7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), pp. 107 to 119, Dubrovnik, Croatia,2003

SMOTEBoost: Improving Prediction of the Minority Class in Boosting

Nitesh V. Chawla [1], Aleksandar Lazarevic [2], Lawrence O. Hall[3], and Kevin W. Bowyer[4]
[1] Business Analytic Solutions, Canadian Imperial Bank of Commerce (CIBC),
BCE Place, 161 Bay Street, 11th Floor, Toronto, ON M5J 2S8, Canada
[2] Department of Computer Science, University of Minnesota,
200 Union Street SE, Minneapolis, MN 55455, USA
[3]Department of Computer Science and Engineering, University of South Florida,
ENB 118, 4202 E. Fowler Avenue, Tampa, FL 33620, USA
[4]Department of Computer Science and Engineering,
384 Fitzpatrick Hall, University of Notre Dame, IN 46556, USA
nitesh.chawla@cibc.ca, aleks@cs.umn.edu, hall@csee.usf.edu, kwb@cse.nd.edu

Abstract. Many real world data mining applications involve learning from imbalanced data sets, where the particular events of interest may be very few when compared to the other classes. Learning from data sets that contain rare events usually produces biased classifiers that have a higher predictive accuracy over the majority class(es), but poorer predictive accuracy over the minority class of interest. SMOTE (Synthetic Minority Over-sampling Technique) is a recent approach that is specifically designed for learning with minority classes. Boosting is a promising ensemble-based learning algorithm that can improve the classification performance of any weak classifier. This paper presents a novel approach for learning from rare classes, based on a combination of the SMOTE algorithm and the boosting procedure. Unlike standard boosting where all misclassified examples are given equal weights, the novel SMOTEBoost approach creates synthetic examples from the minority class, thus indirectly changing the updating weights and compensating for skewed distributions. The SMOTEBoost algorithm applied to several highly and moderately imbalanced data sets shows improvement in prediction performance on the rare class compared to standard boosting and to the SMOTE algorithm used with a single classifier.

1. Motivation and Introduction

The recent, explosive growth of information available on the Web, and in business and scientific fields, has resulted in an unprecedented opportunity to develop automated data mining techniques for extracting useful knowledge from massive data sets. Despite the enormous amount of data, particular events of interest can still be quite rare. Classification of rare events is a common problem in many domains, such as fraudulent transactions, network intrusion detection, Web mining, direct marketing domains and medical diagnostics. For example, in the network intrusion detection domain, the goal is to detect illegitimate use that deviates significantly from normal behavior through constantly monitoring unusual user activity. The number of intrusions on the network is typically a very small fraction of the total traffic. The crucial step in successfully detecting intrusions is to develop a model that describes most known as well as novel unseen attacks, while keeping a low false alarm rate. The challenge of detecting future attacks has led to an increasing interest in intrusion detection techniques based upon data mining [1, 2, 3, 4]. In the example of Web mining applications, it is important to predict those web sessions that are of particular interest from the revenue standpoint. However, these sessions usually occur in a relatively small proportion of the sea of thousands of sessions occurring at a web site on a given day. In medical databases, when classifying the pixels in mammogram images as cancerous or not [5, 6], abnormal (cancerous) pixels represent only a very small fraction of the entire image. The nature of the application requires a fairly high detection rate of the minority class and allows for a small error rate in the majority class since the cost of misclassifying a cancerous patient as non-cancerous can be very high.

In all these scenarios when the majority class typically represents 98-99% of the entire population, a trivial classifier that labels everything with the majority class can achieve 98-99% accuracy. It is apparent that for domains with imbalanced and/or skewed distributions, classification accuracy is not sufficient as a standard performance measure. ROC analysis [7] and metrics such as *precision*, *recall* and *F-value* [8, 9] have been used to understand the performance of the learning algorithm on the minority class.

A confusion matrix as shown in Table 1 is typically used to evaluate performance of a machine learning algorithm. In classification problems, assuming class C as the minority/rare class of the interest, and NC as a conjunction of all the other classes, there are four possible outcomes when detecting class C.

From Table 1, *recall*, *precision* and *F-value* may be defined as follows:

$$Precision = TP / (TP + FP)$$

$$Recall = TP / (TP + FN)$$

$$F\text{-}value = \frac{(1+\beta^2)\cdot Recall \cdot Precision}{\beta^2 \cdot Recall + Precision},$$

where β corresponds to relative importance of *precision* vs. *recall* and it is usually set to 1. The main focus of all learning algorithms is to improve the *recall*, without sacrificing the *precision*. However, the *recall* and *precision* goals are often conflicting and attacking them simultaneously may not work well, especially when one class is rare. The *F-value* incorporates both *precision* and *recall*, and the "goodness" of a learning algorithm for the minority class can be measured by the *F-value*. While ROC curves represent the trade-off between values of TP and FP, the F-value basically incorporates the relative effects/costs of *recall* and *precision* into a single number.

Table 1. Confusion matrix defines four possible scenarios when classifying class "C"

	Predicted Class "C"	**Predicted Class "NC"**
Actual class "C"	True Positives (TP)	False Negatives (FN)
Actual class "NC"	False Positives (FP)	True Negatives (TN)

It is now well known in machine learning that a combination of classifiers can be an effective technique for improving prediction accuracy. As one of the most popular combining techniques, boosting [10] uses adaptive sampling of instances to generate a highly accurate ensemble of classifiers whose individual global accuracy is only moderate. In boosting, the classifiers in the ensemble are trained serially, with the weights on the training instances adjusted adaptively according to the performance of the previous classifiers. The main idea is that the classification algorithm should concentrate on the instances that are difficult to learn. Boosting has received extensive theoretical and empirical study [11, 12], but most of the published work focuses on improving the accuracy of a weak classifier on data sets with well-balanced class distributions. There has been significant interest in the recent literature for embedding cost-sensitivities in the boosting algorithm. CSB [13] and AdaCost boosting algorithms [14] update the weights of examples according to the misclassification costs. Karakoulas and Shawe-Taylor's ThetaBoost adjusts the margins in the presence of unequal loss functions [15]. On the other side, Rare-Boost [9, 16] updates the weights of the examples differently for all four entries shown in Table 1. One of our goals outlined in future research is to have a comprehensive comparison of SMOTEBoost with some of the aforementioned cost-sensitive boosting approaches.

