Contribution of Boosting in Wrapper Models

Marc Sebban, Richard Nock
TRIVIA, West Indies and Guiana University
Campus de Fouillole, 95159 - Pointe à Pitre (France)
{msebban,rnock}@univ-ag.fr

Abstract. We describe a new way to deal with feature selection when boosting is used to assess the relevancy of feature subsets. In the context of wrapper models, the accuracy is here replaced as a performance function by a particular exponential criterion, usually optimized in boosting algorithms. A first experimental study brings to the fore the relevance of our approach. However, this new "boosted" strategy needs the construction at each step of many *learners*, leading to high computational costs.
We focus then, in a second part, on how to speed-up boosting convergence to reduce this complexity. We propose a new update of the instance distribution, which is the core of a boosting algorithm. We exploit these results to implement a new forward selection algorithm which converges much faster using overbiased distributions over learning instances. Speed-up is achieved by reducing the number of weak hypothesis when many identical observations are shared by different classes. A second experimental study on the UCI repository shows significantly speeding improvements with our new update without altering the feature subset selection.

1 Introduction

While increasing the number of descriptors for a machine learning domain would not intuitively make it harder for "perfect" learners, machine learning algorithms are quite sensitive to the addition of irrelevant features. Actually, the presence of attributes not directly necessary for prediction could have serious consequences for the performances of classifiers. That's why feature selection is became a central problem in machine learning. This trend will certainly continue because of the huge quantities of data (not always relevant) collected thanks to new acquisition technologies (the World Wide Web for instance).

In addition, the selection of a *good* feature subset may not only improve performances of the deduced model, but may also allow to build simpler classifiers with higher understanding. To achieve feature selection, we generally use one of the two following approaches, respectively called *filter* and *wrapper* [2, 6, 7]. *Filter* models use a preprocessing step, before the induction process, to select relevant features. The parameter to optimize is often a statistical criterion or an information measure: *interclass distance*, *probabilistic distance* [11], *entropy*

J.M. Żytkow and J. Rauch (Eds.): PKDD'99, LNAI 1704, pp. 214–222, 1999.

[8, 13], etc. The main argument for these methods is that they try to estimate feature relevance regardless of the classifier, as an intrinsic property of the represented concept. The second main trend uses *wrapper* models. These methods assess alternative feature subsets using a given induction algorithm; the criterion to optimize is often the accuracy. In spite of high computational costs, wrapper models have the advantage to provide better accuracy estimates (by holdout, cross-validation or bootstrap) than a statistical criterion or an information measure as used in the filter approach.

In this paper, we propose to challenge the criterion to optimize in wrapper models, replacing the accuracy by the Schapire-Singer's criterion [12] which was not previously tested in the field of feature selection. This approach is motivated by recent theoretical results on the general performances of the algorithm AdaBoost [12]. Boosting consists in training and combining the output of T various base learners. Each of them can return various formulas (decision trees, rules, k-Nearest-Neighbors (kNN), etc.). We show through experimental results that the optimization of this criterion in a forward feature selection algorithm, called FS^{2}BOOST, allows to select more relevant features, and achieves higher classification performances, compared to the classical accuracy criterion.

Despite its interesting properties, using boosting in a feature selection algorithm (notably in a wrapper model) needs to cope with high computational costs. Actually, wrapper models are already known to have high complexity. The worst case of a forward selection algorithm requires $O(p^2)$ estimates, each of them requiring $O(|LS|^2)$ comparisons (using for instance a kNN classifier), where LS is the learning sample and p is the number of features. The use of the boosting procedure increases this complexity, requiring T steps at each stage. However, arguing for the use of Boosting, Quinlan [10] points out that even if Boosting is costly, the additional complexity factor (T) is known in advance and can be controlled. Moreover, it can be useful to choose a fast classifier (such as the kNN) to decrease again this complexity. Nevertheless, this parameter T in FS^{2}Boost deserves investigation.

