International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:9, 2008

Performance Comparison and Evaluation of AdaBoost and SoftBoost Algorithms on Generic Object Recognition

Doaa Hegazy, Joachim Denzler

***Abstract*—SoftBoost is a recently presented boosting algorithm, which trades off the size of achieved classification margin and generalization performance. This paper presents a performance evaluation of SoftBoost algorithm on the generic object recognition problem. An appearance-based generic object recognition model is used. The evaluation experiments are performed using a difficult object recognition benchmark. An assessment with respect to different degrees of label noise as well as a comparison to the well known AdaBoost algorithm is performed. The obtained results reveal that SoftBoost is encouraged to be used in cases when the training data is known to have a high degree of noise. Otherwise, using Adaboost can achieve better performance.**

***Keywords*– SoftBoost algorithm, AdaBoost algorithm, Generic object recognition.**

I. Introduction

Boosting methods and algorithms have been used successfully in many pattern recognition applications such as feature selection [12], face detection [14] and generic object recognition [7], [3]. AdaBoost is one of the most popular boosting algorithms. It generates a combined hypotheses with large margin and works well on data with low degree of noise [10], but not good on data with high degree of noise [15]. In such a case, a large margin on all training data cannot be achieved without affecting the generalization performance [10], [15]. Due to this reason, many variants of AdaBoost appeared to cope with this problem and to trade off the number of margin errors and the size of the margin. This is achieved by restricting the weighting maintained by the algorithm to not concentrate too much on the most difficult (hard to classify) examples [15]. Examples of these algorithms are AdaBoost with soft margin [8] and LPBoost [1].

On the other hand, and as previously mentioned, the hypotheses combination produced by AdaBoost has a large margin on the data. This margin is not necessarily the maximum hard margin. Therefore, many new versions of AdaBoost, which try to provide a maximum hard margin, have been developed such as AdaBoost* [9], TotalBoost [16], and many other algorithms [15]. However, such algorithms are not suitable for real-world applications with noisy data as over-fitting is more problematic for them than the original AdaBoost algorithm [15].

SoftBoost is a newly presented boosting algorithm [15], which combines the previously mentioned two lines of research in a single algorithm, that implements the soft margin idea in a practical boosting algorithm. To the best of our knowledge up to now, no evaluation of this new algorithm exist on a real world object recognition problem.

Therefore, the main objective of this paper is investigate the recognition performance of the SoftBoost algorithm on generic object recognition problem. An additional objective is to evaluate the performance by comparing it to the performance of AdaBoost algorithm using label noisy training data.

The used generic recognition model is an appearance-based one. It employs a combination of two different types of local descriptors (grayscale and color) for the recognition task. These two different descriptors are computed from interest regions extracted using an affine point detector. Learning these descriptors is performed in a weakly supervised manner using boosting.

The paper is organized as follows: in section 2, preliminaries of boosting are given and AdaBoost and SoftBoost are briefly explained . The generic object recognition model is described in section 3. Section 4 presents the evaluation experiments and the results obtained. Finally, conclusions are drawn in section 5.

II. Boosting Algorithms

A. Preliminaries of Boosting

In the boosting setting, a set of N labeled training examples (x_n, y_n) for $n = 1 \ldots N$ are given, where the instances x_n are in some domain χ and the labels $y_n \in \pm 1$. Boosting algorithms maintain a distribution $\mathbf{d}$ on the examples N such that the hard to classify examples receive more weight. In each iteration, the algorithm gives the current distribution to a weak learning algorithm (*weak learner*), which returns a new weak hypothesis $h : \chi \to [-1,1]^N$ with a certain guarantee of performance.

