Biclustering-Driven Ensemble of Bayesian Belief Network Classifiers for Underdetermined Problems

Tatdow Pansombut[1,2], **William Hendrix**[1,2], **Zekai J. Gao**[2,3],
Brent E. Harrison[1,2], **and Nagiza F. Samatova**[1,2,*]

[1]Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695-8206

[2]Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, TN 37831

[3]Department of Computer Science
Zhejiang University
Hangzhou, P.R. China

[*] Corresponding author: samatovan@ornl.gov

Abstract

In this paper, we present BENCH (Biclustering-driven ENsemble of Classifiers), an algorithm to construct an ensemble of classifiers through concurrent feature and data point selection guided by unsupervised knowledge obtained from biclustering. BENCH is designed for underdetermined problems. In our experiments, we use Bayesian Belief Network (BBN) classifiers as base classifiers in the ensemble; however, BENCH can be applied to other classification models as well. We show that BENCH is able to increase prediction accuracy of a single classifier and traditional ensemble of classifiers by up to 15% on three microarray datasets using various weighting schemes for combining individual predictions in the ensemble.

1 Introduction

Ensemble classification is a technique that combines the predictions of some number of "base" classifiers in some manner to produce an aggregate prediction. Ensemble classifiers have been successfully used to increase the stability of "unstable" or "weak" classifiers [Aljamaan and Elish, 2009] [Tao *et al.*, 2006] [Breiman, 1996] [Waske *et al.*, 2010] [Sohn and Dagli, 2003] [Diaz-Uriarte and de Andres, 2006] [Tu *et al.*, 2009] [Opitz and Maclin, 1999] [Bauer and Kohavi, 1999]; to decrease the expected error by reducing the bias or the variance of the predictions [Bauer and Kohavi, 1999]; and to handle high-dimensional [Tao *et al.*, 2006] or underdetermined [Waske *et al.*, 2010] [Tao *et al.*, 2006] datasets.

Typically, ensemble classification is done by creating several classifiers in one of two ways: 1) random sampling of the data points, as in bagging [Breiman, 1996] or boosting [Schapire, 1990], or 2) random sampling of the features, as in random forests [Ho, 1998]. Sampling from the data point space reduces the variance of "unstable" classifiers. In addition, sampling from the feature space has been shown to increase the prediction accuracy of classifiers for a large number of features [Waske *et al.*, 2010].

Several methods have been developed recently that aim to improve the prediction accuracy of an ensemble of classifiers. DECORATE (Diverse Ensemble Creation by Oppositional Relabeling of Artificial Training Example) [Melville and Mooney, 2003] increases the diversity of an ensemble by adding different randomly constructed data points into the training set. Davidson [Davidson, 2004] proposes "bootstrap model averaging," a method that sums the joint probability of the instances and classes and predicts according to the most probable class. COPEN (pairwise COnstraint Projection based ENsemble) [Zhang *et al.*, 2008] couples traditional bagging and boosting with a new resampling method. Instead of directly resampling the data points in the training set, it resamples based on pairwise constraints that specify whether two data points are in the same class or not.

In some real world problem domains, learning to build classifiers with high prediction accuracy can be a difficult task due to the nature of the data available. For example, in biology, recent advances in DNA microarray technology allow the expression levels of thousands of genes to be measured simultaneously under different experimental conditions [Kim and Cho, 2006] [Harrington *et al.*, 2000], so the resulting data contains many more features or genes than data points or experiments. These types of problems, where there are many more features than data points, are called underdetermined or under-constrained problems. Recently, development of classification methods for underdetermined problems, such as those seen in DNA microarray data analysis, has been an active research area (see, for example, [Kim and Cho, 2006], [Peng, 2006], and [Kim and Cho, 2008]).

Underdetermined problems are hard to learn. One reason is that classifiers may take an unreasonably large amount of computing power to analyze the large number of features in the data. For example, the computational complexity of learning Bayesian Belief Network (BBN) structure and parameters [Cooper and Herskovits, 1992] grows superexponentially in number of features [Murphy, 2002]. Also, having too many features can create noise that prevents the learning algorithm from identifying the features that are discriminating with respect to the target concept.

