Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 736508, 13 pages
doi:10.1155/2008/736508

Research Article

Cascaded Face Detection Using Neural Network Ensembles

Fei Zuo[1] and Peter H. N. de With[2,3]

[1] *Philips Research Labs, High Tech Campus 34, 5656 AE Eindhoven, The Netherlands*
[2] *Department of Electrical Engineering, Signal Processing Systems (SPS) Group, Eindhoven University of Technology, 5612 AZ Eindhoven, Den Dolech2, The Netherlands*
[3] *LogicaCMG, 5605 JB Eindhoven, The Netherlands*

Correspondence should be addressed to Fei Zuo, fei.zuo@philips.com

Received 6 March 2007; Revised 16 August 2007; Accepted 8 October 2007

Recommended by Wilfried Philips

We propose a fast face detector using an efficient architecture based on a hierarchical cascade of neural network ensembles with which we achieve enhanced detection accuracy and efficiency. First, we propose a way to form a neural network ensemble by using a number of neural network classifiers, each of which is specialized in a subregion in the face-pattern space. These classifiers complement each other and, together, perform the detection task. Experimental results show that the proposed neural-network ensembles significantly improve the detection accuracy as compared to traditional neural-network-based techniques. Second, in order to reduce the total computation cost for the face detection, we organize the neural network ensembles in a pruning cascade. In this way, simpler and more efficient ensembles used at earlier stages in the cascade are able to reject a majority of nonface patterns in the image backgrounds, thereby significantly improving the overall detection efficiency while maintaining the detection accuracy. An important advantage of the new architecture is that it has a homogeneous structure so that it is suitable for very efficient implementation using programmable devices. Our proposed approach achieves one of the best detection accuracies in literature with significantly reduced training and detection cost.

1. INTRODUCTION

Face detection from images (videos) is a crucial preprocessing step for a number of applications, such as face identification, facial expression analysis, and face coding [1]. Furthermore, research results in face detection can broadly facilitate general object detection in visual scenes.

A key question in face detection is how to best discriminate faces from nonface background images. However, for realistic situations, it is very difficult to define a discriminating metric because human faces usually vary strongly in their appearance due to ethnic diversity, expressions, poses, and aging, which makes the characterization of the human face difficult. Furthermore, environmental factors such as imaging devices and illumination can also exert significant influences on facial appearances.

In the past decade, extensive research has been carried out on face detection, and significant progress has been achieved to improve the detection performance with the following two performance goals.

(1) *Detection accuracy*: the accuracy of a face detector is usually characterized by its receiver operating characteristic (ROC), showing its performance as a trade-off between the false acceptance rate and the face detection rate.
(2) *Detection efficiency*: the efficiency of a face detector is often characterized by its operation speed. An efficient detector is especially important for real-time applications (e.g., consumer applications), where the face detector is required to process one image at a subsecond level.

Tremendous effort has been spent to achieve the above-mentioned goals in face-detector design. Various techniques have been proposed, ranging from simple heuristics-based algorithms to more advanced algorithms based on machine learning [2]. Heuristics-based face detectors exploit empirical knowledge about face characteristics, for instance, the skin color [3] and edges around facial features [4]. Generally speaking, these detectors are simple, easy to implement, and usually do not require much computation cost. However,

it is complicated to translate empirical knowledge into well-defined classification rules. Therefore, these detectors usually have difficulty in dealing with complex image backgrounds and varying illumination, which limits their accuracy.

Alternatively, statistics-based face detectors have received wider interest in recent years. These detectors implicitly distinguish between face and nonface images by using pattern-classification techniques, such as neural networks [5, 6] and support vector machines [7]. The learning-based detectors generally achieve highly accurate and robust detection performance. However, they are usually far more computationally demanding in both training and detection.

To further reduce the computation cost, an emerging interest in literature is to study structured face detectors employing multiple subdetectors. For example, in [8], a set of reduced set vectors are applied sequentially to reject unlikely faces in order to speed up a nonlinear support vector machine classification. In [9], the AdaBoost algorithm is used to select a set of Haar-like feature classifiers to form a single detector. In order to improve the overall detection speed, a set of such detectors with different characteristics are cascaded into a chain. Detectors consisting of smaller numbers of feature classifiers are relatively fast, and they can be used at the first stages in the detector cascade to filter out regions that most likely do not contain any faces. The Viola-Jones face detector in [9] has achieved real-time processing speed with fairly robust detection accuracy. The feature-selection (training) stage, however, can be time consuming in practice. It is reported that several weeks are needed to completely train a cascaded detector. Later, a number of variants of the Viola-Jones detector have also been proposed in literature, such as the detector with extended Haar features [10], the FloatBoost based detector [11], and so forth. In [12], we have proposed a heterogeneous face detector employing three subdetectors using various image features. In [13], hierarchical support vector machines (SVM) are discussed, which use a combination of linear SVMs to efficiently exclude most nonfaces in images, followed by a nonlinear SVM to further verify possible face candidates.

Although the above techniques manage to reduce the computation cost of traditional statistics-based detectors, the detection accuracy of these detectors is also sacrificed. In this paper, we aim to design a face detector with highly accurate performance, which is also computationally efficient for embedded applications.

More specifically, we propose a high-performance face detector built as a cascade of subdetectors, where each subdetector consists of a neural network ensemble [14]. The ensemble technique effectively improves the detection accuracy of a single network, leading to an overall enhanced accuracy. We also cascade a set of different ensembles in such a way that both detection efficiency and accuracy are optimized.

Compared to related techniques in literature, we have the following contributions.

(1) We use an ensemble of neural networks for simultaneously improving accuracy and architectural simplicity. We have proposed a new training paradigm to form an ensemble of neural networks, which are subsequently used as the building blocks of the cascaded detector. The training strategy is very effective as compared to existing techniques and significantly improves the face-detection accuracy.

(2) We also insert this ensemble structure into the cascaded framework with scalable complexity, which yields a significant gain in efficiency with (near) real-time detection speed. Initial ensembles in the cascade adopt base networks that only receive a coarse feature representation. They usually have fewer nodes and connections, leading to simpler decision boundaries. However, since these networks can be executed with very high efficiency, a large portion of an image containing no faces can be quickly pruned. Subsequent ensembles adopt relatively complex base networks, which have the capability of forming more precise decision boundaries. These more complex ensembles are only invoked for difficult cases that fail to be rejected by earlier ensembles in the cascade. We propose a way to optimize the cascade structure such that the computation cost involved can be significantly reduced while retaining overall high detection accuracy.

(3) The proposal in this paper consists of a two-layer classifier architecture including parallel ensembles and sequential cascade based on repetitive use of similar structures. The result is a rather homogeneous architecture, which facilitates an efficient implementation using programmable hardware.

Our proposed approach achieves one of the best detection accuracies in literature, with 94% detection rate on the well-known CMU+MIT test set and up to 5 frames/second processing speed on live videos.

The remainder of the paper is organized as follows. In Section 2, we first explain the construction of a neural network ensemble, which is used as the basic element in the detector cascade. In Section 3, a cascaded detector is formulated consisting of multiple neural network ensembles. Section 4 analyzes the performance of the approach and Section 5 gives the conclusions.

2. NEURAL NETWORK ENSEMBLE

In this section, we present the basic elements of our proposed architecture, which will be reused later to constitute a complete detector cascade. We first present, in Section 2.1, some basic design principles of our proposed neural network ensemble. The ensemble structure and training paradigms will be presented in Sections 2.2 and 2.3.

