Sequential Skewing: An Improved Skewing Algorithm

Soumya Ray SRAY@CS.WISC.EDU
Department of Computer Sciences and Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, WI 53706

David Page PAGE@BIOSTAT.WISC.EDU
Department of Biostatistics and Medical Informatics and Department of Computer Sciences, University of Wisconsin, Madison, WI 53706

Abstract

This paper extends previous work on the *Skewing* algorithm, a promising approach that allows greedy decision tree induction algorithms to handle problematic functions such as parity functions with a lower run-time penalty than Lookahead. A deficiency of the previously proposed algorithm is its inability to scale up to high dimensional problems. In this paper, we describe a modified algorithm that scales better with increasing numbers of variables. We present experiments with randomly generated Boolean functions that evaluate the algorithm's response to increasing dimensions. We also evaluate the algorithm on a challenging real world biomedical problem, that of SH3 domain binding. Our results indicate that our algorithm almost always outperforms an information gain-based decision tree learner.

1. Introduction

In machine learning, greedy algorithms are often employed to learn concepts. These algorithms make a sequence of choices, such as choosing a feature to split on when building a Decision Tree, or choosing an edge to add to a Bayesian Network. Greedy algorithms commit to choices that are locally optimal according to functions such as Information Gain (Quinlan, 1997), in the case of decision trees, or the Bayesian Information Criterion, in the case of Bayesian Networks. Greedy learning strategies have several advantages. They are computationally efficient, simple to implement and often work well in practice.

While greedy learning strategies have many advantages, they are known to suffer from *myopia*. This refers to the fact that these algorithms are easily misled when the locally optimal choice may not be globally optimal. For example, consider a dataset described by a hundred Boolean features, where the target is a parity function over two of those features. A greedy decision tree learner such as ID3 using Information Gain will be very unlikely to choose the correct pair of features, because every feature at the first choice point is equally likely to be locally optimal, even though only two of them are globally optimal choices.

The myopia of greedy learning strategies such as top-down decision tree learners has traditionally been alleviated with the use of *Lookahead* (Norton, 1989). k-step Lookahead performs an exhaustive search over the next k choices that can be made by the learning algorithm, and makes the best choice over this sequence. This approach has the disadvantage that the size of the search space is exponential in k, and the search has to be repeated at each choice point. Therefore, k-step Lookahead is computationally expensive and only feasible for very small values of k. However, if the value of a choice is apparent only after more than k steps, it is possible that the choice will not be considered by the learning algorithm. Therefore, we would actually like k to be large.

Our previous work introduced an approach called *Skewing* (Page & Ray, 2003), which attempts to alleviate the myopia of greedy tree learners by changing the split selection function. We investigated "hard" Boolean functions. In such target functions, no variable has gain even given a complete data set (one copy of each possible example), or given an arbitrarily large sample drawn according to the test distribution. The Skewing approach relies on the following observation.

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

Hard functions are only hard for *some distributions.* If we are able to obtain data drawn according to a different distribution, or *skew* the data we have, hard functions can become easier to learn. Thus, given a large enough dataset, if the "skewed" distribution differs significantly from the original, it is possible to isolate the relevant features from the irrelevant ones, even when the target function is hard. Unlike Lookahead, the Skewing algorithm introduced in that previous work incurs only a constant runtime penalty over a standard tree learner, such as ID3 using Information Gain. This approach was applied to learn decision trees and significant benefits were observed in accuracy compared to ID3 using Gain when learning hard Boolean functions. The approach was able to learn hard functions of several variables given a modest amount of data. And on easier functions, skewing did not harm ID3 performance; consequently, on randomly generated function, skewing resulted in a modest but consistent improvement in performance.

The Skewing approach outlined above, however, has the flaw that its accuracy does not scale well with the number of features, when the training set size is held constant. In experiments on data sets from the UCI repository (Blake & Merz, 1998), Skewing provided only very small gains over the ID3 algorithm using Information Gain (Quinlan, 1983). We hypothesize that the difficulty with large numbers of features might be one reason why the Skewing approach was unable to discover hard targets in the UCI data sets, if in fact there were such hard targets. In the present work, we give an improved Skewing algorithm that scales much better with the number of features, when the training set size is held constant. We empirically evaluate this algorithm, comparing it the previously proposed Skewing algorithm and to ID3 with Information Gain, on both randomly generated Boolean functions and a real-world data set. Our results indicate that our algorithm (i) has accuracy that scales well with the number of features, (ii) incurs a runtime penalty that is linear in the number of variables, and hence runs in time quadratic in the number of attributes (but is still much less computationally expensive than Lookahead), and (iii) is able to effectively learn hard functions over several variables with a modest amount of training data.

