Boosting Chain Learning for Object Detection

Rong Xiao, Long Zhu, Hong-Jiang Zhang
Microsoft Research Asia
49 Zhichun Road, Beijing 100080, P.R. China
{t-rxiao, hjzhang}@microsoft.com

Abstract

A general classification framework, called boosting chain, is proposed for learning boosting cascade. In this framework, a "chain" structure is introduced to integrate historical knowledge into successive boosting learning. Moreover, a linear optimization scheme is proposed to address the problems of redundancy in boosting learning and threshold adjusting in cascade coupling. By this means, the resulting classifier consists of fewer weak classifiers yet achieves lower error rates than boosting cascade in both training and test. Experimental comparisons of boosting chain and boosting cascade are provided through a face detection problem. The promising results clearly demonstrate the effectiveness made by boosting chain.

1. Introduction

Different from the traditional pattern classification problem where decision is made between well-defined classes, the detection problem requires discriminate analysis between the object class and the rest of the world. As a result, the detection algorithm must accommodate the intra-class variance without compromising the discriminability of locating object within cluttered scenes. On the other hand, typical negative samples are usually unavailable for building a training set due to large variance of negative class. Moreover, as the location and scale of target class are unknown, the computation cost for exhaustive search can hardly be avoided. To conclude, there are three issues which are critical for a detection system: training strategy for negative sample collection, robust learning algorithm, and computation cost for evaluation.

Sung and Poggio [10] proposed training schema, called bootstrap, was applied for negative samples collecting. During bootstrap procedure, false detections are collected iteratively into the training set, and a very low false positive rate is achieved after several iterations of learning.

Also, various learning algorithm has been applied to the detection problem. Papageorgiou [1] built a detector by training a Support Vector Machine (SVM) [12] on an over-complete wavelet representation of object classes. Rowley [3] presented a neural network-based face detection system. Roth [2] used a network of linear units, called SNoW learning architecture, which is specifically tailored for learning in the presence of a very large number of features. Schneiderman [4] used naive Bayesian classifier on multi-resolution features from different levels of wavelet transform.

Although, some works, such as [2] and [4] have achieved the best detection accuracy in the literature, both of them are too slow to be applied in real-time applications due to the computation complexity. Thereby, hierarchical classification framework is wildly adopted to build rapid detector. Serra [11] implemented a two-layer detector. The first layer consists of a fast linear SVM that removes large parts of the background. The second layer consists of a more accurate polynomial SVM performs the final face detection. Viola and Jones [7] built a cascade of boosting classifiers on an over-complete set of Haar-like features. In each layer of the cascade, AdaBoost [13] is adapted to integrate the feature selection and classifier design in one boosting procedure. By adopting simple-to-complex strategy, most non-face candidates are rejected in earlier layer of cascade with little computation costs. This structure results in extremely rapid object detector. However, AdaBoost is a sequential forward search procedure using the greedy selection strategy. Its heuristic assumption is the monotonicity. The premise offered by the sequential procedure can be broken-down when the assumption is violated. Stan Li [8] proposed FloatBoost algorithm by incorporating the idea of Floating Search into AdaBoost. Based on FloatBoost, a detector for multi-view face detection [9] is implemented. Although the new detector achieves the better performance with fewer features, the FloatBoost is unstable and computation extensive for learning complicated problem.

In this paper, a new framework, called boosting chain, is proposed for object detection. Different from the boosting cascade, this algorithm integrates the bootstrap training and boosting algorithm into a single learning procedure, and enables the utilization of historical information during boosting cascade training. Also, based on linear recursive feature elimination (RFE) [5] strategy, the redundancy of AdaBoost is removed, which avoid the local minimum with comparable performance. Moreover, during the RFE procedure, an optimized detection rate adjusting for cascade coupling is achieved.

The rest of the paper is organized as follows: Section 2 presented in detail the proposed boosting chain framework. The linear optimization algorithm for boosting chain is presented in Section 3. Section 4 provides the experimental results and conclusion is drawn in Section 5.