2.1. Basic principles

For complex real-world classification problems such as face detection, the usage of a single classifier may not be sufficient to capture the complex decision surfaces between face and nonface patterns. Therefore, it is attractive to exploit multiple algorithms to improve the classification accuracy. In Rowley's

approach [5] for face detection, three networks with different initial weights are trained and the final output is based on the majority voting of these networks. The Viola-Jones detector [9] makes use of the boosting strategy, which sequentially trains a set of classifiers by reweighting the sample importance. During the training of each classifier, those samples misclassified by the current set of classifiers have higher probabilities to be selected. The final output is based on a linearly weighted combination of the outputs from all component classifiers.

For aforementioned reasons, our approach is to start with an ensemble of neural network classifiers. We denote each neural network in the ensemble as a component network, which is randomly initialized with different weights. More important is that we manipulate the training data such that each component network is specialized in a different region of the training data space. Our proposed ensemble has the following new characteristics that are different from existing approaches in literature.

(1) The component neural networks in our proposal are sequentially trained, each of which uses training face samples that are misclassified by its previous networks. Our approach differs from the boosting approach in that the training samples that are already successfully classified by the current network are discarded and not used for the later training. This gives a hard partitioning of the training set, where each component neural network characterizes a specific subregion.
(2) The final output of the ensemble is determined by a decision neural network, which is trained after the component networks are already constructed. This offers a more flexible combination rule than the voting or linear weighting as used in boosting.

The experimental evidence (Section 4.1) shows that our proposed ensemble technique gives quite good performance in face detection, outperforming the traditional ensemble techniques.

2.2. Ensemble architecture

We depict the structure of our proposed neural network ensemble in Figure 1. The ensemble consists of two layers: a set of sequentially trained component networks $\{h_k \mid 1 \le k \le N\}$, and a decision network g. The outputs of the component networks $h_k(\mathbf{x})$ are fed to the decision network to give the final output. The input feature vector $\mathbf{x}$ is a normalized image window of 24×24 pixels.

(1) Component neural network

Each component classifier h_k is a multilayer feedforward neural network, which has inputs receiving certain representations of the input feature vector $\mathbf{x}$ and one output ranging from 0 to 1. The network is trained with a target output of unity indicating a face pattern and zero otherwise. Each network has locally connected neurons, as motivated by [5]. It is pointed out in [5] that, by incorporating heuristics of facial feature structures in designing the local connections of the network, the network gives much better performance (and higher efficiency) than a fully connected network.

We present here four novel base-network structures employed in this paper: FNET-A, FNET-B, FNET-C, and FNET-D (see Figure 2), which are extensions of [5] by incorporating scalable complexity. These networks are used as the basic elements in the final face-detector cascade. The design philosophy for these networks are partially based on heuristic reasoning. The motivation behind the design is illustrated below.

(1) We aim at building a complexity-scalable structure for all these base networks. The networks are constructed with similar structures.
(2) The complexity of the network is controlled by the following structural parameters: the input resolution, the number of hidden layers, and the number of hidden units in each layer.
(3) When observing Figure 2, FNET-B (FNET-D) enhances FNET-A (FNET-C) by incorporating more hidden units which specifically aim at capturing various facial feature structures. Similarly, FNET-C (FNET-D) enhances FNET-A (FNET-B) by using a higher-input resolution and more hidden layers.

In this way, we obtain a set of networks with scalable structures and varying representation properties. In the following, we illustrate each network in more detail.

As shown in Figure 2(a), FNET-A has a relatively simple structure with one hidden layer. The network accepts an 8×8 grid as its inputs, where each input element is an averaged value of a neighboring 3×3 block in the original 24×24 input features. FNET-A has one hidden layer with 2×2 neurons, each of which looks at a locally neighboring 4×4 block from the inputs.

FNET-B (see Figure 2(a)) shares the same type of inputs as FNET-A, but with extended hidden neurons. In addition to the 2×2 hidden neurons, additional 6×1 and 2×3 neurons are used, each of which looks at a 2×8 (or 4×3) block from the inputs. These additional horizontal and vertical stripes are used to capture corresponding facial features such as eyes, mouths, and noses.

The topology of FNET-C is depicted in Figure 2(b), which has two hidden layers with 2×2 and 8×8 hidden neurons, respectively. The FNET-C directly receives the 24×24 input features. In the first hidden layer, each hidden neuron takes inputs from a locally neighboring 3×3 block of the input layer. In the second hidden layer, each hidden neuron unit takes a locally neighboring 4×4 block as an input from the first hidden layer.

FNET-D (see Figure 2(b)) is an enhanced version of both FNET-B and FNET-C, with two hidden layers and additional hidden neurons arranged in horizontal and vertical stripes.

From FNET-A to FNET-D, the complexity of the network is gradually increased by using a finer input representation, adding more layers or adding more hidden units to capture more intricate facial characteristics. Therefore, the networks have an increasing number of connections and consume more computation power.

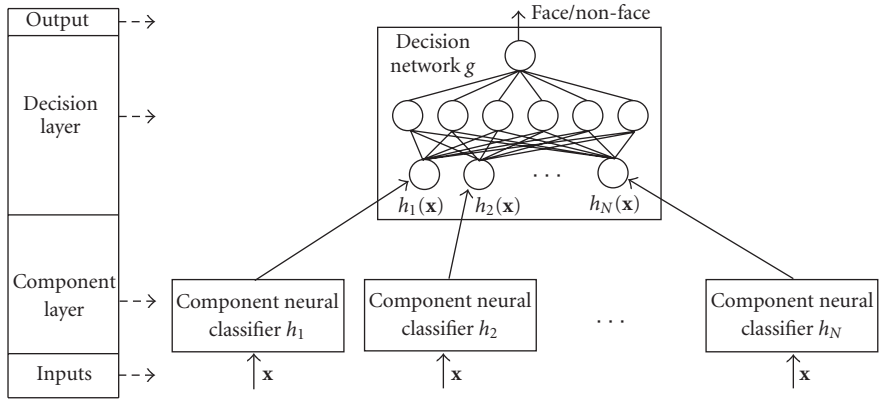
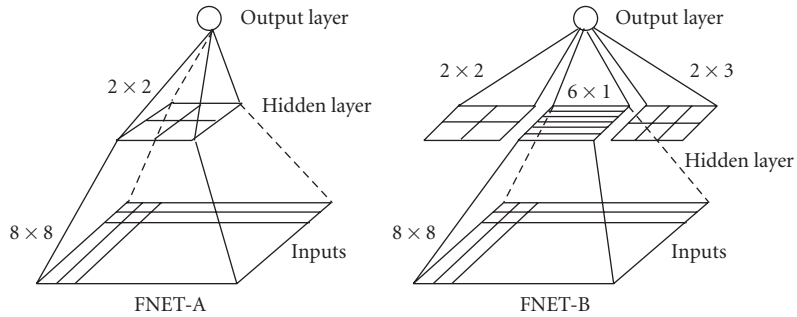
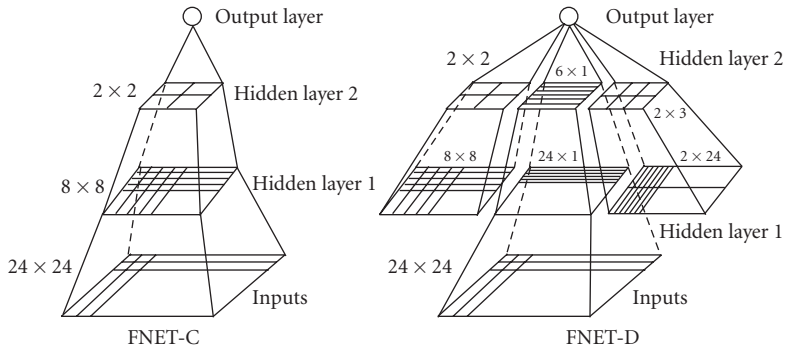


Figure 1: The architecture of the neural network ensemble.



