Scaling Boosting by Margin-Based Inclusion of Features and Relations

Susanne Hoche and Stefan Wrobel

Otto-von-Guericke University, Magdeburg, Germany
{hoche,wrobel}@iws.cs.uni-magdeburg.de

Abstract. Boosting is well known to increase the accuracy of propositional and multi-relational classification learners. However, the base learner's efficiency vitally determines boosting's efficiency since the complexity of the underlying learner is amplified by iterated calls of the learner in the boosting framework. The idea of restricting the learner to smaller feature subsets in order to increase efficiency is widely used. Surprisingly, little attention has been paid so far to exploiting characteristics of boosting itself to include features based on the current learning progress. In this paper, we show that the dynamics inherent to boosting offer ideal means to maximize the efficiency of the learning process. We describe how to utilize the training examples' *margins* - which are known to be maximized by boosting - to reduce learning times without a deterioration of the learning quality. We suggest to stepwise include features in the learning process in response to a slowdown in the improvement of the margins. Experimental results show that this approach significantly reduces the learning time while maintaining or even improving the predictive accuracy of the underlying fully equipped learner.

1 Introduction

Boosting is a method for enhancing learning algorithms by basing predictions on a group of specialized hypotheses. Instead of searching for one highly accurate prediction rule covering a given set of training examples, an ensemble of rules is constructed by repeatedly calling a base learner with a changing distribution of weights for the training examples. Each rule in the ensemble might cover only a small subset of the examples, and all predictions are combined into one accurate joint prediction. Boosting is a popular technique for increasing the accuracy of classification learners and has been developed into practical algorithms that have demonstrated superior performance on a broad range of application problems in both propositional and multi-relational domains [3,19,17,5,7].

However, the iterative nature of boosting implies an amplification of the underlying learner's complexity. Boosting's efficiency is vitally determined by the base learner's efficiency. A standard approach to deal with the issue of efficiency in the presence of large feature sets would be to use a feature selection method in an a priori fashion, and then run boosting with the small selected feature subset. However, deciding a priori on the number of features to be included in

T. Elomaa et al. (Eds.): ECML, LNAI 2430, pp. 148–160, 2002.

the learning process might lead to inferior results since it is often difficult to decide just how many features to include. If too many features are included the learner is unnecessary slow, if too few features are included the learning result might not be sufficiently accurate.

Instead, in this paper we suggest to *actively* determine the right balance between speed and accuracy of a learner based on its learning progress. We propose to monitor the learning success in terms of the development of the training examples' mean *margins* - which are known to be maximized by boosting - and to present step-by-step promising features to the learner whenever the improvement of the margins drops below a certain threshold. The margins' improvement is measured by the ratio of the mean margins' gradients averaged over several iterations and the current gradient of the training examples' mean margins. This ratio increases from one iteration to the next as long as the margins increase significantly. As soon as the ratio starts to decrease, an estimate of the slowdown in the margins' improvements is determined. This estimate predicts the expected decrease of the ratio and is used to determine when to provide a new feature to the learner. Whenever the actual decrease of the ratio is exceeding the predicted decrease by a certain factor, a new feature is included in the learning process. To this end, all features present in the training examples are initially sorted according to their mutual information [25,14] with the examples' class, and new features are provided to the learner in a demand-driven fashion, starting with the top two features and the relations in which they occur.

The evaluation of our approach on various domains shows that our approach significantly reduces learning times while maintaining or even improving predictive accuracy. Although our learner is multi-relational, our experiments indicate that the results apply equally well to boosting in propositional domains.

This paper is organized as follows. In section 2, we review boosting. In section 3, we present our approach to include features and relations into the learning process in a demand-driven fashion. Our experimental evaluation of the approach is described and discussed in section 4. In section 5, we discuss related work and conclude in section 6 with some pointers to future work.

2 Boosting

Boosting is a method for improving the predictive accuracy of a learning system by means of combining a set of base classifiers constructed by a base learner into one single hypothesis [22,20,19]. The idea is to "boost" a weak learner performing slightly better than random guessing into an arbitrarily accurate learner by repeatedly calling the learner on varying probability distributions over the training instances. The probability distribution models the weight associated with each training instance and indicates the influence of an instance when building a base classifier. Initially, all instances have equal influence on the construction of a base hypothesis, i.e. the probability distribution is uniform. In each iterative call of the learner, a base hypothesis is learned with a prediction confidence for each example. The weights of misclassified instances are increased and those of cor-

rectly classified instances are decreased according to the confidence of the learned base hypothesis. Thus, correctly classified instances have less and misclassified instances have more influence on the construction of the base hypothesis in the next iteration. That way, in each new round of boosting the learner is confronted with a modified learning task and forced to focus on the examples which have not yet been correctly classified. Finally, all base hypotheses learned are combined into one strong hypothesis. An instance x is classified by the strong hypothesis by adding up the prediction confidence of each base hypothesis covering x, and classifying x according to the sign of this sum.