In the second part of this paper, we focus on how to speed-up boosting convergence in FS^{2}Boost, to reduce this complexity. We propose a particular update of the instance distribution during boosting. It consists in balancing the distribution not only toward hard examples (as in the original AdaBoost algorithm [12]), but also on examples for which the conditional class distribution is highly in favor of some class against the others. The main point of this new update, which particularly suits to feature selection, is that as the description becomes poorer (*e.g.* by removing features), many examples of different classes may match the same description. Ultimately, descriptions with evenly balanced examples among classes are somewhat useless and can be "forgotten" by the learning algorithm. Applying this new principle, we can avoid to build many base learners, while selecting almost the same feature subsets. We propose an improved extension of our first algorithm, called iFS^{2}Boost, and we compare performances of the two presented algorithms on several benchmarks of the

UCI repository[1]. Our experimental results highlight significant speed-up factors during the selection, without altering the selected feature subsets.

2 A Wrapper Model using Boosting

Wrapper models evaluate alternative feature subsets using a performance criterion which is usually the accuracy over the learning sample; the goal is to find which feature subset allows to increase the prediction accuracy. The accuracy is estimated using an induction algorithm as a core procedure, which builds formulae such as decision trees, induction graphs, neural networks, kNN, etc. This core procedure chosen, there remains to choose an heuristic of search among all the possible subspaces. Here we consider a forward selection algorithm, which often allows to reduce computational costs, avoiding calculations in high dimensional spaces. It is an *a priori* choice, but selecting a backward instead of a forward algorithm would not challenge the framework of our approach. Its principle can be summarized as follows:

> *At each time, add the feature to a current feature set (initialized to* $\emptyset$*) which increases the most the accuracy of a formula built using the core algorithm. If no addition of a new feature increases the accuracy, then stop and return the current feature subset.*

While the wrapper approach is accurate when the core algorithm is the same as the subsequent induction algorithm, it may suffer a drawback that the core step introduces a representational bias. Indeed, not only do we measure the potential of improvement a feature represents, but also the bias according to which the feature could improve the accuracy of a formula built from the concept class of the core algorithm. Such a problem appears because functional dependencies of various nature exist between features, themselves understandable by means of representational biases [2].

For these reasons, recent works have chosen to investigate the properties of a novel kind of algorithms: boosting [12]. Boosting as presented into AdaBoost [12] is related to the stepwise construction of a linear separator into a high dimensional space, using a base learner to provide each functional dimension. Decision tree learning algorithms are well-suited for such a base-learner task, but other kind of algorithms can be chosen. The main idea of boosting is to repetitively query the base learner on a learning sample biased to increase the weights of the misclassified examples; by this mean, each new hypothesis is built on a learning sample which was hard to classify for its predecessor. Figure 1 presents the AdaBoost learning algorithm [12] in the two-classes case.

When there are $k > 2$ classes, k binary classifiers are built, each of them used for the discrimination of one class against all others. The classifier returning

[1] http://www.ics.uci.edu/~mlearn/MLRepository.html

AdaBoost($LS = \{(x_i, y(x_i))\}_{i=1}^{|LS|}$)
Initialize distribution $D_1(x_i) = 1/|LS|$;
For $t = 1, 2, ..., T$
Build weak hypothesis h_t using D_t;
Compute the confidence α_t:

$$\alpha_t = \frac{1}{2} \log \left(\frac{1 + r_t}{1 - r_t} \right) \quad (1)$$

$$r_t = \sum_{i=1}^{m} D_t(x_i) y(x_i) h_t(x_i) \quad (2)$$

Update: $D_{t+1}(x_i) = \frac{D_t(x_i) e^{-\alpha_t y(x_i) h_t(x_i)}}{Z_t}$;
$/* Z_t$ is a normalization coefficient$*/$
endFor
Return the classifier

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Fig. 1. Pseudocode for AdaBoost.

the greatest value gives the class of the observation. Boosting has been shown theoretically or empirically to satisfy particularly interesting properties. Among them, it was remarked [5] that boosting is sometimes immune to overfitting, a classical problem in machine learning. Moreover it allows to reduce a lot the representational bias in relevance estimation we pointed out before. Define the function $F(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$ to avoid problems with the "sign" expression in $H(x)$. [12] have proven that using AdaBoost is equivalent to optimize a criterion which is not the accuracy, but precisely the normalization factor Z_t as presented in figure 1. Using a more synthetic notation, [5] have proven that AdaBoost repetitively optimizes the following criterion:

$$Z = E_{(x,y(x))}(e^{-y(x)F(x)})$$

In a first step, we decided then to use this criterion in a forward selection algorithm that we called FS^{2}Boost (figure 2). We show in the next section the interest of this new optimized criterion thanks to experimental results.