One measure of the performance of a weak hypothesis h with respect to distribution $\mathbf{d}$ is its *edge*, $\gamma_h = \sum_{n=1}^{N} d_n y_n h(x_n)$. When the range of h is ± 1 instead of the range of $[-1,1]$, the edge is just a affine transformation of the weighted error ε_h of hypothesis h: i.e. $\varepsilon_h(\mathbf{d}) = \frac{1}{2} - \frac{1}{2}\gamma_h$. A hypothesis that predicts perfectly has an edge $\gamma = 1$ while a hypothesis that always predicts incorrectly has an edge $\gamma = -1$. A random hypothesis has an edge $\gamma = 0$. The higher the edge, the more useful is the hypothesis for classifying the training examples. The edge of a set of hypotheses is defined as the maximum edge of the set.

Chair of Computer Vision, Institute of Computer Science, Friedrich-Schiller-University, Jena, Germany, hegazy@informatik.uni-jena.de, denzler@informatik.uni-jena.de.

After a hypothesis is received, the algorithm must update its distribution $\mathbf{d}$ on the examples. Boosting algorithms (for separable case) commonly update its distribution $\mathbf{d}$ by placing an edge constraint on the most recent hypothesis. Such algorithms are called *corrective* [9], [15]. In *totally corrective* algorithms, the distribution is updated to have a small edge with respect to all of the previous hypotheses [16], [15]. The final output of the boosting algorithm is always a convex combination of weak hypotheses $f_{\mathbf{w}}(x_n) = \sum_{t=1}^{T} w_t h^t(x_n)$, where h^t is the hypothesis added at iteration t and w_t is its coefficient [15]. The *hard margin* of a labeled examples (x_n, y_n) is defined as $\rho_n = y_n f_{\mathbf{w}}(x_n)$. The margin of the examples are taken to be the minimum margin of the set.

It is convenient to define a N-dimensional vector $\mathbf{u}^m$ that combines the weak hypothesis h^m with the label y_n of the N examples: $u_n^m = y_n h^m(x_n)$. With this notation, the edge of the weak hypothesis t-th hypothesis becomes $\mathbf{d}.\mathbf{u}_t$ and the margin of the n-th example with respect to a convex combination $\mathbf{w}$ of the first $t-1$ hypothesis is $\sum_{t=1}^{t-1} u_n^m w_m$ [15].

B. AdaBoost Algorithm

AdaBoost (Adaptive Boosting) [2] is the most well known boosting algorithm. It is adaptive in that the linear coefficient of the weak hypothesis depends on the weighted error of the weak hypothesis at the time when the weak hypothesis added to the linear combination. AdaBoost is also a *corrective* algorithm. It is observed experimentally that AdaBoost is prone to over-fitting when the training data contains high degree of noise. This is due to the fact that, by training, AdaBoost concentrates too much on outliers and hard to classify examples [10] which in turn affects the generalization performance.

AdaBoost with confidence-rated prediction [11] is used in the evaluations performed in this paper. It differs from the AdaBoost [2](also known as Discrete AdaBoost) that the weak learner of the first computes a weak hypothesis $h : \chi \to \mathbb{R}$. The sign of h is interpreted as the predicted label (-1 or +1) to be assigned to the instance x_i and the magnitude $|h(x)|$ as the confidence of this prediction. Moreover, the method of computing the coefficient of the weak hypothesis is different [11](see algorithm 1). Further details of the algorithm could be found in [11].

C. SoftBoost Algorithm

SoftBoost is a *totally corrective* algorithm which optimizes the *soft* margin and tries to produce a linear combination of hypotheses with the maximum one [15]. The term soft here means that the algorithm does not concentrate too much on outliers and hard to classify examples. It allows them to lie below the margin (i.e. to have wrong predictions) but penalizes them linearly via slack variables.

Therefore, it seems that SoftBoost avoids the problem of over-fitting exist in AdaBoost when using training data with high degree of noise. For the space restrictions, a brief description of the SoftBoost algorithm is given below. Further details about the algorithm could be found in [15].

