Boosted Image Classification: An Empirical Study

Nicholas R. Howe
Smith College
nhowe@cs.smith.edu

May 30, 2002

Abstract

The rapid pace of research in the fields of machine learning and image comparison has produced powerful new techniques in both areas. At the same time, research has been sparse on applying the best ideas from both fields to image classification and other forms of pattern recognition. This paper combines boosting with state-of-the-art methods in image comparison to carry out a comparative evaluation of several top algorithms. The results suggest that a new method for applying boosting may be most effective on data with many dimensions. Effectively marrying the best ideas from the two fields takes effort, but the techniques and analyses developed herein make the task straightforward.

1. Introduction

Researchers have made gratifying progress recently in both machine learning and image comparison. Boosting (Freund & Schapire, 1996), support vector machines (Burges, 1998) and other so-called *large-margin techniques* consistently demonstrate improved performance when applied on top of older, more established machine learning methods. Simultaneously, researchers in the field of image retrieval have devised new representations that allow quick comparisons between images based upon multiple cues – color and texture distributions, for example. Techniques such as color correlograms (Huang et al., 1997), redundant banks of texture filters (De Bonet & Viola, 1997), and others (Howe & Huttenlocher, 2000) have shown measurable improvements over earlier, more simplistic methods such as color histograms (Swain & Ballard, 1991). These two bodies of research combined have the potential to generate powerful pattern recognition algorithms, particularly in the area of image classification. Unfortunately, with a few exceptions, very little current research appeals to both fields by incorporating the best elements of each. The combination of the newer image analysis techniques with the concurrent advances in machine learning turns out to contain subtle yet significant potential pitfalls that have not been adequately addressed to date.

This paper investigates the best way to combine boosting with a range of promising image comparison methods, after the example of Tieu and Viola Tieu and Viola (2000). It examines several candidate approaches, since the best way to incorporate boosting using these techniques has not yet been established. A summary of the findings appears in Section 3. In particular, a novel approach is developed herein using conic decision boundaries that allows boosting to be combined with any of the leading image representations. The experimental results show this method to work better with the high-dimensional image representations that are becoming more common today.

1.1. The Problem

Several characteristics of image comparison techniques make the straightforward application of boosting difficult. Image representations typically exhibit a high number of linearly nonseparable dimensions. This makes the use of machine learning mainstays such as C4.5 (Quinlan, 1993) both slower and less effective than they are in the sorts of problems typically looked at by the learning community. The comparison metrics developed for image retrieval, on the other hand, also form an incomplete foundation for boosting. Oriented towards retrieval rather than classification, they do not address the issue of establishing a classification threshold. More significantly, these techniques are designed to measure image similarities given a *single* target image; they do not necessarily handle a set of target images, possibly with weights indicating their importance. For boosting, incorporating such a weighted set of targets is essential.

The naïve approach to expanding from a single-target technique to a multiple-target technique would be to use some linear combination of the representations of the multiple target, probably the mean or the sum. Unfortunately, this method does not work: typically the combined representation is significantly *worse* at picking out members of the class than many of the individual training examples alone (Howe, 2001). This reflects the complexity of image classes: they tend to be only diffusely clustered in any given image representation, interspersed with non-members of the class, and rife with outliers.

1.2. Boosting in Context

Boosting began as a technique for combining differently-trained classifiers with unique sets of strengths and weaknesses. Properly done, a weighted vote of each classifiers' predictions can reinforce the strengths and cancel out the weaknesses (Schapire, 1990). Thus a classification algorithm that displays marginal success (accuracy slightly better than chance) can be "boosted" into an algorithm with much higher accuracy. AdaBoost (Freund & Schapire, 1996) first provided a widely known algorithmic approach to boosting. Since then, many variants have appeared that seek to address some of its shortcomings, such as intolerance to errors in the training data (Friedman et al., 1998), but AdaBoost continues to be widely used.