(a) Left: structure of FNET-A; right: structure of FNET-B



(b) Left: structure of FNET-C; right: structure of FNET-D

Figure 2: Topology of four types of component networks.

(2) Decision neural network

For the decision network g (see Figure 1), we adopt a fully connected feedforward neural network, which has one hidden layer with eight hidden units. The number of inputs for g is determined by the number of the component classifiers in the network ensemble. The decision network receives the outputs from each component network h_k, and outputs a value y ranging from 0 to 1, which indicates the confidence that the input vector represents a face. In other words,

$$y = g(h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_N(\mathbf{x})). \quad (1)$$

In the following, we present the training paradigms for our proposed neural network ensemble.

2.3. Training algorithms

Since each ensemble is a two-layer system, the training consists of the following two stages.

(i) Sequentially, train N component classifiers h_k ($1 \leq k \leq N$) with a feature sample $\mathbf{x}$ drawn from a training data set $\mathcal{T}$. $\mathcal{T}$ contains a face sample set $\mathcal{F}$ and a nonface sample set $\mathcal{N}$.

(ii) Train the decision neural network g with samples $\langle h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_N(\mathbf{x})\rangle$, where $\mathbf{x} \in \mathcal{T}$.

Let us now present the training algorithm for each stage in more detail.

(1) Training algorithm for component neural networks

One important characteristic of the component-network training is that each network h_k is trained on a subset $\mathcal{F}_k$ of the complete face set $\mathcal{F}$. $\mathcal{F}_k$ contains only face samples misclassified by the previous $k-1$ trained component classifiers. More specifically, suppose the $(k-1)$th component network is trained over sample set $\mathcal{F}_{k-1}$. After the training, the network is able to correctly classify samples $\mathcal{F}_{k-1}^f$ ($\mathcal{F}_{k-1}^f \subset \mathcal{F}_{k-1}$). The next component network (the kth network) is then trained over sample set $\mathcal{F}_k = \mathcal{F}_{k-1} \setminus \mathcal{F}_{k-1}^f$. This procedure can be iteratively carried out until all N component networks are trained. This is also illustrated in Table 1.

In this way, each component network is trained over a subset of the total training set and is specialized in a specific region in the face space. For each h_k, the nonface samples are selected in a bootstrapping manner, similar to the approach used in [5]. According to the bootstrapping strategy, an initial set of randomly chosen nonface samples is used, and during the training, new false positives are iteratively added to the current nonface training set. In this way, more difficult nonface samples are reinforced during the training process.

Up to now, we have explained the training-set selection strategy for the component networks. The actual training of each network h_k is based on the standard backpropagation algorithm [15]. The network is trained with unity for face samples and zero for nonface samples. During the classification, a threshold T_k needs to be chosen such that the input $\mathbf{x}$ is classified as a face when $h_k(\mathbf{x}) > T_k$. In the following, we will elaborate on how the combination of neural networks (h_1 to h_N) can yield a reduced classification error over the training face set.

First, we define the face-learning ratio α_k of the component network h_k as

$$\alpha_k = \frac{|\mathcal{F}_k^f|}{|\mathcal{F}_k|}, \tag{2}$$

where $|\cdot|$ denotes the number of elements in a set. Furthermore, we define β_k as the fraction of the face samples successfully classified by h_k with respect to the total training face samples, given by

$$\beta_k = \frac{|\mathcal{F}_k^f|}{|\mathcal{F}|}. \tag{3}$$

We can see that

$$\begin{aligned}\beta_k = \frac{|\mathcal{F}_k|}{|\mathcal{F}|}\cdot\alpha_k &= \left(1 - \sum_{i=1}^{k-1}\beta_i\right)\alpha_k, \\ &\left(\text{since } |\mathcal{F}_k| = |\mathcal{F}| - \sum_{i=1}^{k-1}|\mathcal{F}_i^f|\right),\end{aligned} \tag{4}$$

$$\begin{aligned}&= \beta_{k-1}\frac{\alpha_k}{\alpha_{k-1}}(1-\alpha_{k-1}), \\ &\left(\text{since } |\mathcal{F}_k| - |\mathcal{F}_k^f| = |\mathcal{F}_{k+1}|\right).\end{aligned} \tag{5}$$

Table 1: Partitioning of the training set for component networks.

Network	Training set	Correctly classified samples
h_1	$\mathcal{F}_1 = \mathcal{F}$	$\mathcal{F}_1^f (\mathcal{F}_1^f \subset \mathcal{F}_1)$
h_2	$\mathcal{F}_2 = \mathcal{F} \setminus \mathcal{F}_1^f$	$\mathcal{F}_2^f (\mathcal{F}_2^f \subset \mathcal{F}_2)$
...	...	...
h_N	$\mathcal{F}_N = \mathcal{F} \setminus \bigcup_{i=1}^{N-1}\mathcal{F}_i^f$	$\mathcal{F}_N^f (\mathcal{F}_N^f \subset \mathcal{F}_N)$

By recursively applying (5), we derive the following relation between β_k and α_k:

$$\beta_k = \alpha_k \times \prod_{i=1}^{k-1}(1-\alpha_i). \tag{6}$$

The $(k+1)$th component classifier h_{k+1} thus uses a percentage of P_{k+1} of all the training samples, and

$$P_{k+1} = 1 - \sum_{i=1}^{k}\beta_i = 1 - \sum_{i=1}^{k}\left(\alpha_i \times \prod_{j=1}^{i-1}(1-\alpha_j)\right). \tag{7}$$

During the sequential training of the component networks, each network has a decreasing number of available training samples P_k. To ensure that each component network has sufficient samples to learn some generalized facial characteristics, P_k should be larger than a performance critical value (e.g., 5% when $|\mathcal{F}| = 6,000$).

Given a fixed topology of component networks, the value of α_k is inversely proportional to threshold T_k. Hence, the larger T_k, the smaller α_k. Equation (7) provides guidance to the selection of a proper T_k for each component network such that P_k is large enough to provide sufficient statistics.

In Table 2, we give the complete training algorithm for component neural network classifiers.

(2) Training algorithm for the decision neural network

In Table 3, we present the training algorithm for the decision network g. During the training of g, the inputs are taken from $\langle h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_N(\mathbf{x})\rangle$, where $\mathbf{x}$ is drawn from the face set or the nonface set. The training also makes use of the bootstrapping procedure as in the training of the component networks to dynamically add nonface samples to the training set (line (5) in Table 3). In order to prevent the well-known overfitting problem during the backpropagation training, we use here an additional face set $\mathcal{V}_f$ and a nonface set $\mathcal{V}_n$ for validation purposes.

(3) Difference between our proposed technique and bagging/boosting

Let us now briefly compare our proposed approach to two other popular ensemble techniques: bagging and boosting. The bagging selects training samples for each component classifier by sampling the training set with replacements. There is no correlation between the different subsets used for the training of different component classifiers. When applied for neural network face detection, we can train N component

Table 2: The training algorithm for component neural classifiers.

Algorithm *Training algorithm for component neural network*

Input: A training face set $\mathcal{F} = \{\mathbf{x}_i\}$, a number of component neural networks N, a decision threshold T_k, an initial nonface set $\mathcal{N}$, and a set of downloaded scenery images $\mathcal{S}$ containing no faces.

