IDIAP COMMUNICATION REPORT

FACE DETECTION USING FERNS

Venkatesh Bala Subburaman Sébastien Marcel

Idiap-Com-01-2011

DECEMBER 2011

Centre du Parc, Rue Marconi 19, P.O. Box 592, CH - 1920 Martigny
T +41 27 721 77 11 F +41 27 721 77 12 info@idiap.ch www.idiap.ch

Face Detection using Ferns

Bala Subburaman Venkatesh and Sébastien Marcel
Idiap Research Institute
P.O. Box 592
CH-1920 Martigny
Switzerland
venkatesh.bala@idiap.ch
sebastien.marcel@idiap.ch

Idiap Communication
December 2008

Abstract

This paper discusses the use of ferns (a set of binary features) for face detection. The binary feature used here is the sign of pixel intensity difference. Ferns were first introduced for keypoint recognition and showed good performance, and improving the recognition speed. Keypoint recognition deals with classification of few hundred different classes, while face detection is a two-class problem with an unbalanced data. For keypoint recognition random pixel pairs proved to be good enough while we used conditional mutual information criteria to select a small subset of informative binary feature to build class conditional densities and a Naive Bayesian classifier is used for face and non-face classification. We compared our approach with boosted haar-like features, modified census transform (MCT), and local binary pattern on a single stage classifier. Results shows that ferns when compared to haar-like features are robust to illumination changes and comparable to boosted MCT feature. Finally a cascade of classifiers was built and the performance on cropped face images and the localization results using Jesorsky measure are reported on XM2VTS and BANCA database.

1 Introduction

Computer vision applications are now ubiquitously found in mobile phones, digital camera, cars, toys, hospitals, airports, and security areas, and this has been possible by the availability of fast processors and robust algorithms. One of the problem in computer vision is face detection. The goal of *face detection* is to detect all faces in an image. This is an easy task for humans, indeed babies are able to recognize human faces very early. However, this task becomes challenging for a machine as the face image captured with a vision sensor, gets altered by pose variations (rotation out-of-plane), camera angle, illumination, facial expressions, occlusions (glasses, sunglasses, hat), age and gender.

There are many closely related problems of face detection. *Facial feature detection* aims at detecting the presence and location of features, such as eyes, nose, nostrils, eyebrow, mouth, lips and ears, with the assumption that there is only one face in an image. *Face recognition* or *face identification* compares an input image (probe) against a database (gallery) and reports a match, if any. *Face authentication* verifies the claim of the identity of an individual in an input image. *Face tracking* methods continuously estimate the location and possibly the orientation of a face in an image sequence, while facial expression recognition concerns identifying the affective states (happy, sad, surprised, etc.) of humans. Clearly, face detection is the first step in any of the above automated system.

In this report we explore the use of fern features for frontal face detection task. Before going into the technique, we give a brief review on different techniques proposed for face detection task in next section, and in Sec. 3, we describe the preliminary experiments conducted for frontal face detection, and finally conclusion is given in Sec. 4.

2 Brief Review

We split this review into two parts, one focusing on classification techniques and the other describing various feature extraction process.

2.1 Classification Techniques

Face detection is a very well explored area and details of different methods are available in two survey papers by Yang et al. [9] and Hjelmas and Low [36]. Face detection methods can be categorized into *feature-based* and *appearance-based* approach. Feature-based approaches uses explicit features such as the eye, nose, mouth, elliptical shape of face, and distance between various features [37]. The limitation with feature-based method is that it is not possible to translate all human knowledge into explicit rules. Furthermore, the extraction of facial features can be difficult as some of the features could be occluded, or altered due to pose and illumination variations. Also, the face can be too small that extracting individual features would not be feasible. Skin color models are also used for the detection of face [24]. Color information helps to reduce the search space by focusing on regions with likely skin color. Unfortunately skin color cannot be used in gray scale images or when the faces are artificially colored (e.g., spectators having country flag painted on face), and mainly in a lot of conditions color is not reliable.

The excellent results achieved by appearance-based approach have lead research in this direction. The advantage of appearance-based methods is that the knowledge about face is learned implicitly by the training algorithm. In this approach a subimage of a fixed size is given to a classifier which takes a decision if it is a face or a non-face. Many different algorithms have been proposed for face detection; Subspace methods [33], Neural Network [27], Support Vector Machines [20], Sparse Network of Winnows [25], Naive Bayes Classifier [28], and Information Theoretical approach [4]. Appearance based methods require the input image to be scanned at every location and scale. As a consequence the number of

windows that needs to be tested easily reaches millions depending on the size of the image and the accuracy of the search. Therefore these methods need a classifier with a high true positive rate (detection rate), but also with an extremely low false positive rate (false alarm), typically of the order of 10^{-6}.

In the early 1990's, Sirovich and Kirby [13] developed a technique using PCA to efficiently represent human faces. Turk and Pentland [33] later proposed to perform Principal Component Analysis (PCA) on training face images and to use the eigenvectors, also called eigenfaces, as a face template. A candidate subwindow region is classified according to the distance computed in PCA subspace after projection. The distance can be interpreted as a measure of faceness. PCA is an intuitive and appropriate way of constructing a subspace for representing an object class in many cases. However, for modeling the manifold of face images, PCA is not necessarily optimal. The face space might be better represented by dividing it into subclasses, and this technique was first applied by Sung and Poggio [30] by modeling the subclasses of face and non-face using mixture of multi-dimensional Gaussians. Each cluster was decomposed in PCA subspace. A set of Mahalanobis-like and Euclidean distance were computed for a new image with respect to 6 face and 6 non-face clusters and given to a trained Multi-Layer Perceptron (MLP) for face/non-face classification.