AdaBoost and boosting algorithms in general require a base learning algorithm, often referred to as a *weak learner*, that can classify any set of weighted instances with better than 50% accuracy. With a two-class system, such weak learners are not hard to develop: nearly any division of the space of possible instances will do. Although the theoretical results place only weak requirements on the base algorithm, empirical experience suggests that more powerful base classifiers tend to work better when boosted (Freund & Schapire, 1996). The base classifier is trained in successive rounds on different subsets or weightings of the initial training data, producing the required set of differently-trained classifiers that can be combined to produce a final, more reliable classification.

The success of boosting has been attributed to its ability to increase the separation (called the *margin*) between positive and negative instances of a class better than a single classifier alone (Schapire et al., 1998). As such it belongs to the broad category of large-margin classifiers, which also includes support vector machines (Burges, 1998). Practical experience with boosting suggests that many successive rounds of boosting can gradually increase the margin without overfitting the training data, even after the error on the training set has been reduced to zero (Schapire et al., 1998).

In spite of the success of boosting in other areas, little work has been done to date in applying it to images. Tieu and Viola Tieu and Viola (2000) use a feature-selection algorithm equivalent to simple boosting, but the focus of their work is elsewhere. Perhaps one reason that so little attention has been devoted to the topic is that researchers working with images have focused mainly on retrieval rather than classification. Only recently have algorithms developed that offer reasonable classification performance on any but the simplest of image categories.

1.3. Two Approaches

In order to apply boosting to most extant image representations designed for retrieval, one must first decide how to adapt a representation designed for pairwise determination of similarity so as to produce a class decision boundary. In doing so, one may decide to adopt an approach that has more of the flavor found in traditional machine learning, or one may opt instead for an approach that retains more of the flavor of the original image retrieval technique. This paper looks at both of these paths. The first approach uses simple single-dimension thresholded decision boundaries, which may be viewed as decision-tree stumps (decision trees with a depth of 1). Intuitively, it looks for dimensions (a.k.a. *features*) demonstrating exceptional values that happen to be highly correlated with membership in the class. The second approach uses the entire vector for comparison. This is achieved using a cosine metric (measuring the angle between two normalized vectors) and conic decision boundaries, for example with the vector representation of a target image at the axis of the cone. Intuitively, it looks for images that are similar enough to the exemplar image according to the cosine metric. One might expect either or both of these approaches to work better, depending upon the underlying image representation chosen.

Regardless of the method, any simple classifier used for boosting must conform to a few simple assumptions. As input it receives two collections of vectors, V_p and V_n, containing respectively the representations of positive and negative training examples of the class to be learned. In addition it receives a collection of weights on these vectors, W_p and W_n, indicating the importance placed upon learning to classify the corresponding training example. From these inputs, the classifier should generate a rule that classifies any image representation $\mathbf{v}$ as either a class member or not; this may be thought of as a function from the space of possible representations $\mathcal{V}$ onto $\{0, 1\}$. Section 2.1 describes several common image feature spaces $\mathcal{V}$ used in this paper.

Two features of most image representations make them somewhat different from many of the types of data typically used with boosting. They tend to be of very high dimension, with some schemes using tens of thousands of dimensions (Howe, 2001; Tieu & Viola, 2000). The correlations between individual dimensions tend to be unknown and presumably highly complicated. Furthermore, any single individual dimension typically has low correlation with any interesting image class: there are few "smoking guns". These considerations have led to the development of the two techniques described below, one which concentrates on individual features, and one which looks at the vector representation as a whole.

1.3.1. FEATURE-BASED BOOSTING

The first method, denoted hereafter as *feature-based boosting*, or *FBoost* for brevity, creates a simple classifier as follows. For each dimension in $\mathcal{V}$, it sorts the values found in V_p and V_n, removing duplicates, to determine a complete set of candidate decision thresholds. It then scores each of these decision thresholds in terms of the weighted error rate it would generate if used as a classifier. The best threshold is computed for each individual dimension, and the best of these becomes the rule used to classify unknown instances.