2.1 Constrained Confidence-Rated Boosting

In this paper, we employ a specific form of constrained boosting, C^2RIB, *Constrained Confidence-Rated ILP-Boosting*, which we introduced in [7] and which forms the basis of the work presented here (C^2RIB^D). In Tables 1 and 2, we give a concise description of the proposed algorithm. Components of the base algorithm C^2RIB in Table 1 are denoted by '$\circ$'. For a definition of the functions in the following explanation, the reader is referred to Table 1. In C^2RIB, the training instances are randomly split into two sets used for specialization and pruning of clauses, respectively. Starting with the target predicate, the refinement operator ρ of the relational learner iteratively refines the clause C maximizing the objective function $\tilde{z}$ until either a clause C' is found with hitherto maximal $\tilde{z}(C')$ that covers only positive examples, or $\tilde{z}$ can not be further maximized. The resulting clause is subject to overfitting on the training data, and thus immediately considered for pruning. The generated hypothesis is compared to the so called default hypothesis, just comprising the target predicate and satisfying all examples. Whichever of these two hypotheses maximizes the objective function z is chosen as the base classifier of the current iteration and its prediction confidence is used to update the probability distribution for the next iteration.

3 Margin-Based Inclusion of Features and Relations

The objective of our work presented here is to accelerate the learning process of the boosted ILP-learner C^2RIB without a deterioration of its prediction accuracy. The idea is to equip the learner at all times with the right amount of power needed to "successfully" perform the learning task, i.e. to start the learner with a few features and relations to be considered for refinement, monitor the learning results and include additional features and relations into the learning process by demand. For this purpose, we exploit the dynamics inherent to boosting, namely that it a) is known to maximize training examples' *margins*, b) is based on combining classifiers specialized on certain fractions of the instance space, and c) works by repeatedly calling a weak learner. Table 2 and the sections of Table 1 marked with '$\bullet$' give a concise description of the algorithm which is detailed in the following. References to Table 2 will be indicated by "T2._".

Table 1. C^2RIB^D Algorithm

Let N denote the number of training instances $e_i = (x_i, y_i) \in E = E^+ \cup E^-$, and $y_i = 1$ for $e_i \in E^+$ and $y_i = -1$ for $e_i \in E^-$. Let p be the target predicate of arity $a(p)$ to be learned, T the total number of iterations of the weak learner, and D a probability distribution over E with D_i^t the probability of e_i in the t-th iteration. For a clause C and a set $\mathcal{S} \subseteq E$, let w_+, w_- be weight functions defined as in 1. and 2. further down this page, and $c(C, \mathcal{S})$ C's prediction confidence on $\mathcal{S}$ defined according to 3.

