# Data Reduction Using Multiple Models Integration

Aleksandar Lazarevic and Zoran Obradovic

Center for Information Science and Technology, Temple University,
Room 303, Wachman Hall (038-24),
1805 N. Broad Street, Philadelphia, PA 19122, USA
{aleks, zoran}@ist.temple.edu

**Abstract.** Large amount of available information does not necessarily imply that induction algorithms must use all this information. Samples often provide the same accuracy with less computational cost. We propose several effective techniques based on the idea of progressive sampling when progressively larger samples are used for training as long as model accuracy improves. Our sampling procedures combine all the models constructed on previously considered data samples. In addition to random sampling, controllable sampling based on the boosting algorithm is proposed, where the models are combined using a weighted voting. To improve model accuracy, an effective pruning technique for inaccurate models is also employed. Finally, a novel sampling procedure for spatial data domains is proposed, where the data examples are drawn not only according to the performance of previous models, but also according to the spatial correlation of data. Experiments performed on several data sets showed that the proposed sampling procedures outperformed standard progressive sampling in both the achieved accuracy and the level of data reduction.

## 1 Introduction

Many existing data analysis algorithms require all the data to be resident in a main memory, which is clearly untenable in many large databases nowadays. Even fast data mining algorithms designed to run in a main memory with a linear asymptotic time may be prohibitively slow, when data is stored on a disk, due to the many orders of magnitude difference between main and secondary memory retrieval time.

While data mining methods are faster when used on smaller data sets, the demand for accurate models often requires the use of large data sets that allow algorithms to discover complex structure and make accurate parameter estimates. Therefore, one of the most important data mining problems is to determine a reasonable upper bound of the data set size needed for building sufficiently accurate model. Oates and Jensen [1] found that increasing the amount of data used to build a model often results in a linear increase in model size, even when additional complexity causes no significant increase in model accuracy. Despite the promise of the better parameter estimation, models built with large amounts of data are often needlessly complex and cumbersome.

Data reduction can also be extremely helpful for data mining from very large distributed databases. In the contemporary data mining community, the majority of the

work for learning in a distributed environment considers only two possibilities: moving all data into a centralized location for further processing, or leaving all data in place and producing local predictive models, which are later moved and combined via one of the standard machine learning methods [2]. With the emergence of new high-cost networks and huge amounts of collected data, the former approach may be too expensive, while the latter too inaccurate. Therefore, reducing the size of databases by several orders of magnitude and without loss of extractable information could speed up the data transfer for a more efficient and a more accurate centralized learning.

In this paper we propose a novel technique for data reduction based on the idea of progressive sampling [3]. Progressive sampling starts with a small sample in an initial iteration and uses progressively larger ones in subsequent iterations until model accuracy no longer improves. As a result, a near-optimal minimal size of the data set needed for efficient learning an acceptably accurate model is identified. Instead of constructing a single predictor on identified data set, our approach attempts to reuse the most accurate and sufficiently diverse classifiers built in sampling iterations and to combine their predictions. In order to further improve achieved prediction accuracy, we propose a weighted sampling, based on a boosting technique [4], where the prediction models in subsequent iterations are built on those examples on which the previous predictor had poor performance. Similar techniques of active or controllable sampling are related to windowing [5], wherein subsequent sampling chooses training instances for which the current model makes the largest errors. However, simple active sampling is notoriously ill behaved on noisy data, since subsequent samples contain increasing amount of noise and performance often decrease as sampling progresses [6].

In addition, both the number and the size of spatial databases are rapidly growing, because huge amounts of data have been collected in various GIS applications ranging from remote sensing and satellite telemetry systems, to computer cartography and environmental planning. Therefore the data reduction of very large spatial databases is of fundamental importance for efficient spatial data analysis. Hence, in this paper we also propose the method for efficient progressive sampling of spatial databases, where the sampling procedure is controlled not only by the accuracy of previous prediction models but also by considering spatially correlated data points. In our approach, the data points that are highly spatially correlated are not likely to be sampled together in the same sample, since they bear less useful data information than two non-correlated data points. The objective of this approach is to further reduce the size of spatial data set and to allow more efficient learning in such domains.