Feature-based boosting represents a fairly traditional way to apply boosting. Tieu and Viola Tieu and Viola (2000) adopt this approach in their work. From a machine-learning viewpoint, FBoost is equivalent to using decision trees with a single branch (also called *decision stumps*) as the base classifier. Friedman, Hastie, & Tibshirani Friedman et al. (1998) present evidence that the simplicity of the decision stumps as compared to full decision trees may be counteracted by a sufficient number of boosting steps.

One might expect feature-based boosting to work best with image representations that use the L_1 metric (also known as the Manhattan distance), since both the feature-based classifier and the L_1 metric rely on a separation in the individual features (dimensions). Such representations include both histograms and correlograms.

1.3.2. VECTOR-BASED BOOSTING

The second method, denoted hereafter as *vector-based boosting*, or *VBoost* for brevity, represents a non-traditional application of boosting concepts with no close analogues known to the author. Implementing it effectively turns out to require some creativity. If only the positive training instances are used as cone axes in creating individual classifiers, then the resulting set of decision boundaries lacks enough variety for effective boosting. (The algorithm quickly reaches a point where none of the available decision boundaries are of high quality.) On the other hand, allowing any axis at all leaves an infinite number of possible decision boundaries to check, with no guide towards finding the best one. A compromise heuristic is therefore used, with reasonable results produced in practice. The algorithm described below consistently generates individual classifiers with greater than 50% accuracy even after many successive rounds of boosting.

Let $\mathbf{v}_p$ and $\mathbf{v}_n$ be the sum of the vectors in V_p and V_n respectively, as weighted by the weights in W_p and W_n. Consider the hyperplane that bisects the angle between $\mathbf{v}_p$ and $\mathbf{v}_n$. Our experience shows that the majority of the weight of positive examples will tend to lie on one side of the hyperplane, while the majority of the weight of negative examples will tend to lie on the other side. (Usually the positive examples are clustered on the $\mathbf{v}_p$ side, but if this is not the case the algorithm simply exchanges the classification labels for that round of boosting.) The dot product with a vector orthogonal to the bisecting hyperplane therefore proves useful in discriminating between positive and negative examples. The heuristic algorithm calculates the orthogonal vector $\mathbf{v}_\perp$ according to Equation 1 and then computes its dot product with all the training vectors.

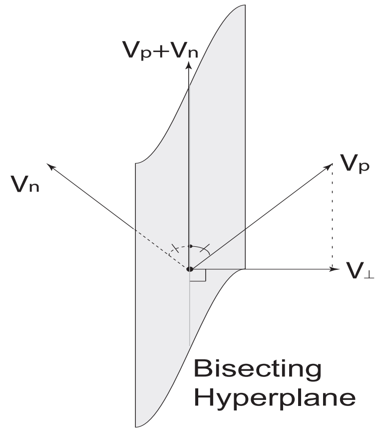

Figure 1. Schematic illustration of $\mathbf{v}_\perp$

$$\mathbf{v}_\perp = \mathbf{v}_p - \frac{\mathbf{v}_n \cdot (\mathbf{v}_p + \mathbf{v}_n)}{\|\mathbf{v}_p + \mathbf{v}_n\|} \qquad (1)$$

Figure 1.3.2 contains a schematic illustration of the situation described above.

As was the case for the previous classifier, the range of dot products between $\mathbf{v}_\perp$ and the elements of the training set offers a finite choice of decision thresholds. The best can be chosen simply by computing the weighted training error for each possibility.

2. Experimental procedure

The experimental procedure described below has three axes of variation: the image representation used, the type of base classfier used for boosting, and the image category used. Of these, the comparisons between image representations and between base classifiers are most interesting. All experiments are performed on the same set, comprising 20,100 images from the Corel photo library. Corel images have been used in many works on image retrieval, and more details on this set of images are available elsewhere (Howe, 2001).