The prevalence of class imbalance in various scenarios has caused a surge in research dealing with rare classes. Several approaches for dealing with rare classes were recently introduced [5, 7, 16, 17, 18, 19, 20, 21, 22]. SMOTE [5] is an algorithm that tackles the problem by generating synthetic minority class examples. It has been shown that in conjunction with an inductive learner, SMOTE may be a very successful technique in modeling the rare or minority classes in a data set. In this paper we propose a novel approach for learning from rare classes, SMOTEBoost, that embeds SMOTE in the boosting algorithm. After each boosting round, we apply the SMOTE algorithm in order to create synthetic examples from the minority class. Experiments performed on data sets from several domains (network intrusion detection, medical applications, etc.) have shown that SMOTEBoost is able to achieve a higher *F-value* than either SMOTE applied to a classifier or just the standard boosting algorithm for all the datasets. While on the other hand both SMOTE applied to a classifier and SMOTEBoost achieve higher *F-value* than a single classifier. We also provide a *precision-recall* analysis of the approaches.

2. Synthetic Minority Oversampling Technique - SMOTE

Researchers have dealt with class imbalance by over-sampling the minority class samples with replacement, and/or under-sampling the majority class [19, 20, 21, 22]. However, the effect of over-sampling is to identify similar but more specific regions in the feature space as the decision region of the minority class [5]. This can lead to over-fitting, with the minority class decision region becoming very specific.

SMOTE (Synthetic Minority Oversampling Technique) was proposed to counter the effect of having few instances of the minority class in a data set [5]. Operating in the "feature space" rather than the "data space" creates synthetic instances of the minority class. By synthetically generating more instances of the minority class, the inductive learners, such as decision trees (e.g. C4.5 [23]) or rule-learners (e.g. RIPPER [24]), are able to broaden their decision regions for the minority class. We deal with nominal (or discrete) and continuous attributes differently in SMOTE. In the nearest neighbor computations for the minority classes we use Euclidean distance for the continuous features and the Value Distance Metric (with the Euclidean assumption) for the nominal features [5, 25, 26]. The new synthetic minority samples are created as follows:

Algorithm *SMOTE*(*T*, *N*, *k*)
Input: Number of minority class samples *T* ; Amount of SMOTE *N*%; Number of nearest neighbors *k*
Output: (*N*/100) * *T* synthetic minority class samples

1. (* *If N is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTEd.* *)
2. **if** *N* < 100
3. **then** Randomize the *T* minority class samples
4. *T* = (*N*/100) * *T*
5. *N* = 100
6. **end if**
7. *N* = (*int*)(*N*/100) (* *The amount of SMOTE is assumed to be in integral multiples of 100* *)
8. *k* = Number of nearest neighbors
9. *numattrs* = Number of attributes
10. *Sample*[ ][ ]: array for original minority class samples
11. *newindex*: keeps a count of number of synthetic generated samples; it is initialized to 0
12. *Synthetic*[ ][ ]: array for synthetic samples
(* *Compute k nearest neighbors for each minority class sample only.* *)
13. **for** *i* ← 1 **to** *T*
14. Compute *k* nearest neighbors for *i*, and save the indices in the *nnarray*
15. *Populate*(*N*, *i*, *nnarray*)
16. **end for**

Populate(*N*, *i*, *nnarray*) (* *Function to generate the synthetic samples.* *)
17. **while** *N* ≠ 0 **do**
18. Choose a random number between 1 and *k*, call it *nn*. This step chooses one of the *k* nearest neighbors of *i*.
19. **for** *attr* ← 1 **to** *numattrs*
20. **if** *attr* == Continuous feature
21. Compute:
dif = *Sample*[*nnarray*[*nn*]][*attr*] - *Sample*[*i*][*attr*]
22. Compute:*gap* = random number between 0 & 1
23. *Synthetic*[*newindex*][*attr*] = *Sample*[*i*][*attr*] + *gap* * *dif*
24. **else**
25. *attr_value* = majority vote for the *attr* values between *i* and *nn.* If no majority then choose at random.
26. *Synthetic[newindex][attr] = attr_value*
23. **end for**
24. *newindex*++
25. *N* = *N* - 1
26. **end while**
27. **return** (* *End of Populate.* *)
End of Pseudo-Code.

Figure 1. The Synthetic Minority Oversampling Technique (SMOTE)

- For the continuous features
  - Take the difference between a feature vector (minority class sample) and one of its k nearest neighbors (minority class samples).
  - Multiply this difference by a random number between 0 and 1.
  - Add this difference to the feature value of the original feature vector, thus creating a new feature vector
- For the nominal features
  - Take majority vote between the feature vector in consideration and its k nearest neighbors for the nominal feature value. In the case of a tie, choose at random.
  - Assign that value to the new synthetic minority class sample.

Using this technique, a new minority class sample is created along the line segment joining a minority class sample and its nearest neighbor. Hence, using SMOTE, more general regions are learned for the minority class, allowing the classifiers to better predict unseen examples belonging to the minority class. A combination of SMOTE and under-sampling creates potentially optimal classifiers as a majority of points from the SMOTE and under-sampling combination lie on the convex hull of the family of ROC curves [5, 7].

