Object Detection Using Feature Subset Selection

Zehang Sun[1], George Bebis[1] and Ronald Miller[2]

[1]Computer Vision Lab. Department of Computer Science, University of Nevada, Reno

[2]Vehicle Design R&A Department, Ford Motor Company, Dearborn, MI

(zehang,bebis)@cs.unr.edu, rmille47@ford.com

Abstract

Past work on object detection has emphasized the issues of feature extraction and classification, however, relatively less attention has been given to the critical issue of feature selection. The main trend in feature extraction has been representing the data in a lower dimensional space, for example, using Principal Component Analysis (PCA). Without using an effective scheme to select an appropriate set of features in this space, however, these methods rely mostly on powerful classification algorithms to deal with redundant and irrelevant features. In this paper, we argue that feature selection is an important problem in object detection and demonstrate that Genetic Algorithms (GAs) provide a simple, general, and powerful framework for selecting good subsets of features, leading to improved detection rates. As a case study, we have considered PCA for feature extraction and Support Vector Machines (SVMs) for classification. The goal is searching the PCA space using GAs to select a subset of eigenvectors encoding important information about the target concept of interest. This is in contrast to traditional methods selecting some percentage of the top eigenvectors to represent the target concept, independently of the classification task. We have tested the proposed framework on two challenging applications: vehicle detection and face detection. Our experimental results illustrate significant performance improvements in both cases.

Keywords

feature subset selection, genetic algorithms, vehicle detection, face detection, support vector machines.

I. Introduction

The majority of real-world object detection problems involve "concepts" (e.g., face, vehicle) rather than specific objects. Usually, these "conceptual objects" have large within class variabilities. As a result, there is no easy way to come up with an analytical decision boundary to separate a certain "conceptual object" against others. One feasible approach is to learn the decision boundary from a set of training examples using supervised learning where each training instance is associated with a class label. Building an object detection system under this framework involves two main steps (1) extracting a number of features, and (2) training a classifier using the extracted features to distinguish among different class instances.

Choosing an appropriate set of features is critical when designing pattern classification systems under the framework of supervised learning. Often, a large number of features is extracted to better represent the target concept. Without employing some kind of feature selection strategy, however, many of them could be either redundant or even irrelevant to the classification task. Watanabe [1] has shown that it is possible to make two arbitrary patterns similar by encoding them with a sufficiently large number of redundant features. As a result, the classifier might not be able to generalize nicely.

Ideally, we would like to use only features having high separability power while ignoring or paying less attention to the rest. For instance, in order to allow a vehicle detector to generalize nicely, it would be necessary to exclude features encoding fine details which might appear in specific vehicles only. A limited yet salient feature set can simplify both the pattern representation and the classifiers that are built on the selected representation. Consequently, the resulting classifier will be more efficient.

In most practical cases, relevant features are not known *a-priori*. Finding out what features to use in a classification task is referred to as *feature selection*. Although there has been a great deal of work in machine learning and related areas to address this issue [2][3] these results have not been fully explored or exploited in emerging computer vision applications. Only recently there has been an increased interest in deploying feature selection in applications such as face detection [4][5], gender classification [6][7], vehicle detection [4][8], image fusion for face recognition [9], target detection [10], pedestrian detection [11], tracking [12], image retrieval [13], and video categorization [14].

Most efforts in the literature have largely ignored the feature selection problem and have focused mainly on developing effective feature extraction methods [15] and employing powerful classifiers (e.g., probabilistic [16], Hidden Markov Models (HMMs) [17] , Neural Networks (NNs) [18], SVMs [19]). The main trend in feature extraction has been representing the data in a lower dimensional space computed through a linear or non-linear transformation satisfying certain properties (e.g., PCA [20], Linear Discriminant Analysis (LDA) [21], Independent Components Analysis (ICA) [22], Factor Analysis (FA)[23], and others [15]). The goal is finding a new set of features that represent the target concept in a more compact and robust way but also providing more discriminative information. Without using effective schemes to select an appropriate subset of features in the computed subspaces, however, these methods rely mostly on classification algorithms to deal with the issues of redundant and irrelevant features. This might be problematic, especially when the number of training examples is small compared to the number of features (i.e., curse of dimensionality problem [15][24]).

We argue and demonstrate the importance of feature selection in the context of two challenging object detection problems: vehicle detection and face detection. As a case study, we have considered the well-known method of PCA for feature extraction and SVMs for classification. Feature extraction using PCA has received considerable attention in the computer vision area [20][25][26]. It represents an image in a low dimensional space spanned by the principal components of the covariance matrix of the data. Although PCA provides a way to represent an image in an optimum way (i.e., minimizes reconstruction error), several studies have shown that not every principal eigenvectors encode useful information for classification purposes. We elaborate more on this issue in Section IV-A).

In this paper, we propose using GAs to search the space of eigenvectors with the goal of selecting a subset of eigenvectors encoding important information about the target concept of interest. This is in contrast to the typical strategy of picking a percentage of the top eigenvectors to represent the target concept, independently of the classification task. The proposed approach has the advantage that it is simple, general, and powerful. An earlier version of this work has appeared in [4] and it relates to our previous work on gender classification [6][7], however, the size of the classes considered here (e.g., object vs. non-object) are larger and in principle quite different from each other.

The rest of the paper is organized as follows: In Section II, we review the problem of feature selection, emphasizing different search and evaluation strategies. An overview of the proposed method is presented in Section III. In Section IV we discuss feature extraction using PCA. In particular, we discuss the problem of understanding the information encoded by different eigenvectors. Section V, presents our approach to choosing an appropriate subset of eigenvectors using genetic search. In particular, we discuss the issues of encoding and fitness evaluation. Section VI presents a brief review on SVMs. Our experimental results and comparisons using genetic eigenvector selection for vehicle and face detection are presented in Sections VII and VIII correspondingly. An analysis of our experimental results is presented in Section IX. Finally, Section X presents our conclusions and plans for future work.

II. Background on Feature Selection

Finding out which features to use for a particular problem is referred to as feature selection. Given a set of d features, the problem is selecting a subset of size m that leads to the smallest classification error. This is essentially an optimization problem that involves searching the space of possible feature subsets to find one that is optimal or near-optimal with respect to a certain criterion. A number of feature selection approaches have been proposed in the literature [27][2][3], [28][15][29]. There are two main components in every feature subset selection system: the search strategy used to pick the feature subsets and the evaluation method used to test their goodness based on some criteria. We review both of them below.

