Directed Reading: Boosting algorithms

Guillaume Lemaître, Miroslav Radojević
Heriot-Watt University, Universitat de Girona, Universite de Bourgogne

December 21, 2009

Abstract

This work gives an overview of the classification methods based on *boosting*. The whole new concept of classifying data using boosting algorithm has evolved from basic principle idea of applying classifier to training data sequentially and weighting items that were wrongly classified as more important ones for the next iteration. This means that boosting performs supervised learning and by using the set of weak learners creates the powerful one. With pioneering work of Discrete AdaBoost, the whole family of algorithms has been developed and successfully applied, being available on commercial cameras today as face detection feature or implemented for applications such as real-time tracking, or various data mining software.

1 Introduction

Boosting as method is not constrained with usage of one specific algorithm. It is known as machine learning meta-algorithm. Common pattern for most boosting algorithms consists of learning weak classifiers[1] so that they become a part of a powerful one. Many boosting algorithms have been proposed. The essential one and historically the most important is the work of Robert Schapire [29] and Yoav Freund [13] introduced at the very beginning, in *Methods* section. Their work was the first provable boosting algorithm. It consisted of calling weak learner three times on three modified distributions, which caused boost in accuracy. Distributions were modified according to classification results, with emphasis on those elements that were misclassified. The idea of successively applying classifiers on the most informative data had not yet introduced adaptive behaviour, but it was a milestone. Many variations came later, usually bringing new understanding to the basis that was previously made, by introducing new learning algorithms and new hypotheses. AdaBoost (AdaptiveBoosting) was the first adaptive. It became popular and significant since it was the first one that used feedback information about the quality of the chosen samples so that it focused more on difficult, informative cases. Further development brings us to algorithms such as *LPBoost*, *TotalBoost*, *BrownBoost*, *GentleBoost*, *LogitBoost*, *MadaBoost*, *RankBoost*. These algorithms will be briefly introduced in *Methods* with their main features and ideas. Section *Boosting algorithm applications* will deal with some real-life implementations of presented methods. Indeed, *boosting* methods are commonly used to detect ob-

[1] classifiers that misclassified less than 50% samples

jects or persons in video sequences. The application the most famous was implemented by Viola and Jones and allowing to detect faces [32]. This application is usually used in videoconference, security system, etc. Section *Comparation* brings out and examines differences or similarities between properties of some algorithms. Last section concludes the story of boosting algorithms and the new ideas they contributed.

2 Boosting History - method backgrounds

Several methods of estimating have preceded boosting approach. Common feature for all methods is that they work out by extracting samples of a set, calculating the estimate for each drawn sample group repeatedly and combining the calculated results into unique one. One of the ways, the simplest one, to manage estimation is to examine the statistics of selected available samples from the set and combine the results of calculation together by averaging them. Such approach is a *jack-knife* estimation, when one sample is left out from the whole set each time to make an estimation [12]. Obtained collection of estimates is averaged afterwards to give the final result. Another, improved method, is *bootstrapping*. Bootstrapping repeatedly draws certain number of samples from the set and processes calculated estimations by averaging, similar to jack-knife [12]. *Bagging* is the further step towards boosting. This time, samples are drawn with replacement and each draw has a classifier C_i attached to it, so that final classifier becomes a weighted vote of $C_i - s$.

Essential Boosting idea is combining together basic rules, creating an ensemble of rules with better overall performance than the individual performances of the ensemble components. Each rule can be treated as a hypothesis, a classifier. Moreover, each rule is weighted so that it is appreciated according to its performance and accuracy. Weighting coefficients are obtained during the boosting procedure which, therefore, involves learning.

Mathematical roots of Boosting originate from probably approximately correct learning (PAC learning) [31, 23]. Boosting concept was applied for real task of optical character recognition using neural networks as base learners [25] . Recent practical implementation focuses on diverse fields, giving answers to questions such as tumor classification [6] or assessment whether household appliances consume energy or not [25].