3. SMOTEBoost algorithm

In this paper, we propose a SMOTEBoost algorithm that combines the Synthetic Minority Oversampling Technique (SMOTE) and the standard boosting procedure. We want to utilize SMOTE for improving the accuracy over the minority classes, and we want to utilize boosting to not sacrifice accuracy over the entire data set. The major goal is to better model the minority class in the data set, by providing the learner not only with the minority class instances that were misclassified in previous boosting iterations, but also with a broader representation of those instances. We want to improve the overall accuracy of the ensemble by focusing on the difficult minority (positive) class cases, as we want to model this class better. The goal is to improve our True Positives (TP).

The standard boosting procedure gives equal weights to all misclassified examples. Since boosting samples from a pool of data that predominantly consists of the majority class, subsequent samplings of the training set may still be skewed towards the majority class. Although boosting reduces the variance and the bias in the final ensemble [11], it might not hold for data sets with skewed class

distributions. There is a very strong learning bias towards the majority class cases in a skewed data set, and subsequent iterations of boosting can lead to a broader sampling from the majority class. Boosting (Adaboost) treats both kinds of errors (FP and FN) in a similar fashion. Our goal is to reduce the bias inherent in the learning procedure due to the class imbalance, and increase the sampling weights for the minority class. Introducing SMOTE in each round of boosting will enable each learner to be able to sample more of the minority class cases, and also learn better and broader decision regions for the minority class. By introducing SMOTE in each round of boosting, we are particularly enhancing the probability of selection for the difficult minority class cases that are dominated by the majority class points.

We also conjecture that introducing the SMOTE procedure also increases the diversity amongst the classifiers in the ensemble, as in each iteration we produce a different set of synthetic examples. The amount of SMOTE is a parameter that can vary for each data sets. It will be useful to know a priori the amount of SMOTE to be introduced for each data set. We observe that it is not really a feature of the class imbalance, and is more dependent on the distribution in the feature space. We believe that utilizing a validation set to set the amount of SMOTE before the boosting iterations can be useful.

The combination of SMOTE and the boosting procedure that we present here is a variant of the AdaBoost.M2 procedure [10]. The proposed SMOTEBoost algorithm, shown in Figure 3, proceeds in a series of T rounds. In every round a weak learning algorithm is called and presented with a different distribution D_t altered by emphasizing particular training examples. The distribution is updated to give wrong classifications higher weights than correct classifications. Unlike standard boosting, where the distribution D_t is updated uniformly for examples from both the majority and minority classes, in the SMOTEBoost technique the distribution D_t is updated such that the examples from the minority class are oversampled by creating synthetic minority class examples (**step 1**). The entire weighted training set is given to the weak learner to compute the weak hypothesis h_t. At the end, the different hypotheses are combined into a final hypothesis h_{fn}.

We used RIPPER [24], a learning algorithm that builds a set of rules for identifying the classes while minimizing the amount of error, as the classifier in our SMOTEBoost experiments. RIPPER is a noise-tolerant rule-learning algorithm based on the separate-and-conquer strategy. It gives comparable results to a decision tree learning algorithm while being more efficient. The Synthetic Minority Oversampling Technique (SMOTE) was employed with different values for the parameter N that specifies the amount of synthetically generated examples. As mentioned earlier, one of the goals outlined in the future work of this paper is identifying a priori the amount of SMOTE applicable for each data set, before initiating the boosting procedure.

- Given: Set S $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ $x_i \in$X, with labels $y_i \in$Y = {1, …, C}, where C_m, C_m < C, corresponds to a minority class.
- Let B = {(i, y): i = 1,…,m, $y \neq y_i$}
- Initialize the distribution D_1 over the examples, such that D_1(i) = 1/m.
- For t = 1, 2, 3, 4, … T
  1. Modify distribution D_t by creating N synthetic examples from minority class C_m using SMOTE algorithm
  2. Train a weak learner using distribution D_t
  3. Compute weak hypothesis h_t: X × Y → [0, 1]
  4. Compute the pseudo-loss of hypothesis h_t:
     $$\varepsilon_t = \sum_{(i,y)\in B} D_t(i,y)(1 - h_t(x_i, y_i) + h_t(x_i, y))$$
  5. Set $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$ and $w_t = (1/2)\cdot(1-h_t(x_i,y)+h_t(x_i,y_i))$
  6. Update D_t : $D_{t+1}(i, y) = (D_t(i,y) / Z_t) \cdot \beta_t^{w_t}$ where Z_t is a normalization constant chosen such that D_{t+1} is a distribution.
- Output the final hypothesis:
  $$h_{fn} = \arg\max_{y\in Y} \sum_{t=1}^{T} (\log \frac{1}{\beta_t}) \cdot h_t(x, y)$$

Figure 3. The SMOTEBoost algorithm

4. Experiments

4.1. Data sets

Our experiments were performed on the four data sets summarized in Table 2. For all data sets, except for the KDD Cup-99 intrusion detection data set, the reported values for *recall*, *precision* and *F-value* were obtained by performing 10-fold cross-validation. For the KDDCup-99 data set however, the separate intrusion detection test set was used to evaluate the performance of proposed algorithms. Unlike the KDDCup-99 intrusion data set that has a mixture of both nominal and continuous features, the remaining data sets (mammography, satimage, phoneme) have all continuous features.