A. Search strategies

Search strategies can be classified into one of the following three categories: (1) optimal, (2) heuristic, and (3) randomized. Exhaustive search is the most straightforward approach to optimal feature selection [15]. However, the number of possible subsets grows combinatorially, which makes the exhaustive search impractical for even moderate size of features. The only optimal feature selection method which avoids the exhaustive search is based on the branch and bound algorithm [27][2]. This method requires the monotonicity property of the criterion function, which most commonly used criterion function do not satisfy.

Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS) are two well-known heuristic feature selection schemes [30]. SFS, starting with an empty feature set, selects the best single feature and then adds that feature to the feature set. SBS starts with the entire feature set and at each step drops the feature whose absence least decreases the performance. Combining SFS and SBS gives birth to the "plus l-take away r" feature selection method [31], which first enlarges the feature subset by adding l using SFS and then deletes r features using SBS. Sequential Forward Floating Search (SFFS) and Sequential Backward Floating Search (SBFS) [32] are generalizations of the "plus l - take away r" method . The values of l and r are determined automatically and updated dynamically in SFFS and SBFS. Since these strategies make local decisions, they cannot be expected to find globally optimal solutions.

In randomized search, probabilistic steps or a sampling process are employed. The Relief algorithm [33] and several extension of it [34] are the typical randomized search approaches. Based on their estimated effectiveness for classification, features are assigned weights in the relief algorithm. Then, features whose weights exceed a user-determined threshold are selected to train the classifier. Recently, GAs [35] have attracted more and more attention in the feature selection area. Siedlecki et al. [36] presented one of the earliest studies of GA-based feature selection in the context of a K-nearest-neighbor classifiers. Yang et al. [29] proposed a feature selection approach using GAs and NNs for classification. A standard GA with rank-based selection strategy was used. The rank-based selection method depends on a predefined parameter $p \in (0.5\ \ 1)$. Specifically, the probability of selecting the highest ranked individual is p and that of the kth highest ranked individual is $p(1-p)^{(k-1)}$. They tested their methods using several benchmark real-world pattern classification problems and reported improved results. However, they used the accuracy on the test set in the fitness function, which is not appropriate since it introduces bias to the final classification. Chtioui et al. [37] investigated a GA approach for feature selection in a seed discrimination problem. Using standard GA operators, they selected the best feature subset from a set of 73 features. Vafaie et al. [38] conducted a comparison between important score (IS)–a greedy-like feature section method and GAs. They represented the feature selection problem using binary encoding and standard GA operators. The evaluation function was solely based on classification performance. Using several real world problems, they found that GAs are more robust at the expense of more computational effort.

B. Evaluation strategies

Each of the evaluation strategies belongs to one of two categories: (1) filter and (2) wrapper. The distinction is made depending on whether feature subset evaluation is performed using the learning algorithm employed in the classifier design (i.e., wrapper) or not (i.e., filter). Filter approaches are computationally more efficient than wrapper approaches since they evaluate the goodness of selected features using criteria that can be tested quickly (e.g., reducing the correlation or the mutual information among features). This, however, could lead to non-optimal features, especially, when the features dependent on the classifier. As a result, classifier performance might be poor. Wrapper approaches on the other hand perform evaluation by training the classifier using the selected features and estimating the classification error using a validation set. Although this is a slower procedure, the features selected are usually more optimal for the classifier employed.

III. Method Overview

Traditionally, there are three main steps in building a pattern classification system using supervised learning. First, some preprocessing is applied to the input patterns (e.g., normalize the pattern with respect to size and orientation, compensate for light variations, reduce noise etc.). Second, feature extraction is applied to represent patterns by a compact set of features. The last step involves training a classifier to learn to assign input patterns to their correct category. In most cases, no explicit feature selection step takes place besides feature weighting performed implicitly by the classifier.

Fig. 1 illustrates the main steps of the approach employed here. The main difference from the traditional approach is the inclusion of a step that performs feature selection using GAs. Feature extraction is carried out using PCA to project the data in a lower-dimensional space. The goal of feature selection is then to choose a subset of eigenvectors in this space, encoding mostly important information about the target concept of interest. We use a wrapper-based approach to evaluate the quality of the selected eigenvectors. Specifically, we use feedback from a SVM classifier to guide the GA search in selecting a good subset of eigenvectors, improving detection accuracy. The evaluation function used here contains two terms, the first based on classification accuracy on a validation set and the second on the number of eigenvectors selected. Given a set of eigenvectors, a binary encoding scheme is used to represent the presence or absence of a particular eigenvector in the solutions generated during evolution.

IV. Feature Extraction Using PCA

Eigenspace representations of images use PCA to linearly project an image in a low-dimensional space [20]. This space is spanned by the principal components (i.e., eigenvectors corresponding to the largest eigenvalues) of the distribution of the training images. After an image has been projected in the eigenspace, a feature vector containing the coefficients of the projection is used to represent the image. We summarize the main ideas below:

Each image $I(x, y)$ is represented as a $N \times N$ vector Γ_i. First the average face Ψ is computed:

$$\Psi = \frac{1}{R} \sum_{i=1}^{R} \Gamma_i \tag{1}$$

where R is the number of faces in the training set. Next, the difference Φ of each face from the average face is computed: $\Phi_i = \Gamma_i - \Psi$. Then the covariance matrix is estimated by:

$$C = \frac{1}{R} \sum_{i=1}^{R} \Phi_i \Phi_i^T = AA^T, \tag{2}$$

where, $A = [\Phi_1 \Phi_2 \ldots \Phi_R]$. The eigenspace can then be defined by computing the eigenvectors μ_i of C. Since C is very large ($N \times N$), computing its eigenvector will be very expensive. Instead, we can compute ν_i, the eigenvectors of $A^T A$, an $R \times R$ matrix. Then μ_i can be computed from ν_i as follows:

$$\mu_i = \sum_{j=1}^{R} \nu_{ij} \Phi_j, j = 1 \ldots R. \tag{3}$$

Usually, we only need to keep a smaller number of eigenvectors R_k corresponding to the largest eigenvalues. Given a new image, Γ, we subtract the mean ($\Phi = \Gamma - \Psi$) and compute the projection:

$$\widetilde{\Phi} = \sum_{i=1}^{R_k} w_i \mu_i. \tag{4}$$

where $w_i = \mu_i^T \Gamma$ are the coefficients of the projection (i.e., eigenfeatures).

The projection coefficients allow us to represent images as linear combinations of the eigenvectors. It is well known that the projection coefficients define a compact image representation and that a given image can be reconstructed from its projection coefficients and the eigenvectors (i.e., basis).