3 Methods

Boosting method uses series of training data, with weights assigned to each training set. Series of classifiers are defined so that each of them is tested sequentially comparing the result of the previous classifier and using the results of previous classification to concentrate more on misclassified data. All the classifiers used are voted according to accuracy. Final classifier, combines weight of the votes of each classifier from the test sequence[22].

Two important ideas have contributed development of Boosting algorithms' robustness. First tries to find the best possible way to modify the algorithm so that its weak classifier produces more useful and more effective prediction results. Second tries to improve the design of a weak classifier. Answers to both concepts result in a large family of boosting methods[30]. Relations between two concepts of optimization and Boosting procedures have been a basis for establishing hew types of Boosting algorithms.

3.1 Basic methods

3.1.1 *Discrete AdaBoost*

Discrete AdaBoost (Adaptive Boost) algorithm takes training data and defines weak classifier functions for each sample of training data. A tree-based classifier has been thoroughly explored and proved to be the one that outcomes low error rates [20]. Classifier function takes the sample as argument and produces value -1 or 1 in case of a binary classification task and a constant value - weight factor for each classifier. Procedure trains the classifiers by giving higher weights to those training sets that were misclassified. Every classification stage contributes with its weight coefficients, making a collection of stage classifiers whose linear combination defines the final classifier [20]. Each training pattern receives a weight that determines its probability of being selected as a training set for an individual component. Inaccurately classified patterns are likely to be used again. The idea of accumulating weak classifiers means adding them so that each time the adding is done, they get multiplied with new weighting factors, according to distribution and relating to the accuracy of classification. At first this was proposed to be without adapting. Discrete AdaBoost or just AdaBoost was the first one that could change weak learners [20].

Early works on this topic have proposed the misconception that AdaBoost has its test error always decreasing with more classifiers added, meaning it is immune to over-fitting, hence it cannot be over-trained so that it starts increasing classification error once. Experiments [21, 27], though, exposed overfitting effects on datasets containing high level of noise. Generally, AdaBoost has shown good performance at classification. Bad feature of Adaptive Boosting is its sensitivity to noisy data and outliers. Boosting has a feature of reducing variance and bias, and a major cause of boosting success is variance reduction.

3.1.2 *RealBoost*

The creators of boosting concept have developed a general version of AdaBoost, which changes the way of expressing predictions. Instead of Discrete AdaBoost classifiers producing -1 or 1, a RealBoost classifiers produce real values. The sign of classifier output value defines which class the element belongs to. Those real values produced by classifier will serve as measure of how confident in prediction we are, so that classifiers implemented later can learn from their predecessors. Difference is that with real value, confidence can be measured instead of having just the discrete value that expresses classification result.

3.2 Weight function modification

3.2.1 *GentleBoost*

GentleBoost algorithm represents modified version of the Real AdaBoost algorithm. It is using adaptive Newton steps in the same manner like later introduced LogitBoost algorithm. The function that assigns weight for each sample in Real AdaBoost [14] is the following:

$$e^{-(r(x,y))} \tag{1}$$

where $r(x, y) = h(x)y$ and:

$$h(x) = \sum_i \ln \frac{1 - \epsilon_i}{\epsilon_i} \tag{2}$$

where ϵ_i is the weighted error of h_i. Minimization of function (1) is achieved using adaptive Newton steps. Real AdaBoost used formula

$$f_m(x) = \frac{1}{2} \log \frac{P_w(y = 1|x)}{P_w(y = -1|x)} \tag{3}$$

for updating the functions. Values obtained from outliers, using logarithm 3 can be unpredictably high, causing large updates. The consequence of this ponderation method is that the increasing number of misclassified samples, causes very fast increase of weight, without boundaries [15]. Friedman et al. introduce a derivated algorithm of Real AdaBoost to create GentleBoost algortihm [19]. The purpose is to make the previous function "gentler" [15]. GentleBoost updates the function using $f_m(x) = P_w(y = 1|x) - P_w(y = -1|x)$ formula with estimated weighted class probabilities. This way, function update stays in a limited range. GentleBoost allows to increase performance of classifier and reduce computation by 10 to 50 times compared to Real AdaBoost [19]. This algorithm usually outperforms Real AdaBoost and LogitBoost at stability.