One solution is to apply existing ensemble methods, such as random forests, to reduce the number of features in the feature space; however, methods that select features randomly may produce unnecessary noise. Another solution might be to select only features that are closely related to the target function using some feature score-based methods, but this solution may not be suitable for biological domain because biological systems and their relationships to the environments are complex [Reiss *et al.*, 2006]. For example, in genetic regulatory networks, genes may be co-expressed across just a subset of experimental conditions [Reiss *et al.*, 2006]. Therefore, we need to take into account not just the values across a subset of features but also a subset of conditions.

Biclustering is an unsupervised learning technique that uncovers common patterns that exist across subsets of features and data points [Tanay *et al.*, 2005]. Biclustering methods have been successfully applied in the biological domain, specifically when applied to network reconstruction and microarray analysis [Bonneau *et al.*, 2006].

To address the problem, in this paper, we present BENCH (Biclustering-driven ENsemble of Classifiers), a method to build an ensemble of classifiers for binary classification problems using knowledge obtained from biclustering. BENCH divides the original dataset into biclusters, which enables it to perform concurrent feature and data point selection. Each bicluster becomes a candidate dataset for constructing a base classifier in the ensemble.

Since biclustering is an unsupervised learning technique, its output (a set of biclusters) does not contain any class information; however, when constructing a classifier, the class label information is essential. BENCH solves this problem by analyzing how well the biclusters correlate with the class labels and by forming candidate datasets to construct classifiers based on bicluster features that distinguish the classes well or distinguish them poorly.

The first type of candidate dataset is one that is enriched with either mostly positive or mostly negative class labels, which we call a *homogeneous* bicluster. Biclusters that are enriched with one class label likely have features that are discriminating with respect to the class label. These features are the ones we select to build a classifier; however, in order to build a good classifier, we need to have both positive and negative class examples. BENCH solves this problem by combining a positively enriched bicluster with a negative one. BENCH identifies positive and negative homogeneous biclusters with overlapping features and constructs a candidate dataset by taking the intersection of the two (discriminating) feature sets and the union of the data point sets. This combined dataset is considered a Feature Inclusion (FI) candidate dataset.

The second type of candidate dataset is one that is enriched with both class labels equally, which we call a *heterogeneous* bicluster. The features in these datasets are ones that are non-discriminating with respect to the class label. Ideally, we would like to remove these types of features. BENCH combines heterogeneous biclusters by taking the complement of the union of the two feature sets and the union of the data points. This approach collects non-discriminating features and forms the Feature Exclusion (FE) candidate datasets by removing these features.

BENCH is not dependent on the type of classifier used, but in this paper we focus on the use of BBNs as our base classifiers. We have chosen BBNs because they have many good characteristics. BBNs are among the most widely used probabilistic models for learning under uncertainty. Furthermore, they are capable of dealing with incomplete data, they are robust against overfitting, they are able to make probabilistic predictions (a property we use later when assigning weights to the classifiers), and they are able to incorporate *prior* knowledge into the learning process. Most importantly, the structure of BBNs can reveal the dependency relationships that exist between features in the problem domain.

1.1 Related Work: BBN Ensemble Classification

In this paper, we show that our method of constructing an ensemble BBN classifier is able to increase the prediction accuracy in microarray datasets by 15% relative to a single BBN and traditional ensemble classifiers. In addition, we are able to reduce the learning time by three orders of magnitude when using only FI classifiers. This result is a significant improvement in the area of BBN learning, especially in light of previous research to improve BBN learning. In this section, we discuss some previous works on ensemble methods to increase the prediction accuracy of BBNs.