The first data set that we used in our experiments was from the KDD Cup 1999 competition [27]. The competition task was to build a network intrusion detector, a

Table 2. Summary of data sets used in experiments

Data set	Number of majority class instances		Number of minority class instances		Number of classes
KDDCup-99 Intrusion	DoS	13027	U2R	136	5
	Probe	2445	R2L	1982	
	Normal	17400			
Mammo-graphy	10923		260		2
Satimage	5809		626		2
Phoneme	3818		1586		2

predictive model capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" connections. The data set represents a modification of the DARPA 1998 Intrusion Detection Evaluation Data [28] prepared by MIT Lincoln Lab and it contains a wide variety of intrusions simulated in a military network environment [29]. The entire data set contains original training data and original test data. The original raw training data corresponds to seven weeks of network traffic and contains around five million network connections. A network connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. The original test data corresponds to two weeks of network traffic and contains around 300,000 network connections. In addition to the normal network connections, the data contains four main categories of attacks:

- DoS (Denial of Service), for example, ping-of-death, teardrop, smurf, SYN flood, etc.;
- R2L (Remote to Local), unauthorized access from a remote machine, for example, guessing password;
- U2R (User to Remote), unauthorized access to local super-user privileges by a local unprivileged user, for example, various buffer overflow attacks;
- Probe, surveillance and probing, for example, port-scan, ping-sweep, etc.

The original training and the original test data set have totally different distributions due to novel intrusions introduced in the test data. Thus, for the purposes of this paper, we modified the data sets in order to make similar distributions for the training and test data. Therefore, we first merged original training and test data sets and then sampled 69,980 network connections from this merged data set in order to reduce the size of the data set. The sampling was performed only from majority classes (normal background traffic and DoS attack category), while other classes remained intact. Finally, the new train and test data sets used in our experiments were obtained by randomly splitting the sampled data set into equal size subsets. The distribution of network connections in the new test data set is given in Table 2.

The second data set used in our experiments is the mammography dataset [6] that contains 11,183 examples with six features and two classes representing calcifications (cancer) and non-calcifications (not-cancer). There are only 260 calcifications in the data set. If we look at predictive accuracy as a measure of goodness of the classifier for this case, the default accuracy would be 97.68% when every sample is labeled non-calcification. However in practice, it is highly desirable for the classifier to predict most of the calcifications correctly.

The third data set is the satimage dataset [30] that contains 6435 examples with 36 features and originally 6 classes. However, we chose the smallest class as the minority class and collapsed the remaining classes into one class as was done in [31]. This procedure gave us a skewed 2-class dataset, with 5809 majority class examples and 626 minority class examples.

Finally, the fourth data set used in our evaluation is the phoneme dataset from the ELENA project [31]. The data set contains 5404 examples with 5 features. The aim of the dataset is to distinguish between nasal sounds (majority class) and oral sounds (minority class), where 3,818 examples are from the majority "nasal" class and 1,586 examples are from the minority "oral" class.

4.2 Results

When experimenting with SMOTE and the SMOTEBoost algorithm, different values for the SMOTE parameter *N*, ranging between 100 and 500, were used for the minority classes. Since the KDD Cup'99 data set has two minority classes U2R and R2L that are not equally represented in the data set, different combinations of SMOTE parameters were investigated for these two minority classes (values 100, 300, and 500 were used for the U2R class while the values 100 and 300 were used for the SMOTE parameter for the R2L class). The values of SMOTE parameters for U2R class were higher than the SMOTE parameter values for R2L class, since R2L class is better represented in KDD-Cup 1999 data set than the U2R class (R2L has larger number of examples, and standard RIPPER achieves a *high F-value*). Our experimental results also showed that the higher values of SMOTE parameters for R2L class could lead to over-fitting and decreasing the prediction

performance on that class. The *precision*, *recall*, and *F-value* were reported for both classes separately.

The experimental results for all four data sets are presented at Figures 3 to 6 and Tables 3 to 7. It is important to note that these figures and tables report only the prediction performance for the minority classes from four data sets, since prediction of majority class was not of interest in this study and no decrease in prediction performance of majority class was observed. Moreover, we report also *precision*, which captures the FP trend.
In the figures we show *precision* and *recall* trends over the boosting iterations, alongside the F-value trends for the representative SMOTE parameter. Analyzing Figures 3 to 6 and Tables 3 to 7, it is apparent that SMOTEBoost achieved higher *F-values* than the other presented methods including standard boosting, SMOTE with the RIPPER classifier and standard RIPPER classifier, although the improvement varied with different data sets. The figures also show that SMOTEBoost consistently gives a higher *recall* than standard boosting for all the data sets. It is SMOTEBoost's apparent improvement in *recall*, while not causing a significant degradation in *precision* that improves the over-all *F-value*. Tables 3 to 7 include the *precision*, *recall*, and *F-value* for the various methods at different amounts of SMOTE (best values are given in bold). These reported values indicate that SMOTE applied with the RIPPER classifier has the effect of improving the *recall* of the minority class due to improved coverage of the minority class examples, while at the same time SMOTE causes the decrease in *precision* due to increased number of false positive examples. Thus, SMOTE is more targeted to the minority class than the standard boosting or RIPPER. On the other hand, the standard boosting is able to improve both *recall* and *precision* of a single classifier, since it gives all errors equal weights; false positives are as important as false negatives in boosting. Our goal is to embed SMOTE within the boosting procedure to additionally improve the *recall* achieved by the boosting procedure, not cause a significant degradation in *precision*, and thus increasing the *F-value*. SMOTEBoost can potentially reach a balance between *precision* and *recall* due to the utilization of both SMOTE and the boosting algorithm thus producing higher *F-values*. SMOTE as a part of SMOTEBoost allows the learners to broaden the minority class scope, while the boosting on the other hand aims at reducing the number of false positives.