Rowley et al. [27] incorporated face knowledge in a retinally connected neural network and reported results on a difficult dataset. To improve the performance, multiple neural networks were trained and the output were combined with an arbitration strategy. Feraud et al. [23] suggested a different neural approach, based on a constrained generative model (CGM). Their CGM is an autoassociative fully connected MLP with three layers of weight. The idea behind this model is to force nonlinear PCA to be performed by modifying the projection of non-face examples to be close to the face examples. Classification is obtained by considering the reconstruction error of the CGM.

Osuna et al. [20] used Support Vector Machine (SVM) for face detection. A SVM with a polynomial kernel function is trained for face/non-face classification. One of the main advantage of SVM is that it can work with few training samples. Colmenarez and Huang [4] proposed a system based on Kullback relative information for face detection. A joint-histogram for each pair of pixels in the training set is used to create probability functions for face and non-face classes. Since pixel values are highly correlated with neighboring pixel values, it is treated as a first order Markov process and the pixel values in the gray-level images are requantized to four levels. The training procedure results in a set of look-up tables with likelihood ratios, which could be used during test by selecting patterns with high likelihood. Schneiderman and Kanade [28] describe a Naive Bayes classifier to estimate the joint probability of local appearance and position of face patterns (subregions of face) at multiple resolutions. At each scale, a face image is decomposed into four rectangular subregions. These subregions are then projected to a lower dimensional space using PCA and quantized into a finite set of patterns, and the statistics of each projected subregions are estimated from the projected samples to encode local appearance. Their method decides that a face is present when the likelihood ratio is greater than the ratio of prior probabilities. They also extended this method with wavelet representation to detect profile faces and cars [29].

Roth et al. [25] used Sparse Network of Winnows (SNoW) learning architecture for face detection. The SNoW learning architecture is a sparse network of linear functions that utilizes the Winnow update rule [17]. Their face detector makes use of Boolean features that encode the positions and intensity value of pixels. Their system shows better performance than other methods discussed above. Garcia and Delakis [8] proposed a convolution neural network for detecting semi-frontal human face in complex images. Their method automatically derives optimal convolution filters that act as feature extractors. This technique has also been used for other object detection tasks [31].

All the above classification technique provide good results for face detection but are computationally expensive as the input pattern has to go through all the calculations before making a decision for face and non-face. In 2001, Viola and Jones [34] proposed Asymmetric Adaboost with cascade architecture

(see Fig. 1) and haar-like features for face detection which achieves good performance in real-time. The key idea for achieving fast detection is that the complexity of the classifier (i.e number of features in each classifier) increases when moving forward in the cascade, and focus more on face like region. With this breakthrough approach, many research papers focuses on boosting framework, cascade architecture, and features which are fast to compute and robust to illumination changes. Many different features and boosting strategies have been developed over last few years.

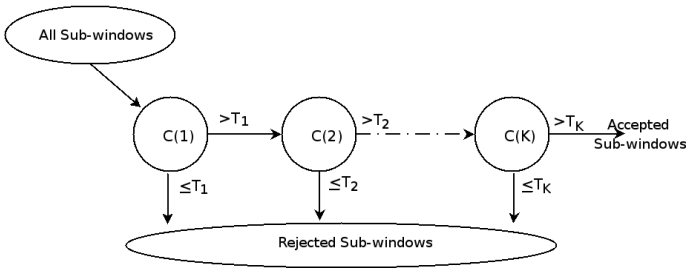

Figure 1: Cascade architecture. A series of classifiers $C(k), k = 1, ..., K$, are applied to every sub-window. A sub-window is rejected if it is lesser than the stage threshold $T(k)$, otherwise it is passed on to the next stage. The thresholds are selected such that large number of background sub-windows are rejected by first few classifiers.

Lienhart et al. [16] introduced rotated haar-like feature and its implementation for fast computation. In their paper they analyze the three boosting algorithms (Discrete, Real and Gentle Adaboost) and empirically show that Gentleboost performs better than the rest with lower computational complexity. They analyzed the effect of feature scaling, input pattern size, basic vs. extended haar-like feature, tree vs. stump decision classifier, and size of training set. They have shown that having large training sets only improve performance slightly, but also that while scanning the image at different scales, the performance can degrade if the scaling is fractional. Stan Z. Li et al. [14] proposed Floatboost technique to backtrack and delete those weak classifiers that are non-effective or unfavorable in terms of error rate, which leads to a strong classifier consisting of fewer weak classifiers.

Jianxin wu et al. [35] proposed a novel cascade learning algorithm based on forward feature selection which is two order of magnitude faster than Viola-Jones approach and yields classifier of equivalent quality. Huang et al. [11] proposed a nested cascade detector in which the confidence of the strong classifier from the previous layer is used as input along with other weak classifiers to the next layer. This reduces the number of layers and features used to reach the same detection and false positive rate.

Most of the above classification methods can be extended to detect multi-view face and different techniques have been proposed: one is to have a separate detector for each pose and run the detector for all the poses; the other option is to have a pose detector and then redirect the input to the estimated pose; another option is to have a detector which jointly detects and estimates the pose of face. Rowley et al. [26] proposed a neural network classifier which estimates the rotation of face, and uses this information to normalize the input pattern and feed it to the upright face detector. Stan Z. Li et al. [15] proposed a detector-pyramid which is composed of several levels from the coarsest view partition at the top to the finest partition at the bottom. At each level of pyramid, the full range of out-of-plane rotation is partitioned into a number of sub-ranges, and the same number of detectors are trained for face detection in that partition, each specialized for a certain view sub-range.

Recently sharing features across different classes of object has gained attention for object classification.