3 Experimental Results: Z versus Accuracy

In this section, the goal is to test the effect of the criterion optimized in the wrapper model. We propose to compare the selected feature relevance using either Z or the accuracy, on synthetic or natural databases. Nineteen problems

FS^{2}BOOST($LS = \{(x_i, y(x_i))\}_{i=1}^{|LS|}$)
1 $Z_0 \leftarrow +\infty$; $E \leftarrow \emptyset$; $S \leftarrow \{s_1, s_2, ..., s_p\}$;
2 ForEach $s_j \in S$
$H \leftarrow$ ADABOOST($LS, E \cup s_i$);
$Z_i \leftarrow Z_{E \cup s_i}(H)$;
select $s_{\min}$ for which $Z_{\min} = \min_i Z_i$;
endFor
3 If $Z_{\min} \ll Z_0$ then
$S = S \backslash \{s_{\min}\}$; $E = E \cup \{s_{\min}\}$; $Z_0 \leftarrow Z_{\min}$;
Goto step 2;
Else return E;

Fig. 2. Pseudocode for FS^{2}BOOST. S is the set of features

were chosen, among them the majority was taken from the UCI repository. A database was generated synthetically with some irrelevant features (called *Artificial*). *Hard* is a hard problem consisting of two classes and 10 features per instance. There are five irrelevant features. The class is given by the *XOR* of the five relevant features. Finally, each feature has 10% noise. The *Xd6* problem was previously used by [3]: it is composed of 10 attributes, one of which is irrelevant. The target concept is a disjunctive normal form over the nine other attributes. There is also classification noise.

Since we know for artificial problems the relevance degree of each feature, we can easily evaluate the effectiveness of our selection method. The problem is more difficult for natural domains. An adequate solution consists in running on each feature subset an induction algorithm (kNN in our study), and compare the "qualities" of the feature subsets with respect to the *a posteriori* accuracies. Accuracies are estimated by a leave-one-out cross-validation. On each dataset, we used the following experimental set-up:

1. the SIMPLE FORWARD SELECTION (SFS) algorithm is applied, optimizing the accuracy during the selection. We compute then the accuracy by cross-validation in the selected subspace.
2. FS^{2}BOOST is run ($T = 50$). We compute also the *a posteriori* accuracy.
3. We compute the accuracy in the original space with all the attributes.

Results are presented in table 1. First, FS^{2}BOOST works well on datasets for which we knew the nature of features: relevant attributes are almost always selected, even if irrelevant attributes are sometimes also selected. On these problems, the expected effects of FS^{2}BOOST are then confirmed. Second, FS^{2}BOOST allows to obtain almost always a better accuracy rate on the selected subset, than on the subset chosen by the simple forward selection algorithm. Third, in the majority of cases, accuracy estimates on feature subsets after FS^{2}BOOST are better than on the whole set of attributes.

Despite these interesting results, FS^{2}BOOST has a shortcoming: its *computational cost*. In the next section, after some definitions, we will show that instead

Database	SFS	FS^{2}Boost	All Attributes
Monks 1	97.9	97.9	81
Monks 2	67.2	67.2	68.3
Monks 2	99.0	99.0	94.4
Artificial	84.7	86.4	84
LED	81.4	90.2	90.2
LED24	81.4	87.2	77.9
Credit	86.1	87.1	76.8
EchoCardio	73	74.8	66.9
Glass2	62.5	73.2	72.0
Heart	82.2	81.7	82.8
Hepatitis	78.7	81.9	82.4
Horse	77.6	86.3	72.2
Breast Cancer	96.4	96.4	96.5
Xd6	79.9	79.9	78.1
Australian	83.8	81.6	78.7
White House	95.7	95.7	91.5
Pima	73.2	73.3	73.0
Hard	58.7	58.7	59.0
Vehicle	72.9	73.7	71.6

Table 1. Accuracy comparisons between three feature sets: (i) the subset obtained by optimizing the accuracy, (ii) the subset deduced by FS^{2}Boost, and (iii) the whole set of features. Best results are underlined.

of minimizing Z, we can speed-up the boosting convergence optimizing another $Z^{'}$ criterion.