A. What Information is Encoded by Different Eigenvectors?

There have been several attempts to understand what information is encoded by different eigenvectors, and the usefulness of this information with respect to various tasks [39][40][41][42]. These studies have concluded that different tasks make different demands in terms of the information that needs to be processed, and that this information is not contained in the same ranges of eigenvectors. For example, the first few eigenvectors seem to encode lighting while other eigenvectors seem to encode local features [42]. We have made similar observations by analyzing the eigenvectors obtained from our data sets. Fig.2, for example, shows some of the eigenvectors computed from our vehicle detection data set. Obviously, eigenvectors 2 and 4 encode more lighting information than others, while eigenvectors 8 and 12 encode more information about some specific local features. Similar comments can be made for the eigenvectors derived from our face detection data set, as shown in Fig. 3. Once again, eigenvectors 2 and 5 seem to encode mostly lighting while eigenvectors 8, 9 and 22 seem to encode mostly local information. Eigenvector 150 seems to encode mostly noise in both cases.

Obviously, the question is how to choose eigenvectors encoding important information about the target concept of interest. The common practice of choosing the eigenvectors corresponding to large eigenvalues might not be the best choice as has been illustrated by Balci et al. [43], Etemad et al. [21], and Sun et al. [6][7]. In [43], PCA features were used with a NN classifier. Using pruning to improve classifier performance, they were also able to monitor which eigenvectors contribute to gender classification. Their experiments showed that not all of the high eigenvectors contributed to gender classification and that some of them had been discarded by the network. In [21], the discriminatory power of eigenvectors in a face recognition task was investigated. They found out that the recognition information of eigenvectors does not decrease monotonically with their corresponding eigenvalues. Many times, there were cases where an eigenvector corresponding to a small eigenvalue had higher discriminatory power than an eigenvector corresponding to a large eigenvalue. In this study, we apply feature selection using GAs to search the space of eigenvectors with the goal of selecting a subset of them encoding important information about the target concept of interest. In [6][7], the problem of selecting a subset of eigenvectors representing mostly gender information was considered. Using an approach similar to the one proposed here, it was illustrated that certain eigenvectors, not necessarily the top ones, were more important for gender classification than others.

V. Genetic Eigenvector Selection

A. A brief review of GAs

GAs are a class of optimization procedures inspired by the biological mechanisms of reproduction. In the past, they have been used to solve various problems including target recognition [44], object recognition [45][46], face recognition [47], and face detection/verification [48]. This section contains a brief summary of the fundamentals of GAs. Goldberg [35] provides a great introduction to GAs and the reader is referred to this source, as well as to the survey paper of Srinivas et al. [49] for further information.

GAs operate iteratively on a population of structures, each one of which represents a candidate solution to the problem at hand, properly encoded as a string of symbols (e.g., binary). A randomly generated set of such strings forms the initial population from which the GA starts its search. Three basic genetic operators guide this search: selection, crossover, and mutation. The genetic search process is iterative: evaluating, selecting, and recombining strings in the population during each iteration (generation) until reaching some termination condition. The basic algorithm, where $P(t)$ is the population of strings at generation t, is given below:

```
t = 0
initialize P(t)
evaluate P(t)
while (termination condition is not satisfied) do
begin
    select P(t+1) from P(t)
    recombine P(t+1)
    evaluate P(t+1)
    t = t+1
end
```

Evaluation of each string is based on a fitness function that is problem-dependent. It determines which of the candidate solutions are better. This corresponds to the environmental determination of survivability in natural selection. Selection of a string, which represents a point in the search space, depends on the string's fitness relative to those of other strings in the population. It probabilistically removes, from the population, those points that have relatively low fitness. Mutation, as in natural systems, is a very low probability operator and just flips a specific bit. Mutation plays the role of restoring lost genetic material. Crossover in contrast is applied with high probability. It is a randomized yet structured operator that allows information exchange between points. Its goal is to preserve the fittest individuals without introducing any new value.

In summary, selection probabilistically filters out solutions that perform poorly, choosing high performance solutions to concentrate on or exploit. Crossover and mutation, through string operations, generate new solutions for exploration. Given an initial population of elements, GAs use the feedback from the evaluation process to select fitter solutions, eventually converging to a population of high performance solutions. GAs do not guarantee a global optimum solution.

However, they have the ability to search through very large search spaces and come to nearly optimal solutions fast. Their ability for fast convergence is explained by the *schema theorem* (i.e., short-length bit patterns in the chromosomes with above average fitness, get exponentially growing number of trials in subsequent generations [35]).

B. Feature Selection Encoding

We have employed a simple encoding scheme where the chromosome is a bit string whose length is determined by the number of eigenvectors. Each eigenvector, computed using PCA, is associated with one bit in the string. If the i^{th} bit is 1, then the i^{th} eigenvector is selected, otherwise, that component is ignored. Each chromosome thus represents a different subset of eigenvectors.

C. Feature Subset Fitness Evaluation

The goal of feature subset selection is to use less features to achieve the same or better performance. Therefore, the fitness evaluation contains two terms: (1) accuracy and (2) the number of features selected. The performance of the SVM is estimated using a validation data set (see Sections VII-A and VIII-A) which guides the GA search. Each feature subset contains a certain number of eigenvectors. If two subsets achieve the same performance, while containing different number of eigenvectors, the subset with fewer eigenvectors is preferred. Between accuracy and feature subset size, accuracy is our major concern. We used the fitness function shown below to combine the two terms:

$$fitness = 10^4 Accuracy + 0.5 Zeros \tag{5}$$

where *Accuracy* corresponds to the classification accuracy on a validation set for a particular subset of eigenvectors, and *Zeros* corresponds to the number eigenvectors not selected (i.e., zeros in the chromosome). The *Accuracy* term ranges roughly from 0.50 to 0.99, thus, the first term assumes values from 5000 to 9900. The *Zeros* term ranges from 0 to $L - 1$ where L is the length of the chromosome, thus, the second term assumes values from 0 to 99 ($L = 200$). Based on the weights that we have assigned to each term, the *Accuracy* term dominates the fitness value. This implies that individuals with higher accuracy will outweigh individuals with lower accuracy, no matter how many features they contain. Overall, the higher the accuracy is, the higher the fitness is. Also, the fewer the number of features is, the higher the fitness is.

Choosing the weights for the two terms of the fitness function is more objective-dependent than application-dependent. When we build a pattern classification system, among many factors, we need to find the best balance between model compactness and performance accuracy. Under some scenarios, we prefer the best performance, no matter what the cost might be. If this is the case, the weight associated with the *Accuracy* term should be very high. Under different situations, we might favor more compact models over accuracy, as long as the accuracy is within a satisfactory range. In this case, we should choose a higher weight for the *Zeros* term.