By sharing the feature across different views (classes) it is possible to reduce the computation time. Huang et al. [10] presented vector boosting for multi-view face detection where the features are shared in the initial stage of the architecture for different views, while at the end of the architecture the features selected are more view specific. Torralbe et al. [32] proposed a multi-task learning procedure, based on boosted decision stumps, that reduces the computations at run-time and complexity at training time, by finding common features that can be shared across classes (views). When the detectors for each class are trained jointly, the selected features are generic edge-like features and have good generalization (classifier performs well on unseen test patterns).

2.2 Feature Extraction

Features are used to describe patterns in a more compact way, such that intra-class variability is decreased and inter-class variability is increased. Features extracted for object detection should be robust to illumination changes, and ideally require as little computation as possible. In pioneering work, gray scale pixel values were used directly by Rowley [27], Feraud [23], and Osuna [20] and complemented by an image preprocessing (histogram equalization, smoothing). This feature extraction mechanism was computationally very expensive. In this subsection we will discuss on different features which are being used for object detection and tracking. We will not be able to cover all types of features but we select a subset which are simple and efficient. We will discuss about haar-like features, Local Binary Patterns (LBP) and its variants, and ferns.

2.2.1 Haar-like features

The first real-time frontal face detection introduced by Viola and Jones used haar-like feature for face detection [34]. Haar-like features (Fig. 2 (a)) are derived from Haar wavelets and a feature is computed by subtracting the sum of pixel values in white and black region. Those features can be efficiently computed by using *integral image* or *summed area table*, first introduced by Crow [5] for texture mapping. At a given location *(x,y)*, in an image, the value of the *integral image* $ii(x,y)$ is the sum of the pixels above and to the left of (x,y):

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

where $i(x', y')$ is the pixel value of the original image at location (x', y'). To compute the sum of the black region shown in Fig. 3, only 4 table access and 3 simple operations are needed. Following Viola and Jones, Lienhart et al. [16] introduced rotated haar-like features (Fig. 2 (b)) which significantly enriched the original haar-like feature and could be calculated also very efficiently.

2.2.2 Local Binary Patterns

Another type of features that are becoming more and more popular in the computer vision community are for instance Local Binary Pattern (LBP) [19], Census Transform [38], and Modified Census Transform (MCT) [7]. Those features are non-parametric operators which captures the local spatial structure of an image. For example, the LBP operator is represented by

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c) 2^p$$

where g_c corresponds to the gray value of the center pixel (x_c, y_c) of a local neighborhood, $g_p (p = 0, ..., P-1)$ corresponds to the gray values of P equally spaced pixels on a circle of radius R $(R > 0)$

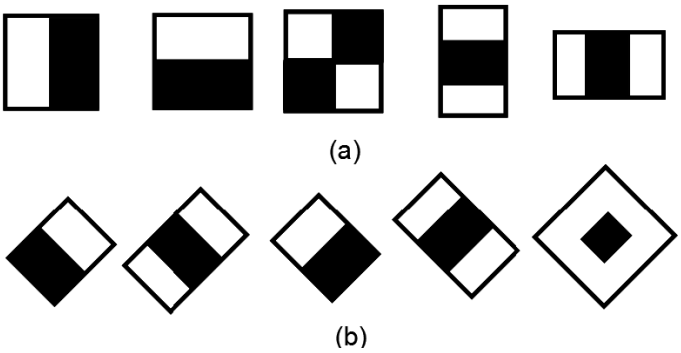

Figure 2: (a) Haar-like features used by Viola and Jones, and (b) rotated haar-like features used by Lienhart in addition to original haar.

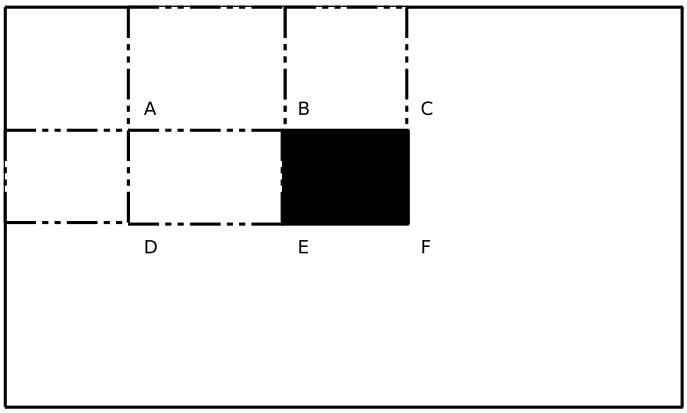

Figure 3: Computation of haar-like feature with the *integral image*. The sum in the black region is given by: $F - C - E + B$

that forms a circularly symmetric set of neighbors. The function s is defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Fig. 4 shows the computation of $LBP_{8,1}$. A modification of $LBP_{8,1}$ was introduced by Ernst et al. [7] called Modified Census Transform (MCT), which can be written as

$$MCT(x_c, y_c) = \sum_{p=0}^{8} s(g_p - g_p^*)2^p$$

where g_p^* is the average value, and g_p consists of center pixel and its 8 neighbors. MCT thus has an extra bit to encode the local structure. The number of binary comparisons for $LBP_{8,1}$, $LBP_{4,1}$, and MCT are 8, 4, and 9 respectively. Fig. 5 shows local primitives detected by $LBP_{8,1}$ operator (spots, line end, edges, and corner). In terms of texture, each LBP code can be regarded as *micro-textons*. These binary features have shown robustness to monotonic gray scale transforms.

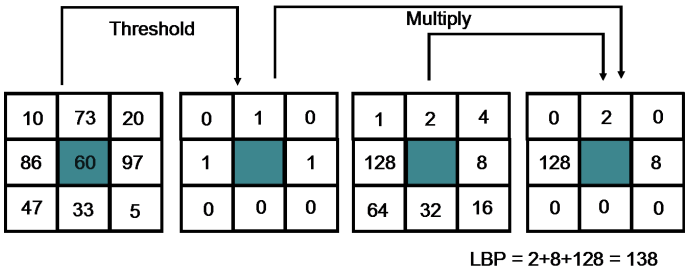

Figure 4: Computation of $LBP_{8,1}$. A LBP code for a neighborhood is produced by multiplying the threshold values with weights given to corresponding pixels and summing up the result.