In the following sections, we first review previous work on the Skewing Algorithm. Next, we describe the new *Sequential Skewing* algorithm. Then we present an empirical evaluation of the algorithms over synthetic and real-world data, and discuss the results.

Algorithm 1 Skewing Algorithm

Input: A matrix D of m data points over n Boolean variables, gain fraction G, number of trials k, skew $\frac{1}{2} < s < 1$
Output: A variable x_i to split on, or -1 if no variable with sufficient gain could be found

$N \Leftarrow$ Entropy of class variable in D
$v \Leftarrow$ Variable with max gain in D
$g \Leftarrow$ Gain of v in D
if $g < G \times N$ **then**
  $v \Leftarrow -1$
for $i = 1$ to n **do**
  $F(i) \Leftarrow 0$
{begin skewing loop}
for $t = 1$ to k **do**
  for $i = 1$ to n **do**
    $V(i) \Leftarrow$ Randomly chosen favored value for x_i
  for $e = 1$ to m **do**
    $W(e) = 1$
    for $i = 1$ to n **do**
      if $t > 1$ **then**
        if $D(e,i) = V(i)$ **then**
          $W(e) \Leftarrow W(e) \times s$
        else
          $W(e) \Leftarrow W(e) \times (1-s)$
  $N \Leftarrow$ Entropy of class variable in D under W
  for $i = 1$ to n **do**
    $E \Leftarrow$ Gain of x_i under distribution W
    if $E \geq G \times N$ **then**
      $F(i) \Leftarrow F(i) + 1$
{end skewing loop}
$j \Leftarrow \arg\max F(i)$
if $F(j) > 0$ **then**
  return x_j
else
  return v

2. Skewing Algorithm

The motivation for the Skewing procedure (Page & Ray, 2003) lies in the following observation. Consider a dataset over a hundred features, $x_1, \ldots, x_{100}$, where the target function is two variable exclusive-or, say $x_{99} \oplus x_{100}$. This task is clearly very difficult for a top-down greedy decision tree learner. Now, suppose the data are distributed differently from uniform. For example, we might suppose all variables are independent as in the uniform distribution, but every variable has probability only $\frac{1}{4}$ of taking the value 0. In this case, with a large enough sample we expect that the class distribution among examples with $x_{99} = 0$ will differ significantly from the class distribution among examples with $x_{99} = 1$. On the other hand, every variable other than x_{99} or x_{100} is likely to have nearly zero gain. Hence unless a highly unlikely sample is drawn, a greedy tree learning algorithm will choose to split on either x_{99} or x_{100}, at which point the remainder of the learning task is trivial.

The desired effect of the skewing procedure is that the skewed data set should exhibit significantly different frequencies from the original data set. To achieve this, the frequency distributions for variables are changed by attaching various weights to the existing examples in a way discussed below. Our previous work observed that, in contrast to skewing, other methods of reweighting (such as boosting) or resampling (such as bagging) did not make hard functions easier to learn (Page & Ray, 2003).

The Skewing procedure initializes the weight of every example to 1. For each variable x_i, $1 \leq i \leq n$, a "favored setting" v_i, either 0 or 1, is randomly, uniformly (independently for each variable) selected. The weight of each example in which x_i takes the value v_i is changed by multiplying it by a constant. At the end of this process, it is likely that each variable has a significantly different weighted frequency distribution than previously, as desired. But this is not guaranteed, because an unfortunate choice of settings could lead to the new frequency distribution being identical to the old one. In addition to this potential difficulty, a second difficulty is that this process can magnify idiosyncrasies in the original data by assigning some data point with an extremely high weight.

