Matrix-Structural Learning (MSL) of Cascaded Classifier from Enormous Training Set

Shengye Yan[1,2], Shiguang Shan[1,2], Xilin Chen[1,2], Wen Gao[1,3], Jie Chen[1,3]
[1]ICT-ISVISION FRJDL, Institute of Computing Technology, CAS, Beijing, 100080, China
[2]Key Laboratory of Intelligent Information Processing, CAS, Beijing, 100080, China
[3]School of Computer Science and Technology, Harbin Institute of Technology, 150001, China
{syyan, sgshan, xlchen, wgao, jchen}@jdl.ac.cn

Abstract

Aiming at the problem when both positive and negative training set are enormous, this paper proposes a novel Matrix-Structural Learning (MSL) method, as an extension to Viola and Jones' cascade learning method for object detection. Briefly speaking, unlike Viola and Jones' method that learn linearly by bootstrapping only negative samples, the proposed MSL method bootstraps both positive and negative samples in a matrix-like structure. Moreover, an accumulative way is further presented to improve the training efficiency of MSL by inheriting features learned previously during training procedure. The proposed method is evaluated on face detection problem. On a positive set containing 230,000 face samples, only 12 hours are needed on a common PC with a 3.20GHz Pentium IV processor to learn a classifier with false alarm rate less than 1/1,000,000. What's more, the accuracy of the learned detector exceeds the state-of-the-art results on the CMU+MIT frontal face test set.

1. Introduction

Object detection is one of the classical problems in computer vision and pattern recognition, with wide potential applications such as visual surveillance, robotics, image retrieval, and intelligent user interfaces. Large variations in pose and individual difference, as well as varying backgrounds and imaging conditions, make this problem particularly challenging.

There have been many important successes over the past several years for some visual patterns such as faces [19], pedestrians [12], and cars [14]. Especially, the face detector of Viola and Jones is among the most influential systems [19] for its high detection speed. The key elements of Viola and Jones' approach include: 1) the cascade structure, which enables the detector to be not only fast but also accurate. 2) The use of AdaBoost [3] to combine weak hypotheses into a strong ensemble. 3) Weak hypotheses designed by thresholding on single simple Haar-like feature.

The large body of literature spawned by this seminal work has tended to focus on alternatives to AdaBoost [6, 9, 11, 16, 18, 20, and 21], Haar-like features [8], coarse-to-fine architecture [5, 7, 13, and 22] or optimization tuning of the cascade architecture [1, 10, 15, 20, and 21]. However, the aspect about the training set has not received adequate attention yet. In this paper, we focus on how to train a cascaded classifier efficiently when both positive and negative training sample sets are enormous. For simplicity, hereinafter, "enormous sample set" is abbreviated to ESS.

Cascade design deal with the negative ESS successfully in company with an efficient classifier structure. The detector is designed as a cascade of sub-classifiers as shown in figure 1. Each sub-classifier deals with some of the non-object examples and makes a decision to either reject the input candidate, i.e. classifying it as non-object, or continue evaluation using the next sub-classifier.

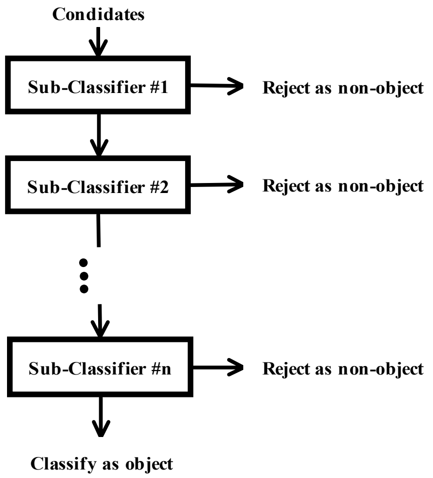

Figure 1: Structure of the cascaded classifier.