D. Initial Population

In general, the initial population is generated randomly, (e.g., each bit in an individual is set by flipping a coin). This, however, would produce a population where each individual contains approximately the same number of 1's and 0's on the average. To explore subsets of different numbers of features, the number of 1's for each individual is generated randomly. Then, the 1's are randomly scattered in the chromosome. In all of our experiments, we used a population size of 2000 and 200 generations. In most cases, the GA converged in less than 200 generations.

E. Selection

Our selection strategy was cross generational. Assuming a population of size N , the offspring double the size of the population and we select the best N individuals from the combined parent-offspring population [50].

F. Crossover

There are three basic types of crossovers: one-point crossover, two-point crossover, and uniform crossover. For one-point crossover, the parent chromosomes are split at a common point chosen randomly and the resulting sub-chromosomes are swapped. For two-point crossover, the chromosomes are thought of as rings with the first and last gene connected (i.e., wrap-around structure). In this case, the rings are split at two common points chosen randomly and the resulting sub-rings are swapped. Uniform crossover is different from the above two schemes. In this case, each gene of the offspring is selected randomly from the corresponding genes of the parents. Since we do not know in general how eigenvectors depend on each other, if dependent eigenvectors are far apart in the chromosome, it is very likely that traditional one-point or two-point crossover will destroy the schemata. To avoid this problem, uniform crossover is used here. The crossover probability used in all of our experiments was 0.66.

G. Mutation

We use the traditional mutation operator which just flips a specific bit with a very low probability. The mutation probability used in all of our experiments was 0.04.

VI. Support Vector Machines

SVMs are primarily two-class classifiers that have been shown to be an attractive and more systematic approach to learn linear or non-linear decision boundaries [51] [52]. Their key characteristic is their mathematical tractability and geometric interpretation. This has facilitated a rapid growth of interest in SVMs over the last few years, demonstrating remarkable success in fields as diverse as text categorization, bioinformatics, and computer vision [53]. Specific applications include text classification [54], speed recognition [55], gene classification [56], and webpage classification [57].

Given a set of points, which belong to either of two classes, *SVM* finds the hyperplane leaving the largest possible fraction of points of the same class on the same side, while maximizing the distance of either class from the hyperplane. This is equivalent to performing structural risk minimization to achieve good generalization [51] [52]. Assuming there are l examples from two classes

$$(x_1, y_1)(x_2, y_2)...(x_l, y_l), \quad x_i \in R^N, y_i \in \{-1, +1\} \tag{6}$$

Finding the optimal hyper-plane implies solving a constrained optimization problem using quadratic programming. The optimization criterion is the width of the margin between the classes. The discriminate hyperplane is defined as:

$$f(x) = \sum_{i=1}^{l} y_i a_i k(x, x_i) + b \tag{7}$$

where $k(x, x_i)$ is a kernel function and the sign of $f(x)$ indicates the membership of x. Constructing the optimal hyperplane is equivalent to find all the nonzero a_i. Any data point x_i corresponding to a nonzero a_i is a support vector of the optimal hyperplane.

Suitable kernel functions can be expressed as a dot product in some space and satisfy the Mercer's condition [51]. By using different kernels, *SVMs* implement a variety of learning machines (e.g., a sigmoidal kernel corresponding to a two-layer sigmoidal neural network while a Gaussian kernel corresponding to a radial basis function (*RBF*) neural network). The Gaussian radial basis kernel is given by

$$k(x, x_i) = \exp(-\frac{\| x - x_i \|^2}{2\delta^2}) \tag{8}$$

The Gaussian kernel is used in this study. Our experiments have shown that the Gaussian kernel outperforms other kernels in the context of our applications.

VII. Vehicle Detection

Robust and reliable vehicle detection in images acquired by a moving vehicle (i.e., on-road vehicle detection) is an important problem with applications to driver assistance systems or autonomous, self-guided vehicles. This is a very challenging task in general. Vehicles, for example, come into view with different speeds and may vary in shape, size, and color. Also, vehicle appearance depends on its pose and is affected by nearby objects. Within-class variability, occlusion, and lighting conditions also change the overall appearance of vehicles. Landscape along the road changes continuously while the lighting conditions depend on the time of the day and the weather.

Research on vehicle detection has been quite active within the last ten years. Matthews et al. [58] used PCA for feature extraction and NNs for classification. Goerick et al. [59] employed Local Orientation Coding (LOC) to encode edge information and NNs to learn the characteristics of vehicles. A statistical model was investigated in [16] where PCA and wavelet features were used to represent vehicle and non-vehicle appearance. A different statistical model was investigated by Weber et al. [60]. They represented each vehicle image as a constellation of local features and used the Expectation Maximization(EM) algorithm [24] to learn the parameters of the probability distribution of the constellations. An interest operator, followed by clustering, is used to identify a small number of local features in vehicle images. In [61], Papageorgiou et al. proposed using Haar wavelets for feature extraction and SVMs for classification. Sun et al. [62] fused Gabor and Haar wavelet features to improve detection accuracy.

Here, we consider the problem of rear-view vehicle detection from gray-scale images. The first step of any vehicle detection system is to hypothesize the locations of vehicles in an image. Then, verification is performed to test the hypotheses. Both steps are equally important and challenging. Approaches to generate the hypothetical locations of vehicles in an images use motion information, symmetry, shadows, and vertical/horizontal edges. Our emphasis here is on improving the performance of the verification step by selecting a representative feature subset.

A. Vehicle Dataset

The images used in our experiments were collected in Dearborn, Michigan during two different sessions, one in the Summer of 2001 and one in the Fall of 2001. To ensure a good variety of data in each session, the images were caught during different times, different days, and on five different highways. The training set contains subimages of rear vehicle views and non-vehicles which were extracted manually from the Fall 2001 data set. A total of 1051 vehicle subimages and 1051 non-vehicle subimages were extracted (see Fig. 4). In [61], the subimages were aligned by wrapping the bumpers to approximately the same position. We have not attempted to align the data in our case since alignment requires detecting certain features on the vehicle accurately. Moreover, we believe that some variability in the extraction of the subimages can actually improve performance. Each subimage in the training and test sets was scaled to 32×32 and preprocessed to account for different lighting conditions and contrast followed the method suggested in [48].

To evaluate the performance of the proposed approach, the average error (ER) was recorded using a three-fold cross-validation procedure. Specifically, we split the training dataset randomly three times by keeping 80% of the vehicle subimages and 80% of the non-vehicle subimages (i.e., 841 vehicle subimages and 841 non-vehicle subimages) for training. The rest 20% of the data was used for validation during feature selection. For testing, we used a fixed set of 231 vehicle and non-vehicle subimages which were extracted from the Summer 2001 data set.