- **Let** $\mathcal{F}$ be the set of features sorted in descending order with respect to their mutual information with the examples' class computed according to equation (2), and $\mathcal{F}'$ the set of features known to the learner, initially comprising the top two features of $\mathcal{F}$.
- **Set** $D_i^1 := \frac{1}{N}$ for $1 \leq i \leq N$
- **For** $t = 1 \ldots T$
  - **Split** E randomly into $\mathcal{G}$ and $\mathcal{P}$ according to D_t s.t. $\sum_{(x_i,y_i)\in\mathcal{G}} D_i^t \approx \frac{2}{3}$
  - $C := p(X_1, \cdots, X_{a(p)})$
  - $\tilde{Z} := 0$
  - **While** $w_-(C, \mathcal{G}) > 0$
    - **Let** $C' := argmax_{C'' \in \rho(C)}\{\tilde{z}(C'')\}$, where $\tilde{z}$ is defined as in 4.
    - **Let** $\tilde{Z}' := \tilde{z}(C')$
    - **If** $\tilde{Z}' - \tilde{Z} \leq 0$ exit loop
    - **Else** $C := C'$, $\tilde{Z} := \tilde{Z}'$
  - $Prunes(C) := \{p(X_1, \cdots, X_{a(p)}) \leftarrow B \mid C = p(X_1, \cdots, X_{a(p)}) \leftarrow BB'\}$
  - **Remove** from $Prunes(C)$ all clauses C' where $c(C', E) \leq 0$
  - **If** $Prunes(C) = \emptyset$ let $C_t := p(X_1, \cdots, X_{a(p)})$
  - **Else**
    - $C' := argmin_{C'' \in Prunes(C)}\{loss(C'')\}$, with $loss(C'')$ defined as in 5.
    - **Let** $C_t := argmax_{C'' \in \{C', p(X_1, \cdots, X_{a(p)})\}}\{(z(C'')\}$, with z defined as in 6.
  - $h_t : X \to \Re$ is the function
    $$h_t(x) = \begin{cases} c(C_t, E) & \text{if e} = (\text{x}, \text{y}) \in \text{E is covered by } C_t \\ 0 & \text{else} \end{cases}$$
  - **Update** the probability distribution: $D_i^{t'} := \frac{D_i^t}{e^{(y_i \cdot h_t(x_i))}}$, $D_i^{t+1} := \frac{D_i^{t'}}{\sum_i D_i^{t'}}, 1 \leq i \leq N$
  - **If** $t > 2$ and $\mathcal{F}' \neq \mathcal{F}$
    - **Let** $H_t := \{h_1, \cdots, h_t\}$, with base classifier h_k of iteration $1 \leq k \leq t$
    - $\mathcal{F}' = CheckLearningProgress(H_t, t, E, N, \mathcal{F}, \mathcal{F}',)$ as detailed in Table 2
- **Construct** the strong hypothesis $H(x) := sign\left(\sum_{C_t:(x,y) \text{ covered by } C_t} c(C_t, E)\right)$

3.1 Margins in the Framework of Confidence-Rated Boosting

In this approach, we monitor the learning success by observing the training examples' mean margins. The margin of an example $e_i = (x_i, y_i)$ under an ensemble H_t of classifiers is a real-valued number $margin(H_t, e_i) \in [-1, 1]$ indicating

Table 1. continue

Function Definitions

1. $w_+(C,\mathcal{S}) =_{def.} \sum_{(x_i,1)\in\mathcal{S} \text{ covered by } C} D_i^t$
2. $w_-(C,\mathcal{S}) =_{def.} \sum_{(x_i,-1)\in\mathcal{S} \text{ covered by } C} D_i^t$
3. $c(C,\mathcal{S}) =_{def.} \frac{1}{2}\ln\left(\frac{w_+(C,\mathcal{S})+\frac{1}{2N}}{w_-(C,\mathcal{S})+\frac{1}{2N}}\right)$.
4. $\tilde{z}(C) =_{def.} \sqrt{w_+(C,\mathcal{G})} - \sqrt{w_-(C,\mathcal{G})}$.
5. $loss(C) =_{def.} (1-(w_+(C,\mathcal{P})+w_-(C,\mathcal{P}))) + w_+(C,\mathcal{P})\cdot e^{(-c(C,\mathcal{G}))} + w_-(C,\mathcal{P})\cdot e^{(c(C,\mathcal{G}))}$
6. $z(C) =_{def.} \left(\sqrt{w_+(C,E)} - \sqrt{w_-(C,E)}\right)^2$

the amount of disagreement of the classifiers in H_t with respect to e_i's class. For the binary case we deal with here, we can define the margin of e_i under H_t as the difference between the sum of the absolute weights of those base classifiers in H_t predicting for e_i its correct class y_i, and the sum of the absolute weights of those base classifiers in H_t predicting for e_i the incorrect class $y \neq y_i$ [24,6].

We define the weight $w(h_k, e_i)$ of a base classifier h_k with respect to an example $e_i = (x_i, y_i)$ as its prediction confidence (as defined in Table 1, 3.) if h_k covers e_i, and 0 otherwise. We define, following [6], the margin of e_i under ensemble $H_t = \{h_1, \cdots, h_t\}$ of t classifiers h_k with weights $w(h_k, e_i)$ as

$$margin(H_t, e_i) = \sum_{h_k\in H_t : h_k(x_i)=y_i} |w(h_k, e_i)| - \sum_{h_k\in H_t : h_k(x_i)\neq y_i} |w(h_k, e_i)| \ . \quad (1)$$

We normalize the prediction confidences of the base classifiers such that the absolute values of the confidences of all base classifiers sum to 1. Consequently, $\sum_{h_k\in H_t:h_k(x_i)=y_i} |w(h_k,e_i)| \in [0,1]$, $\sum_{h_k\in H_t:h_k(x_i)\neq y_i} |w(h_k,e_i)| \in [0,1]$, and $margin(H_t, e_i) \in [-1,1]$ for all $e_i \in E$ and ensembles H_t.

