Boosting First-Order Clauses for Large, Skewed Data Sets

Louis Oliphant[1,3], Elizabeth Burnside[2,3] and Jude Shavlik[1,3]

[1]Computer Sciences Department
[2]Radiology Department
[3]Biostatistics and Medical Informatics Department
University of Wisconsin-Madison

Abstract. Creating an effective ensemble of clauses for large, skewed data sets requires finding a diverse, high-scoring set of clauses and then combining them in such a way as to maximize predictive accuracy. We have adapted the RankBoost algorithm in order to maximize area under the recall-precision curve, a much better metric when working with highly skewed data sets than ROC curves. We have also explored a range of possibilities for the weak hypotheses used by our modified RankBoost algorithm beyond using individual clauses. We provide results on four large, skewed data sets showing that our modified RankBoost algorithm outperforms the original on area under the recall-precision curves.
Keywords: Learning to Rank, Ensembles, Boosting

1 Introduction

Research over the past 15 years has shown an improvement in predictive accuracy by using an ensemble of classifiers [4] over individual classifiers. In the Inductive Logic Programming [6] domain ensembles have been successfully used to increase performance [5,9,10]. Successful ensemble approaches must both learn individual classifiers that work well with a set of other classifiers as well as combine those classifiers in a way that maximizes performance. AdaBoost [8] is a well known ensemble method that does both of these things. AdaBoost learns weak hypotheses iteratively, increasing the weight on previously misclassified examples so successive learners focus on misclassified examples. AdaBoost combines weak hypotheses into a single classifier by using a weighted sum, where each weak hypothesis is weighted according to its accuracy.

While AdaBoost focuses on improving accuracy of the final classifier, other boosting algorithms have been created that maximize other metrics. The objective of Freud and Schapire's RankBoost algorithm [7] is to maximize the correct ordering of pairs in a list. Cortes and Mohri [1] later showed that when using RankBoost with binary classes the objective of the algorithm also maximizes the area under the receiver operator characteristic (AUROC) curve. AUROC is a common metric used to discriminate between classifiers. Davis and Goadrich [3] however demonstrated that AUROC is not a good metric for discriminating between classifiers when working with highly skewed data where the negatives

Presented at the 19th International Conference on Inductive Logic Programming (ILP'09).

Modified RankBoost Algorithm
Given: disjoint subsets of negative, X_0, and positive, X_1, examples
Initialize:

$$skew = \frac{|X_0|}{|X_1|}, \qquad w_1(x) = \begin{cases} \frac{skew}{|X_0|} \; if \; x \in X_0 \\ \frac{1}{|X_1|} \; if \; x \in X_1 \end{cases}$$

for $t = 1, ..., T$:

Train weak learner, h_t, using w_t and $skew$.
Get weak ranking $h_t : X \longrightarrow \mathbb{R}$.
Choose $\alpha_t \in \mathbb{R}$. (see text)
Update

$$w_{t+1}(x) = \begin{cases} \frac{w_t(x) \exp(-\alpha_t h_t(x))}{Z_t^1} \; if \; x \in X_1 \\ \frac{w_t(x) \exp(\alpha_t h_t(x))}{Z_t^0} \; if \; x \in X_0 \end{cases}$$

where

$$Z_t^1 = \sum_{x \in X_1} w_t(x) \exp(-\alpha_t h_t(x)), \qquad Z_t^0 = \frac{1}{skew} \cdot \sum_{x \in X_0} w_t(x) \exp(\alpha_t h_t(x))$$

Output the final ranking: $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$.

Fig. 1. The modified RankBoost algorithm for optimizing area under the recall-precision curve.

outnumber the positives. They recommend using area under the recall-precision curve (AURPC) when working with skewed data.

We demonstrate a modified version of the RankBoost algorithm that works well with skewed data. Its objective function seeks to maximize AURPC. We implement a top-down, heuristic-guided search to find high-scoring clauses for the weak hypotheses and then utilize our modified RankBoost algorithm to combine them into a single classifier. We also evaluate several other possibilities for weak hypotheses that use sets of the best-scoring clauses found during search.

2 Modified RankBoost Algorithm

Freud and Schapire's RankBoost.B algorithm for bipartite data appears in Figure 1, with one modification. We have modified the sum of the weights on the negative set to the skew between the size of the negative set and the size of the positive set. We make this change to expose enough information to our weak learner so that it can optimize the AURPC. We calculate the α parameter using the so-called *Third-Method* of RankBoost where $\alpha = 0.5 \ln\left(\frac{1+r}{1-r}\right)$. In the original algorithm Cortes and Mohri showed that r is a weighted version of AUROC. For our algorithm, since we are trying to optimize for AURPC we have calculated weighted versions of recall and precision and used them in calculating the AURPC. We then use this weighted version of AURPC as our r value.