There has been some research done in applying ensemble learning techniques to Bayesian classifiers. In 2003, Tsymbal, Puuronen, and Patterson [Tsymbal *et al.*, 2003] developed an ensemble classifier for naïve Bayesian classifiers that randomly samples the feature space. They improved upon this method, however, by introducing an interactive refinement technique in which each random subset of the feature space is iteratively improved in order to improve the overall prediction accuracy of the ensemble. They compared the performance of their ensemble of classifiers on a set of 21 datasets from the UCI database against a single naïve Bayesian classifier. In 14 out of the 21 datasets, the ensemble of classifiers performed better than the single classifier. However, their method is limited to naïve BBN classifiers, which makes an assumption about the independence relationships among the features. This assumption is rarely true in biological domain. In addition, none of the 21 datasets used in their experiments are underdetermined.

Wang *et al.* [Wang *et al.*, 2003] applied ensemble classifiers to mining data streams. Data streams consist of data that is constantly being read in from a streaming source. In their method, they divide the data stream into chunks and then use these chunks to train the classifiers. Wang *et al.* also use the predicted error rate to determine the weight of each derived classifier in the ensemble. This method can be viewed as sampling data points from the full streaming dataset in order to create an ensemble. In their experiments, they used several types of base classifiers, naïve Bayesian networks, RIPPER, and the C4.5 decision tree algorithm, on a dataset containing credit card transaction records for a one year period. They show that, with enough classifiers, the ensemble of classifiers will outperform single classifiers with respect to both speed and accuracy. However, this method has not been tested for underdetermined problems.

In 2008, Jing *et al.* [Jing *et al.*, 2008] combined parameter boosting with structure learning to improve the classification accuracy of BBN classifiers. They construct an ensemble of BBN classifiers, starting with an empty set, and the algorithm goes though fixed number of iterations or stops if some criterion is met. At the beginning of each iteration, a training set and the set of corresponding weights for the data points are given to the TAN algorithm to build a BBN classifier. For base classifier i, the TAN algorithm adds the i edges with the highest mutual information to a naïve BBN. The training error of the resulting TAN classifier is then used to determine the weight of the test data points in the next iterations. The algorithm will stop at this point if the training error increases. Jing *et al.* test their ensemble of classifiers using UCI datasets as well as two artificial datasets. According to their results, their boosted BBNs have comparable or reduced average testing error than naïve BBN, TAN, and ELR on the 23 UCI datasets and the simulated datasets. Their method is, however, not designed to handle underdetermined problems.

In each of these methods, researchers applied ensemble techniques, be it sampling the feature space or sampling data points to show an increase in prediction accuracy over single classifiers (naïve Bayes classifiers or BBN classifiers); however, none of the methods concurrently select from both feature and data point spaces or demonstrate success on underdetermined data like BENCH.