The difficulties in the preceding paragraph occur with some data sets combined with some choices of favored settings. Therefore, instead of using skewing to create only a second distribution, k additional distributions, for small k, are created. The k different distributions come about from randomly (without replacement) selecting k different combinations of favored settings for the n variables according to a uniform distribution.

Each of the n variables is scored for each of the $k+1$ weightings of the data (the original data set plus k reweighted versions of this data set). A *gain threshold* is set, and the variable that exceeds the gain threshold for the greatest number of weightings is selected as the split variable. The selected variable is highly likely to be *correct* in the sense that it is actually a part of the target function. Yet, in contrast to lookahead, the run-time has been increased only by a small constant.

Pseudocode for this algorithm is shown in Algorithm 1. It is applicable to binary-valued variables only, with nominal or continuous variables converted to multiple binary ones. The algorithm takes a parameter $\frac{1}{2} < s < 1$. The weight of an example is multiplied by s if x_i takes preferred value v_i in the example, and is multiplied by $1 - s$ otherwise. Hence, if s is $\frac{2}{3}$, the weight of every example in which x_i takes value v_i is effectively doubled relative to examples in which x_i does not take value v_i.

In our previous work, Algorithm 1 was empirically evaluated using Boolean targets of 2 to 6 variables, where the examples were described by 30 variables (the remaining variables were irrelevant to the target). This algorithm was at least as accurate as ID3 over a large, randomly sampled set of Boolean functions, and showed significantly improved accuracy when the sample was drawn from the set of "hard", parity-like Boolean functions. Nevertheless, on UCI data sets skewing provided only insignificant improvements in accuracy over ID3. While the experiments with UCI datasets demonstrated that Skewing does not hurt ID3's performance, other than a small constant cost in run-time, they raise the question of why Skewing does not help significantly on these data sets. Of course, in these data sets one does not know if the target concept is "hard" or not. But in continued work with Algorithm 1, we have observed empirically that the algorithm suffers from the shortcoming that it does not scale well with increasing numbers of irrelevant variables. Since some of the UCI data sets have large numbers of features but only a few hundred examples, we hypothesize that this might be one reason why significant improvements in accuracy over ID3 were not observed in the previous work, even if hard functions were present. The remainder of the paper attempts to address this issue.

3. Sequential Skewing

In this section, we describe the modifications we make to Algorithm 1 in order to improve its scaling properties. As before, we wish the skewed data set to exhibit significantly different frequency distributions from the original. In Algorithm 1, we achieved this by selecting a preferred value for every variable and multiplying the assigned weights. This procedure works well when the example sizes are small. However, when examples are represented by a hundred or more variables, it leads to two problems. First, on any given iteration, it is possible for some data point to get a weight value that is much larger than the others by chance. This can lead to overfitting. Second, underflow problems arise when the example sizes are large. We can avoid these problems if, instead of skewing all the variables simultaneously, we skew one variable at a time. We call this approach "Sequential Skewing". In this

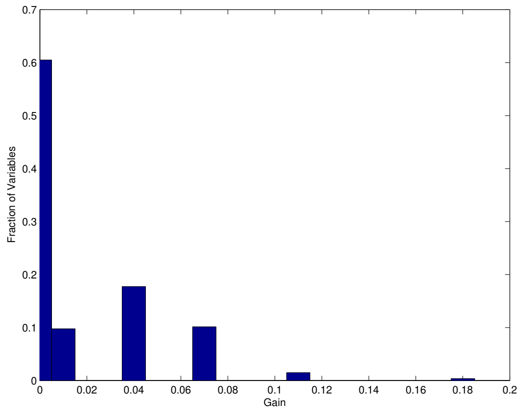

Figure 1. Histogram of gain of relevant variables for a sample of 4-variable hard Boolean targets, when a relevant variable is skewed. Examples are represented by 6 variables, two being irrelevant.

approach, we perform multiple iterations of skewing. In each iteration, we choose a single variable, x_i, and choose a preferred value for it. Each example is now reweighted according to the value taken by x_i in the example. We then calculate the gain of each variable under the new frequency distribution. The variable to split on is the variable that shows maximum gain over all the different skewed distributions.