1. Let $k = 1$, $\mathcal{F}_1 = \mathcal{F}$
2. **while** $k \leq N$
3. Let $\mathcal{N}_k = \mathcal{N}$
4. **for** $j = 1$ to Num_Epochs/* Number of training iterations */
5. Train neural classifier h_k^j on face set $\mathcal{F}_k$ and nonface set $\mathcal{N}_k$ using the backpropagation algorithm.
6. Compute the false rejection rate R_f^j and false acceptance rate R_n^j.
7. Feed h_k^j with randomly cropped image windows from $\mathcal{S}$ and collect misclassified samples in set $\mathcal{B}_j$.
8. Update $\mathcal{N}_k \leftarrow \mathcal{N}_k \cup \mathcal{B}_j$.
9. Select j that gives the maximum value of $(1 - R_f^j)/R_n^j$ for $1 \leq j \leq Num\_Epochs$, and let $h_k = h_k^j$.
10. Feed h_k with samples from $\mathcal{F}_k$, and let $\mathcal{F}_k^f = \{\mathbf{x} \mid h_k(\mathbf{x}) > T_k\}$.
11. $\mathcal{F}_{k+1} = \mathcal{F}_k \setminus \mathcal{F}_k^f$
12. $k = k + 1$

Table 3: The training algorithm for the decision network.

Algorithm *Training algorithm for the decision neural network*

Input: Sets $\mathcal{F}$, $\mathcal{N}$, and $\mathcal{S}$ as used in Table 2. A set of N trained component networks h_k, a validation face set $\mathcal{V}_f$, a validation nonface set $\mathcal{V}_n$, and a required face detection rate R_f.

1. Let $\mathcal{N}_t = \mathcal{N}$
2. **for** $j = 1$ to Num_Epochs/* Number of training iterations */
3. Train decision network g_j on face set $\mathcal{F}$ and nonface set $\mathcal{N}_t$ using the backpropagation algorithm.
4. Compute the false rejection rate R_f^j and false acceptance rate R_n^j over the validation set $\mathcal{V}_f$ and $\mathcal{V}_n$, respectively.
5. Feed the current ensemble (h_k, g_j) with randomly cropped image windows from $\mathcal{S}$ and collect misclassified samples in $\mathcal{B}_j$.
6. Update $\mathcal{N}_t \leftarrow \mathcal{N}_t \cup \mathcal{B}_j$.
7. Let $g = g_j$ so that R_n^j is the minimum value for all values of j with $1 \leq j \leq Num\_Epochs$ that satisfy $R_f^j < 1 - R_f$.

neural classifiers independently using randomly selected subsets of the original face training set. The nonface samples are selected in a bootstrapping fashion similar to Table 2. The final output $g_a(\mathbf{x})$ is based on the average of outputs from component classifiers, given by

$$g_a(\mathbf{x}) = \frac{1}{N}\sum_{k=1}^{N} h_k(\mathbf{x}). \tag{8}$$

Different from the bagging, boosting sequentially trains a series of classifiers by emphasizing difficult samples. An example using the AdaBoost was presented in AdaBoost [15]. During the training of the kth component classifier, AdaBoost alters the distribution of the samples such that those samples misclassified by its previous component classifier are emphasized. The final output g_o is a weighted linear combination of the outputs from the component classifiers.

Different from bagging, our proposed ensemble technique sequentially trains a set of interdependent component classifiers. In this sense, it shares the basic principle with boosting. However, the proposed ensemble technique differs from boosting in the following aspects.

(1) Our approach uses a "hard" partitioning of the face training set. Those samples, already correctly classified by the current set of networks, will not be reused for subsequent networks. In this way, face characteristics already learned by the previous networks are not included in the training of subsequent components. Therefore, the subsequent networks can focus more on a different class of face patterns during their corresponding training stages.
As a result of the hard partitioning, the subsequent networks are trained on smaller subsets of the original face training set. We have to ensure that each network has sufficient samples that characterize a subclass of face patterns. This has also been discussed previously.

(2) We use a decision neural network to make the final classification based on individual outputs from component networks. This results in a more flexible decision function than the linear combination rule used by bagging or boosting.

In Section 4, we will give some examples to compare the performance of the resulting neural network ensembles trained with different strategies.

The newly created ensemble of cooperating neural-network classifiers will be used in the following section as "building blocks" in a pruning cascade.

3. CASCADED NEURAL ENSEMBLES FOR FAST DETECTION

In this section, we apply the ensemble technique into a cascading architecture for face detection such that both the detection accuracy and efficiency are jointly optimized.

Figure 3 depicts the structure of the cascaded neural network ensembles for face detection. More efficient ensemble classifiers with simpler base networks are used at earlier stages in the cascade, which are capable of rejecting a majority of nonface patterns, thereby boosting the overall detection efficiency.

In the following, we introduce a notation framework in order to come to expressions for the detection accuracy and efficiency of cascaded ensembles. Afterwards, we propose a technique to jointly optimize the cascaded face detector for both accuracy and efficiency. Following that, we introduce an implementation of a cascaded face detector using five neural-network ensembles.

3.1. Formulation and optimization of cascaded ensembles

As shown in Figure 3, we assume a total of L neural network ensembles g_i $(1 \le i \le L)$ with increasing base network complexity. The behavior of each ensemble classifier g_i can be characterized by face detection rate $f_i(T_i)$ and false acceptance rate $d_i(T_i)$, where T_i is the output threshold of the decision network in the ensemble. By varying T_i in the interval $[0, 1]$, we can obtain different pairs $\langle f_i(T_i), d_i(T_i)\rangle$ which actually constitute the ROC curve of ensemble g_i. Now, the question is how we can choose a set of appropriate values for T_i such that the performance of the cascaded classifier is optimal.

Suppose we have a detection task with a total of I candidate windows, and $I = F + N$, where F is the number of faces and N is the number of nonfaces. The first classifier in the cascade takes I windows as an input, among which F_1 windows are classified as faces and N_1 windows are classified as nonfaces. Hence $I = F_1 + N_1$. The F_1 windows are passed on to the second classifier for further verification. More specifically, the ith classifier $(i > 1)$ in the cascade takes $I_i = F_{i-1}$ input windows and classifies them into F_i faces and N_i nonfaces. At the first stage, it is easy to see that

$$F_1 = f_1(T_1)F + d_1(T_1)N. \tag{9}$$

More generally, it holds that

$$F_i = f_i(T_1, T_2, \ldots, T_i)F + d_i(T_1, T_2, \ldots, T_i)N, \tag{10}$$

where $f_i(T_1, T_2, \ldots, T_i)$ and $d_i(T_1, T_2, \ldots, T_i)$ represent the face detection rate and false acceptance rate, respectively, of the subcascade formed jointly by the first to the ith ensemble classifiers. Note that it is difficult to express $f_i(T_1, T_2, \ldots, T_i)$ explicitly using $f_i(T_i)$ and $d_i(T_i)$, since the behaviors of different ensembles are usually correlated. In the following, we first define two target functions for maximizing the detection accuracy and efficiency of the cascaded detector. Following this, we propose a solution to optimize both objectives.