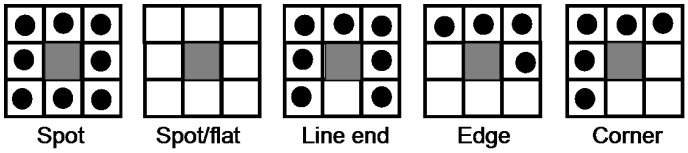

Figure 5: Different texture primitives detected by LBP. In the figure ones are represented by white square and zeros by black circles.

2.2.3 Ferns

Mustafa et al. [21] have shown that simple binary features could be used for patch classification. The binary features they proposed are based simply on the sign of pixel intensity difference. Given an image patch I a binary feature is defined as

$$f_i = \begin{cases} 1 & \text{if } I(m) \leq I(n) \\ 0 & \text{otherwise} \end{cases}$$

where $I(m)$ and $I(m)$ are the pixel intensity values at location m and n. A *fern* is defined as a set of binary features (see Fig. 6), and can basically capture the intensity variations in an image patch (see Fig. 8 for the actual pixel location overlapped on a face image). A set of location pairs (m, n) can be structured vertically, horizontally, diagonally, or circularly in a similar manner to LBP or to any arbitrary shape.

3 Preliminary Work: Frontal Face Detection using Ferns

In [21], a fast patch classification algorithm based on Semi-Naive Bayesian classifier and binary features (ferns) has been proposed for estimating the pose of an object. The main idea was to classify large number of keypoints quickly to estimate the pose of an object. In this section we explore this technique for face detection task. Our main motivations to use this approach was that it does not require any preprocessing of the image and the training time is greatly reduced when compared to boosting.

The binary feature used in our experiment is described in section 2.2.3. In [21], the pixel pair location for computing the binary feature were selected randomly. However, random selection of binary features does not help us to achieve consistent performance for face detection task and it will not be optimal for building a cascade of classifier. In next subsection, we first describe the binary feature selection process,

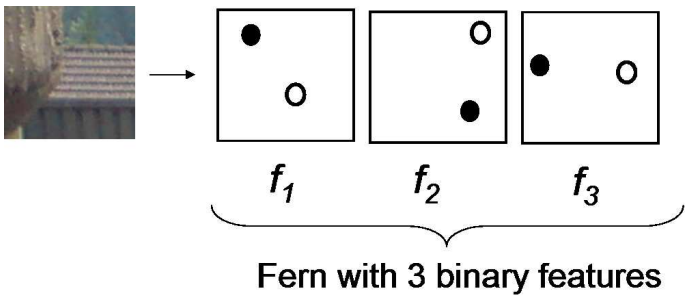

Figure 6: Fern feature. The dark and light circles shown in a white boxes are arbitrary pixel location (m and n). In this figure the fern consists of 3 binary feature (f_1, f_2, and f_3).

then describe Semi-Naive Bayesian classifier using fern features, and finally describe the experiments conducted for face/non-face classification.

3.1 Binary Feature Selection using Conditional Mutual Information

For a image of size $q \times q$ we have $q(q-1)/2$ possible binary features. The main goal of feature selection is to select a small subset of features that carries as much information as possible. The ultimate goal would be to choose $f_1, ..., f_{N_f}$ which minimizes $H(Y|f_1, ...f_{N_f})$, where H is the entropy, Y the class label (0,1), f_i binary feature (0,1), and N_f number of binary features. But this expression can not be estimated with a training set of realistic size as it requires 2^{N_f+1} probabilities (considering all $N_f + 1$ (total number of features plus the class) binary combination).

Fleuret [6] showed that features selected based on conditional mutual information maximization (CMIM) criteria and using a Naive Bayesian classifier gives performance comparable to state-of-art techniques such as boosting or SVM. The features selected using CMIM also show robustness to noisy training data. Conditional mutual information is given by

$$I(Y; f'|f) = H(Y|f) - H(Y|f', f) \tag{1}$$

where $H(Y|f)$ is the conditional entropy which measures the residual uncertainty of Y when f is known, $H(Y|f', f)$ measures the residual uncertainty of Y when f and f' are known, and $I(Y; f'|f)$ gives an estimate on the amount of information shared between Y and f' when f is known. The conditional mutual information is zero if f' and f have the same information about Y, while the value is large if f' and f have different information about Y. To select a feature that carries different information from features that are already selected, the following iterative scheme was proposed

$$f_0 = \arg\max_n I(Y; f_n)$$
$$\forall k, 0 \leq k \leq K-1, f(k+1) = \arg\max_n \underbrace{\{min_{l \leq k} I(Y; f_n|f_l)\}}_{s(n,k)}$$

The score $s(n, k)$ is low if any feature already picked is similar to f_n or if f_n does not contain any information about Y. By taking the feature f_n with maximum score $s(n, k)$, it is ensured that the new feature is different from those that are already selected and also carries information about Y.