The proposed sampling methods applied to several very large data sets indicate that the both a general purpose and a spatial progressive sampling technique can learn faster than the standard progressive sampling [3], and also can outperform the standard progressive sampling in the achieved prediction accuracy.

## 2 Progressive Sampling

Given a data set with $N$ examples, our goal is to determine its minimal size $n_{min}$, for which we aim to achieve a sufficiently accurate prediction model. The modification of

geometric progressive sampling [3] is used in order to maximize accuracy of learned models. The central idea of the progressive sampling is to use a sampling schedule:

$$S = \{n_0, n_1, n_2, n_3, ..., n_k\} \quad (1)$$

where each $n_i$ is an integer that specifies the size of a sample to be provided to a training algorithm at iteration $i$. Here, the $n_i$ is defined as:

$$n_i = n_0 \cdot a^i \quad (2)$$

where $a$ is a constant which defines how fast we increase the size of the sample presented to an induction algorithm during sampling iterations. The relationship between sample size and model accuracy is depicted by a learning curve (Fig. 1). The horizontal axis represents $n$, the number of instances in a given training set, that can vary between zero and the maximal number of instances $N$. The vertical axis represents the accuracy of the model produced by a training algorithm when given a training set with $n$ instances. Learning curves typically have a steep slope portion early in the curve, a more gently sloping middle part, and a plateau late in the curve. The plateau occurs when adding additional data instances is not likely to significantly improve prediction. Depending on the data, the middle part and the plateau can be missing from the learning curve, when $N$ is small. Conversely, the plateau region can constitute the majority of curves when $N$ is very large. In a recent study of two large business data sets, Harris-Jones and Haines [7] found that learning curves reach a plateau quickly for some algorithms, but small accuracy improvements continue up to $N$ for other algorithms.

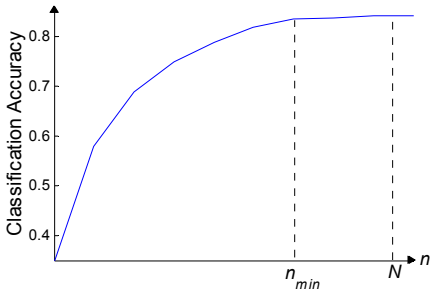


**Fig. 1.** Learning curve

The progressive sampling [3] was designed to increase the speed of inductive learning by providing roughly the same accuracy and using significantly smaller data sets than available. We used this idea to further increase the speed of inductive learning for very large databases and also to attempt to improve the total prediction accuracy.

## 3 Progressive Boosting

The proposed progressive boosting algorithm is based on an integration of Adaboost.M2 procedure [4] into the standard progressive sampling technique described at Section 2. The AdaBoost.M2 algorithm proceeds in a series of $T$ rounds. In each

round $t$, a weak learning algorithm is called and presented with a different distribution $D_t$ that is altered by emphasizing particular training examples. The distribution is updated to give wrong classifications higher weights than correct classifications. The entire weighted training set is given to the weak learner to compute the weak hypothesis $h_t$. At the end, all weak hypotheses are combined into a single hypothesis $h_{fn}$.

Instead of sampling the same number of data points at each boosting iteration $t$, our progressive boosting algorithm (Fig. 2) draws $n_t$ data points ($n_t = n_0 \cdot a^{t-1}$) according to the sampling schedule $S$ (equation 1). Therefore, we start with a small sample containing $n_0$ data points, and in each subsequent boosting round we increase the size of the sample used for learning a weak classifier $L_t$. Each weak classifier produces a weak hypothesis $h_t$. At the end of each boosting round $t$ all weak hypotheses are combined into a single hypotheses $H_t$. However, the distribution for drawing data samples in subsequent sampling iterations is still updated according to the performance of a single classifier constructed in the current sampling iteration.