3.2.2 *MadaBoost*

Domingo and Wanatabe propose a new algorithm, MadaBoost, which is a modification of AdaBoost [10]. Indeed, AdaBoost introduces two main disadvantages. First, this algorithm cannot be used by filtering framework [16]. Filtering framework allows to remove several parameters in boosting methods [34]. Second, AdaBoost is very sensitive to noise [16]. MadaBoost resolves the first problem by limiting the weight of examples with their initial probability. Moreover, filtering framework allows to resolve the problem of noise sensitivity [10]. With AdaBoost, weight of misclassified samples increases until samples are correctly classified [14]. Weighting system in MadaBoost is different. Indeed, variance of sample weights is moderate [10]. MadaBoost is resistant to noise and can progress in noisy environment [10].

3.3 Adaptive *"Boost by majority"*

3.3.1 *BrownBoost*

AdaBoost is a very popular method. However, several experimentations have shown that AdaBoost algorithm is sensitive to noise during the training [8]. To fix this problem, Freund introduced a new algorithm named BrownBoost [16] which makes changing of the weights smooth and still retains PAC learning principles.

BrownBoost refers to *Brownian motion* which is a mathematical model to describe random motions [2]. The method is based on boost by majority, combining many weak learners simultaneously, hence improving the performance of simple boosting [15] [14]. Basically, AdaBoost algorithm focuses on training samples that are misclassified [18]. Hence, the weight given to the outliers is larger than the weight of the good training samples. Unlike AdaBoost, BrownBoost allows to ignore training samples which are frequently misclassified [16]. Thus, this classifier created is trained with non-noisy training dataset [16]. BrownBoost is more performant than AdaBoost on noisy training dataset. Moreover, more training dataset becomes noisy, more BrownBoost classifier created becomes accurate compared to AdaBoost classifier.

3.4 Statistical interpretation of adaptive boosting

3.4.1 *LogitBoost*

LogitBoost is a boosting algorithm formulated by Jerome Friedman, Trevor Hastie, and Robert Tibshirani [19]. It introduces a statistical interpretation to AdaBoost algorithm by using additive logistic regression model for determining classifier in each round. Logistic regression is a way of describing the relationship between one

or more factors, in this case - instances from samples of training data, and an outcome, expressed as a probability. In case of two classes, outcome can take values 0 or 1. Probability of an outcome being 1 is expressed with logistic function. The LogitBoost algorithm uses Newton steps for fitting an additive symmetric logistic model by maximum likelihood [19]. Every factor has a coefficient attached, expressing its share in output probability, so that each instance is evaluated on its share in classification. LogitBoost is a method to minimize the logistic loss, AdaBoost technique driven by probabilities optimization. This method requires care to avoid numerical problems. When weight values become very small, which happens in case probabilities of outcome become close to 0 or 1, computation of the working response can become inconvenient and lead to large values. In such situations, approximations and threshold of response and weights are applied.

3.5 *"Totally-corrective"* algorithms

3.5.1 *LPBoost*

LPBoost is based on Linear Programming [19]. The approach of this algorithm is different compared to AdaBoost algorithm. LPBoost is a supervised classifier that maximizes margin of training samples between classes. Classification function is a linear combination of weak classifiers, each weighted with value that is adjustable. The optimal set of samples is consisted of a linear combination of weak hypotheses which perform best under worst choice of misclassification costs [4]. At first, LPBoost method was disregarded due to large number of variables, however, efficient methods of solving linear programs were discovered later. Classification function is formed by sequentially adding a weak classifier at every iteration and every time a weak classifier is added, all the weights of the weak classifiers present in linear classification function are adjusted (*totally-corrective* property). Indeed, in this algorithm, we update the cost function after each iteration [4]. The result of this point of view is that LPBoost converge to a finite number of iterations and need less iterations than AdaBoost to converge [24]. However, computation cost of this method is more expensive than AdaBoost [24].