In Figure 1, we provide empirical justification for the claim that Sequential Skewing works as expected. Here, we look at complete data sets over 6-variable examples labeled according to several random, 4-variable, parity-like Boolean targets (the other two variables are irrelevant). For such data sets, no variable has gain *a priori*. In the figure, we show a histogram of the fraction of relevant variables that have a given gain when a variable relevant to the target is skewed. We observe that, after the sequential skewing process, there are variables that have nonzero gain, and one among these variables would be chosen by the sequential skewing algorithm as the split variable. Note that given a complete data set, no variable that is *irrelevant* to the target will have gain when either a relevant or an irrelevant variable is skewed.

When the function we are trying to learn is already "easy" according to the original data distribution, and we do not have a complete data set (class assignments for every possible variable combination), the Sequential Skewing approach can sometimes choose the wrong variable. This happens when skewing a single variable causes a variable that does not appear in the target to show high gain by chance. We resolve this issue by inserting a *gain threshold*. If any variable clears this threshold in the *unweighted* data set, we pick that

Algorithm 2 Sequential Skewing Algorithm

Input: A matrix D of m data points over n Boolean variables, gain threshold f, skew $\frac{1}{2} < s < 1$
Output: A variable x_i to split on, or -1 if no variable with sufficient gain could be found

$N \Leftarrow$ Entropy of class variable in D
$v \Leftarrow$ Variable with min entropy split in D
$e \Leftarrow$ Entropy of v in D
if $e < f \times N$ **then**
  return v
if $e = N$ **then**
  $v \Leftarrow -1$
for $i = 1$ to n **do**
  $G(i) \Leftarrow 0$
$maxgain \Leftarrow 0$
{begin skewing loop}
for $t = 1$ to n **do**
  $V \Leftarrow$ Randomly chosen favored value for x_t
  for $e = 1$ to m **do**
    if $D(e,t) = V$ **then**
      $W(e) \Leftarrow s$
    else
      $W(e) \Leftarrow (1-s)$
  $N \Leftarrow$ Entropy of class variable in D under W
  for $i = 1$ to n **do**
    $E \Leftarrow$ Gain of x_i under distribution W
    if $\frac{E}{N} > maxgain$ **then**
      $maxgain \Leftarrow \frac{E}{N}$
      $maxgainvar \Leftarrow x_i$
    if $\frac{E}{N} > G(i)$ **then**
      $G(i) \Leftarrow \frac{E}{N}$
{end skewing loop}
if $maxgain = 0$ **then**
  return v
return $maxgainvar$

variable without entering the skewing procedure.

Unlike Algorithm 1, the number of iterations needed by Sequential Skewing depends on the number of variables. Thus, the time taken by this algorithm to find a split variable is $O(mn^2)$, where m is the number of examples and n is the number of variables. This is less efficient than Algorithm 1 and Information Gain, both of which are $O(mn)$, but much more efficient than k-step Lookahead, which is $O(mn^{2^k-1})$.

4. Experiments

In this section, we present experiments comparing the performance of ID3 using the Information Gain split selection function, the Skewing Algorithm described in Algorithm 1, and the Sequential Skewing Algorithm.

We present experiments using synthetic data, followed by results on a challenging biomedical classification task, that of *SH3 domain binding*. For these experiments, the parameters input to the Sequential Skewing Algorithm were $s = \frac{3}{4}$ and $f = 0.85$. The parameters input to Algorithm 1 were $s = \frac{3}{4}$, $G = 0.05$ and $k = 30$. These parameters were chosen before the experiments were performed and were held constant across all experiments. Improved results could perhaps be obtained by tuning these parameters.

Sequential skewing will perform n skews, where n is the number of variables in the examples. Because ordinary skewing always performs k skews (with $k = 30$ in our experiments), it is possible that sequential skewing will outperform ordinary skewing when $n > 30$ merely because it is permitted more skews. To control for this difference in the following experiments, when $n > 30$, we also run a variant of ordinary skewing that performs n skews rather than $k = 30$ skews. We call this variant "Skewing with n trials".