Increasing the size of training set is one of the good means to obtain classification performance promotion which is in practice often more significant than expected. Furthermore, collecting hundreds of thousands of positive training samples in some applications is not such a difficult job sometimes. Taking frontal face as example, we can easily collect images with frontal faces from various sources such as the explosive WEB, public databases, and digital photo albums. Also, the frontal face sample set can be enlarged by generating new samples with domain knowledge incorporated, i.e. mirroring, in-plane rotation, small changes in translation and scale. Then, given a face set, containing e.g. 10000 frontal face samples, we can obtain up to 800000 samples by mirroring, half pixel shift in 8 directions, and 5 rotations in plane.

Training classifier with such kind of ESS directly is hardly tractable on common personal computers. Sung and Poggio [17] proposed a training scheme, called bootstrap, and applied it for negative training samples collecting from negative ESS. During bootstrap procedure, false detections are collected iteratively into the training set, and a very low false positive rate is achieved after several iterations.

Even with small negative training sample set collected by bootstrap, training time cost is still a very tedious problem for cascade learning, since Boosting cascade learning is very time-consuming. For instance, Viola and Jones reported training time on the order of days or even weeks. If positive ESS is further taken into account, more time are needed obviously.

Intuitively, bootstrap scheme can also be applied to positive samples. For example, one can bootstrap positive sample set by using cascaded classifier as a whole, as illustrated in figure 6. But it is computationally not smart (refer to section 3 for details). Instead, in this paper, we propose a novel method, named Matrix-Structural Learning (MSL), to take advantage of the modularity of cascaded classifier design to improve the efficiency. Briefly speaking, unlike Viola and Jones' methods that learn linearly by bootstrapping only negative samples, the proposed MSL method bootstraps both positive and negative samples in a matrix-like structure.

Moreover, an accumulative way of sub-classifier training is presented to improve the training efficiency of MSL further by inheriting features learned previously during the training procedure, which can greatly reduce the time cost on feature selection of MSL learning.

The proposed method is evaluated on face detection problem. On a positive set containing 230,000 face samples, only 12 hours are needed on a common PC with a 3.20GHz Pentium IV processor to learn a classifier with false alarm rate less than 1/1,000,000. The accuracy of the detector learned exceeds the state-of-the-art results on the CMU+MIT frontal face test set.

The rest of the paper is organized as follows: we first describe in detail of the proposed MSL method in Section 2, then in section 3 an alternative method and a discussion of its computation cost compared with MSL is given. The feature-inheriting technique for speeding up is introduced in Section 4. Experimental setup and results are given in Section 5 and finally conclusion is drawn in section 6.

2. Matrix-structural learning

Boosting cascade proposed by Viola [19] has been proved to be an effective way to detect faces with high speed, but it is time-consuming in training. Based on the modularity of the cascade structure, taking into account a larger positive sample size, a novel matrix-structural learning (MSL) for cascaded classifier is proposed to accelerate cascade training.

For simplicity, the symbols and their denotations used in this paper are listed in figure 2.

C(i,j)	classifier built by *i*th stage learning from *j*th positive training sample set of bootstrap by MSL
C'(i,j)	classifier built by *i*th stage learning from *j*th positive training sample set of bootstrap by LSL
N(i,j)	Negative training set of *C(i,j)* learning
N'(i)	Negative training set of *C'(i,j)* learning
P(i,j)	Positive training set of *C(i,j)* learning
P'(i)	Positive training set of *C'(i,j)* learning
F_j	total feature number of *C(i,j)*
f_{jk}	*k*th feature of *C(i,j)*
H_i	total iterations of *i*th positive bootstrap in MSL
B	stage number of cascade
M_i	iteration number of positive bootstrap of *i*th stage learning in MSL
M	iteration number of positive bootstrap in LSL

Figure 2: Symbols and their denotations used in this paper.