Tables 3 to 7 show the precision, recall, and F-values by varying the amount of SMOTE for each of the minority classes for all four data sets used in our experiments. We report the aggregated result of 25 boosting iterations in the tables. The improvement was generally higher for the data sets where the skew among the classes was also higher. Comparing SMOTEBoost and AdaBoost.M1, for KDD-Cup'99 data set, the (relative) improvement in *F-value* for the U2R class (4.21%) was drastically higher than for the R2L class (0.61%). The U2R class was significantly less represented in the data set than the R2L class (the number of U2R examples was around 15 times smaller than he number of examples from R2L class). In addition, the (relative) improvements in *F-value* for the mammography (2.2%) and satimage (3.4%) data sets were better than for the phoneme data set (1.4%). The much lesser imbalance present in the phoneme data set causes the boosting and the SMOTEBoost to be comparable to each other, while for higher values of the SMOTE parameter *N*, boosting was even better than SMOTEBoost. Since the number of majority class examples is only twice the number of minority class examples in the phoneme data set, increasing the SMOTE parameter *N* to values larger than 200 causes that the minority class to become majority. Hence, the classifiers in the SMOTEBoost ensemble will now tend to over-learn the minority class, causing a higher degradation in *precision* for the minority class and therefore reduction in *F-value*.

Table 3: Final values for *recall*, *precision* and *F-value* for minority **U2R class** when proposed methods are applied on KDDCup-99 intrusion data set. (N_{u2r} corresponds to the SMOTE parameter for U2R class, while N_{r2l} corresponds to the SMOTE parameter for R2L class)

Method		*Recall*	*Precision*	*F-value*	***Method***		*Recall*	*Precision*	*F-value*
Standard RIPPER		57.35	84.78	68.42	Standard Boosting		80.147	90.083	84.83
SMOTE + RIPPER	N_{u2r}=100, N_{r2l}=100	80.15	88.62	84.17	SMOTE-Boost	N_{u2r}=100, N_{r2l}=100	83.8	93.4	**88.4**
	N_{u2r}=300, N_{r2l}=100	74.26	92.66	82.58		N_{u2r}=300, N_{r2l}=100	87.5	88.8	88.15
	N_{u2r}=500, N_{r2l}=100	68.38	86.11	71.32		N_{u2r}=500, N_{r2l}=100	84.6	92.0	88.1

Table 4: Final values for *recall*, *precision* and *F-value* for minority **R2L class** when proposed methods are applied on KDDCup-99 data set. (N_{u2r} corresponds to the SMOTE parameter for U2R class, while N_{r2l} corresponds to the SMOTE parameter for R2L class)

Method		*Recall*	*Precision*	*F-value*	***Method***		*Recall*	*Precision*	*F-value*
Standard RIPPER		75.98	96.72	85.11	Standard Boosting		95.46	96.83	96.14
SMOTE + RIPPER	N_{u2r}=100, N_{r2l}=100	94.50	97.45	95.95	SMOTE-Boost	N_{u2r}=100, N_{r2l}=100	96.1	96.4	96.21
	N_{u2r}=300, N_{r2l}=100	94.40	97.60	95.97		N_{u2r}=300, N_{r2l}=100	96.97	96.5	**96.73**
	N_{u2r}=500, N_{r2l}=100	92.99	97.62	95.25		N_{u2r}=500, N_{r2l}=100	96.5	96.7	96.62

Table 5: Final values for *recall*, *precision* and *F-value* for minority class when proposed methods are applied on *mammography* data set

Method		*Recall*	*Precision*	*F-value*	***Method***		*Recall*	*Precision*	*F-value*
Standard RIPPER		48.12	74.68	58.11	Standard Boosting		59.09	77.05	66.89
SMOTE + RIPPER	N = 100	58.04	64.96	61.31	SMOTE-Boost	N = 100	61.73	76.59	**68.36**
	N = 200	62.16	60.53	60.45		N = 200	62.63	74.54	68.07
	N = 300	62.55	56.57	58.41		N = 300	64.16	69.92	66.92
	N = 500	64.51	53.81	58.68		N = 500	61.37	70.41	65.58

Table 6: Final values for *recall*, *precision* and *F-value* for minority class when proposed methods are applied on *Satimage* data set

Method		*Recall*	*Precision*	*F-value*	***Method***		*Recall*	*Precision*	*F-value*
Standard RIPPER		47.43	67.92	55.50	Standard Boosting		58.74	80.12	67.78
SMOTE + RIPPER	N = 100	65.17	55.88	59.97	SMOTE -Boost	N = 100	63.88	77.71	70.12
	N = 200	74.89	48.08	58.26		N = 200	65.35	73.17	69.04
	N = 300	76.32	47.17	57.72		N = 300	67.87	72.68	**70.19**
	N = 500	77.96	44.51	56.54		N = 500	67.73	69.5	68.6

Table 7: Final values for *recall*, *precision* and *F-value* for minority class when proposed methods are applied on *phoneme* data set

Method		*Recall*	*Precision*	*F-value*	***Method***		*Recall*	*Precision*	*F-value*
Standard RIPPER		62.28	69.13	65.15	Standard Boosting		76.1	77.07	76.55
SMOTE + RIPPER	N = 100	82.18	59.91	68.89	SMOTE -Boost	N = 100	81.86	73.66	**77.37**
	N = 200	85.88	58.51	69.59		N = 200	84.86	76.47	76.47
	N = 300	89.79	56.15	69.04		N = 300	86	66.76	75.16
	N = 500	94.2	50.22	65.49		N = 500	88.46	65.16	75.04