- Given: Set S $\{(x_1, y_1), \ldots, (x_N, y_m)\}$ $x_i \in$ X, with labels $y_i \in$ Y = {1, …, C}
- Let B = $\{(i, y)$: $i = 1,\ldots,N, y \neq y_i\}$. Let $t = 0$.
- Initialize the distribution $D_1$ over the examples, such that $D_1(i) = 1/N$.
- *REPEAT*

1. $t = t + 1$
2. Draw a sample $Q_t$ that contains $n_0 \cdot a^{t-1}$ data instances according to the distribution $D_t$.
3. Train a weak learner $L_t$ using distribution $D_t$
4. Compute the pseudo-loss of hypothesis $h_t$:

$$\varepsilon_t = \frac{1}{2} \cdot \sum_{(i,y) \in B} D_t(i,y)(1 - h_t(x_i, y_i) + h_t(x_i, y))$$

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.
7. Combine all weak hypotheses into a single hypothesis:

$$H_t = \arg\max_{y \in Y} \sum_{j=1}^{t} (\log \frac{1}{\beta_j}) \cdot h_j(x, y)$$

- *UNTIL* (accuracy of $H_t$ is not significantly larger than accuracy of $H_{t-1}$)

8. - Sort the classifiers from ensemble according to their accuracy.
   - *REPEAT* removing classifiers with accuracy less than prespecified threshold *UNTIL* there is no longer improvement in prediction accuracy

**Fig. 2.** The progressive boosting algorithm for data reduction

We always stop the progressive sampling procedure when the accuracy of the hypothesis $H_t$, obtained in the $t$-th sampling iteration, lies in 95% confidence interval of the prediction accuracy of hypothesis $H_{t-1}$ achieved in the ($t$-1)-th sampling iteration:

$$acc(H_t) \in \left[ acc(H_{t-1}),\ acc(H_{t-1}) + 1.645 \cdot \sqrt{\frac{acc(H_{t-1}) \cdot (1 - acc(H_{t-1}))}{N}} \right] \quad (3)$$

where $acc(H_j)$ represents classification accuracy achieved by hypothesis $H_j$ constructed in $j$-th sampling iteration on the entire training set.

It is well known in machine learning theory that an ensemble of classifiers must be both diverse and accurate in order to improve the overall prediction accuracy. Diversity of classifiers is achieved by learning classifiers on different data sets obtained through weighted sampling in each sampling iteration. Nevertheless, some of the classifiers constructed in early sampling iterations may not be accurate enough due to insufficient number of data examples used for learning. Therefore, before combining the classifiers constructed in sampling iterations, we prune the classifier ensemble by removing all classifiers whose accuracy on a validation set is less than some prespecified threshold until the accuracy of the ensemble no longer improves. A validation set is determined before starting the sampling procedure as a 30% sample of the entire training data set. Assuming that the entire training set is much larger than the reduced data set used for learning, our choice of the validation sets should not introduce any significant unfair bias, since only the small fraction of data points from the reduced data set are included in the validation set. When the reduced data set is not significantly smaller than the entire training set, the unseen separated test and validation sets are used for estimating the accuracy of the proposed methods.

Since our goal is to identify a non-redundant representative subset, the usual way of drawing samples with replacement used in the AdaBoost.M2 procedure cannot be employed here. Therefore, the reminder stochastic sampling without replacement [8] is used, where the data examples cannot be sampled more than once. Therefore, as a representative subset we obtain a set of distinct data examples with no duplicates.

## 4 Spatial Progressive Boosting

Spatial data represent a collection of attributes whose dependence is strongly related to a spatial location where observations close to each other are more likely to be similar than observations widely separated in space. Explanatory attributes, as well as the target attribute in spatial data sets are very often highly spatially correlated. It is clear that data redundancy in spatial databases may be partially due to different reasons than in non-spatial data sets and therefore the standard sampling procedures may not be appropriate for spatial data sets.

In the most common geographic information science (GIS) applications the fixed-length grid is regular and therefore the standard method to determine the degree of correlation between neighboring points in such spatial data is to construct a correlogram [9]. The correlogram represents a plot of the autocorrelation coefficient computed as a function of separation distance between spatial data instances (Fig. 3). One of the main characteristics of the spatial correlogram is its range, which corresponds to a distance where spatial dependency starts to disappear, e.g. where the absolute value of the correlogram drops somewhere around 0.1.