B. *Experimental Results*

We have performed a number of experiments and comparisons to demonstrate the importance of feature selection for vehicle detection. First, SVMs were tested using some percentage of the top eigenvectors. We ran several experiments by varying the number of eigenvector from 50 to 200. Using the top 50, 100, 150, and 200 eigenvectors, the average error rates obtained were 18.21%, 10.89%, 10.24%, and 10.80% respectively. Next, we used GAs to select an optimum subset of eigenvectors. For comparison purposes, we also implemented the SFBS feature selection method discussed in Section II. Fig. 5(a) shows the error rates for all the approaches tested here. Using eigenvector selection, the SVM achieved a 6.49% average error rate in the case of GAs, and a 9.07% average error rate in the case of SFBS. In terms of number of eigenvectors contained in the final solution, SFBS kept 87 features, which is 43.5% of the complete feature set, while GAs kept 46 features, which is 23% of the complete feature set.

VIII. Face Detection

Face detection from a single image is a difficult task due to the variability in scale, location, orientation, pose, race, facial expression, and occlusion. Rowley [18] proposed an NN-based face detection method, where pre-processed image intensity values were used to train a multilayer NN to learn the face and nonface patterns from face and nonface examples. Sung et al. [63] developed a system composed of two parts, (i) a distribution-based model for face/nonface representations and (ii) a multilayer NN for classification. SVMs have been applied to face detection by Osuna et al.[19]. In that work, the inputs to the SVM were pre-processed image intensity values such as those used in [18]. SVMs have also been used with wavelet features for face detection in [61]. Recently, Viola et al. [5] developed a face detection system using wavelet-like features and the AdaBoost learning algorithm which combines increasingly more complex classifiers are combined in a cascade. The boosting process they used selects a weak classifier at each stage of the cascade which can been seen as a feature selection process. Two recent comprehensive surveys on face detection can be found in [25][26].

To detect faces in an image, a fixed window is usually run across the input image. Each time, the contents of the window are given to a classifier which verifies whether there is a face in the window or not. To account for differences in face size, the input image is represented at different scales and the same procedure is repeated at each scale. Alternatively, candidate face locations in an image can be found using color, texture, or motion information. Here, we concentrate on the verification step only.

A. Face Dataset

Our training set contains 616 faces and 616 non-faces subimages which were extracted manually from a gender dataset [6] and the CMU face detection dataset [18]. Several examples are shown in Fig. 6. For testing, we used a fixed set of 268 face and non-face subimages which were also extracted from disjoint set of images from the CMU face detection data set. Each subimage in the training and test sets was scaled to 32×32 and preprocessed to account for different lighting conditions and contrast [48].

To evaluate the performance of the proposed approach, we used a three-fold cross-validation procedure, splitting the training dataset randomly three times by keeping 84% of the face subimages and 84% of the non-face subimages (i.e., 516 vehicle subimages and 516 non-face subimages) for training. The rest 16% of the data was used for validation during feature selection.

B. Experimental Results

First, we tested SVMs using a percentage of the top eigenvectors as in the case of vehicle detection. We ran several experiments by varying the number of eigenvectors from 50 to 200. Using the top 50, 100, 150, and 200 eigenvectors, the average error rates obtained were 12.31%, 11.57%, 13.81%, and 14.93% respectively. Next, we used GAs to select an optimum subset of eigenvectors. As in the case of vehicle detection, we compared the results of the GA approach with the SFBS approach. Fig. 5(b) shows the average error rates for all the approaches tested here. Using eigenvector selection, the SVM achieved a 8.21% average error rate in the case of GAs, and a 10.45% average error rate in the case of SFBS. In terms of number of eigenvectors contained in the final solution, SFBS kept 68 features, which is 34% of the complete feature set, while GAs kept 34 features, which is 17% of the complete feature set.

IX. Discussion

To get an idea about the optimal set of eigenvectors selected by GAs (or SFBS) in the context of vehicle/face detection, we computed histograms (see Fig. 7), showing the average distributions of the selected eigenvectors over the three training sets. The x-axis corresponds to the eigenvectors, ordered by their eigenvalues, and has been divided into bins of size 10. The y-axis corresponds to the average number of times an eigenvector within some bin was selected by the GA (or SFBS) approach in the final solution. For example, Fig. 7(a) shows the average distribution of the eigenvectors selected by GAs for vehicle detection. For example, the first bar of each histogram indicates that, on average, 5.7 eigenvectors were selected from the top 10 eigenvectors.

Fig. 7 illustrates that the eigenvector subsets selected by GA approach were different from those selected by the

SFBS approach. As we have discussed in Section II, different eigenvectors seems to encode different kind of information. For visualization purposes, we have reconstructed several vehicle (Fig. 8) and face (Fig. 9) images using the selected eigenvectors only. For comparison purpose, we also reconstructed the same images using the top 50 eigenvectors. Several interesting comments can be made by observing these reconstructions, the experimental results presented in Sections VII and VIII, and the eigenvector distributions shown in Fig. 7:

(1) *The eigenvector subsets selected by the GA approach improve detection performance, both for vehicle and face detection:* Feature subsets selected by GAs yielded an average error rate of 6.49% for vehicle detection, better that the 9.07% obtained using SFBS or 10.24% using a percentage of the top eigenvectors. In the context of face detection, the average error rate using GAs was 8.21%, which is better than the average error rate using a percentage of the top eigenvectors (i.e., 11.57%) or eigenvectors selected by SBFS (i.e., 10.45%).

(2) *The GA solutions found are quite compact:* The final eigenvector subsets found by GAs are very compact - 46 eigenvectors out of 200 for vehicle detection, and 34 eigenvectors out of 200 for face detection. The significant reduction in the number of eigenvectors kept speeds up classification substantially.

(3) *The eigenvectors selected by the GA approach do not encode fine details:* The images shown in the fourth row of Fig. 8 correspond to the reconstructed vehicle images using only the eigenvectors selected by GAs. It is interesting to note that they all look quite similar to each other. As we discussed before, only some general information about vehicles is desirable for vehicle detection. These features can be thought as features representing the "conceptual vehicle", but not individual vehicles. In contrast, the reconstructed images using the top 50 eigenvectors or eigenvector subsets selected by the SFBS approach reveal more vehicle identity information (i.e., more details) as can be seen from the images in the second and third rows. Similar observations can be made by observing the reconstructed face images shown in Fig. 9. The reconstructed faces shown in the last row (i.e., using eigenvectors selected by the GA approach) look more blurry (i.e., have less details) than the original images or the ones reconstructed using the top eigenvectors or those selected by the SFBS approach. Identity information has not been preserved which might be the key to successful face detection.