2.1. Bootstrap positive sample in sub-classifier learning

Bootstrap proposed by Sung and Poggio [17] is a good way to collect a small and representative training set for learning a classifier, which was applied for negative training samples collecting from negative ESS. We argue that it can not only be applied for negative samples, it can also be applied for positive sample. Before discussing how MSL integrates positive sample bootstrap into cascade learning, the bootstrap schema is given firstly in figure 3. This bootstrap method starts with a randomly sampling training set. So we name it randomly starting bootstrap in contrast with the new bootstrap method we proposed in Section 2.2. There are two assumptions in bootstrap schema: 1) selecting a small training sample set and learning with it will take less computation cost in comparison with learning directly with the whole ESS; 2) the selected training samples can well represent the original ESS in terms of classification performance of the final classifier.

A modified version of negative sample bootstrap is used in cascade learning proposed by Viola and Jones [19], i.e.

1) Start with a small set of training samples randomly picked out from ESS;
2) Train a classifier with current training set;
3) Run the classifier on ESS. Collect some samples that the current system wrongly classifies. Add these samples to the training set as new training samples.
4) Return to 2).

Figure 3: Schema for randomly starting bootstrap.

each bootstrap iteration results in a sub-classifier and the false positives of all previous sub-classifiers are collected into a new negative training set with a predefined size. The sub-classifier is constructed by boosting. Each step of the boosting involves a tradeoff between accuracy and speed. Generally speaking, the more features used, the higher detection accuracy achieved. By using suitable number of features, each sub-classifier is adjusted to have a very high target detection rate and moderate false alarm rate to achieve the overall high detection rate and very low false alarm rate of the final strong classifier.

The framework of the proposed MSL method is illustrated in figure 5. In the figure, N_i and P(.) denote the bootstrapped negative and positive training set respectively, and C(.) denotes sub-classifier. As can be seem clearly, MSL is a matrix-like structure with two ***alternated*** bootstrap procedures, i.e. positive and negative sample set bootstrap. While the negative bootstrap is conducted similarly as in Viola and Jones' method, the positive bootstrap is illustrated in figure 4 and described as follows.

Each row in figure 5 is an iteration of positive sample bootstrap, as shown in figure 4. To guarantee the same

1) Start with a small set of positive training samples randomly selected from ESS;
2) Train a strong classifier on current training set with terminal conditions of target detection rate d_{min} and target maximal false alarm rate f_{max};
3) Run the classifier on positive ESS to get a detection rate d. If d exceeds d_{min}, the classifier is the final sub-classifier, train is over; otherwise, collect some samples that the current system wrongly classifies and add these samples into the training set as new training samples, return to 2).

Figure 4: positive sample bootstrap in sub-classifier learning.

classification accuracy performance (detection rate) on both the whole positive ESS and the current training set, the sub-classifier learned from the training set with target accuracy is validated on the whole positive ESS. If the detection rate does not meet the target, new positive samples wrongly classified are collected by bootstrap and added to the training set, until the detection rate on positive ESS meets the target. Thus, in positive sample bootstrap, for the *i*th sub-classifier, the relation between two successive positive training sample sets, say the *j*th and the *(j+1)*th iteration, can be expressed as:

$$P(i,j) \subset P(i,j+1) \quad 0<i<B+1,\ 0<j<M_i$$

where P(.) denotes the bootstrapped positive training set, B the total number of sub-classifiers, and M_i the number of bootstrap of positive samples.