(a) Detection accuracy

The detection accuracy of a face detector is characterized by both its face detection rate and false acceptance rate. For a specific application, we can define the maximally allowed false acceptance rate. Under this constraint, the higher the face detection rate, the more accurate the classifier. More specifically, we use cost function $C_p(T_1, T_2, \ldots, T_L)$ to measure the detection accuracy of the L-ensemble cascaded classifier, which is defined by the maximum face detection rate of the classifier under the condition that the false acceptance rate is below a threshold value T_d. Therefore,

$$\begin{aligned} C_p(T_1, T_2, \ldots, T_L) &= \max f_L(T_1, T_2, \ldots, T_L) \\ &\text{subject to } d_L(T_1, T_2, \ldots, T_L) < T_d. \end{aligned} \tag{11}$$

(b) Detection efficiency

We define the detection efficiency of a cascaded classifier by the total amount of time required to process the I input windows, denoted as $C_e(T_1, T_2, \ldots, T_L)$. Suppose the classification of one image window by ensemble classifier g_i takes t_i time. To classify I candidate windows by the complete L-layer cascade, we need a total amount of time

$$\begin{aligned} &C_e(T_1, T_2, \ldots, T_L) \\ &\quad = \sum_{i=0}^{L-1} F_i t_{i+1} \quad (\text{with } F_0 = I) \\ &\quad = \sum_{i=0}^{L-1} \left(f_i(T_1, T_2, \ldots, T_i)F + d_i(T_1, T_2, \ldots, T_i)N\right) t_{i+1}, \end{aligned} \tag{12}$$

where the last step is based on (10) and we define the initial rates $f_0 = 1$ and $d_0 = 1$.

The performance of a cascaded face detector should be expressed by both its detection accuracy and efficiency. To this end, we combine cost functions C_p (11) and C_e (12) into a unified function C, which measures the overall performance of a cascaded face detector. There are various combination methods. One example is based on a weighted summation of (11) and (12):

$$C(T_1, T_2, \ldots, T_L) = C_p(T_1, T_2, \ldots, T_L) - wC_e(T_1, T_2, \ldots, T_L). \tag{13}$$

We use a substraction for the efficiency (time) component to trade-off against accuracy. By adjusting w, the relative importance of desired accuracy and efficiency can be controlled.[1]

[1] Factor w also compensates for the different units used by C_p (detection rate) and C_e (time).

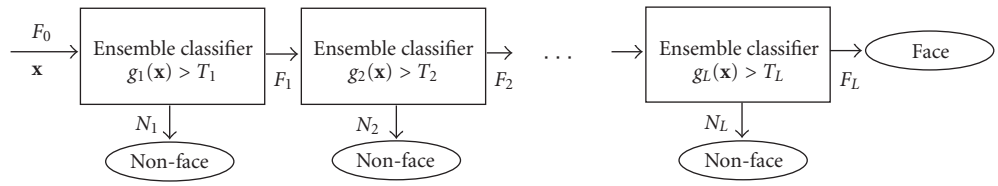


Figure 3: Pruning cascade of neural network ensembles.

Table 4: Parameter selection for the face-detection cascade.

Algorithm *Parameter selection for the cascaded face detection*
Input: F test face patterns and N test nonface patterns. A classifier cascade consisting of L neural network ensembles.
Maximally allowed false acceptance rate T_d.
Output: A set of selected parameters $(T_1^*, T_2^*, \ldots, T_L^*)$.
1. Select $T_L^* = \text{argmax}_{T_L} f_L(T_L)$, subject to $d_L(T_L) \leq T_d$.
2. **for** $k = L - 1$ to 1
3. Select $T_k^* = \text{argmax}_{T_k} C(T_k, T_{k+1}^*, \ldots, T_L^*)$.

In order to obtain a cascaded face detector of high performance, we aim at maximizing the performance goal as defined by (13). For a given cascaded detector consisting of L ensembles, we can optimize over all possible T_i $(1 \leq i \leq L)$ to obtain the best parameters T_i^*. However, this process can be computationally prohibitive, especially when L is large. In the following, we propose a heuristic suboptimal search to determine these parameters.

(c) Sequential backward parameter selection

In Table 4, we present the algorithm for selecting a set of parameters $(T_1^*, T_2^*, \ldots, T_L^*)$ that maximizes (13). Since the final face detection rate $f_L(T_1^*, T_2^*, \ldots, T_L^*)$ is upper bounded by $f_L(T_L^*)$, we first ensure a high detection accuracy by choosing a proper T_L^* for the final ensemble classifier (line 1 in Table 4). Following that, we add each ensemble in a backward direction and choose its threshold parameter T_k^* such that the partially formed cascade from the kth to the Lth ensemble gives an optimized $C(T_k^*, T_{k+1}^*, \ldots, T_L^*)$.

The experimental results show that this selection strategy gives very good performance in practice.

3.2. Implementation of a cascaded detector

We build a five-stage cascade of classifiers with increasing order of topology complexity. The first four stages are based on component network structures FNET-A to FNET-D, as illustrated in Section 2.2. The final ensemble consists of all component networks of FNET-D, plus a set of additional component networks that are variants of FNET-D. These additional component networks allow overlapping of locally connected blocks so that they offer slightly more flexibility than the original FNET-D. Although, in principle, a more complex base network structure can be used and the final ensemble can be constructed following the similar principle as FNET-A to FNET-D, we found, in our experiments, that using our proposed strategy for the final ensemble construction already offers sufficient detection accuracy while still keeping the complexity at a reasonably low level.

In order to apply the face detector to real-world detection from arbitrary images (videos), we need to address the following issues.

(1) Multiresolution face scanning

Since we have no *a priori* knowledge about the sizes of the faces in the input image, in order to select face candidates of various sizes, we need to scan the image at multiple scales. In this way, potential faces of any size can be matched to the 24×24 pixel model at (at least) one of the image scales. Here, we use a scaling factor of 1.2 between adjacent image scales during the search. In Figure 4, we give an illustrating example of the multiresolution search strategy.

(2) Fast preprocessing using integral images

Our proposed face detector accepts an image window preprocessed by zero mean and unity standard deviation, with the aim to reduce the global illumination influence. To facilitate efficient image preprocessing during the multiresolution search, we compute the mean and variance of an image window using a pair of auxiliary integral images of the original input image. The integral image of an image with intensity $P(x, y)$ is defined as

$$I(u, v) = \sum_{x=1}^{u} \sum_{y=1}^{v} P(x, y). \tag{14}$$

As introduced in [9], using integral images can facilitate a fast computation of mean value of an arbitrary window from an image. Similarly, a "squared" integral image can facilitate a fast computation of the variance of the image window.

In addition to the preprocessing, the fast computation of the mean values of image windows can also accelerate the computation of the low-resolution image input for the neural network such as FNET-A and FNET-B.