(4) *Eigenvectors encoding irrelevant or redundant information have not been favored by the GA approach:* This is obvious by observing the reconstructed images in the fourth row of Fig. 8. All of them seem to be normalized with respect to illumination. Of particular interest is the image shown in the fourth column which is much lighter than the rest. It appears that eigenvectors encoding illumination information have not be included in the final eigenvector subset. This result is very reasonable since illumination is not critical for vehicle detection, if not confusing. We can also notice that the reconstructed vehicle images are better framed compared to the original ones, therefore, some kind of implicit normalization has been accomplished with respect to location and size. For face detection, we can observe similar results. Fine details have been removed from the reconstructed face images as shown in Fig. 9. Moreover, we can observe normalization effects with respect to size, location, and orientation. Of particular interest is the face image shown in the fifth column of Fig. 9 which is rotated and illuminated from the right side. These effects have been removed from the reconstructed image shown in the last row. This implies that eigenvectors encoding lighting and rotation have not been included in the final solution.

X. Conclusions

We have investigated a systematic feature subset selection framework using GAs. Specifically, the complete feature set is encoded in a chromosome and then optimized by GAs with respect both to detection accuracy and number of discarded features. To evaluate the proposed framework, we considered two challenging object detection problems: vehicle detection and face detection. In both cases, we used PCA for feature extraction and SVMs for classification. Our experimental results illustrate that the proposed method improves the performance of vehicle and face detection, both in terms of accuracy and complexity (i.e., number of features). Further analysis of our results indicates that the proposed method is capable of removing redundant and irrelevant features, outperforming traditional approaches.

For future work, we plan to generalize the encoding scheme to allow eigenvector fusion (i.e., using real weights) instead of pure selection (i.e., using 0/1 weights). We also plan to investigate qualitatively different types of encodings, for example, linkage learning, inversion operators, and messy encodings [35][64][65], as well as hybrid feature selection schemes to find better solutions faster. Filter-based approaches, for example, are much faster in finding a subset of features. One idea is to run a filter-based approach first and then use the results to initialize the GA or even "inject" some of those solutions to the GA population in certain generations to improve exploration [66]. For fitness evaluation, there are many more options. Since the main goal is to to use fewer features while achieving same or better accuracy, a fitness function containing the two terms used here seems to be appropriate. However, more powerful fitness functions can be formed by including more terms such as information measures (e.g., entropy) or dependence measures (e.g., mutual information, minimum description length).

Acknowledgements: This research was supported by Ford Motor Company under grant No.2001332R, the University of Nevada, Reno under an Applied Research Initiative (ARI) grant, and in part by NSF under CRCD grant No.0088086.

References

[1] S. Watanabe, *Pattern Recognition: Human and Mechanical.* Wiley NY, 1985.

[2] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, 1997.

[3] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, pp. 245–271, 1997.

[4] Z. Sun, G. Bebis, and R. Miller, "Boosting object detection using feature selection," *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 290–296, July 2003.

[5] P. Viola and M. Jones, "Rapid object detection using a boosted cascacd of simple features," *Proc. Computer Vision and Pattern Recogntion*, 2001.

[6] Z. Sun, X. Yuan, G. Bebis, and S. Louis, "Neural-network-based gender classification using genetic eigen-feature extraction,," *IEEE International Joint Conference on Neural Networks*, May 2002.

[7] Z. Sun, G. Bebis, X. Yuan, and S. Louis, "Genetic feature subset selection for gender classification: A comparison study," *IEEE Workshop on Applications of Computer Vision*, December,2002.

[8] Z. Sun, G. Bebis, and R. Miller, "Evolutionary gabor filter optimization with application to vehicle detection," *IEEE International Conference on Data Mining*, pp. 307–314, November 2003.

[9] A. Gyaourova, G. Bebis, and I. Pavlidis, "Infrared and visible image fusion for face recognition," *European Conference on Computer Vision*, May, 2004.

[10] B. Bhanu and Y. Lin, "Genetic algorithm based feature selection for target detection in sar images," *Image and Vision Computing*, vol. 21, no. 7, pp. 591–608, 2003.

[11] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," *IEEE International Conference on Computer Vision*, 2003.

[12] R. Collins and Y. Liu, "On-line selection of discriminative tracking features," *IEEE International Conference on Computer Vision*, 2003.

[13] N. Haering and N. da Vitoria Lobo, "Features and classification methods to locate deciduous trees in images," *Computer Vision and Image Understanding*, vol. 75, no. 1/2, pp. 133–149, 1999.

[14] Y. Liu and J. Kender, "Video frame categorization using sort-merge feature selection," *IEEE Workshop on Applications in Computer Vision*, pp. 72–77, 2002.

[15] A. Jain, R. Duin and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[16] H. Schneiderman and T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition," *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 45–51, 1998.

[17] A. Nefian and M. Hayes III, "Face recognition using an embedded hhm," *IEEE Conference on Audio and Video-based Biometric Person Authentication*, pp. 19–24, 1999.

[18] H. Rowley, S. Baluja, and T. Kanade, “Neural network-based face fetection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 22–38, 1998.

[19] E. Osuna, R. Freund, and F. Girosi, “Training support vector machines: An application to face detection,” *Proc. of Computer Vision and Pattern Recognition*, 1997.

[20] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 3, pp. 71–86, 1991.

[21] K. Etemad and R. Chellappa, “Discriminant analysis for recognition of human face images,” *Journal of the Optical Society of America*, vol. 14, pp. 1724–1733, 1997.

[22] M. Bartlett and T. Sejnowski, “Independent components of face images: a representation for face recognition,” *4th Annual Joint Symposium on Neural Computation*, 1997.

[23] K. Baek and B. Draper, “Factor analysis for background suppression,” *International Conference on Pattern Recognition*, 2002.

[24] R. Duda, P. Hart, and D. Stork, *Pattern Classification.* Jon-Wiley, 2001.

[25] M. Yang, D. Kriegman and N. Ahuja, “Detection faces in images:a survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[26] E. Hjelmas and B. Low, “Face detection:a survey,” *Computer Vision and Image Understanding*, vol. 83, pp. 236–274, 2001.

[27] W.Siedlecki and J.Sklansky, “On automatic feature selection,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 2, no. 2, pp. 197–220, 1988.

[28] A.Jain D. Zongker, “Feature selection: Evaluation, application, and small sample performance,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 153–158, 1997.

[29] J. Yang and V. Honavar, “Feature subset selection using a genetic algorithm,” *A Data Mining Perspective (H. Motoda and H. Liu eds) Chapt 8, Kluwer*, 1998.

[30] T.Marill and D.Green, “On the effectiveness of receptors in recognition systems,” *IEEE Transactions on Information Theory*, vol. 9, pp. 11–17, 1963.

[31] S. Stearns, “On selecting features for pattern classifiers,” *The Third International Conference of Pattern Recognition*, pp. 71–75, 1976.