Our spatial sampling procedure represents a modification of the proposed progressive boosting technique, described in Section 3. The general algorithm for progressive boosting, presented in Fig. 2 still remains the same, but the procedure for sampling the

data examples in subsequent sampling iterations according to the given distribution is adapted to the spatial domain data. In standard sampling without replacement [8] when the data example is sampled once, it cannot be sampled again. In our spatial modification of sampling procedure, when a data instance (shown as o in Fig. 4) is drawn once, not only that instance cannot be sampled again but also all its neighboring points, represented with □ and ◊ in Fig. 4. How many neighbors are excluded from further sampling depends on the degree of correlation and also on the number of data points required to be drawn in current sampling iteration. If the number of points needed to be sampled prevails the number of available data examples for sampling, the farthest square of points (examples denoted as ◊ in Fig. 4) is then included in the set of examples available for sampling. This allows a more uniform sampling across the spatial data set, while still concentrating on more difficult examples for learning.

The spatial progressive boosting employs the same algorithm as one shown in Fig. 2, but uses our modified spatial sampling procedure.

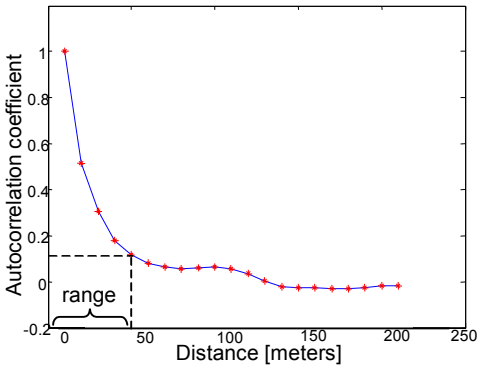


**Fig. 3.** A spatial correlogram with a 40 m range

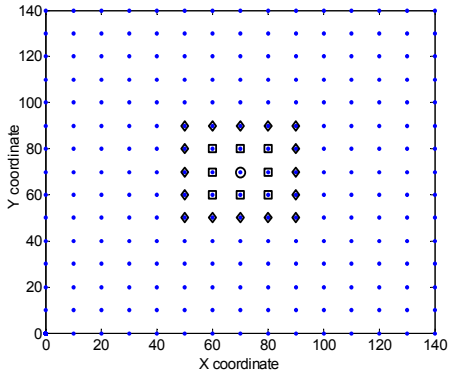


**Fig. 4.** The scheme for sampling data examples in spatial data sets

## 5 Experimental Results

An important issue in progressive sampling based techniques is the type of the model used for training through iterations. We used non-linear 2-layer feedforward neural network (NN) models that generally have a large variance, meaning that their accuracy can largely differ over different weight's initial conditions and choice of training data. In such situations using the progressive sampling procedure may effect in significant errors in the estimation of $n_{min}$. In order to alleviate the effect of neural network instability in our experiments, the prediction accuracy is averaged over 20 trials of the proposed algorithms, i.e. the sampling procedures are repeated 20 times and accuracies achieved at the same sampling round for all 20 trials are averaged. Since it is unlikely that the progressive sampling technique always stops at the same sampling iteration in each of these trials, we simply determined the number of sampling iterations in the first trial of progressive sampling technique, and all other trials for all sampling variants were repeated for such identified number of sampling iterations. To

investigate real generalization properties of built NN models, we tested our classification models on the entire training set and on an unseen data with a similar distribution.

The number of hidden neurons in our NN models was equal to the number of input attributes. The NN classification models had the number of output nodes equal to the number of classes (3 in our experiments), where the class was predicted according to the output with the largest response. Resilient propagation (RP) [10] and Levenberg-Marquardt (LM) [11] algorithms were used for learning, although better prediction accuracies were achieved using the LM learning algorithm, and only those results are reported here. The LM algorithm is a variant of Newton's method, where the approximation of the Hessian matrix of mixed partial derivatives is obtained by averaging outer products of estimated gradients. This is very well suited for small to medium-size NN training through mean squared error minimization.

We performed our experiments on several large data sets. The first data set was generated using our spatial data simulator [12] such that the distributions of generated data resembled the distributions of real life spatial data. A square shaped spatial data of size 5120 meters x 5120 meters sampled on a relatively dense spatial grid (10meters x 10 meters) resulted in 262,144 ($512^2$) training instances. The obtained spatial data stemmed from a homogeneous distribution and had five continuous attributes and three equal size classes.