3.5.2 *TotalBoost*

General idea of Boosting algorithms, maintaining the distribution over a given set of examples, has been optimized. A way to accomplish optimization for TotalBoost is to modify the way measurement of hypothesis' goodness, γ (*edge*) is being constrained through iterations. AdaBoost constrains the edge with the respect to the last hypothesis to maximum zero. Upper bound of the edge is chosen more moderately whereas LPBoost, being a totally-corrective algorithm too always chooses the least possible value[33]. An idea that was introduced in works of Kivinen and Warmuth (1999) is to constrain the edges of all past hypotheses to be at most $\gamma_{adapted}$ and otherwise minimize the relative entropy to the initial distribution. Such methods are called *totally-corrective*. TotalBoost method is "totally corrective", constraining the edges of all previous hypotheses to to maximal value that is properly adapted. It is proven that, with adaptive edge maximal value, measurement of confidence in prediction for a hypothesis weighting increases[33]. Compared with simple boost algorithm that is totally corrective, LPBoost, TotalBoost regulates entropy and moderately chooses γ which has led to significantly less number of iterations [33], helpful feature for proving iteration

bounds.

3.6 *RankBoost*

RankBoost is an efficient boosting algorithm for combining preferences [17] solves the problem of estimating rankings or preferences. It is essentially based on pioneering AdaBoost algorithm introduced in works of Freund and Schapire (1997) and Schapire and Singer (1999). The aim is to approximate a target ranking using already available ones, considering that some of those will be weakly correlated with the target ranking. All rankings are combined into a fairly accurate single ranking, using RankBoost machine learning method. The main product is an ordering list of the available objects using preference lists that are given.

Being a Boosting algorithm, defines RankBoost as a method that works in iterations, calls a weak learner that produces ranking each time, and a new distribution that will be passed to the next round. New distribution gives more importance to the pairs that were not ordered appropriately, placing emphasis on following weak learner to order them properly.

4 Applications

Boosting methods are used in different applications.

4.1 Faces Detection

The most famous application of boosting in image processing is detection of faces. Jones and Viola implemented a method for real-time detection of faces on video sequences [32]. Jones and Viola uses AdaBoost algorithm to classify features obtain Haar Basis functions [32]. The rate of the detector is about 15 frames by second [32]. This rate corresponds to a webcam rate. Hence, this detector is a real-time detector. Moreover, this method is 15 times faster than Rowley-Baluja-Kanade detector [28] which is a famous method of face detection using neural network. This speed allows to implement this method directly in hardware. Recently, Khalil Khattab et al. implemented this method using FPGA hardware [11].

4.2 Classification of Musical Genre

Two methods using boosting classification exist to classify songs in different musical genre like Classical, Electronic, Jazz & Blues, Metal & Punk, Rock & Pop, and World. The first method uses AdaBoost classifier [1] while the second method uses LPBoost classifier [7].

4.2.1 Music classification using AdaBoost

Bergstra and al. suggest a method using AdaBoost to classify music [1]. The principle is to find features, before using the classifier. These features are:

- Fast Fourier Transform Coefficients
- Real Cepstral Coefficients
- Mel Frequency Cepstral Coefficients
- Zero Crossing Rate
- Spectral Spread
- Spectral Centroid
- Spectral Colloff
- Autoregression

AdaBoost is used to classify music with the previous features. Result of the classification on the *Magnatune 6* dataset is 61.3% of good classification compared to the human classification [7]. The number of weak classifiers computed during the training period is 10000 [7].