2. Boosting chain learning

Boosting cascade proposed by Viola [7] has been proved to be an effective way to detect faces with high speed. Based on a thorough analysis of boosting cascade, a naive boosting chain is proposed to accelerate the convergence of cascade training. Moreover, inspired by the similarity between the boosting chain learning and AdaBoost algorithm, a boosting analysis for these phenomena is therefore given. This also derived the improved training algorithm for boosting chain.

2.1. Boosting cascade

During the training procedure, windows which are falsely detected as faces by the previous classifier are processed by successive classifiers. Therefore, the overall false positive (FP) rate F and detection rate D on the training set can be defined as:

$$F = \prod_{i=1}^{M} f_i \,,\quad D = \prod_{i=1}^{M} d_i \qquad (1)$$

where symbol M, f_i, and d_i take the notation in Figure 1.

P	positive training set, $p=\|P\|$
N_i	ith negative training set, $n_i=\|N_i\|$
f_i	maximum false positive rate of ith layer
d_i	minimum detection rate of ith layer
w_j	weighting of sample x_j
F	overall false positive rate
D	overall detection rate
M	number of classifiers used in the cascade.
Φ_i	ith boosting classifier in the cascade
$h_{i,j}$	jth weak learner in ith layer
$\alpha_{i,j}$	parameter for weak learner $h_{i,j}$
b_i	threshold for the boosting classifier Φ_i
m_i	the number of weak learner in Φ_i

Figure 1. **Notation for boosting cascade**

2.1.1 Historical information in cascade training. In each layer of the boosting cascade, the classifier is adjusted to a very high recall ratio to preserve the overall recall ratio. For example, for a 11 layers cascade, to anticipate a overall detection rates at 96% in training set, the recall rate in each single layer will be 99.63% ($\sqrt[11]{0.96} = 0.9963$) on the average.

However, such a high recall rate at each layer is achieved with the penalty of sharply precision decreasing. As shown in Figure 2, value b is computed for the best precision, and value a is the best threshold which satisfies the minimal recall requirement. During the threshold adjustment from value b to value a, the classifier's discriminability in the range $[a,+\infty]$ is lost. As the performance of most weak learner used in the boosting algorithm is near to random guess, such discriminative information discarded between the layers of boost cascade is critical to increase the converge speed of successive classifiers.

2.1.2 Fine tune the boosting cascade. Moreover, suppose the positive rate in the ith layer is p_i, the empirical computation cost of each stage classifier can be defined as:

$$c_i \propto \left(\prod_{t=1}^{i-1} p_t\right)\cdot m_i \approx \left(\prod_{t=1}^{i-1} f_t\right)\cdot m_i \qquad (2)$$

Since target objects are extremely rare, the positive rates of most stage classifier are very close to the FP rates. The overall empirical computation cost can be defined as:

$$C = m_0 + \sum_{i=1}^{M} c_i \qquad (3)$$

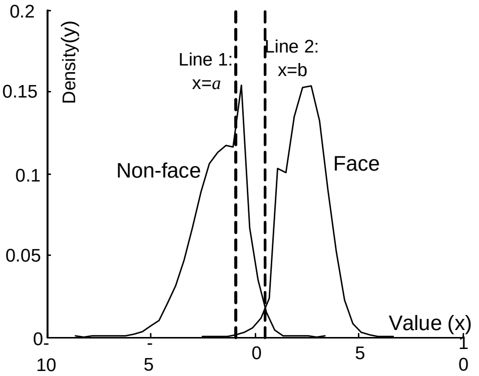

Figure 2. **Adjusting threshold for layer classifier.**

Obviously, given concrete goals for F and D, the detection rate d_i does not affect the overall computation cost, and the smaller f_i and m_i are the less computation cost will be required. Therefore, a set of optimized f_i and m_i will directly improve the detection speed of the cascade. On the other hand, fixed FP rate f_i, the overall detection rate D can be improved by increasing m_i, which corresponding to using more feature in ith classifier for a better detection rate d_i.