All experiments use 5×2-fold cross-validation. For each of five replications, the image set is split in half, with half of the positive instances in each fold. Each fold is used to train a classifier, and its performance is tested on the opposite fold. Comparing results across the five replications provides an estimate of the deviation.

2.1. Image representation

Three image representations come from relatively recent work on ways of describing images that preserve multiple primitive image cues: color, texture, relative location, etc. Correlograms (Huang et al., 1997) assemble statistical information about color co-occurrences on the pixel level. Correlogram features are of the form "the probability that a pixel B at distance x from pixel A has the same color as A." Stairs (Howe & Huttenlocher, 2000) explicitly records the patches of color and texture found in different locations within the image. Thus each feature in Stairs represents the presence or absence of a patch with one discrete combination of color, texture, and location. An unnamed technique introduced by Tieu and Viola (Tieu & Viola, 2000) registers the output of banks of layered color and texture filters. A feature in this representation corresponds to the output of a set of three successively applied texture filters summed over the entire image. Finally, color histograms (Swain & Ballard, 1991) are a well-established representation, included here as a control. Each feature in a color histogram represents the percentage of the image that is of a particular discrete color.

The Tieu-Viola representation is altered somewhat here in order to achieve a practical algorithm. Their original representation stores about 50,000 numbers per image. With a collection of 20,100 images, therefore, the entire data set occupies roughly 8 Gb of memory. While this may easily fit on a disk, it will not fit into memory for efficient processing. An approximation yields the necessary reduction in required memory: the values are normalized by subtracting the mean for each feature and dividing by the standard deviation, after which values differing from the mean by fewer than 2.5 standard deviations are set to zero. This allows the data to be stored as a sparse matrix with about 10^7 elements. Tieu and Viola hypothesize that the success of their method stems from extreme feature values in the range of 4 or more standard deviations, so there is some reason to believe that this approximation will not unduly affect performance. On the other hand, it is problematic for the images that show no features at all with values more than two standard deviations from the mean (roughly one third of the test collection). Tieu and Viola do not address the memory issue in their paper, which looks at only 3000 images.

2.2. Boosting Type

Strictly speaking, only one boosting algorithm, AdaBoost (Freund & Schapire, 1996), is used in all experiments. Rather, the difference comes from the base classifier used, as described in the previous section. Although the base classifier used for boosting makes no difference in theory, the common wisdom and experience holds that some bases make for better boosting than others. Because the two methods used here work quite differently, any disparity in their performance should be instructive. As a control, the experiment includes a third, unboosted classifier. In order to give a more meaningful comparison to the boosted algorithms, this is not simply a single application of the base classifier (which does quite poorly). Rather, the control is a nearest-neighbor classifier using the best exemplars of the class as selected by a greedy additive approach (Howe, 2001).

2.3. Image Category

Five chosen categories reflect a moderate range of difficulty and subject category: *Suns* (226 images of sunrises and sunsets), *Wolves* (110 images of wolves), *Churches* (101 images of churches), *Tigers* (100 images of tigers), and *RaceCars* (100 images of Formula 1 race cars). Note that these choices include both natural and manufactured objects, full scenes and specific entities within scenes.

3. Results

Table 1 shows the mean area under an ROC curve computed for each of the 60 ($4 \times 3 \times 5$) experimental conditions, as a percent of total possible area. The data are grouped by image category, since the most interesting variations in performance show up when the category is held constant. The deviations are generally small: less than 2.5 percentage points in all but three cases.

The results display some interesting trends. First of all, the boosted methods generate higher numbers than the corresponding control in all but two cases, and the difference in each of the latter is not significant. This result is not automatic, since the control case is not a simplistic single application of the boosted classifier, but an effective nearest-neighbor classifier using the best exemplars of the class.

Although the unboosted base classifier does little better than chance, for completeness Table 2 gives the performance of this option. Note that a completely random classification algorithm should generate ROC curves with areas of 50%.[1] A comparison with the numbers in Table 1 shows the clear advantage of boosting.

