A Weight Adjustment Strategy to Prevent Cascade of Boosted Classifiers from Overfitting

Ki-Yeong Park
Center for Integrated Smart Sensors
312, IT Convergence Center
Daehak-ro, Yuseong-gu,
Daejeon 305-701, Republic of Korea
cpky0@kaist.ac.kr

Dong-Seok Kim
Center for Integrated Smart Sensors
312, IT Convergence Center
Daehak-ro, Yuseong-gu
Daejeon 305-701, Republic of Korea
kds1130@kaist.ac.kr

ABSTRACT

We propose a weight adjustment strategy to prevent a cascade of boosted classifiers from overfitting and to achieve an improved performance. In cascade learning, overfitting often occurs due to the iterative applications of bootstrapping. Since false positives that the previous classifier misclassifies are collected as negative examples through bootstrapping, negative examples more similar to positive examples are prepared as stages go on, and thus classifiers become tuned to the positive examples. When overfitting occurs, the classifier cascade shows performance degradation more in the detection rate than in the false alarm rate. In the proposed strategy, the imbalance between the detection rate and the false alarm rate is evaluated by computing the weight ratio of positive examples to negative examples and it is compensated by adjusting the weight ratio prior to boosting at each stage. Experimental results confirm the effectiveness of the proposed strategy. For experiments, face and pedestrian classifier cascades were trained by employing previous approaches and the proposed strategy. By employing the proposed strategy, the detection rate of classifier cascades was significantly improved for both face and pedestrian.

Keywords

AdaBoost, bootstrapping, cascade of boosted classifiers, overfitting, face detection, pedestrian detection

1. INTRODUCTION

Cascade of boosted classifiers is an object detection method popularly employed in real-time systems. Since Viola and Jones introduced a real-time face detector based on classifier cascade [Vio04], the cascade structure has been successfully adopted in detecting various objects such as faces [Li11, Liu12, Cev13], vehicles [Cui10, Siv12], and pedestrians [Che11, Xin11, Hoa12, Pri13], and now it serves as a foundation for modern detectors [Dol12]. Many state-of-the-art object detectors utilize the cascade structure alone or combined with other object detection methods [Dol09, Che11, Xin11].

The success of classifier cascade is mainly due to its fast processing speed. In object detection domain, where a few objects have to be distinguished from an extremely large number of non-objects, classifiers have to be trained to achieve a very high detection rate (e.g., 95%) and an extremely low false alarm rate (e.g., 10^{-6}). This asymmetric performance goal can be efficiently achieved by employing the cascade structure. Classifier cascade achieves a fast processing speed by pre-filtering most of non-objects with simple classifiers at early stages and a high detection accuracy by using more complex classifiers at later stages [Vio01]. Enzweiler and Gavrila compared several pedestrian detectors and reported that the pedestrian detector based on the cascade structure was approximately 20 times faster than the other detectors [Enz09].

Successful cascade learning requires extensive trial-and-error. In cascade learning, each classifier in a cascade is trained just until a given performance goal is achieved. Therefore, the performance of a classifier cascade cannot be simply improved by adopting a more sophisticated algorithm for training each classifier. Furthermore, the detection rate of a classifier cascade is definitely degraded while the false alarm rate will be improved by appending more classifiers to the cascade.

In this paper, we propose a cascade learning strategy to achieve an improved performance. In cascade learning, negative examples required for training each classifier are collected through bootstrapping [Sun98]. Overfitting often occurs due to iterative applications of bootstrapping. As stages go on, negative examples which are more similar to positive examples are collected and classifiers become tuned to the positive examples. In the proposed strategy, the imbalance between the detection rate and the false alarm rate is evaluated at each stage to detect overfitting and the weights of training examples are adjusted to avoid overfitting. The rest of this paper is organized as follows: We briefly review related works in the following subsection and describe conventional cascade learning algorithms in Section 2. The proposed strategy is described in Section 3, and experimental results are presented in Section 4. In Section 5, the conclusions are drawn.