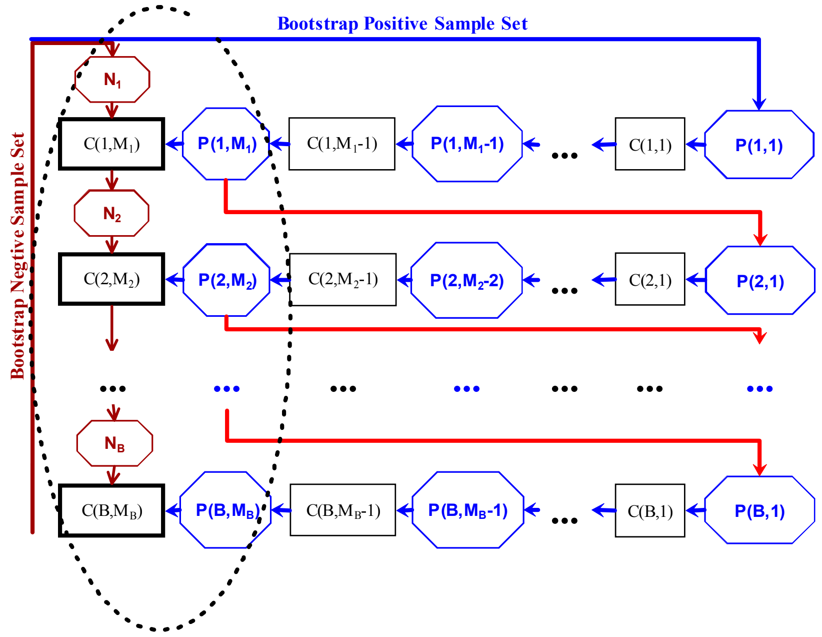

Figure 5: Matrix-Structural Learning of Cascaded Classifier.

2.2. Incremental bootstrap of positive samples for cascade learning

This sub-section describes another important novel point in the proposed MSL, which can greatly speed up the bootstrap procedure.

As we have seen in section 2.1, in each step of positive sample bootstrap, the representation ability of the positive training sample set used by current sub-classifier is gradually enhanced. At the same time, in consideration of the moderate target false alarm rate, there is large intersection between the negative sample sets of two successive stages. So, the positive training sample set of previous sub-classifier still has much representation ability of the positive ESS. Therefore, in our method, the positive training set collected by the bootstrap of previous sub-classifier is used as the initial set of the positive bootstrap for current sub-classifier. Formally, this can be expressed as):

$$P(i,1) = P(i-1, M_{i-1}) \quad 1<i<B+1$$

This is also illustrated as the red arrow-line from P(i-1, M_{i-1}) to P(i, 1) in figure 5. By this means, the iteration (positive bootstrap) stops of each sub-classifier can be impressively reduced. Note that, for the first sub-classifier learning, the initial positive sample set, i.e. P(1, 1), is still constructed by random sampling.

3. An alternative method and computation cost discussion

In contrast with MSL, there is an alternative way to integrate positive sample bootstrap into cascade learning. Cascaded classifier can be treated as a whole in positive sample bootstrap. The method is illustrated in figure 6. Since it is line-structural, for simplicity, we call it LSL. The denotations used in figure 6 of symbols can be referred to figure 2.

To analyze the computation cost of MSL and LSL is not easy. Here we give a simple and rough discussion.

Cascade learning procedure can be considered to consist of two unit procedures. First is the collecting of positive training samples by bootstrap and sub-classifier learning with training set. For simplicity, we denote the procedure by CL. Collecting of positive training samples by bootstrap tests current stage classifier on positive ESS. This procedure spends little time. Sub-classifier learning learns the stage classifier with boosting from current negative and positive training sample set. It takes almost all the computation of CL. Second is the collecting of negative training sample by bootstrap; the procedure is denoted by CN.

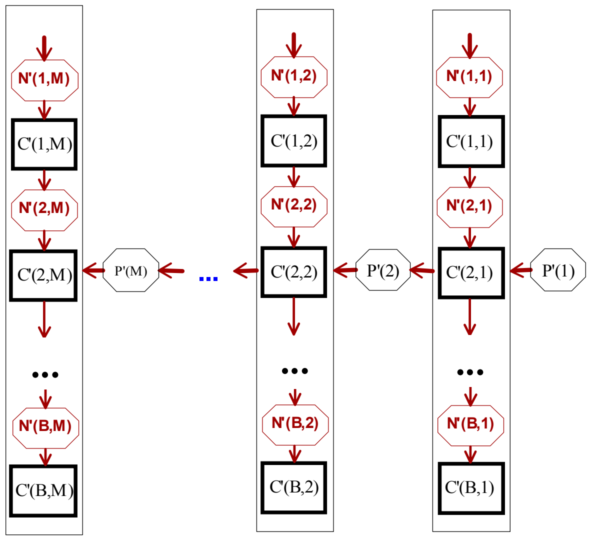

Figure 6: Cascade of Linear Structure Learning.