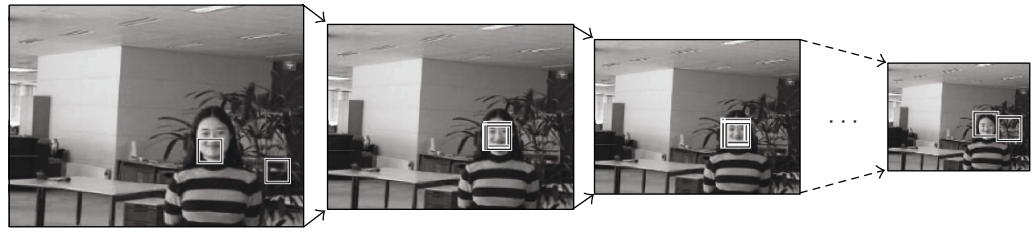
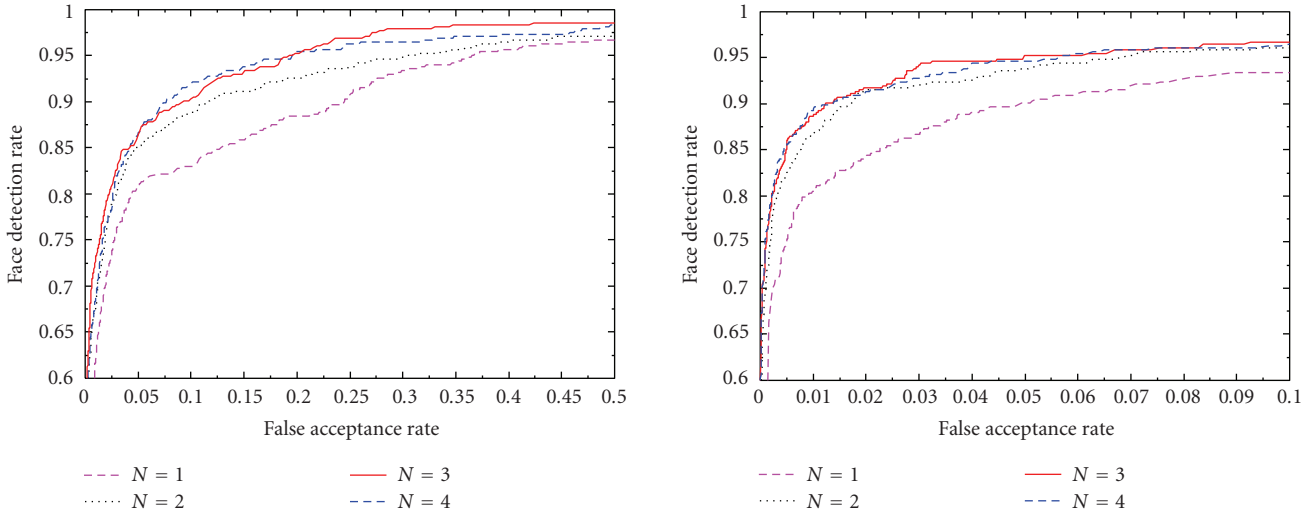


Figure 4: The multiresolution search for face detection.



(a) ROC of *FNET-A* ensembles ($T_k = 0.6$)

(b) ROC of *FNET-C* ensembles ($T_k = 0.5$)

Figure 5: ROC curves of various network ensembles with respect to different N.

(3) Merging multiple detections

Since the trained neural network classifiers are relatively robust with face variations in scale and translation, the multiresolution image search would normally yield multiple detections around a single face. As a postprocessing procedure, we group adjacent multiple detections into one group, removing repetitive detections and reducing false positives.

4. PERFORMANCE ANALYSIS

In this section, we evaluate the performance of our proposed face detector. As a first step, we look at the performance of the new ensemble technique.

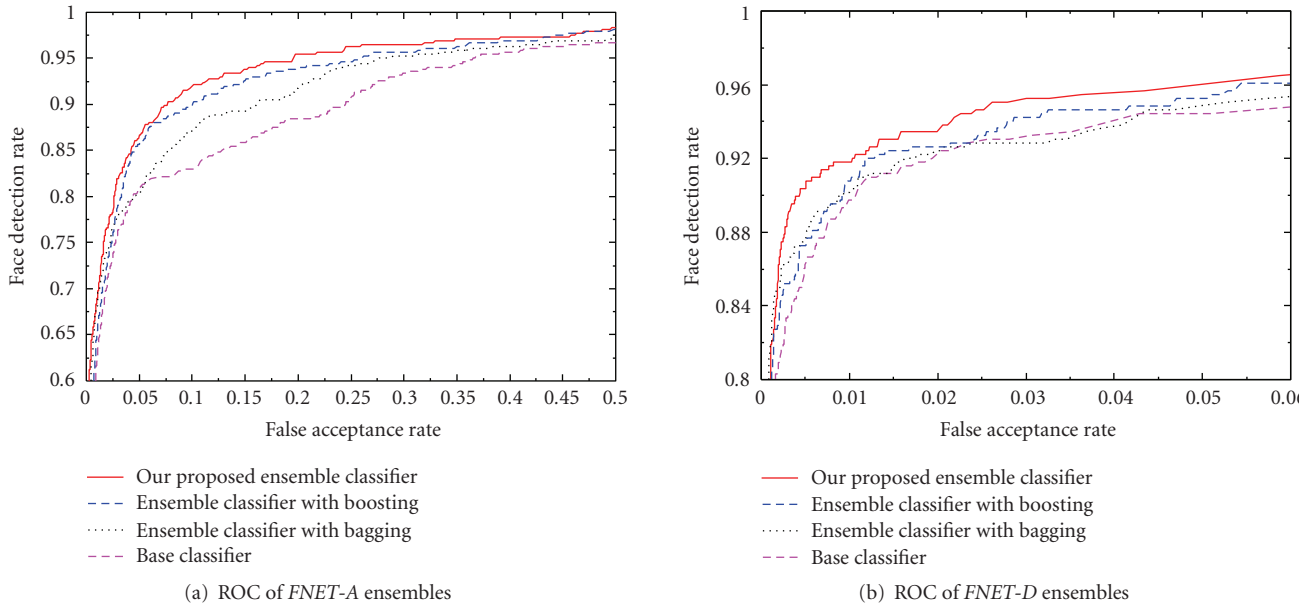
4.1. Performance analysis of the neural network ensemble

To demonstrate the performance of our proposed ensemble technique, we evaluate four network ensembles (FNET-A to FNET-D) (refer to Figure 2) that are employed in the cascaded detection. Our training face set $\mathcal{F}$ consists of 6,304 highly variable face images, all cropped to the size of 24×24 pixels. Furthermore, we build up an initial nonface training set $\mathcal{N}$ consisting of 4,548 nonface images of size 24×24. Set $\mathcal{S}$ comprises of around 1,000 scenery pictures containing no faces. For each scenery picture, we further generate five scaled versions of it, thereby acquiring altogether 5,000 scenery images. Each 24×24 sample is preprocessed to zero mean and unity standard deviation to reduce the influence of global illumination changes.

Let us first quantitatively analyze the performance gain by using an ensemble of neural classifiers. We vary the number of constituting components N and derive the corresponding ROC curve of each ensemble. The evaluation is based on two additional validation sets $\mathcal{V}_f$ and $\mathcal{V}_n$. In Figure 5, we depict the ROC curves for ensembles based on networks FNET-A and FNET-C, respectively. In Figure 5(a), we can see that the detection accuracy of the FNET-A ensemble consistently improves by adding up to three components. However, no obvious improvement can be achieved by using more than three components. Similar results also hold for the FNET-C ensemble (see Figure 5(b)).

Since using more component classifiers in a neural network ensemble inevitably increases the total computation cost during the classification, for a given network topology, we need to select N with the best trade-off between the detection accuracy and the computation efficiency.

As a next performance-evaluation step, we compare our proposed classifier ensemble for face detection with two other popular ensemble techniques, namely, bagging and boosting. We have adopted a slightly different version of



(a) ROC of *FNET-A* ensembles

(b) ROC of *FNET-D* ensembles

Figure 6: ROC curves of network ensembles using different training strategies.

the AdaBoost algorithm [15]. According to the conventional AdaBoost algorithm, the training procedure uses a fixed nonface set and face set to train a set of classifiers. However, we found, from our experiments, that this strategy does not lead to satisfactory results. Instead, we minimize the training error only on the face set. The nonface set is dynamically formed using the bootstrapping procedure.