1.1 Related Work

Multi-exit cascade has been studied as an improved cascade structure [Pha08]. While the information obtained by the previous classifier is discarded prior to boosting at each stage in the Viola-Jones cascade, it is inherited to the subsequent classifier in the multi-exit cascades such as boosting chain [Xia03], nested cascade [Wu04], soft cascade [Bou05], and embedded cascade [Sab12]. By recycling information at each stage, the information redundancy between classifiers will be reduced and more efficient classifier cascade can be constructed [Sun14].

In cascade learning, each classifier is trained to achieve a very low false negative rate (e.g., 0.5%) and a rather high false alarm rate (e.g., 50%). Training a classifier to achieve such an asymmetric goal is not a task typically addressed by machine learning algorithms. In the Viola-Jones' scheme, a very low false negative rate is achieved by adjusting threshold of each classifier [Vio04], however, with the penalty of sharply increased false alarm rate [Xia03]. By adopting an asymmetric AdaBoost, the asymmetric goal can be achieved more effectively [Vio01, Mas07, Sun07, Wu08, Lan12, Wan12]. Masnadi-Shirazi and Vasconcelos [Mas07] presented a theoretically solid asymmetric boosting algorithm based on the statistical view of boosting, and Sun et al. [Sun07] investigated several asymmetric boosting algorithms which assign larger weights to false negatives by manipulating weight update rule. Landesa-Vázquez and Alba-Castro [Lan12] showed that AdaBoost can be used as an asymmetric learning algorithm by manipulating initial weights of training examples instead of manipulating weight update rule.

Cascade learning has several parameters such as the number of classifiers in a cascade and the performance goal for each classifier. Since the processing speed and the performance of a classifier cascade vary non-intuitively with these parameters, a successful cascade learning requires extensive trial-and-error [Sab12]. Several cascade optimization algorithms have been proposed [Sab12, Wan12, Lud13, Pai14]. These algorithms search for optimal trade-off between the detection performance and the processing speed.

2. CONVENTIONAL CASCADE LEARNING ALGORITHM

In this section, AdaBoost algorithm and conventional cascade learning algorithms are described. Among the several variants of AdaBoost algorithms, the gentle AdaBoost [Fri00] is presented, which is also used for describing the proposed strategy in Section 3. For cascade learning algorithms, both the Viola and Jones' algorithm [Vio04] and the multi-exit cascade learning algorithm [Xia03, Pha08] are described.

2.1 AdaBoost

AdaBoost is a machine learning algorithm for constructing a strong classifier as a linear combination of weak classifiers [Fre97]. AdaBoost maintains a distribution of weights over the training examples by increasing the weights of misclassified examples and decreasing those of correctly classified examples at each boosting round. With this weight update rule, AdaBoost focuses on training examples so far misclassified.

For given training examples (x_1, y_1), … (x_N, y_N), where x_i is an example image and y_i is class label (+1, –1 for positive and negative examples, respectively), the weights of training examples are evenly initialized before boosting begins. At each boosting round, a weak classifier is learned from the weight distribution, and the weight of each training example is updated according to the prediction of the weak classifier as

$$w_{t+1}(i) = \frac{w_t(i) \cdot \exp\left(-y_i \cdot h_t(x_i)\right)}{Z_{t+1}}, \qquad (1)$$

where $w_t(i)$ and $w_{t+1}(i)$ are the weights of a training example x_i at the rounds t and t+1, respectively, and Z_{t+1} is the normalization factor. $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction (a value between –1 and +1) for each example. The weak classifier learning and the weight update are repeated until a performance goal is achieved, and the boosted classifier is given as a linear combination of the weak classifiers as

$$H(x_i) = sign\left(\sum_{t=1}^{T} h_t(x_i) + b\right), \qquad (2)$$

where T is the number of weak classifiers in the boosted classifier and b is the bias applied for the classifier. In cascade learning, a positive bias (b > 0) is applied to the boosted classifier to achieve a lower false negative rate, with less weak classifiers, by sacrificing its false alarm rate.

