure. Preliminary experiments indicate that a considerable detection improvement is also obtained when using naive Bayesian classifiers. We therefore believe that the type of learning algorithm is not the reason for the improvement.

7.3.3. Training using relative locations

Our final explanation is that the relative locations at which the samples are taken for training the model may cause COBA's performance. However, it may be argued decision-tree COBA can be transformed into the context detector introduced in Section 7.3.1. It is a mapping from all decision-tree COBA (feature, class) tuples to the appropriate feature in the context detector feature set. Since, the contributions of all local samples are added to obtain the final confidence for a raw object detection, yields a one-shot classifier that predicts a confidence based on a single classification of a 35×35-pixel window.

Our method differs from such a constructed one-shot classifier in two respects. The first difference is that we restrict—or steer—the learning process in such a way that we select a sparse subset of all features in training. This helps improve generalisation. Whether our feature selection is optimal, is an open question. The second difference is that we normalise our features by dividing each feature vector wavelet component by the sum of its elements. Preliminary experiments indicate that this normalisation has a large impact on performance, and that normalising by the wavelet-component sum yields the best detection results.

This line of reasoning indicates that training using relative locations per se does not explain COBA's performance. Rather, it led us to the identification of spatial feature selection and normalization as the main causes for the successful performance of COBA.

7.4. Related and future work

Next to COBA there are other methods that employ context to select interesting regions in images. Below, we briefly discuss two other methods, and their relevance in relation to COBA.

The first method is by Torralba [36]. They use spatial context, but in contrast to COBA they use the context on a more global level: they extract Gabor wavelets at several orientations and frequencies from the entire area of the image, and use this information to estimate a single PDF for the image describing the target-object saliency. Their method highlights a relatively large portion of the image as a region that might contain the target object, and is thus less specific than COBA in locating objects. Because COBA assumes a uniform prior probability over the image of finding the object, Torralba and Oliva's method can be integrated into COBA to weight our object confidence values, thereby effectively creating a three-stage method. This might be an attractive approach, since COBA makes the strong assumption that an object is present within a small distance of the current position.

The second method described in recent work by Kruppa et al. [16] uses a boosted cascade of simple classifiers to detect facial contexts that contain the head and the shoulders. In this respect, their method is comparable to the context detector method we used to be compared with our COBA method. There are two main differences between the method of Kruppa et al. and COBA. First, as stated in Section 1, in contrast to COBA, Kruppa et al. do not use an object-detection stage in conjunction with their context detector, but infer the object location by assuming that the object is in the centre of the detected context. Second, whereas the training region used by COBA is comparable to the context region that Kruppa et al. detect, it should be noted that the approach in COBA is not equivalent to head detection; whereas a head detector classifies an entire context region as either a head or not, COBA bases its probability measure on multiple relative location gradients obtained from windows in the vicinity of the face.

8. Conclusion and recommendations

We proposed an object-detection method named COBA, that uses spatial context to lower the number of false positives. COBA employs two stages; it performs a raw object detection in the first stage, and validates the raw detection using spatial context in the second stage. COBA is independent of the type of the object classifier used.

Human face-detection experiments involving two different types of object classifiers showed that the use of spatial context in COBA lowers the false-positive rate by a factor of up to 8.7 at a detection rate of 80%. Moreover, COBA performs favourably when compared with current methods. The main reason for the improvement obtained using COBA is due to the specific manner in which COBA structures the learning phase of the contextual validation.

Hence, we may conclude that the use of spatial context is a viable means to reduce the number of false positives.

Three recommendations for further research are given below. We believe it is possible to improve COBA by (1) exploiting other contextual cues, such as the location of the horizon or the average image depth [27], (2) taking prior

knowledge on the presence of objects, such as described in [17,36], into account (the current method makes the strong assumption that an object is present in the direct vicinity of the current location), and (3) predicting not only the relative position of the object with respect to the current location, but also the scale.

Acknowledgements

The authors would like to thank the anonymous referees for their constructive criticism, that improved the article considerably. It was a great help in making many details more precise, and thus better understandable for the reader.

The research is carried out within the ToKeN project Eidetic (grant number 634.000.001) of the Netherlands Organisation for Scientific Research (NWO).