Large positive margins (close to $+1$) indicate "confident" correct classification, and small negative margins (close to -1) indicate "confident" incorrect classification. Boosting is known to be especially effective at increasing the margins of the training examples [24,6]. It forces the focus on misclassified instances by increasing their probabilities. Misclassified examples show small or even negative margins. Consequently, the learner is forced to search for base hypotheses which correctly classify these hard examples and thus increase their margins. Since the margins are increasing in the course of iterated calls to the base learner, the gradient of the mean margins can be assumed to be positive and be employed to monitor the quality of the learning process.

The repeated calls of the base learner in the boosting framework allow for a stepwise inclusion of features in the course of iterations. If the learning curve indicates that the learner's current instrumentation is not sufficient any longer,

Table 2. CheckLearningProgress

$CheckLearningProgress(H_t, t, E, N, \mathcal{F}, \mathcal{F}')$ returns $\mathcal{F}''$

1. **Compute** for E the examples' average margin $AM_t = \frac{1}{N}\sum_{i=1}^{n} margin(H_t, e_i)$ according to equation (1)
2. **Let** $gradient(t)$ be the slope of the line determined by the least square fit to the AM_k in $k, 1 \leq k \leq t$
3. **Compute** $trend(t) := \begin{cases} \frac{1}{T_l}\sum_{j=1}^{T_l} gradient(t-j) & \text{if } t > T_l \\ \frac{1}{t-2}\sum_{j=2}^{t-1} gradient(j) & \text{if } t \leq T_l, \end{cases}$
   where T_l denotes the number of iterations over which the gradients are averaged
4. **Compute** $ratio(t) := \frac{trend(t)}{gradient(t)}$
5. **If** $t > 3$:
   (a) **If** $ratio(t-1)$ exhibits a local maximum, estimate the slowdown in the margins' improvement in the form of $predict(x) := a\frac{1}{ln(\frac{x}{b})}$, where a, b are chosen such that $predict(2) = ratio(t-1)$ and $predict(3) = ratio(t)$; $offset := t - 3$
   (b) **If** a, b have already been determined, compute $predict(t) := a\frac{1}{ln(\frac{t-offset}{b})}$
   (c) **Else** $predict(t) := ratio(t)$
   (d) **If** $\frac{predict(t)}{ratio(t)} > \alpha$, select the first element F of $\mathcal{F}$, i.e. the feature with the next greatest mutual information with the training examples' class; $\mathcal{F}'' := \mathcal{F}' \cup \{F\}$
   (e) **Else** $\mathcal{F}'' := \mathcal{F}'$

learning can be continued in the next iteration with an enhanced equipment. Initially, we provide our learner with the target relation to be learned together with two features with the greatest *mutual information* [25,14] with the examples' class and the relations in which these features occur.

In each iteration of boosting, the learning success is monitored in terms of the development of the training examples' mean margins. To this end, we define the gradient $gradient(t)$ of an iteration t as the slope of the line determined by the least square fit to the average margins in each single iteration 1 to t (T2.1, T2.2). We then average the gradients over the last T_l iterations as to smooth temporary fluctuations in the margins' development (T2.3), and compute the ratio of the averaged previous gradients and the gradient of the current iteration (T2.4). The margins' improvement is measured by this ratio which increases from one iteration to the next as long as the margins increase significantly. As soon as the ratio starts to decrease, an estimate for the slowdown in the margins' improvements is determined (T2.5a). This estimate predicts the expected decrease of the ratio and is used to determine when a new feature has to be presented to the learner. The estimate is chosen to be an inverse-logarithm. Whenever the actual decrease of the ratio exceeds the predicted decrease by a certain threshold, a new feature is included into the learning process (T2.5d).