MSL only needs B executions of CN, with B the stage number of cascade. But LSL needs M*B executions of CN, which is M times more than the one in MSL, with M the iteration times of positive sample bootstrap.

While for CL procedure, MSL needs $\sum_{k=1}^{B} M_k$ executions. And LSL needs M*B executions.

Assuming the parameters of the bootstrap, such as initial sample size, maximum amount of new added samples in each bootstrap iteration, are the same in both methods. In LSL, M is the total iterations of positive sample bootstrap roughly according to the difficulty of the negative ESS. While in MSL, the corresponding number is $\sum_{k=1}^{B} M_k$. Generally, M and $\sum_{k=1}^{B} M_k$ are on a near order. If the corresponding positive training sample size of MSL to the one of LSL is roughly thought to be the same, the CL computation cost of LSL is roughly B times of that of MSL.

From the analysis above, one can see that MSL is roughly B times more efficient than LSL.

4. Feature-inheriting technique

In order to further speed up the cascade learning, a feature-inheriting technique is further presented in this section. As mentioned above, during each stage of the learning, many temporary sub-classifiers are built sequentially during the positive sample bootstrap. And, for two successive sub-classifiers, only very small difference between their positive sets is introduced by the bootstrap, and their negative training sets are completely the same. In

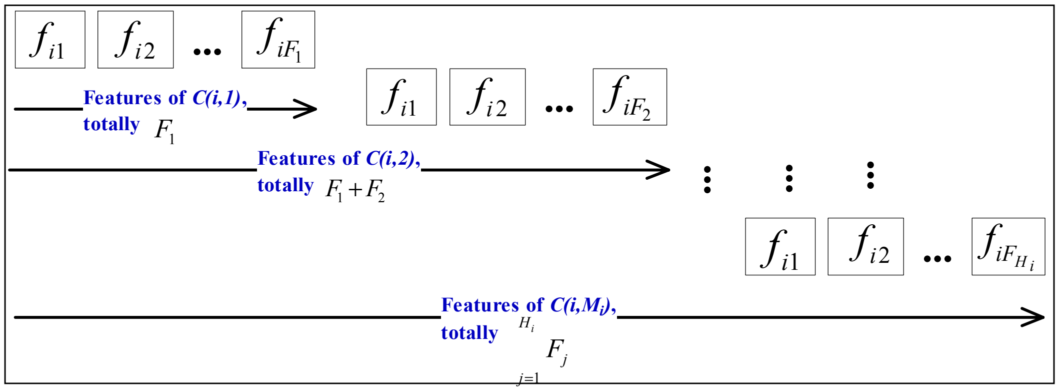
f_{i1}
f_{i2}
f_{iF_1}
Features of C(i,1), totally F_1
f_{iF_2}
Features of C(i,2), totally F_1+F_2
$f_{iF_{H_i}}$
Features of C(i,M_i), totally $\sum_{j=1}^{H_i} F_j$

Figure 7: Feature composition of *i*th layer in cascade.

other word, their training sets are quite similar. Therefore, the features selected by the last sub-classifier can naturally applicable to the current sub-classifier learning. Therefore, except very few new features, most of the features for the current sub-classifier can be inherited directly from the last sub-classifier, which is called by us "feature-inheriting" technique. Intuitively, the computation for feature selection can be greatly reduced by such a technique.

However, feature inheriting does not imply weak classifier inheriting. Due to the difference in training set, the weak classifiers based on these inherited features must be re-trained. Specifically, thresholds and weights for these weak classifiers must be learned based on the current positive and negative training set. Fortunately, this procedure is quite efficient.