3.2 Semi-Naive Bayesian Classifier

Naive Bayesian classifier has been successfully used for various classification tasks. We will be using Semi-Naive Bayesian technique proposed in [21] to build a two class classifier. The idea is to build the class conditional probabilities of binary feature and at run-time use these probabilities to select test input pattern with highest likelihood. We will first define the following notations:
f_i : Binary feature (sign of intensity difference of two pixels).
c_i : Class label
F_j : A Fern defined to be a set of S binary features $\{f_l, ..., f_{l+S}\}$
M : Number of ferns
$N = S \times M$, where N is total number of features in the model.
Given a set of features $f_0, f_1, ..., f_{N-1}$ the idea is to select class c_i such that

$$\hat{c_i} = \arg\max_{c_i} P(C = c_i | f_0, f_1, ..., f_{N-1}) \tag{2}$$

Bayes' Formula yields

$$P(C = c_i | f_1, f_2, ..., f_N) = \frac{P(f_0, f_1, ..., f_{N-1} | C_k) P(C_k)}{P(f_0, f_1, ..., f_{N-1})}$$

Assuming a uniform prior $P(C)$ and since the denominator is simply a scaling factor that is independent from the class the problem reduces to finding

$$\hat{c_i} = \arg\max_{c_i} P(f_0, f_1, ..., f_{N-1} | C = c_i) \tag{3}$$

The joint probability of Equation(3) is not feasible since it would require estimating and storing 2^N entries for each class. One way to simplify the representation is to assume independence between features.

$$P(f_0, f_1, ..., f_{N-1} | C = c_i) = \Pi_j^{N-1} P(f_j | C = c_i).$$

However, this completely ignores the correlation between features. To make the problem tractable while accounting for these dependencies, a compromise is to partition the features into M groups of size $S = \frac{N}{M}$. These groups are defined as *ferns* and the joint probability for feature in each fern is computed. The conditional probability becomes

$$P(f_0, f_1, ..., f_{N-1} | C = c_i) = \Pi_k^{M-1} P(F_k | C = c_i).$$

where $F_k = f_{k,0}, f_{k,1}, ..., f_{k,S-1}, k = 0, ...M-1$. The parameters M and S could be used to tune the performance and memory trade-off.

3.3 Experiments

3.3.1 Database

The database consists of 8744 training and 9232 validation faces, taken from different face database (BANCA [2], Essex, Feret [22], ORL, Stirling and Yale [3]). The face samples are aligned with respect to eye coordinates. For non-face patterns we randomly select around 50000 patterns from different images not containing any face. All the patterns are resized to $q \times q$ pixels, where $q = 19$. For testing the performance of the classifier we have three different test sets. Test set 1 contains 2360 faces with frontal illumination taken from XM2VTS database [18]. Test set 2 contains 1180 faces with side illumination (XM2VTS darkened). Test set 3 contains 580 faces from various sources (web). For non-face test patterns we took MITCBCL [1] dataset and combined both training and test non-face set to obtain 27000 patterns. Samples from these dataset are shown in Fig. 7.

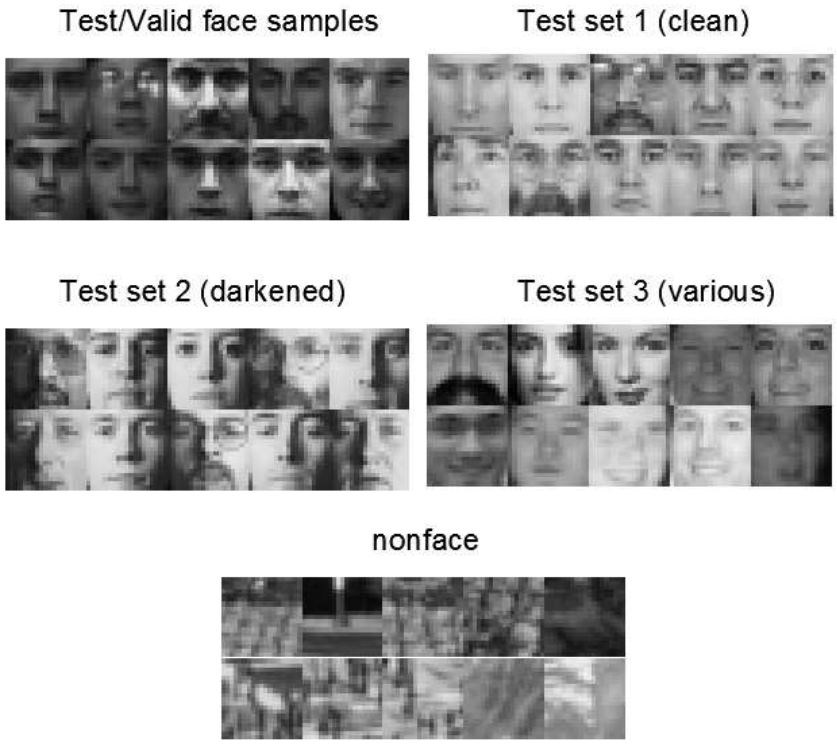

Figure 7: Samples from face and non-face training, validation and test dataset.

3.3.2 Training and Testing

Face detection is a two class problem so we have $C = \{0, 1\}$. From the training face and non-face patterns, we select N features, $N << 64890$ (for $q = 19$ we have 64890 possible binary features) using CMIM criteria as described in section 3.1. There are many ways in which the N features can be grouped in size of S. We decided to split N features into M equal sets, each of size S to form each fern F_j, $F_j = f_{h+0}, ..., f_{h+S-1}$, where $h = j \times S$, $j = 0, ..., M-1$. An example of pixel pair locations of first two ferns ($S = 9$) obtained after selection using CMIM are shown in Fig. 8. Once the selection is done, it becomes possible to build the class conditional probability $P(F_j|C = c_i)$ for all ferns using the training patterns of face and non-face. To test if a pattern is a face or a non-face we check the following condition:

$$\frac{\Pi_k^M P(F_k|C=1)}{\Pi_k^M P(F_k|C=0)} > \tau \tag{4}$$

The value τ is found using the validation set for a given detection rate. The detection rate or true positive rate (TPR) is defined as the percentage of faces that are correctly accepted in a particular data set. False positive rate (FPR) , also called false alarm rate is defined as percentage of non-face patterns accepted as face. Classification error rate is defined as the percentage of total number of miss classification (face classified as non-face and non-face classified as face).