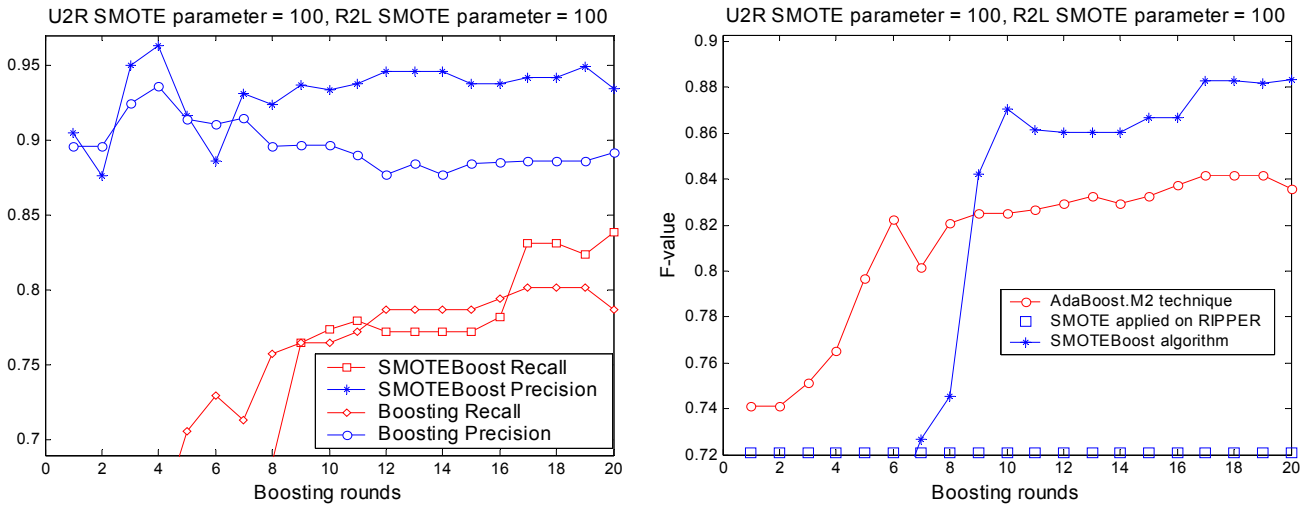

Figure 4. Overall averaged *F-values* for minority **U2R class** when the SMOTEBoost algorithm is applied on KDDCup 1999 data set

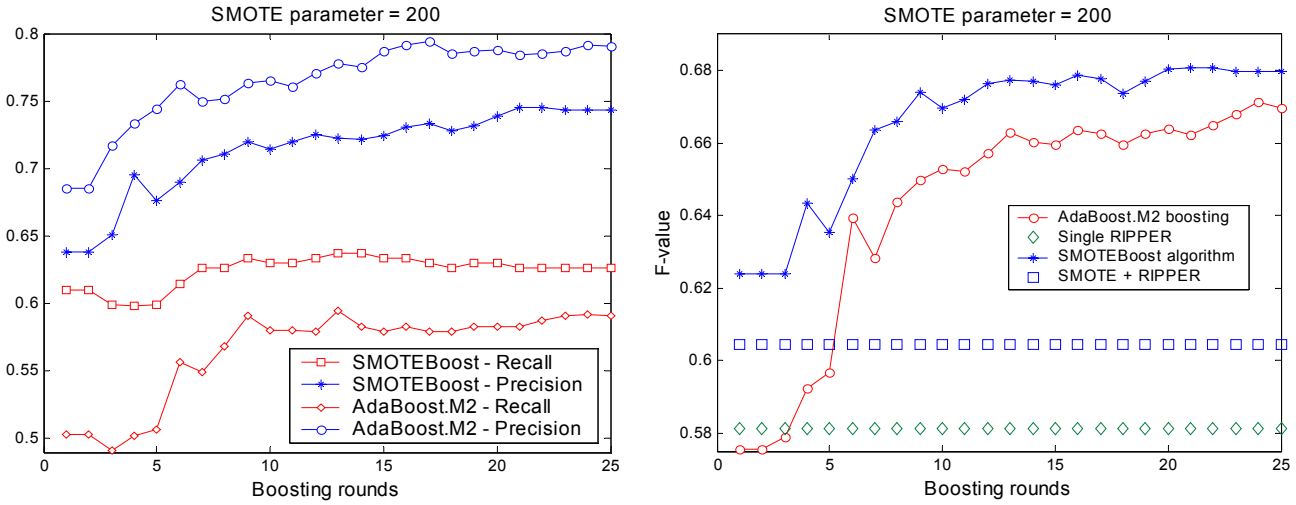

Figure 5. Overall averaged *F-values* for minority class when the SMOTEBoost algorithm is applied on mammography data set

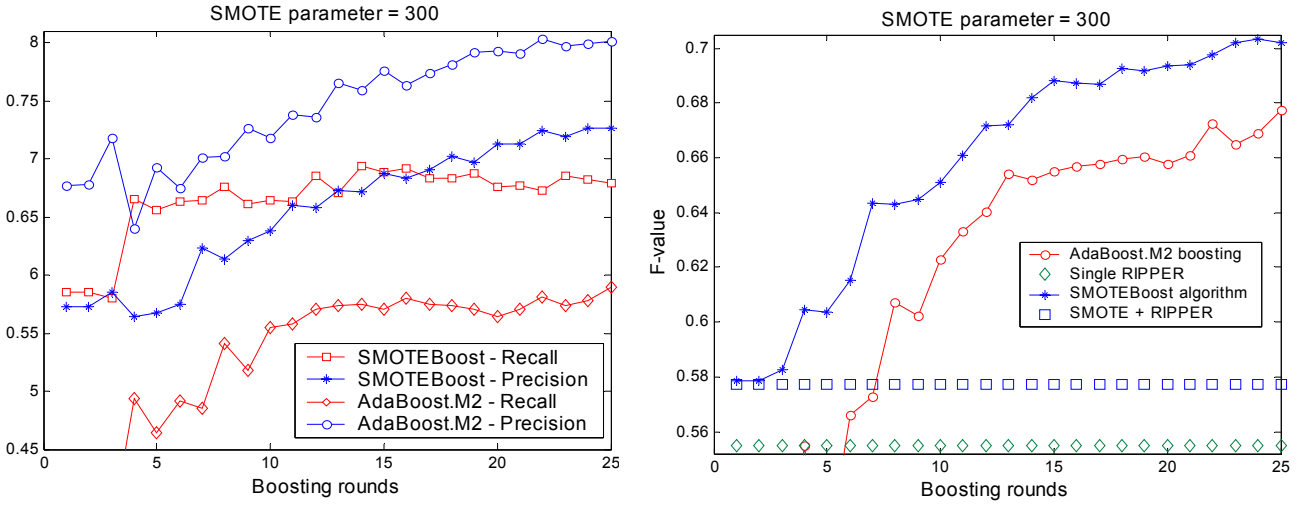

Figure 6. Overall averaged *F-values* for minority class when the SMOTEBoost algorithm is applied on Satimage data set