The second data set was Covertype data, currently one of the largest databases in the UCI Database Repository [13]. This spatial data set contains 581,012 examples with 54 attributes and 7 target classes and represents the forest cover type for 30 x 30 meter cells obtained from US Forest Service (USFS) Region 2 Resource Information System [14]. In Covertype data set, 40 attributes are binary columns representing soil type, 4 attributes are binary columns representing wilderness area, and the remaining 10 are continuous topographical attributes. Since training of a neural network classifier would be very slow if using all 40 attributes representing a soil type variable, we transformed them into 7 new ordered attributes. These 7 attributes were determined by computing relative frequencies of each of 7 classes in each of 40 soil types. Therefore, instead of using a single value for representing each soil type, we used a 7-dimensional vector with values that could be considered continuous and therefore more appropriate for use with neural networks. This resulted in the transformed data set with 21 attributes.

The experiments were also performed on Waveform and LED data sets from the UCI repository [13]. For the Waveform set, 100,000 instances with 21 continuous attributes and three equally sized classes were generated, while for the LED data set 50,000 examples were generated for training and 50,000 examples were generated for testing. Both training and test data sets had 7 binary attributes and 10 classes.

We first performed progressive sampling on all data sets, where in the schedule given in equation (2) we used $a = 2$. Therefore, randomly chosen data samples in subsequent sampling iterations were always twice larger than samples drawn in the previous iterations. Since in our sampling procedures all classifiers constructed in all previous sampling iterations are saved and together with the classifier from the current iteration are combined, we also used the *progressive bagging* scheme, where the classifiers constructed on randomly selected, progressively larger data samples were combined into an ensemble using the same combining weights. Finally, we performed our

proposed progressive boosting technique for data reduction on all sets. The improvement of classification accuracy during the sampling iterations on all considered data sets is shown at Fig. 5.

In order to better compare our proposed sampling techniques with the progressive sampling, we stopped them in the same sampling iteration as we stopped the progressive sampling. In this way, we are able to examine two effects of data reduction techniques. First, we can observe what are the possible improvements in classification accuracy when the same size of data sample, necessary for constructing a sufficiently accurate model in progressive sampling, is used. Second, we are able to compare the level of data reduction by evaluating the sizes of data samples for which we achieve the same classification accuracy. The possible savings in processing time were not reported due to lack of space, although these savings are proportional to the level of data reduction since the time for training NN models is proportional to data set size. All results in Fig. 5 are shown starting from the second or third sampling iteration since all the methods achieved the similar accuracies in a first few iterations.