3.1. Comparing boosting types

Comparisons between the two types of boosting reveal that different approaches work best with different underlying

[1]The apparent worse-than-chance performance of the Tieu-Viola control in Table 1 is interesting in this regard, but does not affect the conclusions of this paper.

Table 1. Per cent area under the ROC curve for two boosting methods and an unboosted control. The best method in each group of three is underlined, and the best method in each cluster of 12 is boldface.

Method	Hist	Corr	Stairs	T-V	
Control	82.7 ± 0.6	86.1 ± 1.1	92.2 ± 0.3	23.6 ± 5.1	Suns
VBoost	93.5 ± 0.5	**98.1** ± 0.4	93.2 ± 1.0	77.5 ± 1.3	Suns
FBoost	96.1 ± 0.4	96.2 ± 0.9	93.8 ± 0.7	75.6 ± 1.9	Suns
Control	71.6 ± 1.9	70.3 ± 1.2	75.3 ± 1.9	51.1 ± 0.6	Church
VBoost	76.9 ± 2.2	79.8 ± 1.7	**81.4** ± 1.6	66.4 ± 2.2	Church
FBoost	78.6 ± 1.2	80.5 ± 1.6	79.6 ± 1.1	62.2 ± 4.0	Church
Control	90.2 ± 1.3	81.0 ± 0.9	88.6 ± 0.3	43.1 ± 1.6	Car
VBoost	93.3 ± 0.7	96.6 ± 0.6	96.3 ± 1.2	68.2 ± 1.1	Car
FBoost	97.3 ± 0.6	**98.0** ± 0.5	96.3 ± 0.6	63.5 ± 1.9	Car
Control	93.5 ± 0.7	87.6 ± 0.3	85.2 ± 1.1	37.0 ± 0.7	Tiger
VBoost	92.5 ± 1.0	96.8 ± 0.5	88.9 ± 0.8	57.8 ± 1.6	Tiger
FBoost	**98.0** ± 0.7	97.7 ± 0.4	92.4 ± 0.6	54.4 ± 3.3	Tiger
Control	86.8 ± 1.5	89.2 ± 1.0	80.7 ± 0.4	48.4 ± 0.6	Wolf
VBoost	85.8 ± 2.1	**91.3** ± 1.8	87.1 ± 0.9	61.4 ± 2.5	Wolf
FBoost	90.7 ± 2.2	89.6 ± 2.0	88.5 ± 1.3	58.0 ± 1.3	Wolf

Table 2. Per cent area under the ROC curve for a single iteration of the base classifier. Random chance will yield an expected score of 50%

Method	Hist	Corr	Stairs	T-V	
VBoost (base)	50.7 ± 1.2	62.3 ± 3.2	52.8 ± 1.7	49.1 ± 2.7	Suns
FBoost (base)	49.6 ± 1.6	54.1 ± 3.0	52.9 ± 2.1	55.2 ± 1.3	Suns
VBoost (base)	49.3 ± 2.0	49.2 ± 4.4	52.2 ± 3.5	50.4 ± 2.6	Church
FBoost (base)	50.3 ± 1.4	51.0 ± 4.0	50.0 ± 2.6	51.7 ± 4.7	Church
VBoost (base)	50.5 ± 3.6	58.4 ± 2.6	48.5 ± 4.4	49.8 ± 4.0	Car
FBoost (base)	47.7 ± 2.3	52.0 ± 4.6	50.1 ± 2.5	49.9 ± 2.3	Car
VBoost (base)	48.4 ± 3.3	50.1 ± 2.8	47.0 ± 1.9	49.4 ± 1.2	Tiger
FBoost (base)	49.2 ± 3.5	50.4 ± 2.0	50.4 ± 2.3	51.0 ± 3.0	Tiger
VBoost (base)	51.6 ± 2.3	51.2 ± 2.6	53.4 ± 2.3	48.7 ± 1.9	Wolf
FBoost (base)	51.6 ± 2.4	49.9 ± 2.5	51.8 ± 3.3	49.1 ± 2.4	Wolf