4.2.2 Music classification using LPBoost

Diethe et al. propose a method using LPBoost to classify music [7]. Features used to allow the classification are:

- Discrete short-term Fourier Transform
- Real Cepstral Coefficients
- Mel Frequency Cepstral Coefficients
- Zero Crossing Rate
- Spectral Spread
- Spectral Centroid
- Spectral Rolloff
- Autoregression

These features are identical to the features used by Bergstra and al. [1]. The difference is the version of boosting algorithm used. Indeed, Diethe et al. used LPBoost to perform the classification. Result on the same dataset as Bergstra, outcomes percentage of good classification of 63.5% [7]. The number of weak classifiers computed during the training period is 585[7]. This number is smaller than the number in AdaBoost version because the principle of LPBoost is that during the training period, LPBoost converge faster than AdaBoost.

4.3 Real-Time Vehicle Tracking

Withopf et al. suggest using GentleBoost to detect and track vehicle in video sequence [35]. Features used to allow the classification are the same used by Viola and Jones for faces detection [32]. Indeed, Haar Basis function are used to find features [35]. Then, GentleBoost is implemented to classify each object on a video sequence like car or no car [35]. Withopf et al. compared results on the same video sequences of boosting method (GentleBoost) with two different other methods which are difference of edges features and trained object tracker [35]. Classification using GentleBoost is more accurate than the obtained using other methods [35].

4.4 Tumor Classification With Gene Expression Data

Dettling et al. propose an algorithm using LogitBoost to classify tumors [5]. Before computing the LogitBoost algorithm, Dettling et al. did a feature selection [5]. Finally, Dettling et al. compared results with a simple AdaBoost algorithm and LogitBoost algorithm [5]. The combination of LogiBoost and features selection gives better results with a better accuracy than AdaBoost [5].

4.5 Film ranking

Example of implementation of RankBoost algorithm [17] can be an algorithm that chooses the list of person's favourite films according to the selection, feedback received during learning process and preferences. Such example suggests whole family of useful applications, especially web interaction based ones. To adjust the method so that it's results can be numerically interpreted films have to be ranked - meaning that

each one gains ordinal number and that the additional tabular information describing numerically the desirable sequence between each instance (film). Tabular information is the one that serves as a source for feedback and decision how similar and qualitative the estimated ranking is. Similarity is measured using criteria function. Criteria function is evaluated as weighted number of disordered pairs in estimated ranking, compared with obtained feedback [17]. RankBoost can be useful in different machine learning problems, even those that do not look like the ones that are be related to ranking, such as sentence-generation system [26] or automati analysis of human language[3].

4.6 Meta-search problem

Useful illustration of ranking using RankBoost [17] is meta-search problem, a task developed by Cohen, Schapire and Singer (1999). Meta-search problem refers to learning a strategy that, takes a query as an input, and generates the ranking of URLs connected with the query positioning those that seem to be more appropriate to the top - quite useful and common concept in everyday usage of internet.

5 Comparison

Boosting algorithms have been compared with other algorithms that share affinities. It is convenient to examine features and originalities of each boosting approach. Overview of strengths and weaknesses of different boosting solutions presented in this section are provided in Table 1.

5.1 GentleBoost

Gentle Boost, as a moderate version of Real AdaBoost and LogitBoost algorithms, shares similar performance with them, even outperforming them considering robustness.

5.2 MadaBoost

Initial probability bounded weight of each instance at MadaBoost changes moderately compared to AdaBoost and the boosting property stays similar to AdaBoost, according to accomplished experiments [10].

5.3 BrownBoost

The cause for AdaBoost noise sensitivity is explained with assigning high weights to noisy examples [9] and over-fitting the noise. BrownBoost tends to isolate noisy data from training set, therefore improving noise robustness compared to AdaBoost.

5.4 LPBoost

LPBoost showed better classification quality and faster solution than AdaBoost [4]. Compared with gradient based methods, LPBoost shows many improvements: finite termination at a globally optimal solution, optimality driven convergence, speed of execution, less weak hypotheses in optimal ensemble [4].