SoftBoost takes as input, a sequence of examples $S = \langle (x_1, y_1), ..., (x_N, y_N) \rangle$ in addition to an accuracy parameter δ and a capping parameter ν (see algorithm 2). This capping parameter specifies how many examples could be mistrusted or, in other words, how many examples are allowed to lie below the margin. The algorithm has a weak learner which provides a hypothesis with an edge with a known guarantee g. The initial distribution $\mathbf{d}^0$ of the algorithm is uniform. In each iteration t, the algorithm prompts the weak learner for a new weak hypothesis, adds it into the constraints set, and updates its distribution $\mathbf{d}^{t-1}$ to $\mathbf{d}^t$ by minimizing the relative entropy $\Delta(\mathbf{d}, \mathbf{d}^0) := \sum_n \left(d_n \ln \frac{d_n}{d_n^0} \right)$ subject to the constraints:

$$\mathbf{d}^{t+1} = \operatorname*{argmin}_{\mathbf{d}} \Delta\left(\mathbf{d}, \mathbf{d}^0\right) \tag{1}$$

$$\text{s.t. } \mathbf{d} \cdot \mathbf{u}^m \leq g - \delta, \quad \text{for } 1 \leq m \leq t,$$

$$\sum_n d_n = 1, \ \mathbf{d} \leq \frac{1}{\nu}\mathbf{1}$$

Input: $S = \langle (x_1, y_1), ..., (x_N, y_N) \rangle$; $x_i \in \chi$, $y_i \in \{-1, +1\}$.
Initialize: $d_1(i)$ *to the uniform distribution.*
for $t = 1, ...T$: **do**
(a) Train weak learner using distribution $\mathbf{d}_t$.
(b) Get weak hypothesis $h_t : \chi \to \mathbb{R}$.
(c) Choose $\alpha \in \mathbb{R}$.
(d) Update:
$d_{t+1}(i) = \dfrac{d_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
where Z_t is a normalization factor(chosen so that d_{t+1} will be a distribution).
end
Output: Final hypothesis: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$.

Algorithm 1: AdaBoost with confidence-rated predictions [11].

III. Generic Object Recognition Model

The generic recognition model consists of three main steps. First, a set of interest regions is detected and extracted from the images using an affine point detector. Second, a combination of two different local descriptors (grayscale and color) is used to describe the extracted interest regions. Finally, learning is performed using a boosting algorithm. Figure 1 provides a semantic view of the main components of the recognition model.

A. Interest Regions Detection

An implementation of the Hessian affine-invariant point detector developed by [5] is used to detect and extract interest regions from the images. The main reason for using this detector is its robustness to view-point and scale changes as reported in [6].