[32] P.Pudil, J. Novovicova and J. Kittler, “Floating search methods in feature selection,” *Pattern Recognition Letter*, vol. 15, pp. 1119–1125, 1994.

[33] K. Kira and L. Rendell, “A practical approach to feature selection,” *The Ninth International Conference on Machine Learning*, pp. 249–256, 1992.

[34] L. Wiskott, J. Fellous, N. Kruger and C. Malsburg, “Estimating attributes: Analysis and extension of relief,” *European Conference on Machine Learning*, pp. 171–182, 1994.

[35] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, 1989.

[36] W. Siedlecki and J. Sklansky, “A note on genetic algorithm for large-scale feature selection,” *Pattern recognition letter*, vol. 10, pp. 335–347, 1989.

[37] Y. Chtioui, D. Bertrand and D. Barba, “Feature selection by a genetic algorithm. application to seed discrimination by artificial vision,” *J. Sci. Food Agric.*, vol. 76, pp. 77–86, 1998.

[38] H. Vafaie and I. Imam, “Feature selection methods: Genetic algorithms vs. greedy-like search,” *International Conference on Fuzzy and Intelligent Control Systems*, 1994.

[39] A. O’Toole, H. Adbi, K. Deffenbacher and D. Valentin, “A low-dimensional representation of faces in the higher dimensions of space,” *Journal of the optical society of America*, vol. 10, pp. 405–411, 1993.

[40] H. Abdi, D. Valentin, B. Edelman and A. O'Toole, "More about the difference between men and women: evidence from linear neural networks and the principal component approach," *Perception*, vol. 24, pp. 539–562, 1995.

[41] D. Valentin and H. Abdi, "Can a linear autoassociator recognize faces from new orientation?," *Journal of the Optical Society of America*, vol. A, 13, pp. 717–724, 1996.

[42] W. Yambor, B. Draper, and R. Beveridge, "Analyzing pca-based face recognition algorithms: Eigenvector selection and distance measures," *2nd Workshop on Empirical Evaluation in Computer Vision*, 2000.

[43] K. Balci and V. Atalay, "Pca for gender estimation: Which eigenvectors contribute?," *International Conference on Pattern Recognition*, p. 11.

[44] A. Katz and P. Thrift, "Generating image filters for target recognition by genetic learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 906–910, 1994.

[45] G. Bebis, S. Louis, Y. Varol, and A. Yfantis, "Genetic object recognition using combinations of views," *IEEE Transactions on Evolutionary Computation*, vol. 6 no. 2, pp. 132–146, 2002.

[46] D. Swets, B. Punch, and J. Weng, "Genetic algorithms for object recognition in a complex scene," *IEEE International Conference on Image Processing*, pp. 595–598, 1995.

[47] C. Liu and H. Wechsler, "Evolutionary pursuit and its application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no.6, pp. 570–582, 2000.

[48] G. Bebis, S. Uthiram, and M. Georgiopoulos, "Face detection and verification using genetic search," *International Journal on Artificial Intelligence Tools*, vol. 9, no. 2, pp. 225–246, 2000.

[49] M. Srinivas, L. Patnaik, "Genetic algorithms: a survey," *IEEE Computers*, vol. 27, no. 6, pp. 17–26, 1994.

[50] L. Eshelman, "The chc adaptive search algoruthm: How to have safe search when engaging in non-traditional genetic recombination," *The Foundation of Genetic Algorithms Workshop*, pp. 265–283, 1989.

[51] V. Vapnik, *The Nature of Statistical Learning Theory.* Springer Verlag, 1995.

[52] C. Burges, "Tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 955–974, 1998.

[53] N. Cristianini, C. Campbell, and C. Burges, "Kernel methods: Current research and future directions," *Machine Learning*, vol. 46, pp. 5–9, 2002.

[54] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[55] N. Smith and M. Gales, "Speech recognition using svms," *Advances in Neural Information Processing Systems 14* (T.G. Dietterich, S. Becker, and Z. Ghahramani, eds.), MIT Press, 2002.

[56] M. Brown, W. Grundy, D. Lin, N. Christianini, C. Sugnet, M. Ares Jr., and D. Haussler, "Support vector machine classification of microarray gene expression data," Tech. Rep. UCSC-CRL 99-09, Department of Computer Science, University California Santa Cruz,CA, 1999.

[57] H. Yu, J. Han, and K. C. Chang, "Pebl: Positive-example based learning for web page classification using svm," *8th Int. Conf. Knowledge Discovery and Data Mining*, 2002.

[58] N. Matthews, P. An, D. Charnley, and C. Harris, "Vehicle detection and recognition in greyscale imagery," *Control Engineering Practice*, vol. 4, pp. 473–479, 1996.

[59] C. Goerick, N. Detlev and M. Werner, "Artificial neural networks in real-time car detection and tracking applications," *Pattern Recognition Letters*, vol. 17, pp. 335–343, 1996.

[60] M. Weber, M. Welling, and P. Perona, “Unsupervised learning of models for recognition,” *European Conference on Computer Vision*, pp. 18–32, 2000.

[61] C. Papageorgiou and T. Poggio, “A trainable system for object detection,” *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[62] Z. Sun, G. Bebis, and R. Miller, “Improving the performance of on-road vehicle detection by combining gabor and wavelet features,” *The IEEE Fifth International Conference on Intelligent Transportation Systems* , September, 2002, Singapore.

[63] K. Sung and T. Poggio, “Example-base learning for view-based human face detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39–51, 1998.

[64] D. Goldberg, B. Korb, and K. Deb, “Messy genetic algorithms: Motivation, analysis, and first results,” *Technical Report TCGA Report No. 89002, The Clearinghouse for Genetic Algorithms, University of Alabama, Tuscaloosa.*

[65] H. Kargupta, “Search, polynomial complexity, and the fast messy genetic algorithm,” *PhD Thesis, University of Illinois, CS Dept.*

[66] Sushil J. Louis and Gan Li, “Combining robot control strategies using genetic algorithms with memory,” *Lecture Notes in Computer Science, Evolutionary Programming VI*, vol. 1213, pp. 431–442, 1997.

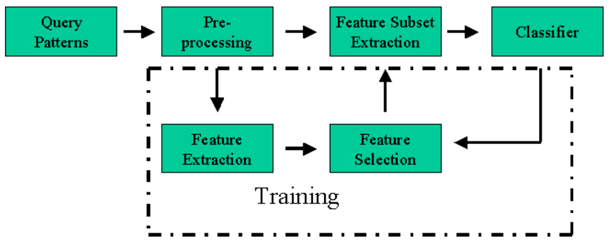


Fig. 1. Main steps involved in building an object detection system using feature subset selection.