image representations. For the histograms, feature-based boosting consistently performs better, achieving significant differences on four of the five image classes. (Underscores indicate the best boosting method in each category.) For the Tieu-Viola method, in contrast, vector-based boosting consistently performed better, although the difference was only statistically significant on one set (*RaceCar*). The remaining representations show mixed results. For correlograms, *FBoost* does significantly better on two categories and *VBoost* on one. For Stairs, *FBoost* does significantly better on only one category, with the remaining categories splitting the advantage and showing no statistically significant results either way.

Although the sparsity of statistical significance in the results makes it difficult to be sure, it appears that vector length may be an important factor governing which method works better. The representations are listed in the table from left to right in order of increasing vector length: histograms (128 dimensions), correlograms (512), Stairs (19,200) and Tieu-Viola (46,875). The success of *VBoost* appears to increase slightly with the number of dimensions, doing worst on the smallest representation and best on the largest. This is fortunate, since choosing the best feature on vectors with many dimensions is quite time-consuming: *VBoost* not only works better in these cases; it runs faster as well. If the advantage of vector-based boosting for high-dimensional representations holds up, it may ultimately prove a fruitful approach in other areas besides image classification. On the other hand, further research may ultimately prove that the apparent trend stems from idiosyncracies of the histogram and Tieu-Viola representations. For example, the individual dimensions in Tieu-Viola may be more intercorrelated than in the other representations.

3.2. Comparing image representations

Several other trends reveal themselves in Table 1. First, although the Tieu-Viola representation improves the most under boosting, the ultimate performance using it does not match that of the other three representations. This may be in part because of the representational restrictions imposed, as described in Section 2.1. However, without some such approximation, it is difficult to imagine how to apply the technique to larger data sets.

Of the other three representations, correlograms display the best overall score (indicated in bold) for three of the five image categories, with histograms and Stairs sharing honors on the remaining two. However, neither of the latter two scores are significantly better statistically than the corresponding correlogram score. Therefore, it appears that correlograms are the best choice by a small margin, if one can somehow determine the best form of boosting to use in a given situation.

4. Conclusion

Boosting improves the performance of image classification virtually across the board. Two different methods described herein for constructing boostable classifiers from a base image representation both yield better results than an unboosted control. Of these, the method based upon individual features works better when the number of dimensions in the image representation is small, while the method based upon the entire vector appears to work better when the number of dimensions is large. The former represents traditional applications of boosting, while the latter uses a novel type of decision boundary for the base classifier. Regardless, this work shows that systematic approaches exist for combining advanced image representations with boosting, and that such combinations generate superior classifiers.

References

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*, 121–167.

De Bonet, J. S., & Viola, P. (1997). Structure driven image database retrieval. *Advances in Neural Information Processing*, *10*.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 148–156).

Friedman, J., Hastie, T., & Tibshirani, R. (1998). *Additive logistic regression: a statistical view of boosting* (Technical Report). Dept. of Statistics, Stanford University.

Howe, N. R. (2001). *Analysis and representations for automatic comparison, classification and retrieval of digital images*. Doctoral dissertation, Cornell University.

Howe, N. R., & Huttenlocher, D. P. (2000). Integrating color, texture, and geometry for image retrieval. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. "II:239–246"). Los Alamitos, CA: IEEE Computer Society.

Huang, J., Ravi Kumar, S., Mitra, M., Zhu, W. J., & Zabih, R. (1997). Image indexing using color correlograms. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 762–768). Los Alamitos, CA.

Quinlan, J. R. (1993). *Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*, 197–227.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*, 1651–1686.

Swain, M., & Ballard, D. (1991). Color indexing. *International Journal of Computer Vision*, *7*, 11–32.

Tieu, K., & Viola, P. (2000). Boosting image retrieval. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 228–235).