2 Method

2.1 Overview

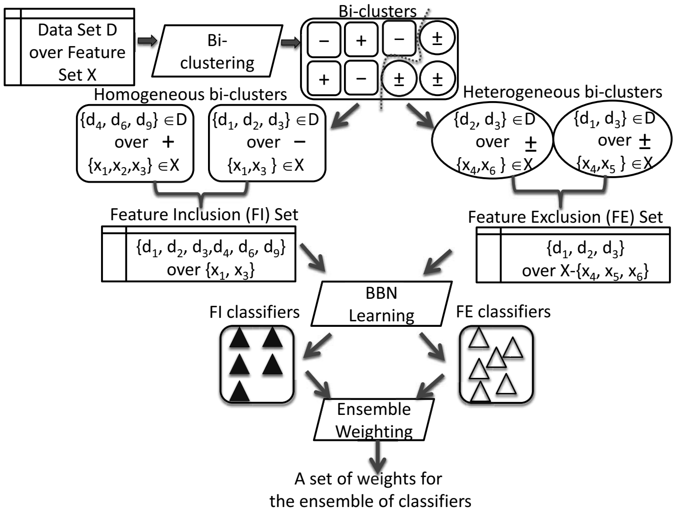

Data Set D over Feature Set X
Bi-clustering
Bi-clusters
−
+
−
±
+
−
±
±
Homogeneous bi-clusters
Heterogeneous bi-clusters
$\{d_4, d_6, d_9\} \in D$ over + $\{x_1, x_2, x_3\} \in X$
$\{d_1, d_2, d_3\} \in D$ over − $\{x_1, x_3\} \in X$
$\{d_2, d_3\} \in D$ over ± $\{x_4, x_6\} \in X$
$\{d_1, d_3\} \in D$ over ± $\{x_4, x_5\} \in X$
Feature Inclusion (FI) Set
$\{d_1, d_2, d_3, d_4, d_6, d_9\}$ over $\{x_1, x_3\}$
Feature Exclusion (FE) Set
$\{d_1, d_2, d_3\}$ over X-$\{x_4, x_5, x_6\}$
BBN Learning
FI classifiers
FE classifiers
Ensemble Weighting
A set of weights for the ensemble of classifiers

Figure 1: BENCH Learning Algorithm method flowchart

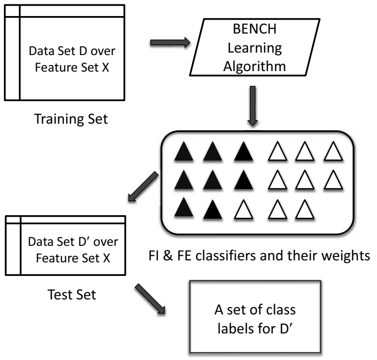

Data Set D over Feature Set X
Training Set
BENCH Learning Algorithm
FI & FE classifiers and their weights
Data Set D′ over Feature Set X
Test Set
A set of class labels for D′

Figure 2: BENCH method flowchart

In this paper, we aim to improve the prediction accuracy of a BBN classifier by combining feature selection and ensemble learning methods. Our strategy is to build an ensemble of BBN classifiers constructed from discriminating and non-discriminating features identified through biclustering.

Figure 1 and Figure 2 give a basic outline of our algorithm. Before applying the BENCH procedure, we need to process the data into a form suitable for biclustering and BBNs. Specifically, we replace the missing values for a feature with the average value of that feature, and we discretize the values for each feature into three equal-width bins representing up-regulation, down-regulation, and no change. Once we have the preprocessed data, we divide the data into biclusters. We then assess these biclusters for their enrichment relative to the class labels, and we use the resulting homogeneous and heterogeneous biclusters to form candidate datasets that we use to train the BBN classifiers. Lastly, we choose a voting scheme to determine how the results from the individual BBNs are combined into a final answer. The individual steps of this procedure are described in the following sections.

2.2 Biclustering Data

Biclustering is an unsupervised learning technique that seeks to identify a subset of features that exhibit similar behavior across a set of data points. We apply the results of biclustering to the problem of feature selection in a supervised fashion—by uncovering regions of the data with strong similarity, biclusters enable discovery of features that discriminate well or poorly across some subsets of the data points.

In our experiments, we use the implementation of the Samba biclustering algorithm available in EXPANDER [Ulitsky *et al.*, 2010]. Samba (Statistic-Algorithmic Method for Bicluster Analysis) combines a statistical data model and graph-theoretic approach to find the most significant biclusters in gene expression microarray data [Tanay *et al.*, 2002]. Samba has been shown to perform well in terms of its ability to recover biologically relevant biclusters on both real and synthetic microarray datasets as well as its robustness on synthetic microarray datasets [Preli *et al.*, 2006]. Samba was shown to perform better than four other common biclustering algorithms: Cheng and Church's algorithm [Cheng and Church, 2000], the Order Preserving Submatix (OPSM) algorithm [Ben-dor *et al.*, 2002], Iterative Signature Algorithm (ISA) [Ihmels *et al.*, 2004], and xMotifs [Murali and Kasif, 2003].

2.3 Calculating Bicluster Enrichment

Having identified a set of biclusters in the data, we assess how well each cluster can predict the class label. We classify each bicluster as homogeneously positive, homogeneously negative, or heterogeneous with respect to the class label. We make this classification based on the likelihood of achieving the same distribution of positive and negative class labels through random choice; i.e., if a bicluster has a statistically significant number of positive (negative) instances, we can classify it as homogeneous.