To achieve the asymmetric performance goal more efficiently, an asymmetric AdaBoost algorithm can be adopted. As Landesa-Vázquez and Alba-Castro [Lan12] showed, AdaBoost can be directly used as an asymmetric learning algorithm. Boosting becomes to focus more on reducing the false negative rate when larger initial weights are assigned to the positive examples as

$$w(i) = \begin{cases} \dfrac{\gamma}{2N_+}, & \text{for } y_i = +1 \\ \dfrac{(1-\gamma)}{2N_-}, & \text{for } y_i = -1 \end{cases}, \tag{3}$$

where $\gamma \in (½, 1)$ is the asymmetric parameter, and N_+ and N_- are the number of positive examples and that of negative examples, respectively.

2.2 Viola-Jones Cascade Learning

AdaBoost [Fre97, Fri00] is used to train each classifier in a cascade. Each classifier is trained to achieve a very low false negative rate and a rather high false alarm rate. In object detection domain where there exist only a few objects contrary to an extremely large number of non-objects in an image, it is possible to construct a simple classifier with a very low false negative rate by sacrificing its false alarm rate [Vio01]. Cascade of boosted classifiers achieves both fast processing speed and high accuracy by discarding negatives with these simple classifiers at early stages and by using more complex classifiers at later stages [Vio04].

Negative examples for training each classifier are prepared through bootstrapping. False positives which the previous classifier misclassifies are collected and used as negative examples for training the subsequent classifier [Sun98]. By the iterative applications of bootstrapping, cascade learning becomes to face more difficult negative examples as stages go on.

The number of classifiers in a cascade can be determined from a given goal for the detection rate and the false alarm rate [Vio04]. For example, a detection rate of 95% and a false alarm rate of 6×10^{-6} can be achieved by constructing a 10-stage cascade and training each classifier to achieve a detection rate of 99.5% ($0.995^{10} \approx 0.95$) and a false alarm rate of 30% ($0.3^{10} \approx 6 \times 10^{-6}$). One thing we have to remember in the cascade design is that the detection rate of a classifier cascade as well as its false alarm rate decrease as more classifiers are appended to the cascade, which makes the cascade optimization much complicated.

2.3 Multi-exit Cascade Learning

Xiao et al. [Xia03] proposed the multi-exit cascade structure called boosting chain where each classifier inherits score from its previous classifier. In the Viola-Jones cascade learning, weights of training examples are evenly initialized before boosting begins at each stage, and thus information obtained by its previous classifier is discarded. For a multi-exit cascade, each classifier is trained after the weights of training examples are initialized and then updated according to the predictions of the weak classifiers of its previous classifier [Xia03]. Weight of each training example is initialized and updated as

$$w_{M+1}(i) = \frac{w_1(i) \cdot \exp\left(-y_i \cdot \sum_{t=1}^{M} h_t(x_i)\right)}{Z_{M+1}}, \tag{4}$$

where $w_1(i)$ is evenly initialized weight, $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction for each training example, M is the number of weak classifiers in the previous classifier, and Z_{M+1} is the normalization factor. Boosting proceeds with the weights initialized and updated.

3. PROPOSED CASCADE LEARNING STRATEGY

In this section, the proposed cascade learning strategy is presented, which evaluates the imbalance between the detection rate and the false alarm rate and compensates it to avoid overfitting.

3.1 Overfitting in Cascade Learning

While a classifier trained to be overly complex may classify the training examples perfectly, it is unlikely perform well on new patterns. This situation is known as overfitting [Dud01]. Even though it is often believed that AdaBoost does not suffer from overfitting [Fri00], cascade learning which employs AdaBoost to train each classifier in the cascade often undergoes overfitting. In cascade learning, false positives which the previous classifier misclassifies are collected and used as negative examples for training the subsequent classifier [Sun98, Vio04]. As stages go on and the bootstrapping is iterated, negative examples which are more similar to positive examples are collected, and thus the generalization of the classifier cascade becomes degraded.