4 Speeding-up Boosting Convergence

Let $S = \{(x_1, y(x_1)), (x_2, y(x_2)), ..., (x_m, y(x_m))\}$ be a sequence of training examples, where each observation belongs to $\mathcal{X}$, and each label y_i belongs to a finite label space $\mathcal{Y}$. In order to handle observations which can belong to different classes, for any description x_p over $\mathcal{X}$, define $|{x_p}^+|$ (resp. $|{x_p}^-|$) to be the cardinality of positive (resp. negative) examples having the description x_p; note that $|x_p| = |{x_p}^-| + |{x_p}^+|$. We make large use of three quantities, $|{x_p}^{max}| = \max(|{x_p}^+|, |{x_p}^-|)$, $|{x_p}^{min}| = \min(|{x_p}^+|, |{x_p}^-|)$ and $\Delta(x_p) = |{x_p}^{max}| - |{x_p}^{min}|$. The optimal prediction for some description x is the class hidden in the "max" of $|{x_p}^{max}|$, which we write $y(x_p)$ for short. Finally, for some predicate P, define as $[\![\mathrm{P}]\!]$ to be 1 if P holds, and 0 otherwise; define as $\pi(x, x')$ to be the predicate "x' and x share identical descriptions", for arbitrary descriptions x and x'.

We give here indications on speeding-up Boosting convergence for the biclass setting. In the multiclass case, the strategy remains the same. The idea is to replace Schapire-Singer's Z criterion by another one, which integrates the notion

of similar descriptions belonging to different classes. This kind of situation often appears in feature selection, notably at the beginning of the SFS algorithm or also when the number of features is small according to a high cardinality of the learning set. More precisely, we use

$$E_{x \sim D'_t}\left[e^{-y(x')(\alpha_t h_t(x'))}\right]$$

with

$$D'_t(x') = \frac{\sum_{x_p} D_t(x')[\![\pi(x,x')]\!]\frac{\Delta(x_p)}{|x_p|}}{\sum_{x''}\left(\sum_{x_p} D_t(x'')[\![\pi(x,x')]\!]\frac{\Delta(x_p)}{|x_p|}\right)}$$

In other words, we minimize a weighted expectation with distribution favoring the examples for which the conditional distribution of the observations projecting onto it is greatly in favor of one class against the others. Note that when each possible observation belongs to one class (*i.e.* no information is lost among the examples), the expectation is exactly Schapire-Singer's Z.

As [12] suggest, for the sake of simplicity, we can fold temporarily α_t in h_t so that the weak learner scales-up its votes to $\mathbb{R}$. Removing the t subscript, we obtain the following criterion which the weak learner should strive to optimize:

$$Z' = E_{x \sim D'}\left[e^{-y(x')h(x')}\right]$$

Optimizing Z' instead of Z at each round of Boosting is equivalent to (i) keeping strictly AdaBoost's algorithm while optimizing Z', or (ii) modifying AdaBoost's initial distribution, or its update rule. With the new Z' criterion, we have to choose in our extended algorithm iFS^{2}Boost,

$$\alpha'_t = \frac{1}{2}\log\left(\frac{1+r'_t}{1-r'_t}\right)$$

where

$$r'_t = \sum_{x'} D'_t(x')y(x')h_t(x') = E_{x \sim D'_t}\left[y(x')h_t(x')\right]$$

5 Experimental Results: Z' versus Z

We tested here 12 datasets with the following experimental set-up:

1. The FS^{2}Boost algorithm is run with T base learners. We test the algorithm with different values of T ($T = 1, .., 100$), and we search for the minimal number T_Z which provides a stabilized feature subset $FS_{stabilized}$, *i.e.* for which the feature subset is the same for $T = T_Z, .., 100$.

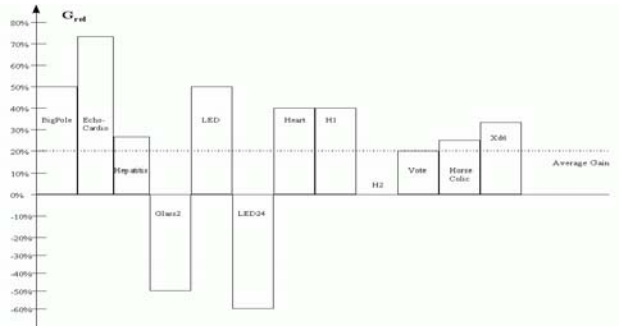

Fig. 3. Relative Gain G_{rel} of weak learners. The dotted line presents the average gain

2. iFS^{2}Boost is also run with different values of T and we search for the number $T_{Z'}$ which provides a $FS'_{stabilized}$ feature subset.