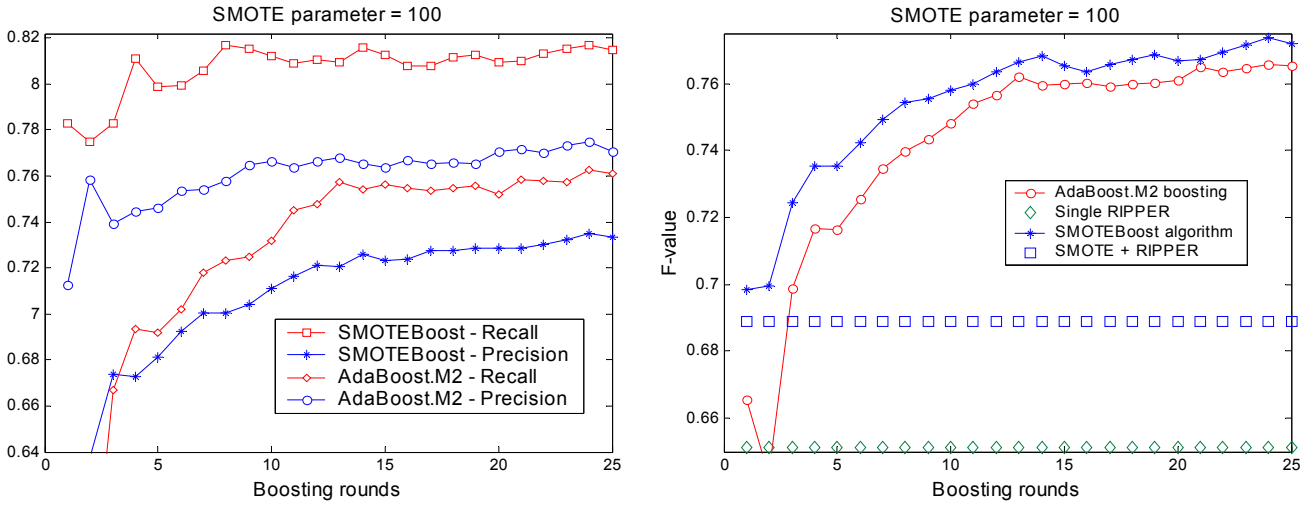

Figure 7. Overall averaged *F-values* for minority class when the SMOTEBoost algorithm is applied on phoneme data set

5. Conclusions

A novel approach for learning from the rare class is presented. The proposed SMOTEBoost algorithm is based on the integration of the SMOTE algorithm within the standard boosting procedure. Experimental results from several imbalanced data sets indicate that the proposed SMOTEBoost algorithm can result in better prediction of minority classes, without hurting the prediction performance of the majority class. Data sets used in our experiments contained different degrees of imbalance and different sizes, thus providing a diverse test bed.

The SMOTEBoost algorithm successfully utilizes the benefits from both the boosting procedure and the SMOTE algorithm. While boosting improves the predictive accuracy of classifiers by focusing on difficult examples that belong to all the classes, the SMOTE algorithm improves the performance of a classifier only on the minority class examples. Therefore, introducing SMOTE in the boosting algorithm forces the boosting algorithm to focus more on difficult examples that belong to the minority class than to the majority class. SMOTEBoost implicitly increases the weights of the misclassified minority class instances (false negatives) in the distribution D_t by increasing the number of minority class instances using the SMOTE algorithm. Therefore, in the subsequent boosting iterations SMOTEBoost is able to create broader decision regions for the minority class compared to the standard boosting.

We conclude that the SMOTEBoost can construct an ensemble of diverse classifiers and reduce the bias of the classifiers. SMOTEBoost combines the power of SMOTE in vastly improving the *recall* with the power of boosting in improving the *precision*. The overall effect is better F-value. As a part of future work, we would like to be able to dictate the amount of SMOTE for each data set. This will not only be useful when deploying SMOTE as an independent approach, but also for combining SMOTE and boosting. We would also like to compare SMOTEBoost with other cost and distribution sensitive boosting algorithms in the literature. Another open question we would like to address is performance of SMOTEBoost in the presence of mislabeling noise. Since, boosting is a weak procedure in the presence of noise, what is the effect of SMOTEBoost in that scenario? Does SMOTEBoost make boosting stronger or weaker? If we SMOTE between a noisy example (a negative example labeled as positive) and "correct" examples, will it lead to driving the noise towards the correctly labeled examples? However, in the presence of mislabeling of positive class examples as negative class examples, we believe that SMOTEBoost might mitigate the effect of that noise; it reduces the weights of the majority (negative) class cases by focusing more on the minority (positive) class cases.

6. Acknowledgements

This research was partially supported by the United States Department of Energy through the Sandia National Laboratories ASCI VIEWS Data Discovery Program contract number DE-AC04-76DO00789 and by Army High Performance Computing Research Center contract number DAAD19-01-2-0014. The content of the work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by AHPCRC and the Minnesota Supercomputing Institute.