Even though the overfitting occurs due to the bootstrapping iterations, it will not be solved by reducing the number of bootstrapping iterations. The bootstrapping iterations can be reduced by designing the cascade to have less number of classifiers and each classifier to achieve a lower false alarm rate (to reject more negatives) as we described in Subsection 2.2. However, if each classifier in a cascade achieves a lower false alarm rate, negative examples even more similar to the positive examples will be collected through bootstrapping. This will worsen the overfitting problem.

3.2 Proposed Cascade Learning Strategy

When overfitting occurs, the classifier cascade shows performance degradation more in the detection rate than in the false alarm rate, since each classifier is trained with true positives (positive examples) and false positives (bootstrapped negative examples). In the proposed strategy, the imbalance between the detection rate and the false alarm rate of the previous classifier is evaluated and it is compensated by adjusting the weights of training examples. The multi-exit cascade structure described in Subsection 2.3 is employed in the proposed strategy. In the multi-exit cascade learning, the weights of training examples are updated using the weak classifiers of the previous classifier before boosting begins at each stage [Xia03]. The imbalance between the detection rate and the false alarm rate can be evaluated from the updated weight distribution by computing the weight ratio of positive examples to negative examples, and it can be compensated by adjusting the weight distribution.

Fig. 1 shows the proposed learning algorithm for each classifier in a cascade. This classifier learning has to be repeated to construct a classifier cascade. After the weights of training examples are initialized and updated, the ratio of the sum of positive example weights to that of negative example weights is computed as

$$r_M = \frac{\sum_{y_i=+1} w_1(i)\cdot \exp\left(-\sum_{t=1}^{M} h_t(x_i)\right)}{\sum_{y_i=-1} w_1(i)\cdot \exp\left(\sum_{t=1}^{M} h_t(x_i)\right)}, \qquad (5)$$

where $w_1(i)$ is an evenly initialized weight, $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction (a value between –1 and +1) for example x_i, and M is the number of weak classifiers used in the previous classifier.

Since each classifier except the first classifier in a cascade is trained with bootstrapped negative examples (false positives), the sum of negative example weights is larger than that of positive examples weights ($r_M < 1$) after the weight update. When larger weights are given to the negative examples at the weight initialization, boosting will focus more on reducing the false alarm rate [Lan12]. To prevent boosting from focusing more on reducing the false alarm rate, the weight ratio has to be adjusted to be balanced. The weight of each training example is adjusted by adding a bias b_w to the weight update rule of the multi-exit cascade as

$$w_{M+1}(i) = \frac{w_1(i)\cdot \exp\left(-y_i \cdot \left(\sum_{t=1}^{M} h_t(x_i) + b_w\right)\right)}{Z_{M+1}}, \qquad (6)$$

where Z_{M+1} is the normalization factor. The bias b_w for the weight adjustment is computed as

$$b_w = \frac{1}{2}\cdot \ln\left(\frac{r_M}{W_a}\right), \qquad (7)$$

where W_a is the weight adjustment factor devised for experimental purpose, which is a desired weight ratio of positive examples to negative examples. $W_a = 1$ should be used to compensate the imbalance between the detection rate and the false alarm rate. If the weights are adjusted with $W_a < 1$, boosting will focus more on reducing the false alarm rate, and vice versa. To confirm this, several different weight adjustment factors will be tested in our experiments.

After the weights of training examples are adjusted, boosting is proceeded as in the conventional cascade learning. Since the weights are updated using the weak classifiers of the previous classifier before boosting begins, the prediction of each boosted classifier has to be computed by summing the predictions of all the weak classifiers used in the previous classifier as well as in the current classifier as

$$H(x_i) = sign\left(\sum_{t=1}^{M+k} h_t(x_i) + b\right), \qquad (8)$$

where M is the number of weak classifiers used in the previous classifier, k is the number of newly learned weak classifiers, and b is the bias applied to the current classifier to reduce the false negative rate by sacrificing its false alarm rate.