3.2 Mutual Information between Features

Initially, all features in the given training examples are sorted according to their mutual information [25,14] with the examples' class. The mutual information $MI(F_1, F_2)$ between two features F_1, F_2 is defined as the difference between the entropy of F_1 and the entropy of F_1 given F_2 [27], i.e. as the amount of information about the possible values $(f_{11}, f_{12}, ...)$ of feature F_1 that is obtained when the value $f \in \{f_{21}, f_{22}, ...\}$ of feature F_2 is known. To compute the mutual information between class C and feature F_j, we estimate the probability distributions of C and F_j from the training data, ignoring missing values, as follows:

- The probability $p(C = c)$ of any training example being of class c is estimated as the fraction $\frac{|E^c|}{|E|}$ of training examples from E belonging to c.
- The probability $p(F_j = f_i)$ that the nominal feature F_j takes value f_i is estimated as the fraction $\frac{|F_j = f_i|}{|E|}$ of training examples for which feature F_j takes value f_i.
- The joint probability $p(C = c) \wedge (F_j = f_i)$ is derived from the probabilities of the two single events.

The mutual information between a feature F_j and the class C of an example can then be defined as

$$\begin{aligned} MI(C, F_j) &= E(C) - E(C|F_j) \\ &= \sum_{i=1}^{m_j} \sum_{c=1}^{k} p(C = c, F_j = f_i) \ \ln \ \frac{p(C = c, F_j = f_i)}{p(C = c)p(F_j = fi)} \end{aligned} \tag{2}$$

with k possible classes and m_j possible values of feature F_j. For features F_j with continuous values, we estimate the probability distribution by discretizing the values of F_j with an entropy based method [4] and using the resulting interval $[d_1, ..., d_{m_i}]$ to estimate the probability of F_j taking a value in the interval $I_i := [d_i, d_{i+1}), 1 \leq i < m_i - 1$, and $I_{m_{i-1}} := [d_{m_i - 1}, d_{m_i}]$ respectively, as the fraction $\frac{|F_j \in I_i|}{|E|}$ of training examples for which feature F_j takes a value in $I_i, 1 \leq i \leq m_i - 1$. Note that this way of sorting features according to their mutual information with respect to classification assumes independence of the features and may thus result in inferior performance in domains with highly correlated features.

4 Empirical Evaluation

To evaluate our approach, we performed experiments on data sets differing in the number of features and the total number of examples. We determine prediction accuracy and learning time for each dataset for both the base case C^2RIB and for C^2RIB^D described in this paper, and compare the results to those of other systems (Tables 3 to 5). For C^2RIB^D, we also indicate the average number of features included in the learning process. In all experiments, 1) the

Table 3. Accuracy, standard deviation and learning time in minutes for SLIPPER [3], C^2RIB and C^2RIB^D on five propositional domains

Domain	♯ Ex.	♯ Features	Train / Test	SLIPPER [3] Acc StdD	Time	C²RIB Acc StdD	Time	C²RIB^D Acc StdD	Time	♯ sel. Feat.
breast-wisc	699	9	10CV	95.8 n/a	n/a	96.1 ±1.5	9	95.4 ±1.7	5.1	5.8
horse-colic	368	23	10CV	85.0 n/a	n/a	81.0 ±8.4	3.6	83.7 ±5.7	0.9	2
hypothyroid	3163	25	10CV	99.3 n/a	n/a	95.2 ±0.69	39.1	96.6 ±2.9	20.8	11.8
mushroom	8124	22	10CV	99.8 n/a	n/a	99.3 ±3.0	144	99.6 ±0.16	71.4	4.8
splice-junction	3190	60	10CV	94.1	n/a	53.13 ±3.0	289	88 ±4.7	13.6	4.4

base learner is invoked $T = 100$ times, 2) the gradients of the examples' mean margins are averaged over the last $T_l = 10$ iterations and, 3) the threshold α is set to 1.01 (see 5d in Table 2). The value 1.01 has been empirically determined on the domain of Mutagenicity [26], and has not been modified for subsequent experiments on the other domains in order to ensure proper cross validation results.

We chose three different types of domains in order to get an assessment of our learner 1) on propositional tasks, and 2) on general knowledge and data mining tasks and 3) on ILP benchmark and classic Machine Learning problems. The first set of experiments comprises five propositional domains from the UCI-repository [16]. We compare our approach to the propositional constrained confidence-rated booster SLIPPER [3] which served as a basis for C^2RIB. Predictive accuracies are estimated by 10-fold-cross validation.[1] As can be seen from Table 3, C^2RIB performs in four domains on par with or slightly weaker than SLIPPER. C^2RIB^D reduces C^2RIB's learning time[2] up to one order of magnitude with a superior predictive accuracy in four domains, and without a significant deterioration of predictive accuracy in the one domain where only few features are present. C^2RIB shows a poor performance on the splice-junction dataset, most likely due to the great number of features. However, C^2RIB^D clearly outperforms C^2RIB both in accuracy and learning time.