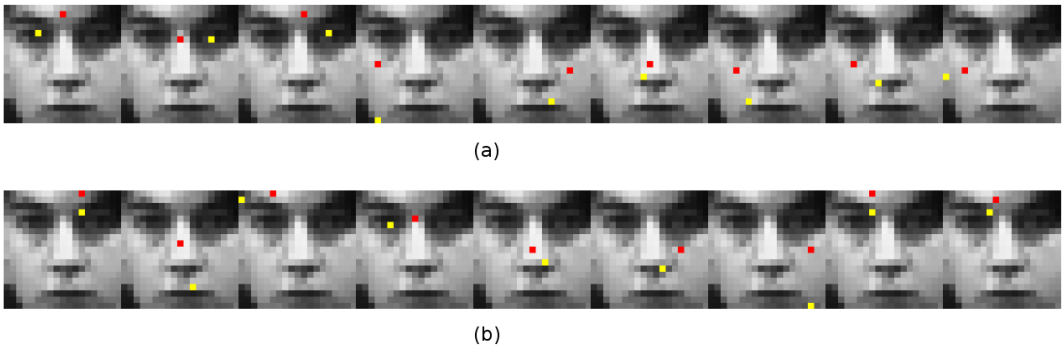

Figure 8: Fern features overlapped on face image. (a) shows the pixel pair location for first fern, and (b) shows pixel pair location for second fern. Both the ferns shown here have $S = 9$.

3.3.3 Results for Single Stage Classifier

To compare the performance of ferns and other features (MCT, haar-like feature, and LBP) we plot the receiver operating characteristic (ROC) curves and look at the TPR and FPR on the test dataset (ref. Sec. 3.3.1). MCT, LBP and haar-like features are trained using boosting techniques. To have a fair comparison among different techniques we selected equivalent number of features in each case. For our approach, we tried different values of S and M and selected $S = 9$ and $M = 10$, which performed better (see Fig. 9). We have a total of 90 binary features for $S = 9$ and $M = 10$. Therefore we set all other techniques to have equivalent to 90 binary features: the number of features for MCT is 10, 11 for $LBP_{8,1}$, and 22 for $LBP_{4,1}$. In the case of haar-like features it was difficult to decide on the exact number of features due to different ways the features were computed. We decided to take approximately 50 haar-like features based on computation cost between the binary features, and 100 features to see roughly the performance when more features are added.

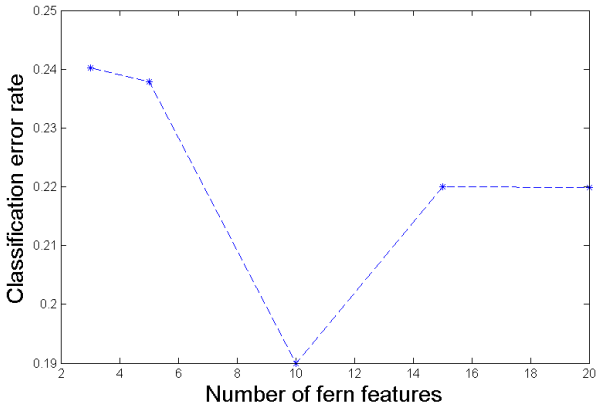

Figure 9: Classification error rate vs. number of fern features (3,5,10,15, and 20) for $S = 9$

The performance between MCT, ferns, $LBP_{8,1}$, $LBP_{4,1}$, and haar-like features are shown in Figs. 10, 11, and 12, and Tab. 1. To obtain TPR and FPR in Tab. 1, the decision threshold for each classifier was

obtained by setting the detection rate to 98% on the validation dataset. From the ROC curves and the Tab. 1, we see that haar-like features when compared to binary features are not so robust to changes in illumination. The performance of ferns and MCT features are comparable across different test data set.

Table 1: True and False positive rate for Test sets in %. The first column shows the feature type, column second, third and fourth show the corresponding TPR for test set 1, test set 2, and test set 3 respectively, and finally the last column shows FPR obtained on non-face test set.

	Test set 1	Test set 2	Test set 3	FPR
MCT	98.22	96.78	95.69	12.25
ferns	99.53	99.15	96.9	16.9
$LBP_{8,1}$	97.88	97.03	96.55	22.80
$LBP_{4,1}$	99.32	95.76	95.34	20.12
haar 100	99.41	89.14	94.83	8.53
haar 50	99.58	86.65	93.28	21.01

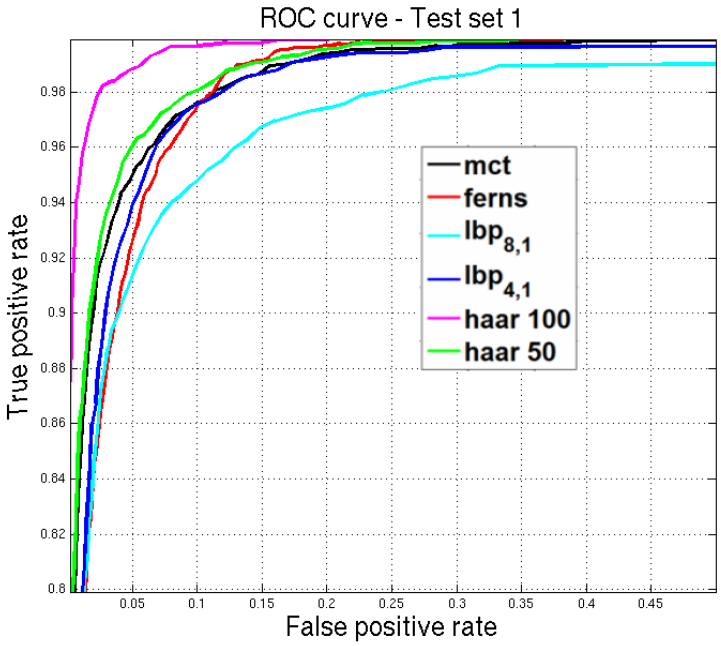

Figure 10: ROC curve for Test set 1 (XM2VTS normal).