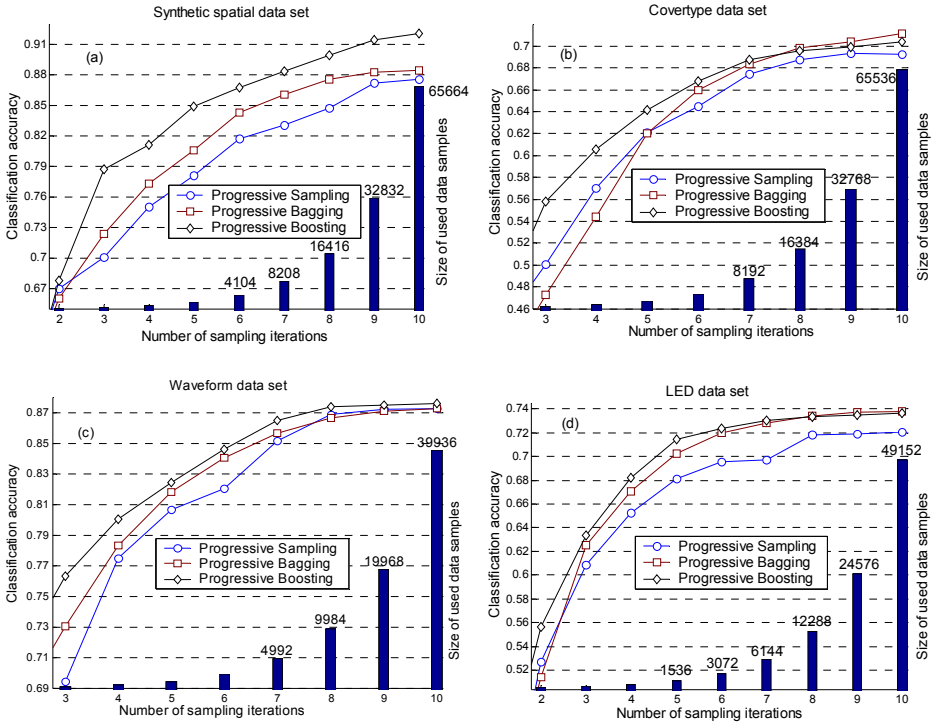


**Fig. 5.** The classification accuracy as a function of sample size for different progressive sampling techniques on four domains

Analyzing the charts in Fig. 5, it is evident that the sampling methods involving the proposed model integration showed improvements both in prediction accuracy and in achieved data reduction as compared to the standard progressive sampling. The improvement in achieved final prediction accuracy was evident for synthetic spatial,

Covertype and LED data set, while the experiments performed on Waveform data sets resulted in similar final prediction accuracy for all proposed variants of sampling techniques probably due to high homogeneity of data. However, during the sampling (iterations 3 to 7, Fig. 5) progressive boosting was consistently achieving better prediction accuracy than progressive bagging, although this difference was fairly small. The dominance of progressive boosting can be explained by the fact that the sampling procedure employed in progressive boosting attempted to rank sampling data examples from those that are more difficult for learning to those that are easier. Therefore, all advantages of standard boosting were also integrated in our progressive boosting technique.

It is also evident that for the same level of data reduction (the same sampling iteration that corresponds to training data of the same size) the achieved prediction accuracy was significantly higher when using progressive boosting and even progressive bagging instead of standard progressive sampling (Fig. 5). In addition, the same prediction accuracy was achieved with much smaller data sets when using progressive boosting and bagging for data reduction instead of relaying on standard progressive sampling. For example, the prediction accuracy on the synthetic spatial data (Fig. 5a) that was achieved by progressive sampling technique with 65,664 examples (10 iterations), was also achieved by the progressive boosting with 8,208 examples (7 iterations). Hence, the gain of these three iterations was an about eight times smaller data set needed for progressive boosting as compared to the progressive sampling.

The level of data reduction for different sampling techniques may be compared if we measure the minimum data sets needed for achieving the same accuracy. This prediction accuracy is determined when no further significant improvements in accuracy, obtained by progressive sampling, is observed. For easier comparison, the size of a reduced data set used to obtain this accuracy by progressive sampling served as a basic reduction level, and then we compared the enhancements of other data reduction techniques. Table 1 shows the level of data reduction for three used data sets.

**Table 1.** The size of the data sets used for successful learning and their percentage of the original data set size when different sampling techniques are employed

| **Method ↓ Data set→** | Synthetic Spatial | Covertype | LED | Waveform |
|---|---|---|---|---|
| Progressive Sampling | 65,664 (25.1 %) | 32,768 (5.6 %) | 12,288 (25 %) | 9,984 (9.9%) |
| Progressive Bagging | 16,416 (6.3 %) | **8,192 (1.4 %)** | **3,072 (6.1 %)** | 9,984 (9.9%) |
| Progressive Boosting | **8,208 (3.1 %)** | **8,192 (1.4 %)** | **1,536 (3.1 % )** | 9,984 (9.9%) |

It is evident from Table 1 that both sampling methods with model integration achieved better reduction performance than the standard progressive sampling. In model integration methods the reduced data set was four to eight times smaller than the reduced data set identified through standard progressive sampling. The only exception was the reduction of Waveform data sets (Table 1), where no additional reduction was achieved by combining different classifiers again due to high homogeneity of data. Nevertheless, when employing progressive boosting and progressive bagging techniques, there is an additional requirement to store all the previously constructed classifiers, or to save all data sets used for constructing these classifiers. Usually, storing only the constructed classifiers is beneficial when

ally, storing only the constructed classifiers is beneficial when employing an ensemble to make a prediction on an unseen data set with a similar distribution. However, very often there is a need for storing all necessary data examples needed for constructing all the classifiers. Since we use geometric progressive sampling, where the data sample in subsequent sampling iteration is twice larger from the sample used at the previous iteration, the total size of all previous data samples cannot be larger than the size of the data sample used in the current sampling iteration. Therefore, even in this case, according to Table 1 we can still achieve a better level of data reduction than the standard progressive sampling.