References

[1] C. Papageorgiou, T. Poggio, A trainable system for object detection, International Journal of Computer Vision 38 (1) (2000) 15–33.
[2] K.K. Sung, T. Poggio, Example-based learning for view-based human face detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1) (1998) 39–51.
[3] H. Schneiderman, T. Kanade, A statistical method for 3D object detection applied to faces and cars, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2000.
[4] P. Viola, M. Jones, Robust real-time object detection, in: Proceedings of the Second International Workshop on Statistical and Computational Theories of Vision—Modeling, Learning, Computing, and Sampling, Vancouver, Canada, 2001.
[5] E.O. Postma, H.J. van den Herik, P.T.W. Hudson, SCAN: a scalable neural model of covert attention, Neural Networks 10 (6) (1997) 993–1015.
[6] F.O. Postma, SCAN: a neural model of covert attention, PhD thesis, Rijksuniversiteit Limburg, Maastricht, The Netherlands, 1994.
[7] I. Biederman, On the semantics of a glance at a scene in: M. Kubovy, J. Pomerantz (Eds.), Perceptual Organization, Erlbaum, Hillsdale, NJ, 1981.
[8] M.B. Lewis, A.J. Edmonds, Face detection: mapping human performance, Perception 32 (2003) 903–920.
[9] A.L. Yarbus, Eye Movements and Vision, Plenum Press, New York, 1967.
[10] M.I. Posner, S.E. Peterson, The attention system of the human brain, Annual Review of Neuroscience 13 (1990) 25–43.
[11] S.E. Palmer, Vision Science, Photons to Phenomenology, MIT Press, Cambridge, MA, 1999.
[12] G.W. Humphreys, V. Bruce, Visual Attention, Erlbaum, Hillsdale, NJ, 1989. Chapter 5.
[13] C.W. Eriksen, Y.Y. Yeh, Allocation of attention in the visual field, Journal of Experimental Psychology: Human Perception and Performance 11 (5) (1985) 583–597.
[14] C.W. Eriksen, J.S. James, Visual attention within and around the focus of attention, Perception and Psychophysics 40 (4) (1986) 225–240.
[15] C. Koch, S. Ullman, Shifts in selective visual attention: towards the underlying neural circuitry, Human Neurobiology 4 (1985) 35–41.
[16] H. Kruppa, M.C. Santana, B. Schiele, Fast and robust face finding via local context, in: Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Nice, France, 2003.
[17] A. Torralba, A. Oliva, Statistics of natural image categories, Network Computation in Neural Systems 14 (2003) 391–412.
[18] L. Chengjun, A Bayesian discrimination features method for face detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (6) (2003) 725–740.
[19] H. Schneiderman, T. Kanade, Object detection using the statistics of parts, International Journal of Computer Vision 56 (3) (2004) 151–177.
[20] R. Lienhart, L. Liang, A. Kuranov, An extended set of haar-like features for rapid object detection, Technical Report, Intel Research, 2002.
[21] Y. Freund, R. Shapire, Experiments with a new boosting algorithm in: L. Saitta (Ed.), Proceedings of the Thirteenth International Conference on Machine Learning, Bari, Italy (1996), pp. 148–156.
[22] B. ter Haar Romeny, L.M.J. Florack, Front-end vision: a multiscale geometry engine, in: Proceedings of the IEEE International Workshop on Biologically Motivated Computer Vision, Lecture Notes in Computer Science, Springer, Heidelberg, Germany, 2000.
[23] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, first ed., MIT Press, Cambridge, MA, 2001.
[24] D.C. Montgomery, G.C. Runger, Applied Statistics and Probability for Engineers, Wiley, New York, 1994.
[25] N. Gershenfeld, The Nature of Mathematical Modeling, Cambridge University Press, Cambridge, MA, 1999.
[26] J.J. Verbeek, N. Vlassis, B. Kröse, Efficient greedy learning of gaussian mixture models, Neural Computation 15 (2) (2003) 469–485.
[27] A. Torralba, A. Oliva, Depth estimation from image structure, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (9) (2002) 1226–1238.
[28] J.R. Quinlan, C4.5: programs for machine learning The Morgan Kaufmann Series in Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.
[29] E. Hjelmås, B.K. Low, Face detection: a survey, Computer Vision and Image Understanding 83 (3) (2001) 236–274.
[30] A.M. Martinez, R. Benavente, The AR face database, Technical Report CVC #24, Electrical and Computer Engineering, Purdue University, 1998.
[31] C. Vinette, F. Gosselin, P.G. Schyns, Spatio-temporal dynamics of face recognition in a flash: it's in the eyes, Cognitive Science 28 (2004) 289–301.
[32] G.C. Cawley, MATLAB support vector machine toolbox (v0.50β), University of East Anglia, School of Information Systems, Norwich, Norfolk, UK, 2000. Available from: http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox.
[33] B. Rifkin, M. Nadermann, P. Moreno, Svmfu: a fast, flexible support vector machine classification algorithm, Research abstract, MIT CBCL, 2001.
[34] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Content-based image retrieval at the end of the early years, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (12) (2000) 1349–1380.
[35] T.V. Pham, M. Worring, A.W.M. Smeulders, Face detection by aggregated bayesian network classifiers, Pattern Recognition Letters 23 (2002) 451–461.
[36] A. Torralba, Contextual modulation of target saliency in: T.G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2002.