As our weak learners, we use a greedy hill-climbing algorithm to find one clause. We begin with the most general clause. All legal literals are considered to extend the clause. The extension that improves the weighted AURPC the most

is selected and added to the clause. The process repeats until no improvement can be found or some time limit or clause-length limit is reached. Each weak hypothesis, $h_t(x)$, is the best scoring individual clause found during this search. The weak hypothesis maps an example, x, to the range $\{0, 1\}$ where the mapping is 1 if the example is covered, 0 otherwise. The clause is then weighted by α. The final score for an example is the weighted sum of all clauses that cover the example. We call this learner RankBoost.Clause.

We have also explored other possibilities for both the weak learner and the optimization function used by the greedy hill climber. When working with large data sets the costly time step is often evaluating clauses against the training data. Our goal in developing other weak learners was to create more accurate models without increasing the number of clauses evaluated on training data. One method of doing this is to retain more than just the best clause found during search. Taking an idea from the Gleaner algorithm [9] which retains the best clause found for each range of recall space, we have developed several other weak learners. Here we report on a second weak hypothesis, RankBoost.Path, that contains all clauses along the path from the most general clause to the highest scoring clause. The weak hypothesis, $h_t(x)$, maps an example, x, to the range $[0, 1]$ by finding the most specific of these clauses that covers the example. The example is mapped to the real value corresponding to the fraction of the total AURPC covered by this clause. The total AURPC, r, is the area under the entire path rather than just the end clause, as is illustrated in Figure 2.

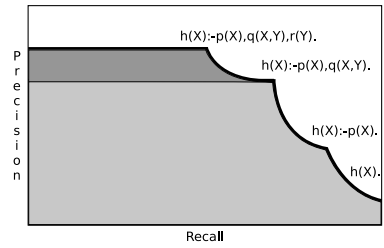

Fig. 2. Area under the recall-precision curve for a path of clauses learned during hill climbing. The total grayed area is the total AURPC, r. If h(X):-p(X),q(X,Y) is the most specific clause in the path to cover an example then $h_t(x)$ maps the example to the value light gray area divided by total grayed area.

When calculating the AURPC we followed the algorithm outlined by Davis and Goadrich [3] with two modifications to improve accuracy and increase speed. First, we use a weighted version of recall and precision. Second, when calculating the area between two points in recall-precision space, A and B, Davis and Goadrich suggest a discretized version that estimates the area under the curve. We calculate it exactly using a closed form solution to the integral for the curve

between the two points,

$$\int_{TP_A}^{TP_B} \frac{x}{x + FP_A + s(x - TP_A)}\, dx$$

where TP is the true positive weight and FP is the false positive weight. Parameter s is the local skew of false positives to true positives, $s = \frac{FP_B - FP_A}{TP_B - TP_A}$.

3 Experimental Results

We modified Aleph [12] to incorporate RankBoost and our modified versions, RankBoost.Clause and RankBoost.Path. We compared these algorithms using AUROC and AURPC on four large, skewed data sets. Two data sets come from the biomedical information extraction (IE) domain. The other two data sets come from the medical diagnosis domain.

Protein Localization data set consists of text from 871 abstracts taken from the Medline database. The task is to find all phrase pairs that specify a protein and where it localizes in a cell. The data set comes from Ray and Craven [11]. Additional hand annotation was done by Goadrich et al. [9] The data set contains 281,071 examples with a positive/negative skew of 1:149.

Gene Disease data set also comes from Ray and Craven [11]. We utilized the ILP implementation by Goadrich et al. [9] The task is to find all phrase pairs showing genes and their associated disorder. the data set contains 104,192 examples with a positive/negative skew of 1:446.

Mammography1 data set is described by Davis et al. [2] It contains 62,219 findings. The objective with this data set is to determine if a finding is benign or malignant given descriptors of the finding, patient risk factors, and radiologist's prediction. The positive/negative skew is 1:121.

Mammography2 is a new data set that has the same task as Mammography1, however the data was collected from mammograms from a second institution, the University of Wisconsin Hospital and Clinics. The data set consists of 30,405 findings from 18,375 patients collected from mammograms at the radiology department. The positive/negative skew is 1:86.

We ran 10 fold cross-validation for the mammography data sets and 5 for the IE data sets. We ran each fold 10 times using a different random seed and averaging results for each fold. We then calculated average AURPC, AUROC, and standard deviation across the different folds. Also, to compare how quickly the ensembles converged, we created learning curves with the x-axis showing the number of clauses evaluated and the y-axis showing the average AURPC.

Table 1 shows average AURPC and AUROC results with standard deviations for ensembles containing 100 weak learners for RankBoost and RankBoost.Clause. These two algorithms are the best to directly compare since they contain the same type of weak learners. RankBoost outperforms RankBoost.Clause when comparing AUROC on three of the four data sets. The AUROC scores are

Data set	AUROC		AURPC	
	RankBoost	RankBoost.Clause	RankBoost	RankBoost.Clause
Mammography 1	$\mathbf{89.9 \pm 4.2}$	88.1 ± 5.8	18.5 ± 5.7	$\mathbf{32.9 \pm 7.6}$
Mammography 2	92.5 ± 2.0	$\mathbf{96.7 \pm 1.1}$	16.2 ± 7.4	$\mathbf{41.3 \pm 10.6}$
Protein Localization	$\mathbf{98.9 \pm 0.1}$	97.9 ± 0.7	40.4 ± 7.9	40.5 ± 8.6
Gene Disease	$\mathbf{98.2 \pm 0.9}$	95.4 ± 2.4	32.9 ± 10.7	$\mathbf{46.6 \pm 11.9}$