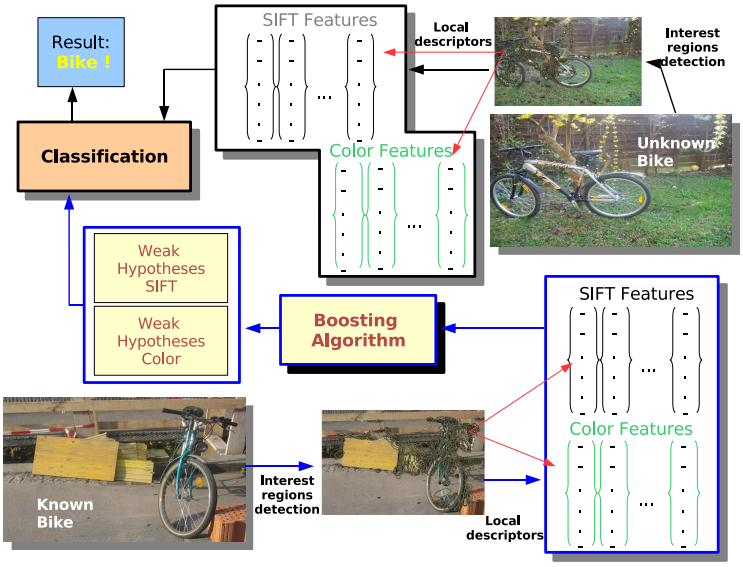

Fig. 1: The used generic recognition model

Input: $S = \langle (x_1, y_1), ..., (x_N, y_N) \rangle$, desired accuracy δ, and capping parameter $\nu \in [1, N]$.
Initialize: d_n^0 *to the uniform distribution.*
for $t = 1, ...$ **do**
(a) Send $\mathbf{d}^{t-1}$ and $\{\mathbf{u}_1, ..., \mathbf{u}_{t-1}\}$ to the weak learner and obtain hypothesis h^t which has edge at least g w.r.t. $\mathbf{d}^{t-1}$.
Set $u_n^t = h^t(x_n) y_n$.
(b) Update

$$\mathbf{d}^t = \underset{\mathbf{d}}{\operatorname{argmin}} \Delta(\mathbf{d}, \mathbf{d}^0)$$

$$\text{s.t. } \mathbf{d} \cdot \mathbf{u}^m \leq g - \delta, \quad \text{for } 1 \leq m \leq t,$$

$$\sum_n d_n = 1, \ \mathbf{d} \leq \frac{1}{\nu} 1$$

(c) If above infeasible or $\mathbf{d}^t$ contains a zero then $T = t - 1$ and break.
end
Output: $f_{\mathbf{W}}(x) = \sum_{m=1}^{T} \mathbf{W}_m h^m(x)$,where the coefficients $\mathbf{W}_m$ maximize the soft margin over the hypotheses set $\{h^1, ..., h^t\}$ using the LP problem (1) in [15].

Algorithm 2: SoftBoost with accuracy parameter δ and capping parameter ν [15].

B. Local Descriptors

As mentioned previously, a combination of two local descriptors is used.
SIFT descriptors [4]: are used as the grayscale descriptors. SIFT descriptors are 3D histograms of gradient locations and orientations, where locations are quantized into a 4x4 location grid and the gradient angle is quantized into 8 orientations. The resulting descriptor is of length 128.
Local color descriptor [13] : The authors in [13] introduced a set of local color descriptors with different criteria such as photometric robustness, geometric robustness, photometric stability and generality. Among those descriptors, we choose to use opponent angle color descriptors as they are robust with respect to both geometrical variations caused by changes in viewpoint, zoom and object orientations and photometric variations caused by shadows, shading and specularity. The opponent color descriptor depends on the angle between the opponent color derivatives and is of length 37 bins. Details about the construction of the opponent angle color descriptors are given in [13].

C. The Learning Model

In learning step, a classifier which predicts whether the image contains an instance of the object category or not is computed. Since more than one feature type is used for learning, each training image I is represented by a set of features $\{F_{i,j}(t_{i,j}, v_{i,j}), j = 1...n_i\}$ where n_i is the number of features in image I_i (which varies from image to another according to the number of interest points detected in the image), $t_{i,j}$ indicates the type of the feature (s for SIFT, c for color) and $\mathbf{v}_{i,j}$ is the feature vector. The boosting algorithm puts weights on the training images and requires construction of a weak hypothesis h_k which, relative to the weights, has discriminative power. The algorithm is run for a certain number of iterations (the actual number of iterations processed by the SoftBoost algorithm depends on the solution of the embedded optimization problem). In each iteration k, one feature of each type is selected as a weak classifier and the weights of the training images are updated. A linear combination of the weak

hypothesis together with their weights is used as a strong hypothesis to classify new images.