The second set of experiments was conducted on datasets subject of the data mining competitions PKDD Discovery Challenge 2000 [1] (classification of loans, where Task AC is based on all loans, and Task A only on the closed loans from

[1] However, in [3], single training- and test set splits are used for hypothyroid, mushroom and splice-junction.

[2] Learning times for SLIPPER are not known to us.

Task AC), and KDD Cup 2001, Task2 [2] (prediction of gene functions). The predictive accuracy is estimated by 10-fold-cross validation, and the results are compared to Progol [15] and RELAGGS [13], a transformation-based approach to ILP, combined with SVM^{light} and C4.5rules, respectively, run on the propositionalized data. For Task AC, Progol was run for 2 days, and discontinued without any results. Prediction accuracies of C^2RIB and C^2RIB^D are, for Task AC, notedly lower than the ones obtained by RELAGGS/C4.5rules, however still in the range of standard deviation of the accuracies obtained by RELAGGS/SVM^{light}, as holds for Task A. However, learning times of C^2RIB and C^2RIB^D are lower than the ones of the other systems. For Task AC, C^2RIB^D speeds up C^2RIB's learning time by factor 2. For Task A, C^2RIB^D seems to be penalized for sorting the features in the presence of few examples. For the gene function

Table 4. Accuracy, standard deviation and learning time for Progol [15], RELAGGS [13], C^2RIB and C^2RIB^D on some data mining competition domains

Domain	♯ Ex. Train / Test	♯ Features	Progol Acc StdD Time	RELAGGS SVM^{light} Acc StdD Time	RELAGGS C4.5rules Acc StdD Time	C^{2}RIB Acc StdD Time	C^{2}RIB^D Acc StdD Time	♯ sel. Feat.
PKDD DS 2000, AC	682 10CV	24	n/a n/a 2 days	90.8 ±3.2 23 min	94.1 ±3.2 23 min	88.9 ±3.4 20 min	88.9 ±3.4 9.5 min	10
PKDD DS 2000, A	234 10CV	24	45.7 ±10.5 hrs	88.0 ±5.3 10 min	88.0 ±6.5 10 min	86.3 ±6.1 3.6 min	86.7 ±6.6 4.2 min	10.2
KDD Cup 2001, Task2	1243 862/381	49	92.2 24 min	92.2 ≈ 2 min	n/a n/a	91.1 53 min	91.5 27 min	5

prediction task, C^2RIB and C^2RIB^D were ran on the original KDD Cup 2001 training-test-data partition and the results were compared to Progol[3] and RELAGGS/SVM^{light}.[4,5] Again, learning time is reduced by factor 2 in the demand-driven approach C^2RIB^D. It slightly improves C^2RIB's predictive accuracy which is on par with the other systems' accuracies.

Finally, we evaluated our approach on the two ILP benchmark problems Mutagenicity [26] (prediction of mutagenic activity of 188 molecules (description $\mathcal{B}_4$)) and QSARs, Quantitative Structure Activity Relationships, [9,10] (prediction of a greater-activity relationship between pairs of compounds based on

[3] L. Peña Castillo, unpublished, 2002
[4] M.-A. Krogel, unpublished, 2002
[5] RELAGGS won Task of KDD Cup 2001.

Table 5. Accuracy, standard deviation and learning time in minutes for C^2RIB and C^2RIB^D in comparison to other systems on two ILP benchmark and one artificial domain

Domain	♯ Ex. Train/ Test	♯ Features	FOIL Acc StdD Time	Fors Acc StdD Time	Progol Acc StdD Time	C²RIB Acc StdD Time	C²RIB^D Acc StdD Time	♯ sel. Feat.
Mutagenicity	188 10CV	18	82.0 [26] ±3.0 n/a	89.0 [8] ±6.0 n/a	88.0 [26] ±2.0 307	88.0 ±3.4 7	88.8 ±5.2 1.53	6
QSARs	2788 5CV	12	82.9 ±2.7 0.7	n/a n/a n/a	79.8 ±3.7 372	83.4 ±2.9 91	83.3 ±1.9 70	11.8
Eastbound Trains	55 / 6	9	n/a n/a n/a	n/a n/a n/a	77.78 [18] ±6.43 1.15	83.3 ±0 0.44	89.6 ±8.6 0.1	6.25