We calculate this likelihood by using the hypergeometric probability, with a significance threshold of 0.05. That is, if the probability of selecting at least as many positive data points as in the bicluster with a random sample of the same size is less than 0.05, we classify the bicluster as homogeneously positive. If this probability is greater than 0.95, then we label the bicluster as homogeneously negative (as the positive and negative probabilities are complementary), and we label biclusters with intermediate values as heterogeneous.

2.4 Combining Biclusters

Homogeneous Biclusters

For the homogeneous biclusters, we want to combine the homogeneously positive and negative biclusters to form training sets with features that discriminate between the class labels. To do so, we first compare all of the homogeneously positive biclusters with all of the homogeneously negative biclusters. If the features in a positive and a negative bicluster overlap significantly (i.e., the number of common features exceeds some user-defined threshold α), we form the candidate datasets by taking the intersection of the features and the union of the data points. We remove any duplicate datasets and use the remaining candidates as our FI datasets. If no FI datasets are formed due to a large value of α, then the user is expected to reduce the value of α; otherwise, BENCH will proceed without FI classifiers (see Section 3.6).

Heterogeneous Biclusters

In contrast to the homogeneous clusters, the features of the heterogeneous clusters do not discriminate well between the class labels. Therefore, in combining the heterogeneous biclusters, we want to form datasets that avoid these non-discriminating features. We first compare all the heterogeneous biclusters with all other heterogeneous biclusters to find a set of heterogeneous biclusters with at least β common features. For each heterogeneous bicluster, we compare it with every other heterogeneous bicluster, combining the biclusters in a greedy fashion. That is, if the bicluster have at least β common features, we combine them by taking the union of the data points and the union of the features. When we have finished combining one heterogeneous bicluster, we form an FE dataset by using the data points of the combined bicluster and the complement of the features. As before, we remove any duplicate datasets that result from this procedure. If no FE datasets are formed due to a large value of β, then the user is expected to reduce the value of β; otherwise, BENCH will proceed without FE classifiers (see Section 3.6).

2.5 Creating and Weighting the Ensemble

Once the candidate FI and FE datasets have been generated from the original set of biclusters, each dataset is used to train a base classifier that will become part of the ensemble. In this paper, we use BBNs as a base classifier, but our technique can be applied to other classifiers as well.

Once the ensemble is created, we wish to weigh each classifier in the ensemble. In order for the ensemble to make a prediction, each classifier is given a weighted vote, and the class with the most votes is the prediction of the ensemble. We tested three possible weighting schemes: a simple majority voting scheme in which every classifier is given equal weight, a training error–based method in which every classifier is weighted based on its training error, and a confidence-based method in which each classifier is weighted by that model's associated confidence value. The results of these tests appear in Section 3.4.

3 Results

In order to test the results obtained by BENCH, we compare the performance of BENCH with that of other classification methods. We used ten-fold cross validation for our experiments.

We built BBN classifiers using the K2 algorithm implementation provided by WEKA [M. Hall, 2009].

3.1 Data

In this paper, we perform experiments on three sets of two-class microarray data, Leukemia, Lymphoma, and Colon Cancer. (BENCH is not limited to microarray data, though; it can handle any two-class dataset with features suitable for biclustering.) The Leukemia dataset contains 72 measurements for the expression of 7129 genes, corresponding to samples taken from bone marrow and peripheral blood. Out of these samples, 47 samples are classified as acute lymphoblastic leukemia (ALL), and 25 samples are classified as acute myeloid leukemia (AML). The Lymphoma dataset contains 40 mRNA samples of normal and malignant lymphocytes. These samples, which analyze expression level for 4026 genes, are classified based on patient survival: 22 patients survived, and 18 did not. The Colon Cancer dataset contains 62 samples of colon tissue measuring the activity of 1996 genes. Of the 62 samples, 40 were cancerous, and 22 samples were normal.