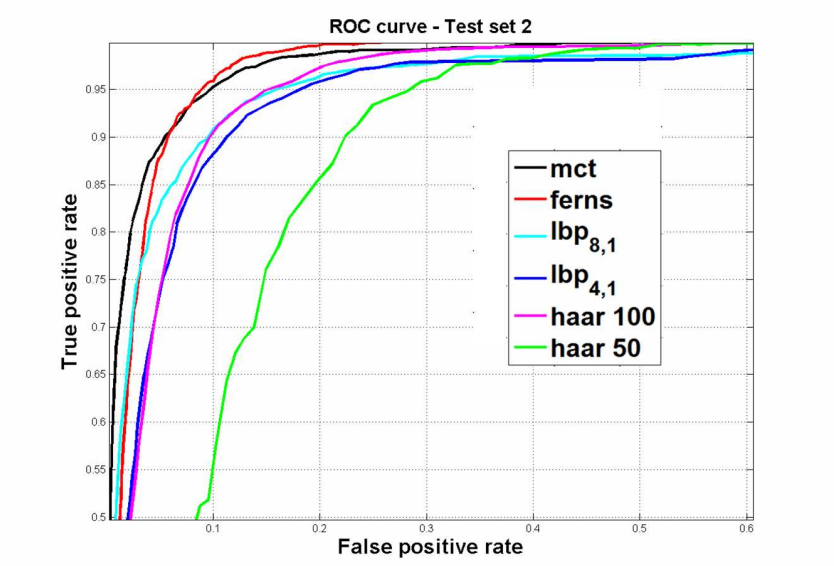

Figure 11: ROC curve for Test set 2 (XM2VTS darkened).

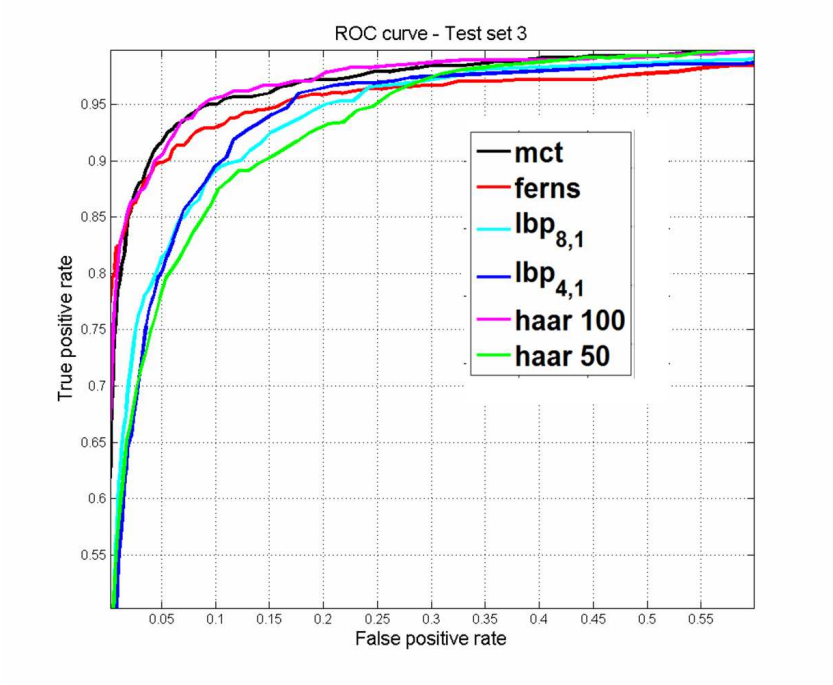

Figure 12: ROC curve for Test set 3 (various sources).

3.3.4 Cascade Model Architecture and Results

Finally, we built a cascade of classifiers to reduce the false positive rate while maintaining high true positive rate. The hyper-parameters of the cascade were selected empirically. A 40 stage cascade with a total of 800 fern feature ($S = 9$) was necessary to obtain a reasonable false positive rate of 10^{-4} by keeping the detection rate at each stage to 99.91% on training data set. The same face training and validation dataset as described in section 3.3.1 were used, while non-face samples were obtained by bootstrapping at each stage from a set of 1734 images containing no faces. We look at the performance of the cascade model on cropped faces and also by scanning full images from the BANCA (English corpus) and the XM2VTS databases. We used sliding window approach at different scales (pyramid scan) to detect faces from an image. The accuracy of localization on the detection result is given by measuring the difference between the ground truth eye center location with the estimated one [12]. The localization accuracy measure is given by

$$d_{eye} = \frac{\max(d(C_l, C_l^*), d(C_r, C_r^*)}{d(C_l, C_r)} \tag{5}$$

where $d(a, b)$ is the Euclidean distance between points a and b. C_l and C_r are the true left and right eye centers, and C_l^* and C_r^* are the estimated left and right eye centers.

Table 2: Detection rate for various test dataset in % (rejection rate = 98.4% on non-face test set). (Note we had only cropped faces for Test set 3)

	cropped faces	localization result $d_{eye} < 0.25$ (full image scan)
Test set 1	99.06	94.9
Test set 2	98.13	85.0
Test set 3	92.07	NA
BANCA Controlled	91.1	77.9
BANCA Degraded	86.77	35.24
BANCA Adverse	95.3	86.97

The results of cascade detector are shown in Tab. 2. As it can be seen from the table, there is a degradation in performance when the detector is applied on full images. We will take up this problem of scanning in our future research. Some detection results are shown in Fig. 13.