However, for a given detection task, the problem of finding the optimized set of f_i and m_i task is a major challenge for cascade classification. In this section, a new framework, called boosting chain, is proposed to improve the convergence of each stage classifier, and an optimizing algorithm for f_i and m_i will be discussed in the next section.

2.2. Naive boosting chain learning

As defined in Figure 1, the *i*th boosting classifier in the cascade is:

$$\Phi_i = sign[\sum_{t=0}^{m_i} \alpha_{i,t} h_{i,t}(x) - b_i] \tag{4}$$

In order to utilize historical information in *i*th layer, define:

$$h_{i+1,0}(x) = \sum_{t=0}^{m_i} \alpha_{i,t} h_{i,t}(x) \tag{5}$$

Then, initialized by the new feature $h_{i+1,0}(x)$, the boosting classifier Φ_{i+1} can be learned from the training set P and N_{i+1} .

$$\Phi_{i+1} = sign[\sum_{t=0}^{m_{i+1}} \alpha'_{i+1,t} h_{i,t}(x) - b'_{i+1}] \tag{6}$$

In order to uniform the presentation, we set $\alpha_{i+1,t} = \alpha'_{i+1,t} / \alpha'_{i+1,0}$, and $b_{i+1} = b'_{i+1} / \alpha'_{i+1,0}$, where $t = 0,1,2,...,m_{i+1}$. Combined with equation (4), equation (6) could be rewritten as:

$$\Phi_{i+1} = sign[\sum_{k=i}^{i+1} \sum_{t=1}^{m_k} \alpha_{k,t} h_{k,t}(x) - b_{i+1}] \tag{7}$$

Therefore, the *i*th classifier is "linked" into the (*i+1*)th classifier. Generally, applying this procedure repeatedly for *i=1,…,M* yields:

$$\Phi_i = sign[\sum_{k=1}^{i} \sum_{t=1}^{m_k} \alpha_{k,t} h_{k,t}(x) - b_i] \tag{8}$$

By this means, the boosting cascade is linked into a "chain" structure with multiple exits for negative patterns. The evaluation of boosting chain could be done in following manner:

```
1. Given an example $x$, evaluate the boosting chain with
   $M$ node
2. Initialize $s = 0$
3. Repeat for $i = 1$ to $M$:
   a)  $s = s + \sum_{t=1}^{m_i} \alpha_{i,t} h_{i,t}(x)$
   b)  if ($s < b_i$) then exit with negative response.
4. Exit with positive response.
```

Figure 3. **Evaluate the boosting chain**

2.3. Boosting chain learning with bootstrap

Similar to the boosting cascade, a set of images without object of target class are regarded as the source of the negative samples. After the training procedure of each node classifier, the boosting chain is evaluated over the whole image set, and any positive predicts, which are considered as FPs, are collected to form the negative training set to train the next node classifier. The whole training procedure could be illustrated in Figure 4.

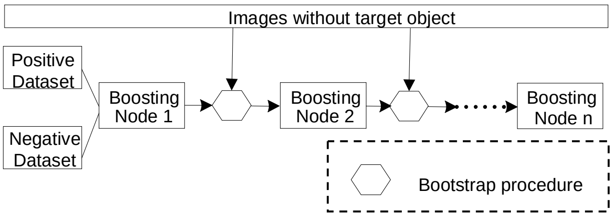

Figure 4. **Boosting chain learning with bootstrap**

Consequently, under boosting chain structure, previous classifier is a prefix of the later classifier. Such correspondence can be roughly expressed as:

$$\Phi_1 \subset \Phi_2 \subset \cdots \subset \Phi_M \tag{9}$$

The last node classifier Φ_M contains all features used in the boosting chain. According to Equation (8) it's very similar to the standard boosting classifier. The only difference is the training strategy. Different from the Adaboost, boosting chain is learned in *M* step with one positive training set and *M* different negative training set.