- Given training examples (positive examples and bootstrapped negative examples):
  - Initialize the weights of examples evenly.
  - Update the weights of examples using the weak classifiers of the previous classifier and compute the weight ratio r_M using the equation (5).
  - Compute the bias b_w for weight adjustment with the weight adjustment factor $W_a = 1$ using the equation (7).
  - Adjust the weights of examples with the bias b_w using the equation (6).
- Repeat the following process until a given performance goal is achieved:
  - Proceed with the conventional boosting (Train a weak classifier and update the weights with the predictions of the weak classifier).
  - Determine the bias b that is applied to the boosted classifier to achieve the goal for the false negative rate as in the equation (2).
- Output the boosted classifier, which is a linear combination of all the weak classifiers learned at the previous stage as well as at this stage as in the equation (8).

Figure 1. The proposed classifier learning algorithm for constructing a classifier cascade

4. EXPERIMENTAL RESULTS

For experiments, we trained classifier cascades for face and pedestrian by employing the Viola-Jones cascade (VJ), the multi-exit cascade (Multi-exit), the Viola-Jones cascade with the asymmetric AdaBoost (Asymmetric), and the proposed strategy (Proposed). Each classifier in the cascades was trained using the gentle AdaBoost [Fri00] with the same set of Haar-like features [Vio04], and was trained to achieve the same performance goal: a detection rate of 99.5% and a false alarm rate of 50%. The classifier cascades employing the asymmetric AdaBoost [Lan12] were trained with the asymmetry parameter γ = 4/5 in the equation (3) to assign four times of weights to positive examples. The weight adjustment factors W_a = 0.5 and 2.0 were also tested in the experiments, which assign twice weights to negative examples and to positive examples, respectively.

Face examples were obtained by cropping the images in Labeled Faces in the Wild-a [Hua07, Wol11]. All of the 13,233 face examples were resized to 18 x 22, and 2,000 examples of them were used for training and the rest was used for test. Pedestrian examples were cropped from test images in the Daimler Stereo Pedestrian Detection Benchmark Dataset [Kel11], and were resized to 14 x 28. Among 13,714 pedestrian examples, 5,000 examples were used for training and the rest was used for test. For test, 1,000,000 negative examples were prepared by randomly cropping from more than 8,000 images which do not contain any objects.

4.1 Impact of Weight Adjustment

Fig. 2 shows the learning curves of the classifier cascades evaluated at each stage with test examples. The false negative rate and the false alarm rate of the classifier cascades are presented separately to show the imbalance between them. The performance goal for the cascade learnings is also presented in the figure (Goal).

The imbalance problem is obviously observed in the face cascade learnings. All the cascades employing the previous approaches overachieved the false alarm rate goal while they failed to achieve the false negative rate goal. In case of the pedestrian cascade learnings, all the cascades employing the previous approaches underachieved both the false negative rate and the false alarm rate goals. However, the degradation in the false negative rate was more severe.

The experimental results show that the performance imbalance problem can be solved by employing the proposed strategy. When larger weights were assigned to negative examples (W_a = 0.5), the false negative rate was similar to or worse than that of cascades employing the conventional approaches, and it was improved when the same or larger weights were assigned to positive examples ($W_a \geq 1.0$). The biggest improvement on the false negative rate was achieved when larger weights were assigned to positive examples (W_a = 2.0). However, in this case, there was degradation in the false alarm rate. Moreover, the pedestrian classifier cascade failed to reduce the false alarm rate anymore at 10th stage as shown in Fig. 2(b).