7. References

1. W. Lee and S. J. Stolfo. Data Mining Approaches for Intrusion Detection. *Proceedings of the 1998 USENIX Security Symposium*, 1998.
2. E. Bloedorn, et al., Data Mining for Network Intrusion Detection: How to Get Started, *MITRE Technical Report*, August 2001.
3. J. Luo, Integrating Fuzzy Logic With Data Mining Methods for Intrusion Detection, *Master's thesis, Department of Computer Science, Mississippi State University*, 1999.
4. D. Barbara, N. Wu, S. Jajodia, Detecting Novel Network Intrusions Using Bayes Estimators, *First SIAM Conference on Data Mining*, Chicago, IL, 2001.
5. N. Chawla, K. Bowyer, L. Hall, P. Kegelmeyer, SMOTE: Synthetic Minority Over-Sampling Technique, *Journal of Artificial Intelligence Research*, vol. 16, 321-357, 2002.
6. K. Woods, C. Doss, K. Bowyer, J. Solka, C. Priebe, and P. Kegelmeyer, Comparative Evaluation of Pattern Recognition Techniques for Detection of Microcalcifications in Mammography, *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7(6), pp. 1417-1436, 1993.
7. F. Provost and T. Fawcett, Robust Classification for Imprecise Environments, *Machine Learning*, vol. 42/3, pp. 203-231, 2001.
8. M. Buckland and F. Gey, The relationship between recall and precision, *Journal of the American Society for Information Science*, 45(1):12--19, 1994.
9. M.Joshi, V. Kumar, R. Agarwal, Evaluating Boosting Algorithms to Classify Rare Classes: Comparison and Improvements, *First IEEE*

International Conference on Data Mining, San Jose, CA, 2001.

10. Freund, Y., and Schapire, R. E.: Experiments with a New Boosting Algorithm, *Proceedings of the Thirteenth International Conference on Machine Learning*, 325-332, 1996.
11. Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: A Statistical View of Boosting, *The Annals of Statistics*, 38(2):337-374, 2000.
12. Mason, L., Baxter, J., Bartlett, P. and Frean., M.: Function Gradient Techniques for Combining Hypotheses, *Advances in Large Margin Classifiers*, (Smola, A., Bartlett, P., Scholkopf, B. and Schuurmans, D., Eds.), MIT Press, chapter 12, 2000.
13. K. Ting, A Comparative Study of Cost-Sensitive Boosting Algorithms, *Proceedings of Seventeenth International Conference on Machine Learning*, 983-990, Stanford, CA, 2000.
14. W. Fan, S. Stolfo, J. Zhang and P. Chan, AdaCost: Misclassification Cost-Sensitive Boosting, *Proceedings of Sixteenth International Conference on Machine Learning*, Slovenia, 1999.
15. Karakoulas, G. and Shawe-Taylor, J. (1999). Optimizing Classifiers for Imbalanced Training Sets. In Kearns, M., Solla, S., and Cohn, D., editors. *Advances in Neural Information Processing Systems* 11, MIT Press.
16. M.Joshi, R. Agarwal, V. Kumar, Predicting Rare Classes: Can Boosting Make Any Weak Learner Strong?, *Proceedings of Eight ACM Conference ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, 2002.
17. M.Joshi, R. Agarwal, PNrule: A New Framework for Learning Classifier Models in Data Mining (A Case-study in Network Intrusion Detection), *First SIAM Conference on Data Mining*, Chicago, IL, 2001.
18. P. Chan, S. Stolfo, Towards Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection, *Proceedings of Fourth ACM Conference ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 164-168, New York, NY, 1998.
19. M. Kubat, R. Holte, and S. Matwin, Machine Learning for the Detection of Oil Spills in Satellite Radar Images, *Machine Learning*, vol. 30, pp. 195-215, 1998.
20. N. Japkowicz, The Class Imbalance Problem: Significance and Strategies, *Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000): Special Track on Inductive Learning*, Las Vegas, Nevada, 2000.
21. D. Lewis and J. Catlett, Heterogeneous Uncertainty Sampling for Supervised Learning, *Proceedings of the Eleventh International Conference of Machine Learning*, San Francisco, CA, 148-156, 1994.
22. C. Ling and C. Li, Data Mining for Direct Marketing Problems and Solutions, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, NY, 1998.
23. J. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufman, 1992.
24. W. Cohen, Fast Effective Rule Induction, *Proceedings of the Twelfth International Conference on Machine Learning*, Lake Tahoe, CA, 115-123, 1995.
25. C. Stanfill and D. Waltz, ``Toward Memory-based Reasoning," *Communications of the ACM*, vol. 29, no. 12, pp. 1213-1228, 1986.
26. S. Cost and S. Salzberg, ``A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features," *Machine Learning*, vol. 10, no. 1, pp. 57-78, 1993.
27. KDD-Cup 1999 Task Description, http://kdd.ics.uci.edu/databases/kddcup99/task.html
28. R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. P. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation, *Proceedings DARPA Information Survivability Conference and Exposition (DISCEX) 2000*, Vol 2, pp. 12-26, IEEE Computer Society Press, Los Alamitos, CA, 2000.
29. Defense Advanced Research Projects Agency. DARPA Intrusion Detection Evaluation, http://www.ll.mit.edu/IST/ideval/index.html
30. C. Blake and C. Merz, UCI Repository of Machine Learning Databases http://www.ics.uci.edu/~mlearn/~MLRepository.html, Department of Information and Computer Sciences, University of California, Irvine, 1998.
31. F. Provost, T. Fawcett, and R. Kohavi, The Case Against Accuracy Estimation for Comparing Induction Algorithms, *Proceedings of the Fifteenth International Conference on Machine Learning*, 445-453, Madison, WI, 1998.
32. ELENA project, ftp.dice.ucl.ac.be in the directory pub/neural-nets/ELENA/databases