Actually, such similarity could be simply interpreted by the sampling procedure of the Adaboost algorithm. Given a very large negative training set, the initial training set N_0 could be selected by random sampling. After several step of learning, classifier Φ_0 is obtained. At this point, most samples in N_0 are classified correctly with small weight, and samples which could not be classified correctly will have large weight. By extending this conclusion to the whole training set, negative training set N_1 is collected by random sampling on the samples with large weight. Based on the new negative training set, the training procedure is continued, and new classifier Φ_1 is learned after several step of learning. With the similar strategy, classifier $\Phi_1,\Phi_2,…,\Phi_M$ are learned as well.

Therefore, based on the sampling interpreting, naive boosting chain learning algorithm could be improved with minor modification on weighting schema and training strategy.

Firstly, the positive sample weights are directly introduced into the substantial learning procedure. For negative samples, collected by bootstrap method, their weights are adjusted according to the classification errors of each previous weak classifier. Similar to the equation used in boosting training procedure [13], the adjusting could be done by:

$$w_j^{i+1} \leftarrow w_j^0 \exp[-y_j \sum_{k=1}^{i} \sum_{t=1}^{m_k} \alpha_{k,t} h_{k,t}(x)], \tag{10}$$

where y_j is the label of sample x_j, w_j^0 is the initial weight for sample x_j, and i is the current node index.

Secondly, the initial weak learner $h_{i,0}(x)$ is no longer required, and the successive training is directly based on the previous boosting classifier. The algorithm description will be shown in Figure 5.

```
1. Initialize: $i$=0, $F_0$=1,$\Phi$={}
   $w_j=1/p$ for all positive sample $x_j$, $w_j=1/n_i$ for all negative sample $x_j$;
2. While $F_i>F$
   a) $i=i+1$
   b) Taining $\Phi_i$ to meet the $f_i$ and $d_i$ requirements on validation set.
      − Using initial weights $w_j$, training set $P$ and $N_i$
      − Train a node classifier $\Phi_i$
   c) Node classifier optimization (in Section 3)
   d) $F_i=F_{i-1}*f_i$,    $\Phi=\Phi\cup\{\Phi_i\}$
   e) Evaluate boosting chain $\Phi$ on non-face image set, and put false detections into the set $N_{i+1}$
   f) For each sample $x_j$ in set $N_{i+1}$, update weight $w_j$ for $\Phi_{i+1}$ according to Equation (10). The weights of missing positive sample are set to zero, and the weights of remaining positive samples are kept unchanged.
```

Figure 5. **The pseudo-code for learning a boosting chain.**

Based on this strategy, the boosting chain could be regarded as a variant of AdaBoost learning algorithm with similar generalization performance and error bound.

3. Boosting chain optimization

In each step of boosting chain, performance at the current stage involves a tradeoff between accuracy and speed. The more features used the higher detection accuracy achieved. At the same time, classifiers with more features require more time to evaluate. The naive optimization method used by Viola is to simply adjust threshold for each classifier to achieve the balance between the targeted recall and false positive rates. However, as mentioned before, this method frequently results in a sharp increase in false positive rate. To address this issue, a new algorithm based on a linear model for boosting optimization is proposed.

3.1 The linear model for boosting optimization

For simplicity, following abbreviation is used: $T=m_i$, $h_j(x)=h_{i,j}$ (x), $\alpha_j=\alpha_{i,j}$, $b=b_i$, and $\alpha=\{\alpha_1,\alpha_2,...\ \alpha_T\}$. Then, the final decision function of AdaBoost in Equation (4) could be regarded as the linear combination of weak learners $\{h_1(x), h_2(x), \ldots, h_t(x)\}$.

Each weak learner $h_i(x)$ will be determined after the boosting training. When it is fixed, the weak learner maps the sample x_i from the original feature space F to a point

$$x_i^* = h(x_i) = \{h_1(x_i), h_2(x_i),..., h_T(x_i)\} \qquad (11)$$

in a new space F^* with new dimensionality T. Consequently, the optimization of α parameter can be regarded as finding an optimal separating hyper-plane in the new space F^*.