The above technique results in quickly an initial "strong" classifier for the current sub-classifier learning based on which some new features can then be added one by one via boosting until the target performance is satisfied. Intuitively, this learning procedure is like an incremental learning.

The feature composition of the *i*th sub-classifier in the cascade is illustrated in figure 7. If *FS (C)* denotes the feature set of classifier *C*, the relation of two feature sets of successive sub-classifier during positive sample bootstrap procedure can be expressed as:

$$FS(C(i,j)) \subset FS(C(i,j+1)) \quad 0<i<B+1,\ 0<j<M_i+1$$

5. Experiments

In this section, a face detector based on MSL is implemented, and performance comparisons are made with Boosting cascade and MSL cascade.

5.1. Experimental setup

23,608 face samples are collected from various sources, such as WEB, FERET, and BioID. Most faces in the sample set have the variation of out-of-plane rotation within range of [-40°, 40°]. Totally 236,080 24×24 grayscale face samples are generated from the original 23,608 face images with manually labeled eyes by following transformation: mirroring, in plane rotation of -12°,-6°, 0°, and 6° 12°. Some examples are shown in figure 8.

As for the negative samples, 15,000 images without faces are collected; extra 15,000 images without faces are generated by in-plane rotation of 45° to get more negative samples. So, there are totally 30,000 images for collecting the non-face samples.

The testing set consists of the standard MIT+CMU frontal face database, which is composed of 130 images containing 507 frontal faces. All experiments are conducted on a common PC with a 3.20GHz Pentium IV processor.

Feature pool consists of 31,728 Haar-like features of five types, as showed in figure 9.

The basic sub-classifier learning method is RealBoost proposed in [5], an improved version of AdaBoost. And the basic cascade learning is nested cascade proposed in [5], also an improved version of Viola and Jone's method[19]. Minimum detection rate and maximum false alarm rate are set to 0.9998 and 0.4 respectively.

For the negative bootstrap, the number of re-sampled non-face samples for each stage is fixed to 10,000. While for the positive bootstrap, the size of the starting face sample set is 3000 and maximally 500 new samples can be added for each iteration.

To detect faces with various scales, test images are down-sampled with scale coefficient of 0.8.

5.2. Training efficiency investigation

The training computation cost partly depends on the training set size. So, the number of training face samples of some stages is shown in Table1. One can see that in the first stage, only 3612 training face samples are used. With stage increasing, training face number increases too. This can be explained by the fact that the corresponding non-face samples are more and more similar to faces and difficult to distinguish from faces.

Stage Number	1	2	8	11	15	19
Number of final training face samples	3612	3695	6629	8260	10498	12899

Table 1: Training face sample number in final training set of some stages

Stage Number	1	2	4	6	8	10	12	14	16
Incremental bootstrap	5	4	4	4	5	5	7	7	6
Random starting bootstrap	5	6	7	8	11	17	20	22	24

Table 2: Number of iterations for randomly starting bootstrap and incremental bootstrap in some stages

Bootstrap iteration number	1	2	3	4	5	6	Total
Number of newly selected features	38	49	39	42	41	46	216
Number of training face samples	10087	10291	10392	10482	10533	10602	——

Table 3: Number of newly selected feature and corresponding number of training face samples of MSL without using feature-inheriting technique for the 15th stage

Bootstrap iteration number	1	2	3	Total
Number of newly selected features	26	13	1	40
Number of training face samples	10087	10447	10498	——

Table 4: Number of newly selected feature and corresponding number of training face samples of MSL with feature-inheriting technique for the 15th stage.

Figure 8: some examples of face samples.

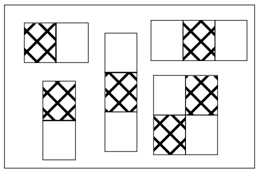
Figure 9: Five types of candidate Haar-like features.