5.5 *Totally-corrective* algorithms

Unlike AdaBoost algorithms where the same hypothesis can be chosen many times, LPBoost and TotalBoost select a base hypothesis once so that the edge of hypothesis affects distribution management afterwards. *Totally-corrective* algorithms need less hypotheses when there are many redundant features[33], but demand more computation.

Method	Pros	Cons
Discrete Ada Boost	simple; adaptive; test error consistently decreases as more classifiers are added; fairly immune to overfitting; decent iteration bound	sensitive to noisy data and outliers, cannot be used in boosting by filtering framework
Real Ada Boost	better suited for frameworks with histograms viewed as weak learners; converges faster than AdaBoost	sensitive to noisy data and outliners
Gentle Boost	increases performance of a classifier; reduce computation by 10 to 50 times	number of misclassified samples increases
Brown Boost	adaptive and uses "boost by majority" principle; performs better on noisy datasets	since the noisy examples may be ignored, only the true examples will contribute to the learning process
Logit Boost	good performance on noisy datasets	numerical problems when calculating z variable for logic regression
Mada Boost	one version of MadaBoost has an adaptive boosting property; works under filtering framework; resistant to some noise types due to belonging to statistical query model of learning [10]; improves accuracy	assumes edge is decreasing - advantages of the weak hypotheses are monotonically decreasing; boosting speed is slower than AdaBoost
Rank Boost	introduces usage of boosting algorithms for ranking; as it is a boosting algorithm (meta-algorithm), there is a possibility of combining different ranking algorithms together yielding a higher precision; effective algorithm for combining ranks	choice of weak learner defines algorithms ability to generalize successfully
LP Boost	has a possibility of minimizing misclassification error and maximizing a margin between training samples of different classes; fast convergence due to *totally-corrective* property; terminates at globally optimal solution, fast algorithm in general	more computation cost compared to AdaBoost; sensitive to in-correctness of the base learning algorithms; small amount of misclassification costs at the early stage can cause problems
Total Boost	fast convergence accomplished by minimizing entropy; suitable for small number of features selection; same iteration bound as AdaBoost	higher computation costs compared to AdaBoost

Table 1: Advantages and disadvantages of boosting methods

5.6 RankBoost

Performance of RankBoost on film preferences task has been compared with three other classification methods: a regression algorithm, a nearest-neighbour algorithm, a vector similarity algorithm. Regression method assumes linear combination of already existing scores for films is used for obtaining the scores for particular user selection. Nearest neighbour finds a viewer with the most similar preferences and suggests its preferences for particular user selection. Vector similarity takes two instances, expresses them as vector, and searches for vector differences. Values that measure disagreement, precision, average precision and predicted rank of top were used for as a criterion for performance comparison. RankBoost showed considerably better performance compared to regression and nearest neighbour for all four performance measures. RankBoost also outperformed vector similarity when the feature set size was larger. For medium and large feature sizes, RankBoost achieved the lowest disagreement and the highest average precision, predicted rank of top. RankBoost, according to its boosting feature, showed the highest potential of improving its performance with the increase of the number of features [17].

6 Conclusion

The progress of boosting machine learning algorithms presented in this overview showcases the original approach to classification, its variations, improvements and application. It is clear that milestone method, AdaBoost, has become a very popular algorithm to use in practise. It emerged to have plenty of versions, each giving different contribution to algorithm performance. It has been interpreted as a procedure based on functional gradient descent (AdaBoost), as an approximation of logistic regression (LogitBoost), or enhanced with arithmetical improvements of calculation of weight coefficients (GentleBoost and MadaBoost). It was connected with linear programming (LPBoost), Brownian motion (BrownBoost), entropy based methods for constraining hypothesis goodness (TotalBoost). Finally, boosting was used for such implementations as ranking the features (RankBoost). Boosting principle or some of its features, was improved with an innovative solution for each method. Depending on method, that could refer to additional equation, equation modification or different approach to solving optimization. Presented development has improved the knowledge and understanding of boosting, opening many possibilities for involvement of boosting in solving diverse and attractive practical problems like classification, tracking, complex recognition or comparation.