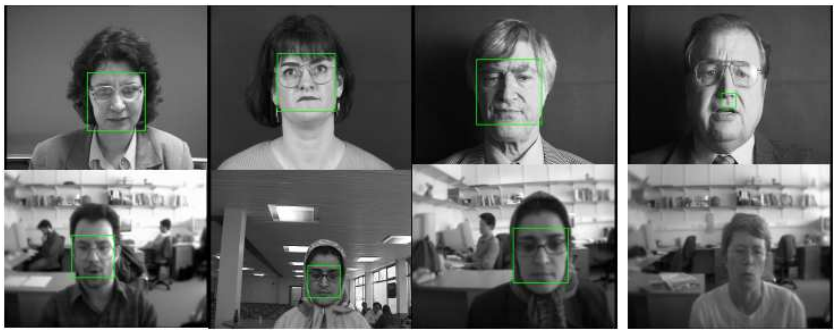

Figure 13: Some examples for face detection results on XM2VTS and BANCA database. The last column shows one false detection and one missed face.

4 Conclusion

From preliminary experiment results, we see that fern features have a potential for face detection. We think that they can be further combined with MCT and LBP to form a richer set of binary features. In our experiments we noticed that though the classifier performs very well on the cropped face images, it performed poorly when scanned on full image. The parameters of scanning such as scale, x and y step size, and fusing of multiple detection can affect drastically the performance of face detection. We would like to explore alternative ways of scanning an image which will minimize the number of misses and false detections.

References

[1] CBCL Face Database 1. In *MIT Center For Biological and Computation Learning*, http://www.ai.mit.edu/projects/cbcl.

[2] E. Bailly, S. Bengio, F. Bimbot, M. Hamouz, J. Kittler, J. Mariethoz, J. Matas, K. Messer, V. Popovici, F. Poree, B. Ruiz, , and J.-P. Thiran. The BANCA database and evaluation pro-

tocol. In *4th International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA)*, pages 625–638, Guilford, UK, 2003.

[3] P. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. In *Fourth European Conference on Computer Vision*, pages 45–58, Cambridge, UK, 1996.

[4] A. Colmenarez and T. Huang. Face detection with information-based maximum discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition*, page 782, Los Alamitos, CA, USA, 1997. IEEE Computer Society.

[5] F. C. Crow. Summed-area tables for texture mapping. In *11th Annual Conference on Computer Graphics and Interactive Techniques*, pages 207–212, New York, USA, 1984. ACM.

[6] F. Fleuret. Fast binary feature selection with conditional mutual information. In *Journal of Machine Learning Research (JMLR)*, volume 5, pages 1531–1555, 2004.

[7] B. Froba and A. Ernst. Face detection with the modified census transform. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 91–96, May 2004.

[8] C. Garcia and M. Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, 2004.

[9] E. Hjelmas and B. K. Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, September 2001.

[10] C. Huang, H. Ai, Y. Li, and S. Lao. High-performance rotation invariant multiview face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):671–686, 2007.

[11] C. Huang, H. Ai, Bo Wu, and S. Lao. Boosting nested cascade detector for multi-view face detection. In *17th International Conference on Pattern Recognition*, volume 2, pages 415–418, Washington, DC, USA, 2004. IEEE Computer Society.

[12] O. Jesorsky, K. Kirchberg, and R. Frischholz. Robust face detection using the Hausdorff distance. In *3rd International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA)*, pages 90–95, Halmstad, Sweden, 2001.

[13] M. Kirby and L. Sirovich. Application of the Karhunen-Loéve procedure for the characterisation of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:103–108, 1990.

[14] S. Z. Li and Z. Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1112–1123, 2004.

[15] S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *7th European Conference on Computer Vision*, pages 67–81, London, UK, 2002. Springer-Verlag.

[16] R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *25th Pattern Recognition Symposium*, pages 297–304, Madgeburg, Germany, 2003.

[17] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning.*, 2(4):285–318, 1988.

[18] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Audio- and Video-based Biometric Person Authentication, AVBPA'99*, pages 72–77, Washington, D.C., March 1999.

[19] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29(1):51–59, January 1996.

[20] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, page 130, Washington, DC, USA, 1997.

[21] M. Özuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, USA, 2007.

[22] P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The feret evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090 – 1104, 2000.

[23] F. Raphaël, B. Olivier, and C. Daniel. A constrained generative model applied to face detection. *Neural Processing Letters*, 5(2):11–19, 1997.

[24] H. Rein-Lien, M. Abdel-Mottaleb, and A.K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):696–706, 2002.

[25] D. Roth, M. Yang, and N. Ahuja. A SNoW-based face detector. *Advances in Neural Information Processing Systems 12*, pages 855–861, 2000.

[26] H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 1998.

[27] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

[28] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, page 45, Washington, DC, USA, 1998. IEEE Computer Society.

[29] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 01, page 1746, Los Alamitos, CA, USA, 2000. IEEE Computer Society.

[30] K. K. Sung and T. Poggio. Learning a distribution-based face model for human face detection. In *Neural Networks for Signal Processing - Proceedings of the IEEE Workshop*, pages 398–406. IEEE, Piscataway, NJ, USA, 1995.

[31] A. Szarvas, M. Yoshizawa, M. Yamamoto, and J. Ogata. Pedestrian detection with convolutional neural networks. In *IEEE Intelligent Vehicles Symposium*, pages 224–229, June 2005.

[32] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

[33] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[34] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

[35] J. Wu, J. Rehg, and M. Mullin. Learning a rare event detection cascade by direct feature selection. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

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

[37] K. Yow and R. Cipolla. Feature-based human face detection. *Image and Vision Computing*, 15(9):713–735, 1997.

[38] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *European Conference on Computer Vision*, pages 151–158, 1994.