To investigate the effect of incremental bootstrap, the number of iterations for randomly starting bootstrap and incremental bootstrap in some stages is gave in Table2. It can be seen that incremental bootstrap reduces the iterations more and more with stage number increasing. Compared with randomly starting bootstrap, incremental bootstrap has more stable number of iterations range from 4 to 8.

To show the efficiency of feature-inheriting technique, two experiments are conducted taking the 15th stage as example. The number of newly selected features and the corresponding number of training face samples in each bootstrap iteration are showed in Table3 and Table4, respectively. In the experiments, MSL without using feature-inheriting technique selects totally 216 new features while MSL with feature-inheriting technique selects only 40 new features. The number of newly selected features is largely reduced attribute to the feature-inheriting technique. The final quantities of features in the stage classifiers learned by the two methods are 46 and 40 respectively. It is a little surprising that fewer features are learned by the stage classifier learned with feature-inheriting technique. We believe this can owe to the sequence forward learning mode of boosting.

5.3. Experimental results

Training the cascaded detector totally spent only about 12 hours with a false alarm rate 1/1, 000, 000 by MSL along with feature-inheriting sub-classifier learning. Considering that 236,080 face samples are used for cascaded classifier learning, it is indeed quite efficient. In contrast, 17 hours were needed to train a detector based on RealBoost learning with a constant 10000 training face samples randomly selected from the same 236,080 samples.

To get a sense of the accuracy of the classifier, face detector learned with MSL was tested on the CMU+MIT dataset comprising 130 images containing 507 faces. The ROC curves are given in figure 10. In order to construct a complete ROC curve, the last classifier layers are removed or adjusted by its threshold. Of particular interest is the improvement over [5], which is the baseline cascade learning method of MSL. This improvement can be attributed directly to the advantages of enormous positive sample set, since the two detectors are otherwise very comparable. The result reported in [23] is also listed for it used a different method of manifold to sub-sample face ESS. The results reported in [1] and [13] are comparable to our system. Nevertheless, an optimization strategy of ROC points for cascade is utilized in [1]. We believe if we use some optimizations in adjusting thresholds of the cascade, the performance of our method will be improved also. Note that, in [13], results are reported on a reduced dataset with 5 images containing hand-drawn faces removed.

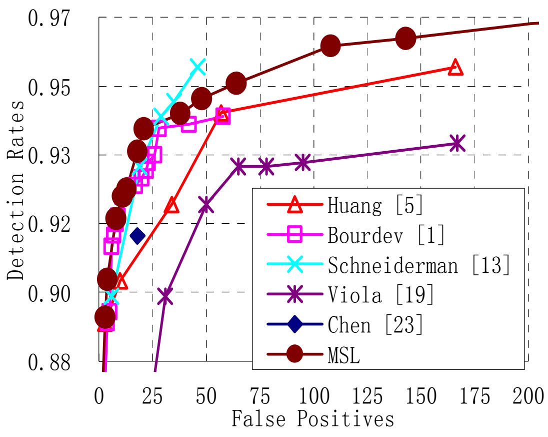

Figure 10: Performance comparison on CMU+MIT face set.

6. Conclusions

A novel learning method for cascaded classifier, called MSL, is proposed to learn from both enormous positive and enormous negative sample set for object detection. In MSL, boosting cascade and bootstrapping positive samples are integrated into single learning procedure in an efficient way, which not only provides a theoretically higher efficiency of learning with enormous training set, but also improves classifier performance by validating learned classifier on enormous positive sample set. Moreover, by the proposed feature-inheriting technique in MSL learning, more efficient cascade training is achieved.

The experimental results on standard MIT+CMU frontal face test set have shown the robustness and superiority of the proposed method. Also we believe the learning method presented in this paper can be applied to other classification problems in computer vision.