3.2 Classifier Adjusting

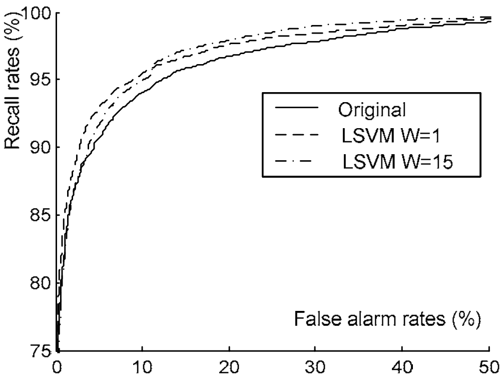

Figure 6. **The ROC curves comparing the original Boosting chain algorithm with the LSVM optimization algorithm with different weights.**

According to [12], the solution for finding optimized hyper-plane can be obtained by resolving the following quadratic programming problem:

Maximize:

$$L(\beta) = \sum_{i=1}^{n} \beta_i - \frac{1}{2}\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j (h(x_i)\cdot h(x_j)) \qquad (12)$$

subject to the constraints $\sum_i^n \beta_i y_i = 0$ and $C_i \geq \beta_i \geq 0$, $i=1,\ldots,n$. Coefficient C_i is set according to the trade-off constant C and classification risk w over the training set:

$$C_i = \begin{cases} wC & \text{if } x_i \text{ is a face pattern} \\ C & \text{otherwise} \end{cases} \qquad (13)$$

where classification risk w is determined by the detection rate requirement, The detection rate could be improved by increasing the value o*f* w at the cost of false positive rate.

The solution of this maximization problem is denoted by $\beta^0 = (\beta_1^0, \beta_2^0 ..., \beta_n^0)$. Then the optimized α will be given by $\alpha = \sum_{i=1}^{n} \beta_i y_i h_t(x_i)$.

By adjusting classification risk w *and* the bias term b, the optimized result will be found. Figure 6 shows the empirical comparison of boosting optimization strategies on face classification problem with 12000 faces and 14000 non-faces. The experimental results reveal that the LSVM optimization algorithm is more effective than original brute force adjusting strategy.

3.3 Boosting redundancy reduction

As AdaBoost is a sequential forward search procedure using the greedy selection strategy, redundancy during the learning procedure can not be avoided. FloatBoost adopt the backtrack strategy. It deletes unfavorable weak classifiers from the ensemble when a new weak classifier is added. Although FloatBoost provides a promising way to reduce the redundancy during the boosting training, such strategy is conflict with the boosting weight schema, and the performance of learning procedure is unstable, which will be shown in Figure 10.

According to the linear model of the boosting classifier, the result classifier could be expressed as:

$$f(x)=\sum_{i=1}^{T}\alpha_i h_i(x)+b \qquad (14)$$

The gradient direction of $f(x)$ over $h_i(x)$ is:

$$\nabla_{h_i(x)} f(x)=\alpha \qquad (15)$$

```
1.  Training a linear SVM classifier over the set {$h_i(x)$},
    $i$=1,…,M, and weight $w$.
2.  Sort the classifier parameter vector $\alpha$ by value.
    Suppose the new index will be $i_1,i_2,…,i_M$
3.  k=1,…,N, N is the const for feature elimination.
        a)  remove the feature $h_{ik}$,
        b)  compute current learning accuracy $p_k$
        c)  put back feature $h_{ik}$
4.  Remove the feature $h_{ik}$, with the largest $p_k$
5.  M=M-1, and go to 1.
```

Figure 7. n-level boosting feature reduction algorithm.

Therefore, in the most time, the smaller α_i is the less significant the feature $h_i(x)$ will be. With this heuristic, backtrack the feature selection procedure on each step is unnecessary. To remove the redundancy in boosting procedure, a top-down schema is more favorable in this situation. By incorporating the idea of linear FRE, a new boosting redundancy reduction algorithm is proposed, and reduction algorithm is shown in Figure 7.