4.1. Synthetic Data Experiments

In the first set of experiments with synthetic data, examples are generated according to a uniform distribution over 30, 100 and 200 binary variables. Target functions are drawn by randomly generating DNF formulae over subsets of 6 of these variables. The number of terms in each target is drawn randomly, uniformly from between 1 and 25, and each term is drawn by choosing for each variable whether it will appear negated, unnegated, or not at all (all with equal probabilities). All targets are ensured to be satisfiable. Examples that satisfy the target are labeled *positive*, and all other examples are labeled *negative*. Figures 2 to 4 show learning curves for different example sizes. Each point on each curve is the average over 100 runs, each with a different target and with a different sample of the specified sample size. The second set of experiments is identical to the first, except that the target functions were drawn from the set of functions that can be described entirely by variable co-references, or equalities among variables, together with the standard logical connectives *and*, *or*, and *not*. For such functions, even given a complete data set, no variable has gain. Figures 5 to 7 show learning curves for different example sizes for this experiment. In each figure, "Gain" represents the results for ID3 with Information Gain, "Seq. Skewing" represents Algorithm 2, "Skewing" represents Algorithm 1, and "Skewing(n)" represents Skewing with n trials.

We find that the figures fit our expectations. We observe that on random Boolean functions, Sequential

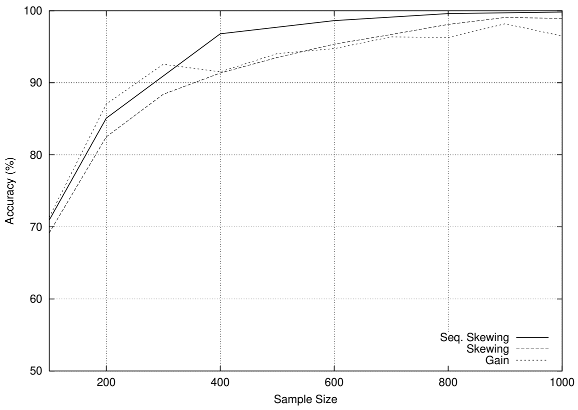

Figure 2. Random Targets, 30-variable examples

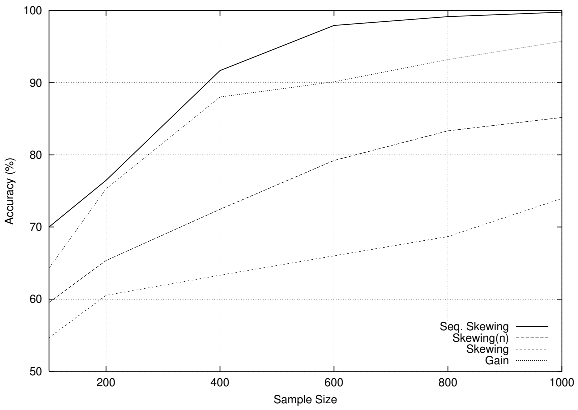

Figure 3. Random Targets, 100-variable examples

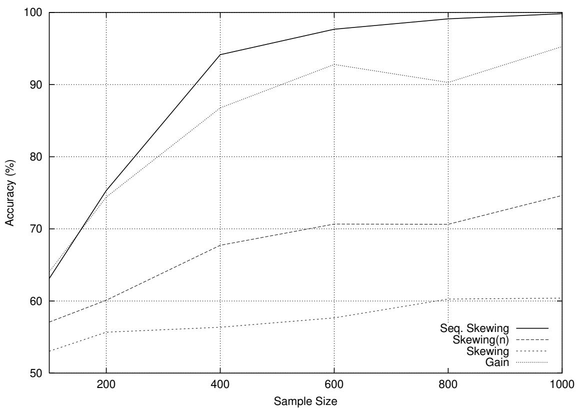

Figure 4. Random Targets, 200-variable examples

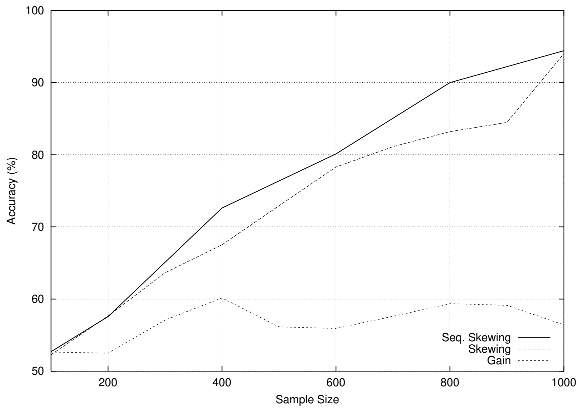

Figure 5. Hard Targets, 30-variable examples

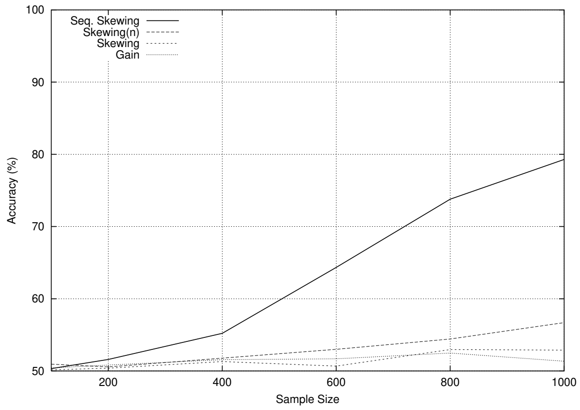

Figure 6. Hard Targets, 100-variable examples

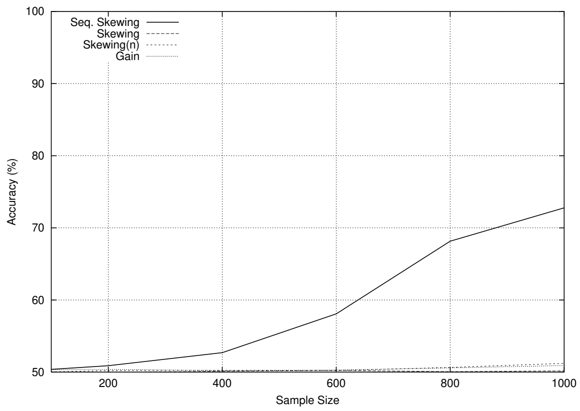

Figure 7. Hard Targets, 200-variable examples