their structure), and on the artificial problem of Eastbound Trains[6] proposed by Ryszard Michalski (prediction of trains' directions based on their properties). For the two ILP domains, predictive accuracy is estimated by 10- and 5-fold-cross validation, respectively, and results are compared to FOIL [21], Fors [8] and Progol. For the Eastbound Trains, the data is split into one training and test set partition, and the results are averaged over 8 iterations of the experiment. Predictive accuracy of C^2RIB is higher than or on par with the one of the other learners. C^2RIB^D significantly outperforms C^2RIB both in terms of predictive accuracy and learning time in two of the three domains, indicating that our approach seems to be superior in classical, highly structured ILP domains.

5 Related Work

The idea of selecting smaller feature subsets and shifting the bias to a more expressive representation language is common in multi-relational learning. The work probably most related to our work is [12], where AdaBoost [22] is combined with MOLFEA, an inductive database for the domain of biochemistry [11]. In [12], AdaBoost is employed to identify particularly difficult examples for which MOLFEA constructs new special purpose structural features. AdaBoost re-weighting episodes and MOLFEA feature construction episodes are alternated. In each iteration, a new feature constructed by MOLFEA is presented to a propositional learner, the examples are re-weighted in accordance to the base classifier learned by it, and a new feature is constructed by MOLFEA based on the modified weights. In contrast, our approach actively decides when to include new features

[6] The examples were generated with the Random Train Generator available at http://www-users-cs-york.ac.uk/∼stephen/progol.html

from the list of ranked existing features with the central goal of including new features only when absolutely necessary in order to be maximally efficient. This means that in principle the two approaches could be easily combined, for example by calling a generator of new features whenever the list of existing features has been exhausted.

[23] propose a wrapper model utilizing boosting for feature selection. In their approach, alternative feature subsets are assessed based on the underlying booster's optimization criterion. The feature subset optimal according to this criterion is then presented as a whole to a learner. In contrast, we use a criterion of mutual information once before we start the boosting process to establish a feature ranking, and utilize the characteristics of our boosted learner to actively decide when to include a new feature. However, it would be interesting to combine both approaches.

6 Conclusion

In this paper, we have proposed an approach to boosting a weak relational learner which starts off with a minimal set of features and relations and is - by demand - stepwise strengthened. Our work is based on C^2RIB [7], a fast weak ILP-learner in a constrained confidence-rated boosting framework. The quality of the current learning results is measured in terms of the gradient of the training examples' mean margins, and the learner is strengthened whenever the learning curve drops under a certain threshold. To that purpose, features occurring in the training examples are sorted according to their mutual information with the examples' class and by and by provided to the learner together with the relation in which they occur. We showed that learning times are significantly reduced while the predictive accuracy is comparable to those of other learning systems and, in the majority of cases, superiour to those of the "fully equipped" learner C^2RIB. These results are encouraging, especially since all experiments were conducted without optimizing parameters.

One question for further work is whether one could expect to even gain a higher predictive accuracy by repeatedly evaluating the features' ordering and taking into account the examples' weights under the current probability distribution. In each iteration, the learner is presented a different training set, emphasizing the hard examples more and more. A stronger influence of so far misclassified examples on the feature ranking could support the induction of correct classifiers for those examples that are particularly difficult to learn.

Another question for further research is whether it is possible to determine automatically for every domain a) an optimal threshold to which the deviation of the current from the expected decrease of the ratio of the average and the current gradient should be compared and b) the number of iterations over which the gradients should be averaged. It is also part of the future work to investigate other approaches to feature selection, and make use of the accelerated learning time to incorporate more standard elements of "full-blown" ILP-learners and to determine the right balance between speed and accuracy of the learning system.

This work was partially supported by DFG (German Science Foundation), project FOR345/1-1TP6. We would like to thank L. Peña Castillo and M. Krogel for providing their results on the KDD Cup 2001, L. Peña Castillo for reviewing previous versions of this paper, and J. Kaduk for many inspiring discussions.