4. Experimental Results

In this Section, a face detector based on boosting chain is implemented, and performance comparisons are made to AdaBoost cascade and FloatBoost cascade, which are two most relevant face detectors in the literature.

4.1 Experimental Setup

Figure 8. Some samples in the face training data.

More than 12000 image without faces and 10000 face images were collected by cropping from various sources, such as AR, Rockfeller, FERET, BioID and from WEB. Most faces in the training set have the variation of both in-plane and out-of-plane rotation within the range of [-30°, 30°]. A total number of about 80000 face training samples with size of 20x20 are generated from the 10000 face images by following random transformation: mirroring, four-direction shift with 1 pixels, in-plane rotation within 15 degrees and scaling within 20% variations, where 12000 face samples are used in boosting training and other positive samples are used in boosting chain optimization.

The testing set consists of the standard MIT+CMU face database, which composed of 125 grayscale images containing 483 labeled frontal faces. And all experiments are tested over a 1.5 Ghz Pentium 4 computer.

4.2 Performance comparisons

Three detectors based on boosting chain, FloatBoost cascade and Adaboost cascade are implemented on the same training set. The FP-Detection rate curve over the MIT-CMU test is shown in Figure 9. And the average numbers of features used in each detector are listed in Table 1.

Table 1. Average number of features used in face detection on MIT-CMU Test set.

Boosting Chain	FloatBoost Cascade	Boosting Cascade
18.1	18.9	22.5

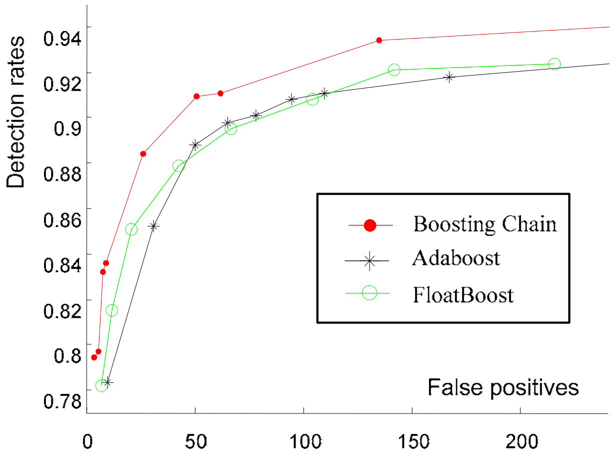

Figure 9. Detection rates for various numbers of false positives on the MIT+CMU test set. All detectors are constructed in 11 layer cascade.

In order to sidestep any differences resulting from the underlying infrastructure systems of detector [6], a training set of 18000 images (8000 faces and 10000 non-faces) and a test set of 15000 images (5000 faces, and 10000 non-faces) are used to evaluate these algorithms. The images are 20*20 grayscale and aligned by eye center. By fixing the detection rate to 95%, the FP rates under different features are shown in Figure 10.

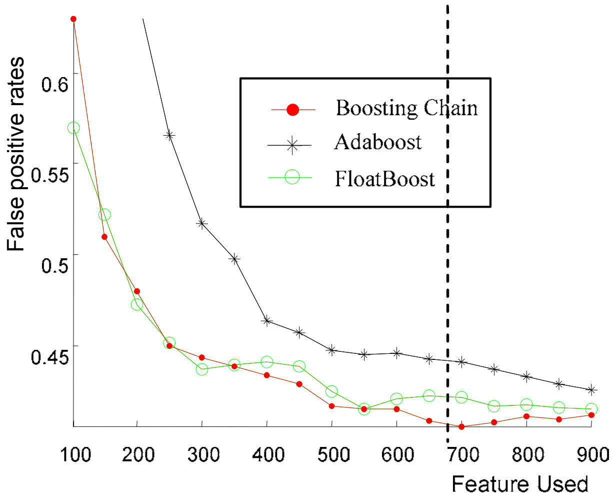

Figure 10. FP rates for various numbers of features on the testing set. The boosting chain used here only contains 1 layer. The best result is achieved by Boosting chain algorithm with around 700 features