Skewing is at least as accurate as ID3 using Information Gain, over a range of example sizes. We further observe that the accuracy of Algorithm 1 on a sample of random Boolean functions drops sharply once examples with many features are considered. This may be because of two reasons: first, the gain threshold parameter, f, of Algorithm 2 was not used in Algorithm 1, which may render this algorithm more susceptible to overfitting. Secondly, as was observed in initial Skewing work, there is a "crossover point" in terms of training set size below which Algorithm 1 performs worse than ID3 with Gain. This crossover point can be seen in Figure 2, at a sample size of 400 examples. We believe that the sample size at which the crossover occurs increases not only as a function of the target size, but also the example size. These issues contribute to the poor accuracy of Algorithm 1 as the example size increases. While permitting n trials does raise the performance of Algorithm 1, nevertheless Skewing with n trials does not perform as well as Sequential Skewing.

When the sample is drawn from problematic functions, we find that the Sequential Skewing algorithm outperforms ID3 by a large margin. We further observe that while the Skewing algorithm described in Algorithm 1 shows good accuracy when the examples are described by 30 variables, its improvement disappears quickly as the size of the examples increases. Even allowing the algorithm n iterations of Skewing does not significantly improve its accuracy in these experiments. However, the Sequential Skewing approach continues to show high accuracy in this situation. Sequential Skewing achieves the highest accuracy overall in each experiment, and retains high accuracy as the example size increases and the training set size is held constant.

4.2. SH3 Domain Binding Experiments

Here, we present results from a biomedical classification task, that of *SH3 binding*. A major part of working out the "circuitry" of an organism—the metabolic, signaling and regulatory pathways—is identifying which proteins interact with one another. Such protein-protein interactions, much like drug-receptor bindings, are based primarily on smaller electrostatic interactions (opposite charges attracting) and hydrophobic interactions (two fatty, or "water-fearing," groups of atoms interacting to keep each other from their environment).