References

[1] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and adaboost for music classification. *Mach. Learn.*, 65(2-3):473–484, 2006.

[2] Robert Brown. A brief account of microscopical observations made in the months of june, july and august, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies. No note, 1828.

[3] Michael Collins. Discriminative reranking for natural language parsing. In *Proc. 17th International Conf. on Machine Learning*, pages 175–182. Morgan Kaufmann, San Francisco, CA, 2000.

[4] Ayhan Demiriz, Kristin P. Bennett, and John S. Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002.

[5] M. Dettling and P. Bhlmann. Boosting for tumor classification with gene expression data. *bioinformatics*, Vol. 19 no. 9:1061 – 1069, 2003.

[6] Marcel Dettling and Peter Bühlmann. Finding predictive gene groups from microarray data. *J. Multivar. Anal.*, 90(1):106–131, 2004.

[7] T. Diethe and J. Shawe-Taylor. Linear programming boosting for classification of musical genre. Technical report, Presented at the NIPS 2007 workshop Music, Brain & Cognition, 2007.

[8] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. In *Bagging, boosting, and randomization. Machine Learning*, pages 139–157, 1998.

[9] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. In *Bagging, boosting, and randomization. Machine Learning*, pages 139–157, 1998.

[10] Carlos Doming and Osamu Watanabe. Madaboost: A modification of adaboost. In *Proc. of ACM 13th Annual Conference on Computational Learning Theory*, 2000.

[11] Khalil Khattab Julien Dubois and Johel Miteran. Cascade boosting-based object detection from high-level description to hardware implementation. *EURASIP Journal on Embedded Systems*, Article ID 235032:12, 2009.

[12] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience Publication, 2000.

[13] Yoav Freund. Boosting a weak learning algorithm by majority. In *COLT '90: Proceedings of the third annual workshop on Computational learning theory*, pages 202–216, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[14] Yoav Freund. Boosting a weak learning algorithm by majority. *Inf. Comput.*, 121(2):256–285, 1995.

[15] Yoav Freund. An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3):293–318, 2001.

[16] Yoav Freund. An adaptive version of the boost by majority algorithm. *Mach. Learn.*, 43(3):293–318, 2001.

[17] Yoav Freund, Raj Iyer, Robert E. Schapire, Yoram Singer, and G. Dietterich. An efficient boosting algorithm for combining preferences. In *Journal of Machine Learning Research*, pages 170–178, 2003.

[18] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55:119–139, 1996.

[19] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[20] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Special invited paper. additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–374, 2000.

[21] Adam J. Grove and Dale Schuurmans. Boosting in the limit: maximizing the margin of learned ensembles. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 692–699, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[22] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

[23] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994.

[24] Jure Leskovec and John Shawe-Taylor. Linear programming boosting for uneven datasets. In *ICML*, pages 456–463, 2003.

[25] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. pages 118–183, 2003.

[26] Owen Rambow, Monica Rogati, and Marilyn A. Walker. Evaluating a trainable sentence planner for a spoken dialogue system. In *ACL*, pages 426–433, 2001.

[27] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for adaboost. *Mach. Learn.*, 42(3):287–320, 2001.

[28] Henry Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. In *Computer Vision and Pattern Recognition '96*, June 1996.

[29] Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, 1990.

[30] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions, 1999.

[31] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[32] Paul Viola and Michael J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, 2004.

[33] Manfred K. Warmuth, Jun Liao, and Gunnar Rätsch. Totally corrective boosting algorithms that maximize the margin. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 1001–1008, New York, NY, USA, 2006. ACM.

[34] O. Watanabe. Algorithmic aspects of boosting, 2002.

[35] D. Withopf and B. Jhne. Learning algorithm for real-time vehicle tracking. *IEEE Intelligent Transportation Systems Conference*, 1-4244-0094-5:516–521, 2006.