For ten datasets, we note that the use of Z' in iFS^{2}Boost allows to save on some weak hypothesis, without modifying the selected features (i.e. $FS_{stabilized} = FS'_{stabilized}$). In average, our new algorithm requires 3.5 learners less than FS^{2}Boost. These results confirm the speeder convergence of iFS^{2}Boost, without alteration of the selected subspace. What is more surprising is that for two databases (*Glass2* and *LED24*), iFS^{2}Boost needs more learners than FS^{2}Boost and we obtain $FS_{stabilized} \neq FS'_{stabilized}$. We could intuitively think that, Z' converging faster than Z, we should not meet such a situation. In fact, we can explain this phenomenon analyzing the speed-up factor of iFS^{2}Boost. Actually, the number $|x_p|$ of instances sharing a same description and belonging to different classes is independent from a subset to another, and the gain $G = T_Z - T_{Z'}$ is directly dependant of $|x_p|$. Thus, at a given step of the selection, iFS^{2}Boost can exceptionally select a weak relevant feature for which the speed-up factor is higher than for a strongly relevant one. In that case, iFS^{2}Boost will require supplementary weak hypothesis to correctly update the instance distribution. Nevertheless, this phenomenon seems to be quite marginal.

Improvements of iFS^{2}Boost can be more dramatically presented by computing the relative gain of weak learners $G_{rel} = \frac{T_Z - T_{Z'}}{T_{Z'}}$. Results are presented in figure 3. In that case, we notice that iFS^{2}Boost requires in average 22.5% learners less than FS^{2}Boost, that confirms the positive effects of our new approach, without challenging the selected subset by FS^{2}Boost.

6 Conclusion

In this article, we linked two central problems in machine learning and data mining: *feature selection* and *boosting*. Even if these two fields have the common

aim to deduce from feature sets powerful classifiers, as far as we know few works tried to share their interesting properties. Replacing the accuracy by another Z performance criterion optimized by a boosting algorithm, we obtained better results for feature selection, despite high computational costs. To reduce this complexity we tried to improve the proposed FS^{2}Boost algorithm, introducing a speed-up factor in the selection. In the majority of cases, improvements are significant, allowing to save on some weak learners. The experimental gain represents on average more than 20% of the running time. Following a remark of [10] on Boosting, improvements of this magnitude without degradation of the solution, would be well worth the choice of iFS^{2}Boost, particularly on large domains where feature selection becomes essential. We still think however that time improvements are possible, but with possibly slight modifications of the solutions. In particular, investigations on computationally efficient estimators of boosting coefficients are sought. This shall be the subject of future work in the framework of feature selection.

References

1. D. Aha and R. Bankert. A comparative evaluation of sequential feature selection algorithms. In *Fisher and Lenz Edts, Artificial intelligence and Statistics*, 1996.
2. A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Issue of Artificial Intelligence*, 1997.
3. W. Buntine and T. Niblett. A further comparison of splitting rules for decision tree induction. *Machine Learning*, pages 75–85, 1992.
4. Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, pages 119–139, 1997.
5. J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression : a Statistical View of Boosting. draft, July 1998.
6. G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Eleventh ICML conference*, pages 121–129, 1994.
7. R. Kohavi. Feature subset selection as search with probabilistic estimates. *AAAI Fall Symposium on Relevance*, 1994.
8. D. Koller and R. Sahami. Toward optimal feature selection. In *Thirteenth International Conference on Machine Learning (Bari-Italy)*, pages 284–292, 1996.
9. P. Langley and S. Sage. Oblivious decision trees and abstract cases. In *Working Notes of the AAAI94 Workshop on Case-Based Reasoning*, pages 113–117, 1994.
10. J. Quinlan. Bagging, boosting and c4.5. In *AAAI96*, pages 725–730, 1996.
11. C. Rao. *Linear statistical inference and its applications*. Wiley New York, 1965.
12. R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the Eleventh Annual ACM Conference on Computational Learning Theory*, pages 80–91, 1998.
13. M. Sebban. On feature selection: a new filter model. In *Twelfth International Florida AI Research Society Conference*, pages 230–234, 1999.
14. D. Skalak. Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *11th International Conference on Machine Learning*, pages 293–301, 1994.