References

1. P. Berka. Guide to the financial Data Set. In: *A. Siebes and P. Berka, editors, PKDD2000 Discovery Challenge*, 2000. 155
2. J. Cheng, C. Hatzis, H. Hayashi, M.-A. Krogel, Sh. Morishita, D. Page, and J. Sese. KDD Cup 2001 Report. I*SIGKDD Explorations, 3(2):47-64*, 2002. 156
3. W. Cohen and Y. Singer. A Simple, Fast, and Effective Rule Learner. *Proc. of 16th National Conference on Artificial Intelligence*, 1999. 148, 155
4. U. M. Fayyad, and K. B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *Proc. of 13th Int. Joint Conf. on AI*, 1993. 154
5. Y. Freund, and R. E. Schapire. Experiments with a New Boosting Algorithm. *Proc. of 13th International Conference on Machine Learning*, 1996. 148
6. A. J. Grove, and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. *Proc. of 15th National Conf. on AI*, 1998. 152
7. S. Hoche, and S. Wrobel. Relational Learning Using Constrained Confidence-Rated Boosting. *Proc. 11th Int. Conf. on Inductive Logic Programming (ILP)*, 2001. 148, 150, 158
8. A. Karalic. *First Order Regression*. PhD thesis, University of Ljubljana, Faculty of Computer Science, Ljubljana, Slovenia, 1995. 157
9. R. D. King, S. Muggleton, R. A. Lewis, and M. J. E. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proc. of the National Academy of Sciences of the USA* 89(23):11322-11326, 1992. 156
10. R. D. King, A. Srinivasan, and M. Sternberg. Relating chemical activity to structure: An examination of ILP successes. *New Generation Computing, Special issue on Inductive Logic Programming* 13(3-4):411-434, 1995. 156
11. S. Kramer, and L. De Raedt. Feature construction with version spaces for biochemical applications. *Proc. of the 18th ICML*, 2001. 157
12. S. Kramer. Demand-driven Construction of Structural Features in ILP. *Proc. 11th Int. Conf. on Inductive Logic Programming (ILP)*, 2001. 157
13. M.-A. Krogel , and S. Wrobel. Transformation-Based Learning Using Multirelational Aggregation. *Proc. 11th Int. Conf. on Inductive Logic Programming (ILP)*, 2001. 156
14. W. J. McGill. Multivariate information transmission. *IRE Trans. Inf. Theory*, 1995. 149, 153, 154
15. S. Muggleton. Inverse Entailment and Progol. *New Gen. Computing*, 13, 1995. 156
16. P. M. Murphy, and D. W. Aha. UCI repository of machine learning databases. University of California-Irvine, Department of Information and Computer Science, 1994. http://www1.ics.uci.edu/ mlearn/MLRepository.html 155
17. D. Opitz, and R. Maclin. Popular Ensemble Method: An Empirical Study. *Journal of Artificial Intelligence Research 11, pages 169-198*, 1999. 148

18. L. Peña Castillo, S. Wrobel. On the Stability of Example-Driven Learning Systems: a Case Study in Multirelational Learning. *Proceedings of MICAI 2002*, 2002. 157
19. J. R. Quinlan. Bagging, boosting, and C4.5. *Proc. of 14th Nat. Conf. on AI*, 1996. 148, 149
20. J. R. Quinlan. Boosting First-Order Learning. *Algorithmic Learning Theory*, 1996. 149
21. J. R. Quinlan and R. M. Cameron-Jones. FOIL: A Midterm Report. In P. Brazdil, editor, *Proc. of the 6th European Conference on Machine Learning*, 667: 3-20, 1993. 157
22. R. E. Schapire. Theoretical views of boosting and applications. *Proceedings of the 10th International Conference on Algorithmic Learning Theory*, 1999. 149, 157
23. M. Sebban, and R. Nock. Contribution of Boosting in Wrapper Models. In: *J. M. Zytkow, and J. Rauch, eds, Proc. of the PKDD'99*, 1999. 158
24. R. E. Schapire, Y. Freund, P.Bartlett, and W. S. Lee. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. *The Annals of Statistics, 26(5):1651-1686*, 1998. 152
25. C. E. Shannon. A mathematical theory of communication. *Bell. Syst. Techn. J.*, 27:379-423, 1948. 149, 153, 154
26. A. Srinivasan, S. Muggleton, M. J. E. Sternberg, and R. D. King. Theories for mutagenicity: A study in first-order and feature-based induction. *Artificial Intelligence*, 1996. 155, 156, 157
27. D. Wettschereck. *A Study of Distance-based Machine Learning Algorithms*. PhD thesis, Oregon State University,Computer Science Department, Corvallis, USA 1994. 154