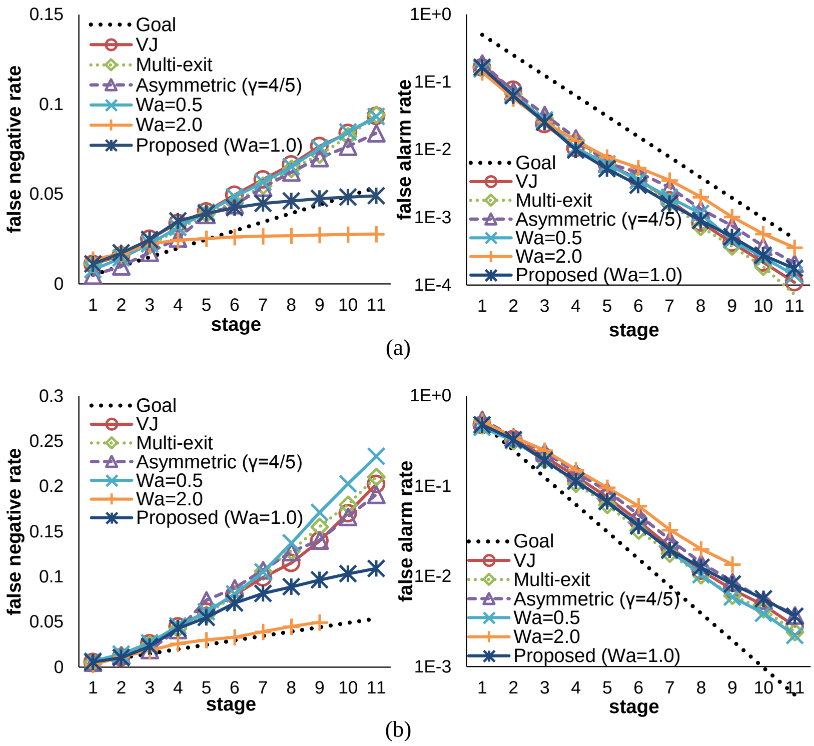

Figure 2. Learning curves of classifier cascades. (a) Face, (b) pedestrian.

These experimental results confirm that the weight ratio should be balanced ($W_a = 1$) to achieve a higher detection rate without sacrificing the false alarm rate.

The figure also shows that the performance improvement gained by adopting the multi-exit cascade structure or the asymmetric AdaBoost is marginal. By adopting a sophisticated algorithm for training each classifier, the performance goal for each classifier can be achieved with less number of weak classifiers. However, the performance of the classifier cascade is hardly improved, since each classifier is trained just until a given goal is achieved.

4.2 Detection Performance Comparison

We compared the performance of the classifier cascades employing the proposed strategy against that of the classifier cascades employing the previous approaches. Fig. 3 shows the receiver operating characteristic (ROC) curves for the classifier cascades. Even though all the classifier cascades were trained to achieve the same performance goal with the same set of training examples and the same set of Haar-like features, those employing the proposed strategy show significantly improved detection rate at the same false alarm rate.

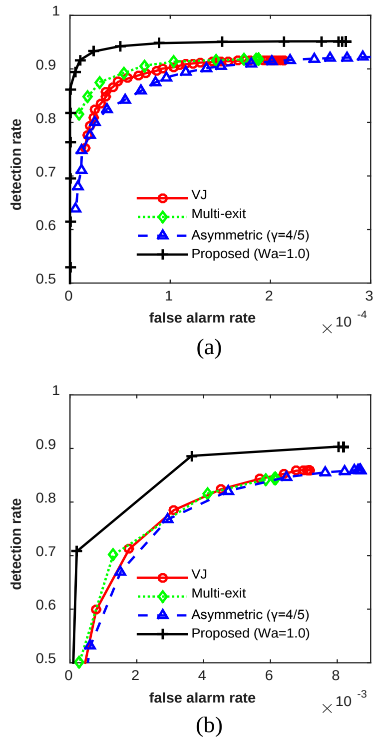

Figure 3. ROC curves of classifier cascades. (a) Face, (b) pedestrian.