We also performed experiments with pruning inaccurate classifiers constructed in progressive boosting iterations (Fig. 6). For geometric sampling schedule we again used $a = 2$. When pruning inaccurate classifiers, we always eliminated those classifiers that harmed the overall classification accuracy on the validation set. Again, the accuracies on the entire training set are shown starting from second iteration, since there was no pruning at the first iteration (Fig. 6).

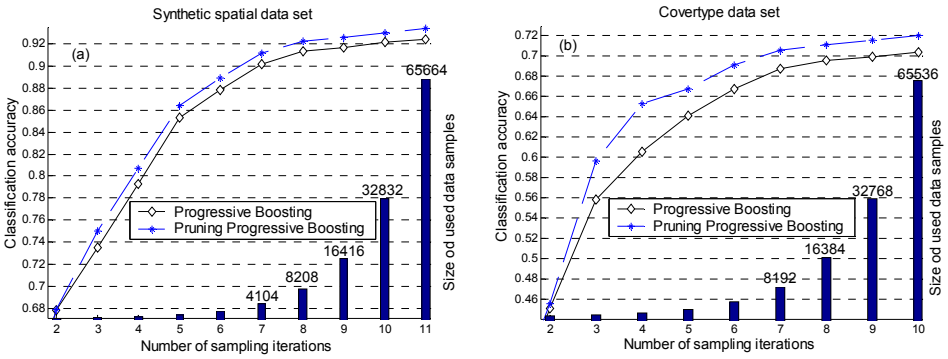


**Fig. 6.** The classification accuracy during the sampling iterations of progressive boosting and pruning progressive boosting

Results from the experiments presented in Fig. 6 indicate that pruning progressive boosting outperformed the progressive boosting technique both in achieved accuracy and in the level of data reduction for synthetic spatial and Covertype data sets. The enhancements of pruning progressive boosting on Waveform and LED data sets was insignificant as compared to the progressive boosting technique, and therefore these results are not presented here. It is evident from Fig. 6 that for synthetic spatial and Covertype data set the same prediction accuracy may be achieved much faster when pruning classifiers than without pruning. For example, accuracy of 92% for synthetic spatial data set was achieved by progressive boosting without pruning with 65,664 examples (iteration 11), while similar accuracy was achieved when pruning progressive boosting with 8,208 example (iteration 8), thus resulting in an eight times smaller data set. The same results can be observed for Covertype data set, where pruning progressive boosting again caused eight times smaller data set for the comparable prediction accuracy.

Finally, we performed the experiments for sampling spatial data using our proposed technique for spatial progressive boosting. Since the positions of data examples in-

cluded in the form of x and y coordinates were only available for the synthetic spatial data set, but not for Covertype data set, the results are reported only for the synthetic spatial data (Fig. 7). The shown accuracy starts from the third sampling iteration due to similar performance of spatial and non-spatial sampling in the first two iterations.

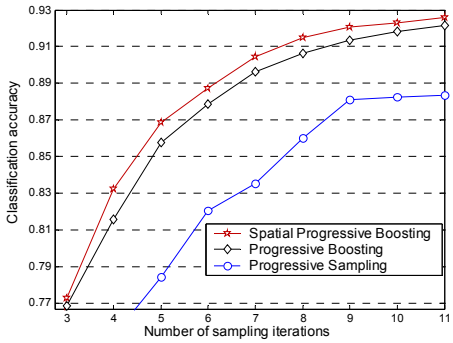


**Fig. 7**. The classification accuracy during the sampling iterations of spatial progressive boosting and standard progressive boosting on synthetic spatial data set

Fig. 7 shows that the spatial progressive boosting method, starting from the fourth iteration outperformed the regular progressive boosting in achieved prediction accuracy. In addition, for achieving accuracy of 92%, spatial progressive boosting needed four times smaller data set than the regular progressive boosting. One of the reasons for such a successful reduction of this data set is possibly in its high spatial correlation among observed attributes and a relatively dense spatial grid (10 x 10 meters).