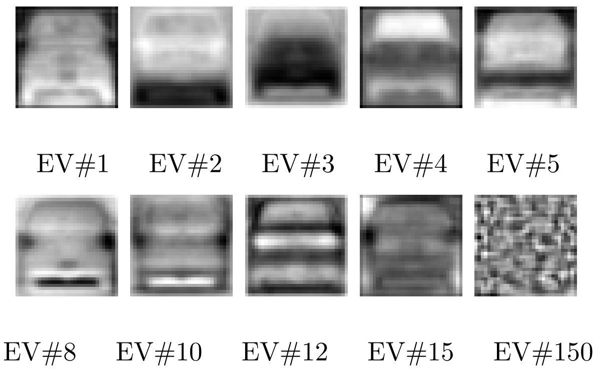


Fig. 2. Eigenvectors corresponding to the vehicle detection dataset

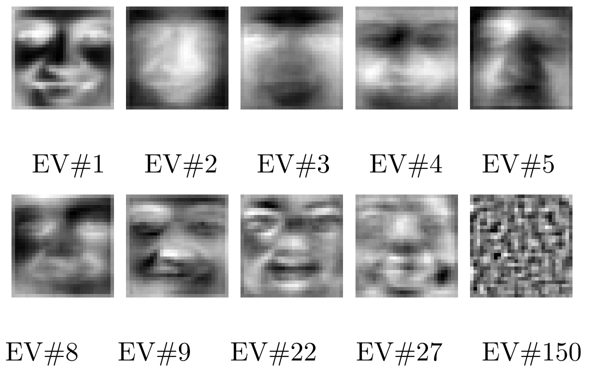


Fig. 3. Eigenvectors corresponding to the face detection dataset

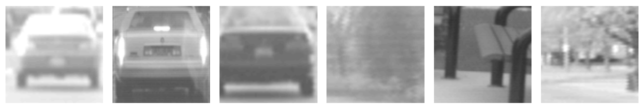
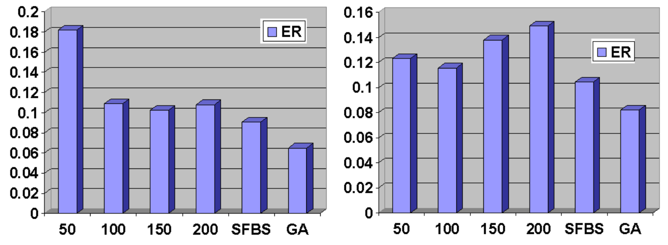


Fig. 4. Examples of vehicle and non-vehicle images used for training.



(a) (b)

Fig. 5. Detection error rates of various methods: (a) vehicle detection results, (b) face detection results.

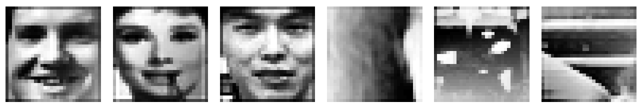
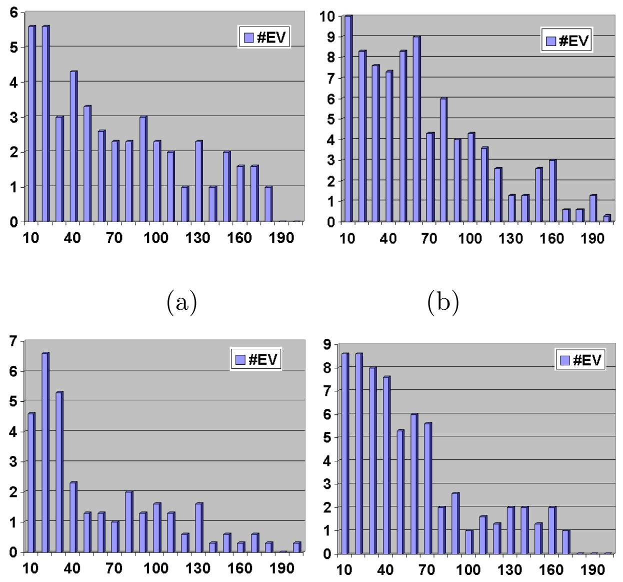


Fig. 6. Examples of face and non-face images used for training.



(a) (b)

(c) (d)

Fig. 7. The distributions of eigenvectors selected by (a) GAs for vehicle detection , (b) SFBS for vehicle detection, (c) GAs for face detection, (d) SFBS for face detection.

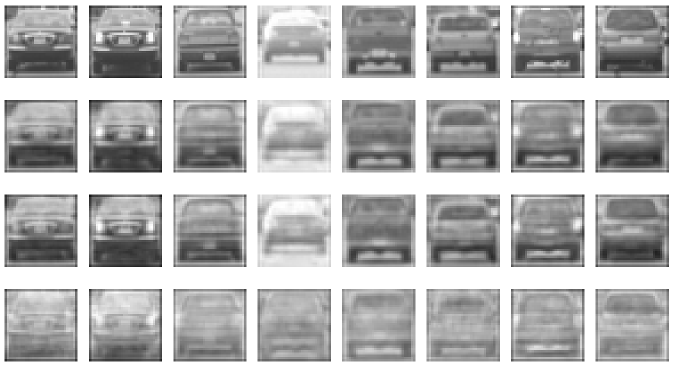


Fig. 8. Reconstructed images using the selected eigenvectors. (first row): original images; (second row): using the top 50 eigenvectors; (third row): using the eigenvectors selected by SFBS; (fourth row): using the eigenvectors selected by GAs.

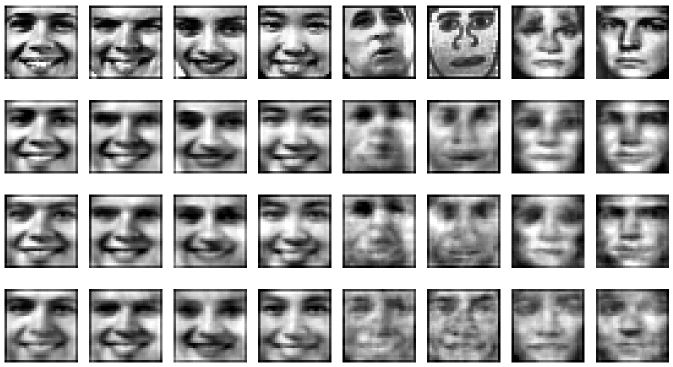


Fig. 9. Reconstructed images using the selected eigenvectors. (first row): original images; (second row): using the top 50 eigenvectors; (third row): using the eigenvectors selected by SFBS; (fourth row): using the eigenvectors selected by GAs.