Many of the important protein-protein interactions occur when a short segment of one protein, 6-10 amino acid residues long , here called the "ligand," interacts with a "domain" on the other protein. A domain is

Experiment	Information Gain			Skewing			Sequential Skewing		
	Acc	Wted Acc	Size	Acc	Wted Acc	Size	Acc	Wted Acc	Size
Replicated/Pruned	65.15	47.16	46.75	67.84	49.62	46.14	**74.47**	**58.84**	34.88
Replicated/Unpruned	80.6	48.8	89.71	83.5	51.9	95.2	82.3	51.6	80.14
Unreplicated/Unpruned	80.2	51.3	104	83.2	50.58	116	82.26	51.43	93.25

Figure 8. Experiment Results on the SH3 binding problem. For each splitting criterion, we show the average over 8 folds of percentage Accuracy(Acc) and Weighted Accuracy (Wted Acc), and tree size in nodes. The accuracy figures in bold show a statistically significant improvement over Information Gain at $p = 0.06$ according to a two-tailed paired t-test.

a longer segment (30-60 residues long), variations of which appear in a variety of proteins. Therefore, one way to predict protein-protein interactions is to predict what possible ligands will bind to which specific instantiations, or variations, of a given domain. Here, we investigate the binding properties of *SH3 domains*, which are implicated in cancer. Ligand-domain binding is a process where we may expect hard functions to arise naturally. For instance, binding may occur if some atoms on the domain have charges of the opposite sign to those of some atoms on the ligand, and will not occur if the charges are of the same sign.

We investigate SH3 domains from 8 proteins using data generated by an experiment performed by Sparks et al (1996). From their work, we obtain, for each SH3 domain, a complete list of ligands that bind to that domain. We then generate a sample of non-binding ligands from peptides (short sequences of amino acid residues) of length 8. These peptides are based on the same position-dependent frequency distribution as the positives, and therefore can be considered to be "near misses". Next, we align the domains and locate the positions in the domains which are believed to be important for binding, following Brannetti et al (2000). We construct each data point by juxtaposing these domain positions from each protein with a proposed ligand, and label it according to whether the ligand binds to the domain. Thus, each data point is a sequence of 33 amino acids of which the first 25 represent amino acids in the domain, and the last 8 represent amino acids in the ligand. Each amino acid is then translated into a 7-digit binary code, where each digit represents a feature of that amino acid, such as charge or hydrophobicity. The final data set consists of 897 data points, 97 positive, each data point being described by 231 binary-valued features. This is thus a fairly high-dimensional data set. The added difficulty is that the classes are substantially imbalanced.

In our experiments, we perform 8-fold cross validation as follows. For each fold, all the examples corresponding to one protein constitute the test set. The examples corresponding to the other 7 proteins form the training set.[1] Thus, on average, each training set has 785 examples, 85 of which are positive.

We compare ID3 with Information Gain as the splitting criterion against Skewing and Sequential Skewing in our experiments. We report the average accuracy, weighted accuracy and the tree size for the three methods for each experiment. Weighted accuracy is defined as the average of the true positive and true negative rates (this is equivalent to a misclassification cost that is inversely proportional to the ratio of classes). Since the domain is imbalanced, it is easy to achieve high accuracy by always predicting "negative". Thus, weighted accuracy may be the most informative measure of performance on this data set.

We perform three experiments on this domain. In our first experiment, we replicate the positive examples so that there are equal numbers of positives and negatives in the training set. Further, we hold aside a prune set of 150 examples that is used to greedily post-prune the trees generated by all algorithms. However, holding aside a prune set exacerbates the data sparsity problem.Therefore, in our second experiment, we replicate the positives, but do not hold out a prune set, or prune the trees produced. Finally, we investigate the effect of learning the trees without either replicating the positives or pruning. We present the results of these experiments in Figure 8.

In our experiments, Sequential Skewing consistently outperforms Information Gain. Because of the small size of the data set and the fact that we could only carry out 8-fold cross validation (this was dictated by the number of proteins for which we had data), we obtained statistical significance only for some of our results, according to a two-tailed paired t-test. The significant values are shown in bold in Figure 8. We note that weighted accuracy is the most important measure on this data set, and Sequential Skewing achieves

[1]Much as with protein secondary structure prediction, performing ordinary cross-validation gives overly optimistic results. To estimate performance on a new protein, one should instead perform "leave-one-protein-out" cross-validation as done here.

the best weighted accuracy overall. Further, we observe that not only does Sequential Skewing have better accuracy and weighted accuracy in general, it also constructs smaller trees on average. We believe therefore that these trees may generalize better to other domains, and provide more insight about the SH3 binding problem. Overall, Sequential Skewing with replicated positives and pruned trees provides the highest weighted accuracy on this task.