Table 1. Average AUROC and AURPC percentages with standard deviations for several large, skewed data sets using the RankBoost and RankBoost.Clause algorithms. Bold indicates statistically significant improvement at 5% confidence level.

high and close together. This makes it difficult to distinguish one algorithm from another. However when comparing AURPC the difference between the two algorithms is large. RankBoost.Clause outperforms RankBoost on three of the four data sets. The variance is much larger for AURPC scores than for AUROC scores because when recall is close to 0 variance in precision values is high.

Learning curves on the four data sets appear in Figure 3. Each graph shows the AURPC on the y-axis by the number of clauses considered on the x-axis. Each curve extends until 100 weak hypotheses have been found. We do this as a way of showing that the various algorithms do different amounts of work to produce 100 hypotheses, a fact that would be lost if we simply extended all three to the full width of the x-axis.

Our RankBoost.Path algorithm reaches an AURPC of 0.44 on the Protein Localization data set after less than 20,000 clauses searched. The Gleaner algorithm [9] takes over 100,000 clauses to surpass this level of performance. On the Gene Disease data set our RankBoost.Clause algorithm reaches 0.48 AURPC after 45,000 clauses searched while the Gleaner algorithm does not reach this level of performance even after 10 million clauses searched. We have additional results, not shown because of space constraints, using other weak learners that combine variations of RankBoost.Clause and RankBoost.Path that demonstrate other properties such as higher asymptotic performance but slower convergence.

4 Conclusion and Acknowledgments

Large, skewed data sets provide an interesting challenge for ILP. We have modified the RankBoost algorithm [7] to optimize area under the recall-precision curve, a metric often used in these types of data sets. We evaluated several types of weak learners using our modified RankBoost algorithm compared with standard RankBoost on four large, skewed data sets. Due to space constraints we show results for two types of weak learners. Our modified RankBoost algorithm outperforms standard RankBoost when comparing AURPC. We gratefully acknowledge the funding from USA grants R01 LM07050 and 1 R01 CA127379, Houssam Nassif, David Page, and Ryan Woods and our anonymous reviewers for their comments and suggestions.

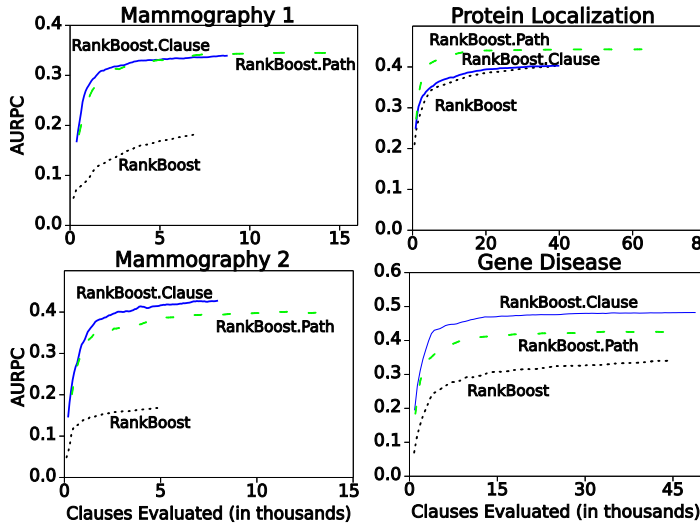

Fig. 3. Learning curves for RankBoost, RankBoost.Clause and RankBoost.Path on four large, skewed data sets.

References

1. C. Cortes and M. Mohri. AUC optimization vs. error rate minimization. In *NIPS*, 2003.
2. J. Davis, E. Burnside, I. Dutra, D. Page, and V. Costa. An integrated approach to learning Bayesian networks of rules. In *ECML*, 2005.
3. J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *ICML*, 2006.
4. T. Dietterich. Ensemble methods in machine learning. In *Workshop on Multiple Classifier Systems*, 2000.
5. I. Dutra, D. Page, V. Costa, and J. Shavlik. An empirical evaluation of bagging in inductive logic programming. In *ILP*, 2002.
6. S. Dzeroski and N. Lavrac. An introduction to inductive logic programming. In *Relational Data Mining*, 2001.
7. Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *ICML*, 1998.
8. Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *ICML*, 1996.
9. M. Goadrich, L. Oliphant, and J. Shavlik. Gleaner: Creating ensembles of first-order clauses to improve recall-precision curves. *Machine Learning*, (1-3), 2006.
10. J. R. Quinlan. Relational learning and boosting. In *Relational Data Mining*, 2001.
11. S. Ray and M. Craven. Representing sentence structure in hidden Markov models for information extraction. In *IJCAI*, 2001.
12. A. Srinivasan. The aleph manual version 4. *http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/*, 2003.