4.3 Discussions

From experiment results shown in Figure 9, 10 and Table 1, it is seen the performance of proposed approach in following aspects:

1. From the Detection-FP rate curve shown in Figure 9, the boosting chain approach outperforms Adaboost cascade and FloatBoost cascade. It works especially well at higher recall rate. This property will greatly enhance the efficiency of the post-filtering procedure.
2. In Table 1, the boosting chain algorithm again achieves the best performance.
3. From the experimental results in Figure 10, the boosting chain classifier outperforms Adaboost and FloatBoost. Although, it seems the curve of FloatBoost is very close to the curve of boosting chain, boosting chain is stable due to the LSVM optimization, and its global minimum could be expected.
4. Specially, as feature began to be eliminated from the original feature space, the false alarm rate of boosting chain classifier is kept dropping, and reaches its minimum at the point with around 700 features. If further feature reduction is processed, the performance of boosting chain is gradually degraded. This phenomena could be explained that the boosting chain reach the intrinsic dimension of boosting linear model at the point with 700 features.

In summary, the experimental results from two test set reveal the robustness and efficiency of proposed framework.

5. Conclusions

In this paper, a novel framework for rapid object detection has been presented. In this framework, boosting cascade and bootstrap training are integrated into a single learning procedure, which not only provide a theoretical foundation for cascade training, but also improve classifier performance by incorporating historical knowledge of cascade learning. Moreover, based on a linear analysis model for boosting classifier, a classifier adjusting and redundancy reduction algorithm is also proposed.

The experiment results from most testing sets have shown the robustness and superiority of the proposed framework. Also, we believe the generic framework presented in this paper can be applied to other classification problems in computer vision.

References

[1] C. P. Papageorgiou, M. Oren, and T. Poggio. "A general framework for object detection". *Proc. of International Conf. on Computer Vision*, 1998.

[2] D. Roth, M. Yang, and N. Ahuja. "A snowbased face detection". *Neural Information Processing*, 12,2000

[3] H. A. Rowley, S. Baluja, and T. Kanade. "Neural network-based face detection". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), pages 22-38.

[4] H. Schneiderman and T. Kanade. "A Statistical Method for 3D Object Detection Applied to Faces and Cars". *Proc. IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, 2000

[5] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. BIOWulf Technical Report, 2000.

[6] M. Alvira, and R. Rifkin, "*An Empirical Comparison of SNoW and SVMs for Face Detection*", CBCL Paper #193/AI Memo #2001-004, Massachusetts Institute of Technology, Cambridge, MA, January 2001.

[7] P. Viola and M. Jones. "Robust real time object detection". *IEEE ICCV Workshop on Statistical and Computational Theories of Vision*, Vancouver, Canada, July 13, 2001.

[8] S. Z. Li, *et al.* "Statistical Learning of Multi-View Face Detection". *Proc. of the 7th European Conf. on Computer Vision*. Copenhagen, Denmark. May, 2002.

[9] S.Z. Li, Z.Q. Zhang, Harry Shum, H.J. Zhang. "FloatBoost Learning for Classification". In Proceedings of The 16-th Annual Conference on Neural Information Processing Systems (NIPS). Vancouver, Canada, December 9-14, 2002.

[10] T. Poggio and K. K. Sung. "Example-based learning for view-based human face detection". *Proc. of the ARPA Image Understanding Workshop*, II: 843-850. 1994.

[11] T. Serre, *et al*. "Feature selection for face detection". *AI Memo 1697*, Massachusetts Institute of Technology, 2000

[12] V. N. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, Inc., New york, 1998.

[13] Y. Freund and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". *Journal of Computer and System Sciences*, 55(1):119--139, August 1997.