As shown in Figure 6, it can be seen that, for complex base network structures such as FNET-D, our proposed neural-classifier ensemble produces the best results. For a base network with relatively simple structures such as FNET-A, our proposed ensemble gives comparable results with respect to the boosting-based algorithm. It is worth mentioning that, for the most complex network structure FNET-D, bagging or boosting only give a marginal improvement as compared to using a single network while our proposed ensemble gives much better results than the other techniques. This can be explained by the following reasoning.

The training strategy adopted by the boosting technique is mostly suitable for combining weak classifiers that may only work slightly better than random guessing. Therefore, during the sequential training as in boosting, it is beneficial to reuse the samples that are correctly classified by its previous component networks to reinforce the classification performance. For a neural network with simple structures, the use of boosting can be quite effective in improving the classification accuracy of the ensemble. However, when training strong component classifiers, which can already give quite accurate classification results in a stand-alone operation, it is less effective to repeatedly feed the samples that are already learned by the preceding networks. Neural networks with complex structures (e.g., FNET-C and FNET-D) are such strong classifiers, and for these networks, our proposed strategy is more effective and gives better results in practice.

4.2. Performance analysis of the face-detection cascade

We have built five neural network ensembles as described in Section 3.2. These ensembles have increasing order of structural complexity, denoted as g_i $(1 \leq i \leq 5)$. As the first step, we evaluate the individual behavior of each trained neural network ensemble. Using the same training sets and validation sets as in Section 4.1, we obtain the ROC curves of different ensemble classifiers g_i as depicted in Figure 7. The plot at the right part of the figure is a zoomed version where the false acceptance rate is within $[0, 0.015]$.

Afterwards, we form a cascade of neural network ensembles from g_1 to g_5. The decision threshold of each network ensemble is chosen according to the parameter-selection algorithm given in Table 4. We depict the ROC curve of the resulting cascade in Figure 8, and the performance of the Lth (final) ensemble classifier is given in the same plot for comparison. It can be noticed that, for false acceptance rates below 5×10^{-4} for the given validation set which is normally required for real-world applications, the cascaded detector has almost the same face detection rate as the most complex Lth stage classifier. The highest detection rate that can be achieved by the cascaded classifier is 83%, which is only slightly worse than the 85% detection rate of the final ensemble classifier. The processing time required by the cascaded classifier drastically drops to less than 5% compared to using the Lth stage classifier alone, when tested on the validation sets $\mathcal{V}_f$ and $\mathcal{V}_n$. For example, a full detection process on a CMU test image of 800×900 pixels takes around two minutes by using the Lth stage classifier alone. By using the cascaded detector, only four seconds are required to complete the processing.

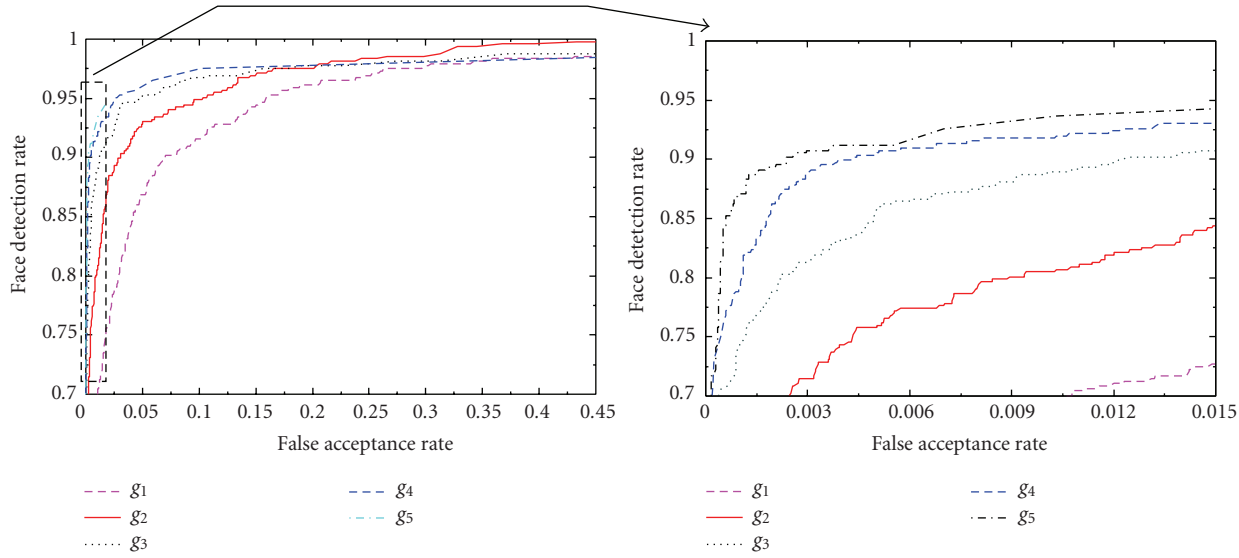


Figure 7: ROC curves of individual ensemble classifiers for face detection.

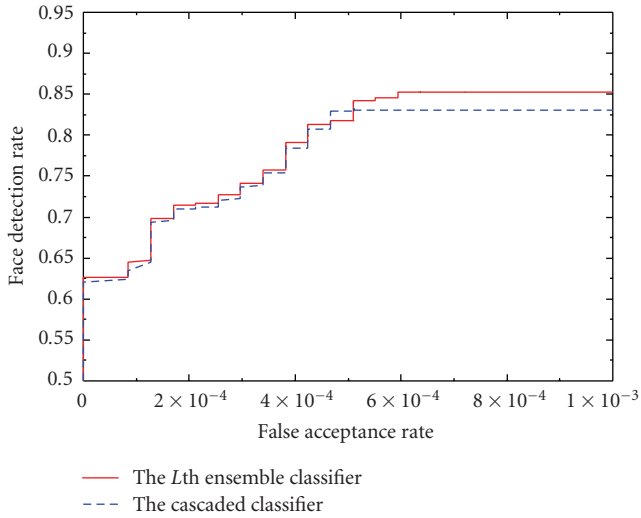


Figure 8: Comparison between the final ensemble classifier (the Lth ensemble classifier) and the cascaded classifier for face detection.

Table 5: Data sets used for the evaluation of our proposed face detector.

Data set	No. of images (sequences)	No. of faces
CMU + MIT	130	507
WEB	98	199
HN2R-DET	46	50

In our implementation, we train each ensemble independently and then build up a cascade. A slightly different strategy is to sequentially train the ensembles such that the subsequent ensemble detectors are only fed with the nonface samples that are misclassified by the previous ensemble detectors. This strategy was adopted by the Viola-Jones detector in [9]. When this strategy is used in the neural ensemble cascade in our case, our experiments show that such a training scheme leads to slightly worse results than with the independent training. This may be due to the relatively good learning capability of subsequent ensemble classifiers, which is less dependent on the relatively "easy" nonface patterns to be pruned. More study is still needed to arrive to a solid explanation.

Another benefit offered by the independent training is the saving of the training time.[2] This is because, during the cascaded training, it takes longer time to collect nonface samples during the bootstrapping training for more complex ensembles, considering the relatively low false acceptance rate of the partially formed subcascade.

4.3. *Performance analysis for real-world face detection*

In this subsection, we apply our cascaded face detector on a number of real-world test sets and evaluate its detection accuracy and efficiency. Three test sets containing various images and video sequences are used for our evaluation purposes, which are listed in Table 5. The CMU + MIT set is the most widely-used test set for benchmarking face-detection algorithms [5], and many of the images included in this data set are of very low quality. The WEB test set contains various images randomly downloaded from the Internet. The HN2R-DET set contains various images and video sequences we have collected using both a DV camera and a web camera during several test phases in the *HN2R* project [16].