5. CONCLUSIONS

We propose a weight adjustment strategy to achieve an improved performance in cascade learning. Cascade learning often underachieves the detection rate goal even when it overachieves the false alarm rate goal due to overfitting. In the proposed strategy, the weight ratio of positive examples to negative examples is computed to evaluate the imbalance between the detection rate and the false alarm rate, and it is adjusted to be balanced to prevent cascade learning from overfitting.

Since both the detection rate and the false alarm rate definitely decrease as more classifiers are appended to the classifier cascade, maintaining a higher detection rate is far more important than achieving a lower false alarm rate in cascade learning. By adopting the proposed strategy, an improved performance can be achieved by preventing the degradation in the detection rate at later stages, which often occurs in cascade learning.

Experimental results confirm the effectiveness of the proposed strategy. For experiments, face and pedestrian classifier cascades were trained by employing previous approaches and the proposed strategy. Even though each classifier cascade was trained to achieve the same performance goal with the same set of training examples and the same set of features, the performance of the classifier cascades employing the proposed strategy is significantly improved.

6. ACKNOWLEDGMENTS

This work was supported by the Center for Integrated Smart Sensors funded by the Ministry of Science, ICT & Future Planning as Global Frontier Project (CISS-2013M3A6A6073718).

7. REFERENCES

[Bou05] Bourdev, L. and Brandt, J., Robust object detection via soft cascade. In CVPR, vol. 2, pp. 236–243, 2005.

[Cev13] Cevikalp, H., Triggs, B., and Franc, V., Face and landmark detection by using cascade of classifiers. in FG, pp. 1–7, 2013.

[Che11] Cheng, W. C. and Jhan, D. M., A cascade classifier using Adaboost algorithm and support vector machine for pedestrian detection. in SMC, pp. 1430–1435, 2011.

[Cui10] Cui, J., Liu, F., Li, Z., and Jia, Z., Vehicle localisation using a single camera. in IV, pp. 871–876, 2010.

[Dol09] Dollar, P., Tu, Z., Perona, P., and Belongie, S., Integral channel features. in BMVC, pp. 1–11, 2009.

[Dol12] Dollar, P., Wojek, C., Schiele, B., and Perona, P., Pedestrian detection: an evaluation of the state of the art. TPAMI, vol. 34, no. 4, pp. 743–761, 2012.

[Dud01] Duda, R. O., Hart, P. E., and Stork, D. G., Pattern classification. 2nd ed., John Wiley & Sons, Inc., 2001.

[Enz09] Enzweiler, M. and Gavrilla, D. M., Monocular pedestrian detection: survey and experiments. TPAMI, vol. 31, no. 12, pp. 2179–2195, 2009.

[Fre97] Freund, Y. and Schapire, R. E., A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting. J. Comp. Syst. Sci., vol. 55, pp. 119–139, 1997.

[Fri00] Friedman, J., Hastie, T., and Tibshirani, R., Additive logistic regression: a statistical view of boosting. Ann. Stat., vol. 28, no. 2, pp. 337–407, 2000.

[Hoa12] Hoang, V., Vavilin, A., and Jo, K. H., Pedestrian detection approach based on modified Haar-like features and AdaBoost. In ICCAS, pp. 614–618, 2012.

[Hua07] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E., Labeled faces in the wild: a database for studying face recognition in unconstrained environments. University of Massachusetts, Amherst, Technical Report 07–49, 2007.

[Kel11] Keller, C., Enzweiler, M., and Gavrila, D. M., A new benchmark for stereo-based pedestrian detection. in IV, pp. 691–696, 2011.

[Lan12] Landesa-Vázquez, I. and Alba-Castro, J. L., Shedding light on the asymmetric learning capability of AdaBoost. Pattern Recognit. Lett., vol. 33, no. 3, pp. 247–255, 2012.

[Li11] Li, J., Wang, T., and Zhang, Y., Face detection using SURF cascade. in ICCV Workshops, pp. 2183–2190, 2011.