Acknowledgements

This paper is partially supported by National Natural Science Foundation of China under contract No.60332010, and No.60673091, Hi-Tech Research and Development Program of China (No.2006AA01Z122), 100 Talents Program of CAS, Program of New Century Excellent Talents in University (NCET-04-0320), Natural Science Foundation of Beijing municipal under contract No. 4061001 and 4072023, and ISVISION Technology Co. Ltd.

References

[1] L. Bourdev, J. Brandt, Robust object detection via soft cascade. CVPR2005, vol.2, pp 236- 243.

[2] S. C. Brubaker, M. D. Mullin, and J. M. Rehg. Towards Optimal Training of Cascaded Detectors. ECCV2006, LNCS 3951, pp. 325 ff.

[3] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55(1) (1997) 119–139.

[4] X. W. Hou, C. L. Liu, and T. N. Tan, Learning Boosted Asymmetric Classifiers for Object Detection.CVPR 2006.

[5] C. Huang, H. Z. Ai, B. Wu, S. H. Lao, Boosting Nested Cascade Detector for Multi-View Face Detection, in 17th Inter. Conf. on Pattern Recognition (ICPR 2004), Cambridge, UK, 23-26 Aug.,2004.

[6] S.Z. Li, L. Zhu, Z. Q. Zhang, A. Blake, H. J. Zhang, H. Shum, Statistical Learning of Multi-View Face Detection. In Proceedings of the 7th European Conference on Computer Vision. Copenhagen, Denmark. May, 2002.

[7] R. Lienhart, L. Liang, and A. Kuranov, A detector tree of boosted classifier for real time object detection and tracking. IEEE International Conference on Multimedia & Expo (ICME2003).

[8] R. Lienhart, J. Maydt. An Extended Set of Haar-like Features for Rapid Object Detection IEEE ICIP 2002,Vol. 1, pp 900~903, 2002.

[9] C. Liu and H.Y. Shum. Kullback-Leibler Boosting. CVPR2003, pp. 587-594.

[10] H. T. Luo, Optimization Design of Cascaded Classifiers. CVPR2005.

[11] S.W. Lyu , Infomax Boosting. CVPR 2005.

[12] S. Munder and D.M. Gavrila. An Experimental Study on Pedestrian Classification. IEEE Trans. on PAMI, 28(11):1863–1868, 2006.

[13] H. Schneiderman, "Feature-centric evaluation for efficient cascaded object detection," CVPR2004, vol. 2, pp. 29-36.

[14] H. Schneiderman, T. Kanade, Object detection using the statistics of parts. Int. J. Comput. Vision, 56(3):151–177, 2004.

[15] J. Sochman, J. Matas. WaldBoost–Learning for Time Constrained Sequential Detection. CVPR2005.

[16] J. Sun, J. M. Rehg, A. F. Bobick. Automatic Cascade Training with Perturbation Bias. CVPR (2) 2004: 276-283.

[17] K.K. Sung and T. Poggio. Example-based learning for view-based human face detection. IEEE Trans. on PAMI, 20(1):39–51,1998.

[18] P. Viola, Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade NIPS 2001:1311-1318.

[19] P. Viola and M. Jones, Rapid Object Detection Using a Boosted Cascade of Simple Features. CVPR2001.

[20] J. X. Wu, M. D. Mullin, and J. M. Rehg. Linear asymmetric classiffier for cascade detectors. In ICML '05: Proceedings of the 22nd international conference on Machine learning, pages 988–995, New York, NY, USA, 2005. ACM Press.

[21] J. X. Wu, J. M. Rehg, M. D. Mullin. Learning a Rare Event Detection Cascade by Direct Feature Selection, NIPS 2004.

[22] R. Xiao, L. Zhu, and H.-J. Zhang, Boosting chain learning for object detection. In ICCV, pages 709–715, 2003.

[23] J. Chen, R. Wang, S. Yan, S. Shan, X. Chen, and W. Gao. How to Train a Classifier Based on the Huge Face Database? IEEE International Workshop on AMFG2005, LNCS 3723, pp. 84 – 95.