We also ran C5.0 (www.rulequest.com) on this data, with a differential cost file that stipulated a false negative misclassification penalty of 10 units. This algorithm achieved a average weighted accuracy of 46.52%, with an average tree size of 76.5 nodes. Comparing this to the Replicated/Pruned experiment, we observe that C5.0 is outperformed by Sequential skewing. The accuracy difference is significant at $p = 0.02$ according to a two-tailed paired t-test.

5. Conclusion

Functions that are difficult for greedy decision tree learners, including parity-like functions, appear in the real world. For example, in biomedical domains, cases exist where expression of a gene or survival of an organism may be an exclusive-or function of the expression of other genes or groups of genes; a particular example was described in our earlier work (Page & Ray, 2003). Ligand-receptor binding, whether for protein-protein interactions as discussed in the present paper or for drug-target interactions, is frequently controlled by hard functions of two or more variables. Skewing is a promising approach that efficiently addresses such hard functions. Previous work on skewing introduced an algorithm that did not scale sufficiently well to hundreds of features, with only hundreds of examples, to be broadly applicable to real-world problems. This paper has taken a major step forward in making skewing a practical approach to learning hard functions in real-world domains. Sequential skewing significantly improves the ability of skewing to handle large numbers (hundreds) of features with reasonable numbers (again, hundreds) of examples, as demonstrated in the present paper with both real and synthetic data. Sequential skewing increases the time complexity by a factor of n, the number of features, over ordinary skewing, but the resulting time complexity remains lower than that of even two-step lookahead.

Much room remains for research into skewing. We are looking at ways to improve the scaling behavior further to handle thousands of features with only hundreds of data points. This is desirable, for example, for working with gene expression data. Another interesting direction is to use Skewing as a feature selection algorithm. Since the full data set is used to select features, and Skewing can capture relevant features that are informative in conjunction with other features, it should outperform Gain-based selection methods. However, the disadvantage is that variables are evaluated in a context which may be different from their context in the target function, so their utility may not be apparent. We are currently evaluating this approach. Besides this, we are also investigating approaches that incorporate Skewing directly into greedy learners other than decision tree learners. Other important future directions are extending this approach to handle real-valued features, and multi-class or real-valued prediction problems. Besides the algorithmic issues, much work remains to be done in exploring the biomedical application domains in which we expect functions that are hard for greedy learners to arise.

Acknowledgements

The first author was supported by NIH Grant 1R01 LM07050-01 and by grants from the University of Wisconsin Graduate School. The second author was supported by NSF grant 9987841 and by grants from the University of Wisconsin Graduate School and Medical School. The authors thank Brian Kay and Beverly Seavey for discussions on the task of SH3 binding.

References

Blake, C., & Merz, C. (1998). UCI repository of machine learning databases.

Brannetti, B., Via, A., Cestra, G., Cesareni, G., & Helmer-Citterich, M. (2000). SH3-SPOT: an algorithm to predict preferred ligands to different members of the SH3 gene family. *Journal of Molecular Biology*, *298*, 313–28.

Norton, S. (1989). Generating better decision trees. *IJCAI-89* (pp. 800–805). Los Altos, CA: Kaufmann.

Page, D., & Ray, S. (2003). Skewing: An efficient alternative to lookahead for decision tree induction. *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, San Francisco, CA.

Quinlan, J. (1983). Learning efficient classification procedures and their application to chess end games. In R. Michalski, J. Carbonell and T. Mitchell (Eds.), *Machine learning i*, chapter 15, 463–482. Morgan Kaufmann Publishers.

Quinlan, J. (1997). *C4.5: Programs for machine learning*. Kaufmann.

Sparks, A., Rider, J., Hoffman, N., Fowlkes, D., Quillam, L., & Kay, B. (1996). Distinct ligand preferences of Src homology 3 domains from Src, Yes, Abl, Cortactin, p53bp2, PLCγ, Crk, and Grb2. *Proceedings of the National Academy of Sciences*, *93*, 1540–4.