Weak learner
In each iteration, the weak learner selects the most discriminant feature vector for each different feature type. The most discriminant feature vector here is the one which has the maximum edge on the training data. This means that the weak learner selects two weak hypotheses in each iteration: h_k^s and h_k^c for SIFT and color features respectively. Each weak hypothesis consists of two components: a feature vector $\mathbf{v}_k^x$ and a similarity distance threshold θ_k^x (where $x = s$ or c for SIFT and color respectively). The threshold θ_k^x measures if an image contains a feature $\mathbf{v}_{i,j}$ that is similar to $\mathbf{v}_k^x$. The similarity between $\mathbf{v}_{i,j}$, which belongs to the image I_i, and $\mathbf{v}_k^x$ is measured using Euclidean distance. This similarity threshold is calculated as in [7].

IV. Experiments and Results

To investigate the recognition performance of SoftBoost algorithm, a set of experiments is performed using it as the base learning algorithm in the recognition model. Moreover, some experiments using AdaBoost algorithm are performed in order to establish a performance comparison and evaluation of both algorithms.

A. Experimental settings

The experimental settings are divided as follows:
- The used dataset: Graz02 database (http://www.emt.tugraz.at/∼ pinz/data/) is used in all evaluation experiments. Graz02 is a difficult dataset which consists of three generic classes: cars, bikes and persons, in addition to the background class (the counter-class).
- Training and Test images: For training, 300 examples (images) are used: 150 images of objects class and 150 images of counter-class (negative). For testing, 150 new examples are used: 75 images of objects class and 75 images of counter-class.
- SoftBoost algorithm: The value of guarantee g is set equal to γ^* in all experiments, where γ^* is the value of linear programming problem $P2$ presented in [15]. The value of accuracy parameter δ used in all experiments is 0.0001. This value is used based on performing experiments on bikes dataset, with values of $\delta \in \{0.1, 0.01, 0.001, 0.0001, 0.00001\}$. The best generalization performance is achieved at $\delta = 0.0001$.
- AdaBoost algorithm: The AdaBoost algorithm is run for a number of iterations T= 150. We conclude this number by experiments where T is varied from 10 to 300. After T = 150, the test error remains constant.

B. Experiments Using Noise Free Data

Using SoftBoost Algorithm: an important parameter of the SoftBoost algorithm is the capping parameter ν. It represents the number of training examples which are allowed to have wrong predictions in order to obtain high generalization performance. Therefore, the optimal value of ν should be selected. This is accomplished by using a 5-fold cross-validation for each object class. This results to three optimal values of ν, one for each object class. Training is performed afterwards using the selected values of ν and the generalization rates are given in table I.

Using AdaBoost Algorithm: Training the three object classes is performed using T iterations specified in section 4.1 and the resultant generalization rates are shown in table I. When comparing SoftBoost and AdaBoost, it is clear that AdaBoost outperforms SoftBoost in two of the three used object classes (cars and persons) while SoftBoost achieves higher generalization rates on the bikes class.
Concerning the general performance of the used recognition model, table II presents a comparison of the used recognition model to the recognition model presented in [7]. The model of [7] combines three point detectors with four different local descriptors for the recognition task. AdaBoost is also used as the underlaying learning algorithm. The results displayed in table II proves the generalization ability of our model in general. Although the learning algorithm is changed between SoftBoost and AdaBoost, the achieved results in both cases better than the results of [7].

TABLE I: Generalization rates of SoftBoost vs AdaBoost

Dataset	AdaBoost	SoftBoost	optimal ν
Bikes	76.00	80.00	40%
Cars	80.00	78.67	70%
Persons	84.00	82.67	30%

TABLE II: Recognition rates of our model using the Graz02 dataset compared to the results of Opelt [7].