[2] The complete training takes, roughly, a few hours in our experimental setup (P-IV PC 3.0 GHz).

Table 6: Comparison of different face detectors for the CMU + MIT data set.

Detector	Detection rate	No. of false positives
1. Single neural network [5]	90.9%	738
2. Multiple neural networks [5]	84.4%	79
3. Bayes statistics [18]	94.4%	65
4. SNoW [19]	94.8%	78
5. AdaBoost [9]	88.4%	31
6. FloatBoost [11]	90.3%	8
7. SVM [7]	89.9%	75
8. Convolutional network [6]	90.5%	8
9. Our approach [14]	93.6%	61

(1) Detection accuracy

First, we compare our detection results to reported results from the literature on the CMU + MIT test set. The comparison results are given in Table 5.[3] It can be seen that our approach for face detection is among one of the best performing techniques in terms of detection accuracy.

Using the WEB data set, we achieve a face detection rate of 93% with a total of 29 false positives. For the HN2R-DET set, which captures indoor scenes with relatively simple background, a total of 98% detection rate is achieved with zero false positives.

(2) Detection efficiency

Furthermore, we have evaluated the efficiency gain by using a cascaded detector. For the CMU + MIT test set, the five ensembles in the cascade reject 77.2%, 15.5%, 6.2%, 1.1%, and 0.09% of all the background image windows, respectively. For a typical image of size 320×240, using a cascade can significantly reduce the computation of the final ensemble by 99.4%, bringing the processing time from several minutes to a subsecond level. When processing video sequences of 320×240 resolution, we achieve a 4-5 frames/second detection speed on a Pentium-IV PC (3.0 GHz). The detection is frame-based without the use of any tracking techniques.

The proposed detector has been integrated into a real-time face-recognition system for consumer-use interactions [17], which gives quite reliable performance under various operation environments.

(3) Training efficiency

The state-of-the-art learning-based face detectors such as the Viola-Jones detector [9] usually takes weeks to accomplish due to the large amount of features involved. The training of our proposed face detector is highly efficient, taking usually only a few hours including the parameter tuning. This is because the cascaded detector involves only five stages, each of which can be trained independently. For each stage, only a limited number of component networks need to be trained due to the relatively good learning capacity of multilayer neural networks (Section 2). As a result, the computation cost is kept low, which offers the advantages for applications where frequent updates of detection models are necessary.

5. CONCLUSIONS

In this paper, we have presented a face detector using a cascade of neural-network ensembles, which offers the following distinct advantages.

First, we have used a neural network ensemble for improved detection accuracy, which consists of a set of component neural networks and a decision network. The experimental results have shown that our proposed ensemble technique outperforms several existing techniques such as bagging and boosting, with significantly better ROC performance for more complex neural network structures. For example, as shown in Figure 6(b), by using our proposed technique, the false rejection rate has been reduced by 23% (at the false acceptance rate of 0.5%) as compared to bagging and boosting.

Second, we have used a cascade of neural network ensembles with increasing complexity, in order to reduce the total computation cost of the detector. Fast ensembles are used first to quickly prune large background areas while subsequent ensembles are only invoked for more difficult cases to achieve a refined classification. Based on a new weighted cost function incorporating both detection accuracy and efficiency, we use a sequential parameter-selection algorithm to optimize the defined cost. The experimental results have shown that our detector has effectively reduced the total processing time from minutes to a fraction of a second, while maintaining similar detection accuracy as compared to the most powerful subdetector in the cascade.

When used for real-world face-detection tasks, our proposed face detector in this chapter is one of the best performing detectors in detection accuracy, with 94.4% detection rate and 61 false positives on the CMU+MIT data set

[3] Techniques 3, 4, 7, and 8 and our approach use a subset of the test sets excluding hand-drawn faces and cartoon faces, leaving 483 faces in the test set. If we further exclude four faces using face masks or having poor resolution, as we do not consider these situations in the construction of our training sets, we can achieve a 94.4% face-detection rate with the same number of false positives. Note that not all techniques listed in the table uses the same size of training faces and the training data size may also vary.

(see Table 6). In addition, the cascaded structure has greatly reduced the required computation complexity. The proposed detector has been applied in a real-time face-recognition system operating at 4-5 frames/second.

It is also worth pointing out the architectural advantages offered by the proposal. In our detector framework, each subdetector (ensemble) in the cascade is built upon similar structures, and each ensemble is composed of base networks of the same topology. Within one ensemble, the component networks can simultaneously process an input window. This structure is most suitable to be implemented in parallelized hardware architectures, either in multiprocessor layout or with reconfigurable hardware cells. Additionally, the different ensembles in a cascade can be implemented in a streamlined manner to further accelerate the cascaded processing. It is readily understood that these features are highly relevant for embedded applications.

REFERENCES

[1] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: a literature survey," *ACM Computing Surveys*, vol. 35, no. 4, pp. 399–458, 2003.

[2] M.-H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[3] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color pixel classification: analysis and comparison," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 1, pp. 148–154, 2005.

[4] B. Fröba and C. Küblbeck, "Real-time face detection using edge-orientation matching," in *Proceedings of the 3rd International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA '01)*, vol. 2091 of *LNCS*, pp. 78–83, Springer, Halmstad, Sweden, June 2001.

[5] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[6] 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*, vol. 26, no. 11, pp. 1408–1423, 2004.

[7] B. Heisele, T. Poggio, and M. Pontil, "Face detection in still gray images," Tech. Rep. 1687, Massachusetts Institute of Technology, Cambridge, Mass, USA, 2000, AI Memo.

[8] S. Romdhani, P. Torr, B. Schölkopf, and A. Blake, "Computationally efficient face detection," in *Proceedings of the 18th IEEE International Conference on Computer Vision (ICCV '01)*, vol. 2, pp. 695–700, Vancouver, BC, Canada, July 2001.

[9] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[10] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proceedings of the International Conference on Image Processing (ICIP '02)*, vol. 1, pp. 900–903, Rochester, NY, USA, September 2002.

[11] S. Z. Li and Z. Zhang, "FloatBoost learning and statistical face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1112–1123, 2004.

[12] F. Zuo and P. H. N. de With, "Fast human face detection using successive face detectors with incremental detection capability," in *Image and Video Communications and Processing*, vol. 5022 of *Proceedings of SPIE*, pp. 831–841, Santa Clara, Calif, USA, January 2003.

[13] Y. Ma and X. Ding, "Face detection based on hierarchical support vector machines," in *Proceedings of the 16th International Conference on Pattern Recognition (ICPR '02)*, vol. 1, pp. 222–225, Quebec, Canada, August 2002.

[14] F. Zuo and P. H. N. de With, "Fast face detection using a cascade of neural network ensembles," in *Proceedings of the 7th International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS '05)*, vol. 3708 of *LNCS*, pp. 26–34, Antwerp, Belgium, September 2005.

[15] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley-Interscience, New York, NY, USA, 2nd edition, 2000.

[16] HomeNet2Run, http://www.hitech-projects.com/euprojects/hn2r/.

[17] F. Zuo and P. H. N. de With, "Real-time embedded face recognition for smart home," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 1, pp. 183–190, 2005.

[18] H. Schneiderman and T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '98)*, pp. 45–51, Santa Barbara, Calif, USA, June 1998.

[19] M.-H. Yang, D. Roth, and N. Ahuja, "A SNoW-based face detector," in *Proceedings of Advances in Neural Information Processing Systems (NIPS '99)*, vol. 12, pp. 862–868, Denver, Colo, USA, November-December 1999.