## 6 Conclusions

Several new sampling procedures based on the progressive sampling idea are proposed. They are intended for an efficient reduction of very large and possibly spatial databases. Experimental results on several data sets indicate that the proposed sampling techniques can effectively achieve similar or even better prediction accuracy while obtaining a better data reduction than the standard progressive sampling technique. Depending on the data set, accuracy comparable to relying on the whole data set was achieved using 1.4% to 6.1% of the original data.

The question that naturally arises from this paper is a possible gain when comparing the proposed sampling techniques with the procedure of first performing the progressive sampling and then applying some of the methods for combining classifiers (bagging, boosting). First, our sampling techniques are faster since they do not require additional algorithm of combining classifiers. Second, our algorithms provide a better diversity of combined classifiers, since during the sampling iterations some of the instances difficult for learning were naturally included in the reduced data set by our algorithms while these may not be included in a final data set when performing standard progressive sampling. Finally, when using our algorithms, only a small number

of data examples that are relatively easy for learning will be included in the reduced data set, unlike the progressive sampling where this number cannot be controlled. Our future work will address the significance of the difference between these two methods.

One of the possible drawbacks of our proposed sampling techniques that will be also carefully investigated in our future work, is an increased time required for controlled sampling as compared to random sampling. For reduction of heterogeneous data sets we are currently experimenting with radial basis functions, while for spatial data reduction different similarity information will be explored. In addition, we are also extending the proposed methods to regression-based problems.

**Acknowledgment.** The authors are grateful to Dragoljub Pokrajac for providing synthetic data and for his useful comments. Work in part supported by INEEL LDRD Program under DOE Idaho Operations Office Contract DE-AC07-99ID13727.

## 7 References

1. Oates, T., Jansen, D.: Large Datasets Lead to Overly Complex Models: An Explanation and a Solution, *Proc. Fourth International Conference On Knowledge Discovery and Data Mining*, (1998), 294-298
2. Grossman R, Turinsky A. A Framework for Finding Distributed Data Mining Strategies That Are Intermediate Between Centralized Strategies and In-Place Strategies, *KDD Workshop on Distributed Data Mining*, (2000)
3. Provost, F., Jensen, D., Oates, T.: Efficient Progressive Sampling, *Proc. Fifth Int'l Conf. On Knowledge Discovery and Data Mining*, (1999), 23-32
4. Freund, Y., and Schapire, R. E.: Experiments with a New Boosting Algorithm, in *Proc. of the 13th International Conference on Machine Learning*, (1996) 325-332
5. Quinlan, J. R.: Learning Efficient Classification Procedures and their Application to Chess and Games, In Michalski, R., Carbonell, J., Mitchell, T. (eds.): Machine Learning. An Artificial Intelligence Approach, (1983), 463-482
6. Fürnkranz, J.: Integrative windowing, *J. Artificial Intelligence & Research* 8, (1998), 129-164
7. Harris-Jones, C., Haines, T.: Sample Size and Misclassification: Is More Always Better?, *Proc. Second International Conference on the Practical Application of Knowledge Discovery and Data Mining*, (1998)
8. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Reading, MA, (1989)
9. Cressie, N.A.C., *Statistics for Spatial Data,* John Wiley & Sons, Inc., New York, 1993.
10. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm, *Proc. of the IEEE International Conference on Neural Networks*, (1993), 586–591
11. Hagan, M., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks* 5, (1994) 989-993
12. Pokrajac D, Fiez T, Obradovic Z.: A Spatial Data Simulator for Agriculture Knowledge Discovery Applications, in review
13. Murphy, P.M., Aha, D.W., *UCI Repository of Machine Learning Databases,* Department of Information and Computer Science, University of California, Irvine, CA, (1999)
14. Blackard, J., *Comparison of Neural Networks and Discriminant Analysis in Predicting Forest Cover Types*, Ph.D. dissertation, Colorado State University, Fort Collins, (1998)