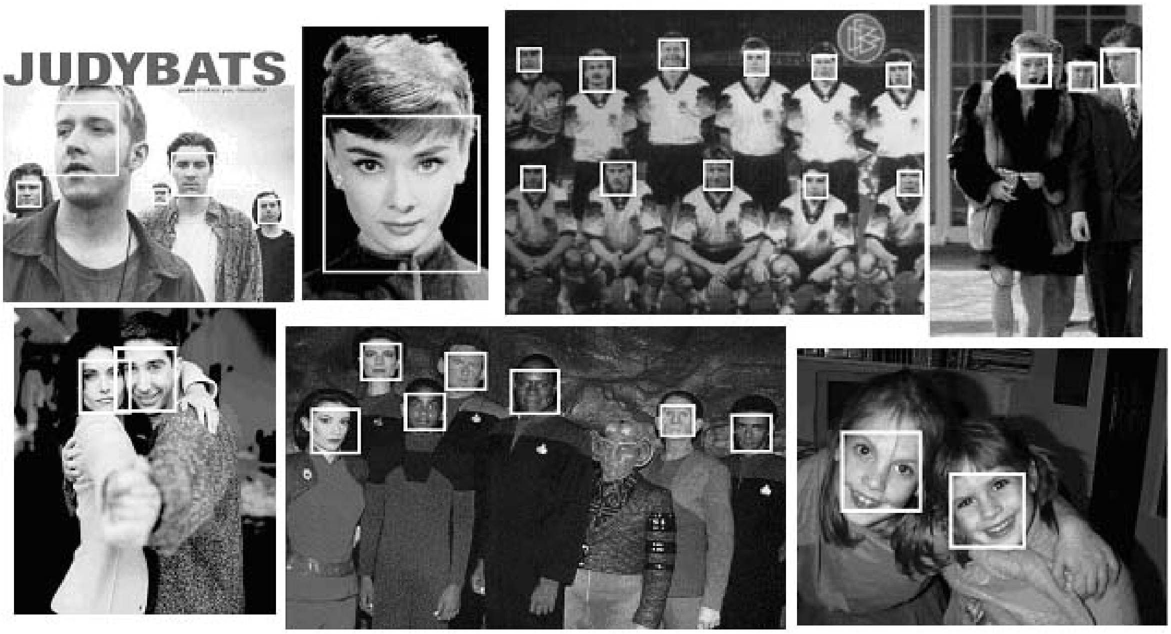

Figure 11: Sample experiment results using our method on MIT-CMU Test Set

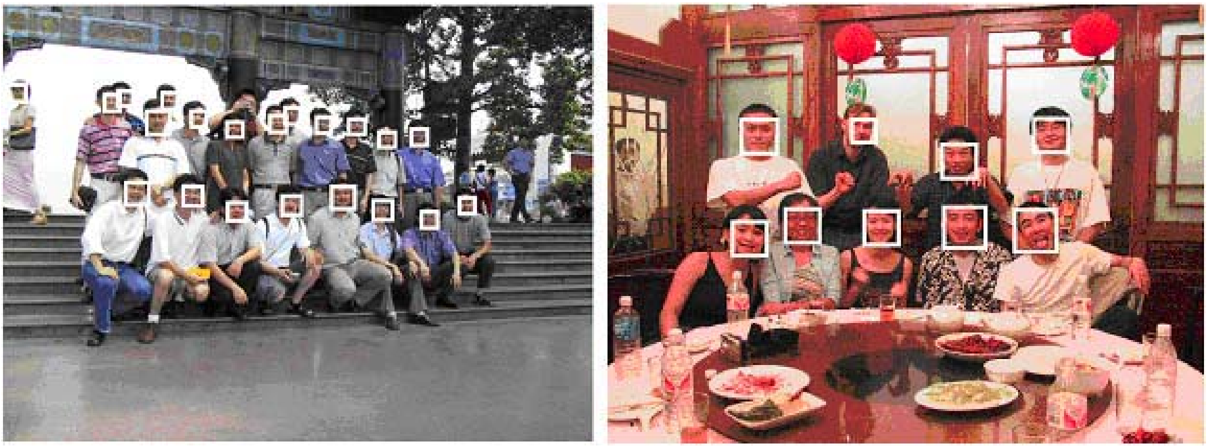

Figure 12: Sample experiment results using our method on digital photos