Dataset	AdaBoost	SoftBoost	[7]
Bikes	76.00	80.00	77.80
Cars	80.00	78.67	70.5
Persons	84.00	82.67	81.2

C. Experiments Using Noisy Data

Actually, *noise* could be observed at various levels of abstraction in learning and recognition process, including high intra-class variability, partial occlusion, background clutter, varying illumination and added Gaussian noise to the test images. In fact, Graz02 database is already quite noisy in this respect. Additionally, more difficulties could be added to the training data by the presence of outliers or miss-labeled (label noisy) patterns. The label noise means that a pattern is clearly a member of one class and its label corresponds to the alternate class. The label noisy patters or examples cause the boosting to concentrate on them during training, which in turn, deteriorates the final hypothesis and thus the generalization performance of the algorithm.
In this set of experiments, label noise is added by assigning wrong labels to a percentage n of the training examples, where $n = 10, 30$ and 50% respectively. This means that three different degrees of label noise are given. The test examples

are left unchanged. For training using SoftBoost algorithm, a 5-fold cross-validation is used to select the optimal value of ν at each level of noise for each class. This results to the estimations of nine values of ν.
Table III presents the generalization rates of both AdaBoost and SoftBoost algorithms on this set of experiments. Generally, the performance of SoftBoost exceeds the performance of AdaBoost when the training data contains high degree of label noise (i.e. 50%). At small degrees of label noise (i.e.10% and 30%), AdaBoost achieves better performance.

TABLE III: Generalization rates of AdaBoost and SoftBoost using training data with different degrees of label noise.

Dataset	noise degree	AdaBoost	SoftBoost	optimal ν
Bikes	10%	78.67	74.67	70%
	30%	74.67	74.67	70%
	50%	60.00	58.67	30%
Cars	10%	77.33	70.67	30%
	30%	76.00	73.33	30%
	50%	60.00	68.00	50%
Persons	10%	82.67	82.67	40%
	30%	80.00	77.33	30%
	50%	52.00	66.67	50%

V. Conclusions

A performance investigation and evaluation of the SoftBoost algorithm on the generic object recognition problem has been presented. Moreover, a comparative analysis of the performance of SoftBoost and AdaBoost algorithms has been performed. Experiments on Graz02 dataset revealed that, SoftBoost has a good performance and almost the same as AdaBoost, if not lower in some cases, using noise free data. The term *noise free* means recognition using already difficult real data of object classes without imposing more difficulties (label-noise) on it. However, SoftBoost shows more robust performance than AdaBoost using training data with high degree of label noise.

References

[1] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002.
[2] Yoav. Freund and Robert E. Schapire. A decision theoretic generalization of online learning and application to boosting. *Computer and System Sciences*, 55(1):119–139, 1997.
[3] Doaa Hegazy and Joachim Denzler. Boosting local colored features for generic object recognition. *Pattern Recognition and Image Understanding*, 18:323–327, 2008.
[4] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.
[5] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *7th European Conference on Computer Vision ECCV02*, pages 128–142, 2002.
[6] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *Int. J. Comput. Vision*, 73(3):263–284, 2007.
[7] Andreas Opelt, Axel Pinz, Michael Fussenegger, and Peter Auer. Generic object recognirion with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):416–431, 2006.
[8] Gunnar. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, March 2001.
[9] Gunnar Rätsch and Manfred K. Warmuth. Efficient margin maximizing with boosting. *Journal of Machine Learning Research*, 6:2131–2152, 2002.
[10] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. In *Proc. 14th International Conference on Machine Learning*, pages 322–330. Morgan Kaufmann, 1997.
[11] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.
[12] Zehang Sun, George Bebis, and Ronald Miller. Boosting object detection using feature selection. page 290, 2003.
[13] Joost van de Weijer and Cordelia Schmid. Coloring Local Feature Extraction. In *8th European Conference on Computer Vision ECCV06*, volume 2, pages 334–348, 2006.
[14] Paul Viola and Michael Jones. Rapid object detection uisng a boosted cascade of simple features. In *IEEE Computer Scociety Conference on Computer Vision and Pattern Recognition CVPR01*, volume 1, pages 511–518, 2001.
[15] Manfred K. Warmuth, Karen Glocer, and Gunnar Rätsch. Boosting algorithms for maximizining the soft margin. *Advances in Neural Information Processing Systems (NIPS'08)*, 2008.
[16] 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.