[Liu12] Liu, S., Dong, Y., Liu, W., and Zhao, J., Multi-view face detection based on cascade classifier and skin color. in CCIS, vol. 1, pp. 56–60, 2012.

[Lud13] Ludwig, O., Nunes, U., Ribeiro, B., and Premebida, C., Improving the generalization capacity of cascade classifiers. IEEE Trans. Cybernetics, vol. 43, no. 6, pp. 2135–2146, 2013.

[Mas07] Masnadi-Shirazi, H. and Vasconcelos, N., Asymmetric boosting. in ICML, pp. 609–619, 2007.

[Pai14] Paisitkriangkrai, S., Shen, C., and van den Hengel, A., Asymmetric pruning for learning cascade detectors. IEEE Trans. Multmed., vol. 16, no. 5, pp. 1254–1267, 2014.

[Pha08] Pham, M.-T., Hoang, V.-D. D., and Cham, T.-J., Detection with multi-exit asymmetric boosting. in CVPR, pp. 1–8, 2008.

[Pri13] Prioletti, A., Møgelmose, A. Grisleri, P., Trivedi, M. M., Broggi, A., and Moeslund, T. B., Part-based pedestrian detection and feature-based tracking for driver assistance: real-time, robust algorithms, and evaluation. IEEE Trans. Intelli. Transport. Syst., vol. 14, no. 3, pp. 1346–1359, 2013.

[Sab12] Saberian, M. J. and Vasconcelos, N., Learning optimal embedded cascades. TPAMI, vol. 34, no. 10, pp. 2005–2018, 2012.

[Siv12] Sivaraman, S. and Trivedi, M. M., Real-time vehicle detection by parts for urban driver assistance. in ITSC, pp. 1519–1524, 2012.

[Sun98] Sung, K.-K. and Poggio, T., Example-based learning for view-based human face detection. TPAMI, vol. 20, no. 1, pp. 39–51, 1998.

[Sun07] Sun, Y., Kamel, M. S., Wong, A., and Wang, Y., Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit., vol. 40, no. 12, pp. 3358–3378, 2007.

[Sun14] Sun, C., Hu, J., and Lam, K. M., A biased selection strategy for information recycling in boosting cascade visual-object detectors. Pattern Recognit. Lett., vol. 40, pp. 11–18, 2014.

[Vio01] Viola, P. and Jones, M., Fast and robust classification using asymmetric AdaBoost and a detector cascade. NIPS 14, MIT Press, pp. 1311–1318, 2001.

[Vio04] Viola, P. and Jones, M., Robust real-time face detection. IJCV, vol. 57, no. 2, pp. 137–154, 2004.

[Wan12] Wang, P., Shen, C., Barnes, N., and Zheng, H., Fast and robust object detection using asymmetric totally corrective boosting. IEEE Trans. Neural Networks Learn. Syst., vol. 23, no. 1, pp. 33–46, 2012.

[Wol11] Wolf, L., Hassner, T. and Taigman, Y., Effective face recognition by combining multiple descriptors and learned background statistics. TPAMI, vol. 33, no. 10, pp. 1978–1990, 2011.

[Wu04] Wu, B., Ai, H., Huang, C., and Lao, S., Fast rotation invariant multi-view face detection based on real AdaBoost. in AFGR, pp. 79–84, 2004.

[Wu08] Wu, J., Brubaker, S. C., Mullin, M. D., and Rehg, J. M., Fast asymmetric learning for cascade face detection. TPAMI, vol. 30, no. 3, pp. 369–382, 2008.

[Xia03] Xiao, R., Zhu, L., and Zhang, H., Boosting chain learning for object detection. in ICCV, vol. 1, pp. 709–715, 2003.

[Xin11] Xin, Y., Xiaosen, S., and Li, S., A combined pedestrian detection method based on Haar-like features and HOG features. in ISA, pp. 1–4, 2011.

 ISBN 978-80-86943-65-7