3.2 Overall BENCH Performance

We tested BENCH's performance against two well-known ensemble learning techniques: bagging and random forests. We also included the results from a single BBN classifier to act as a baseline for our algorithm, since we use BBNs as the base classifiers for BENCH. For this experiment, we chose to use the confidence-based weighting scheme for BENCH, and we have included two results from BENCH in the figure in order to compare bagging with BENCH on a more even footing, one result using the full 59 classifiers produced by BENCH and the other using a random sample of five classifiers. In the following experiments, *% Accuracy* is defined as the ratio of the number of correctly classified data points to the total number of data points in the test set. A data point is correctly classified if the predicted class label matches the actual class label of that data point. Figure 3 shows the results of ten-fold cross validation on the Leukemia dataset.

Since the performance of bagging and random forests depends on the number and size of the individual datasets chosen, we tried several different values for our comparison to BENCH. Only the most accurate result from our trials appears in Figure 3. In order to provide a basis for comparison, the figure also includes information on the number of data points used in the classifier, the number of features used in the classifier, and the number of classifiers used, in the form "data points/features/classifiers." The numbers shown for the number of features and data points used in the BENCH run are averages taken over all classifiers and all folds.

As one can see from the figure, BENCH outperforms bagging by 15.71% and random forests by 14.29%. Interestingly, both bagging and random forests actually perform worse than the single BBN classifier on this dataset; however, BENCH still outperforms the single BBN classifier by about 14.28%.

Note that although using BENCH with BBN as a base classifier shows a significant performance improvement over the single BBN classifier, it does not exceed the best results for these datasets reported in the literature. For example, the best reported accuracy for the Leukemia data is 97.22% (using leave-one-out cross validation) [Peng, 2006], and the best accuracy for the Colon Cancer dataset is 88.87% [Kim and Cho, 2008] (again, using leave-one-out cross validation). However, as the performance of BENCH depends on the base classifier being used (see Section 4), a direct comparison of these results to BENCH using BBNs as a base classifier may not be apt. What we demonstrate here is that by using biclustering to identify sets of discriminating features over a subset of data points, we can build an ensemble that outperforms the base classifier as well as traditional ensemble methods such as bagging and boosting.

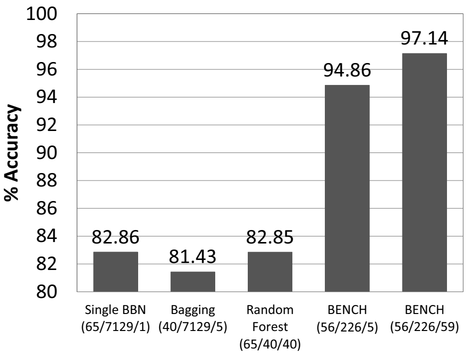


Figure 3: Comparison of prediction accuracy of BENCH to single and ensemble classifiers on the Leukemia dataset

3.3 A Note on Using Ten-Fold Cross Validation

The problem of error estimation for small sample data received a systematic treatment by [Braga-Neto and Dougherty, 2004], [Braga-Neto, 2007], and [Yousefi *et al.*, 2010]. Specifically, they concluded that although the bias is low, the variance is very high for small datasets when using cross validation. The use of bootstrapping and bolstered resubstitution was suggested in [Braga-Neto and Dougherty, 2004] and [Braga-Neto, 2007]; however, due to the high computational cost of bootstrapping experiments and the broad acceptance of cross validation by the data mining and AI communities, we report the prediction accuracy along with the bias and variance. In our results, the bias and variance were 0.0012 and 0.0096 for Leukemia, 0.0012 and 0.0043 for Lymphoma, and 0.0541 and 0.0174 for the Colon Cancer datasets, respectively. Note that validating BENCH with respect to bolstered resubstitution is a topic for future research.

3.4 Different Weighting Schemes

One factor that influences the results of BENCH is the weight assigned to each classifier for voting on the class. In this section, we test the three weighting schemes introduced earlier: majority voting, training error–based voting, and confidence-based voting. The experiment shows that the choice of weighting scheme had no bearing on prediction accuracy for the Leukemia or Colon Cancer datasets; however, the confidence-based method performed slightly better (7–10%) than the other two weighting schemes on the Lymphoma dataset. Combining the weighting schemes by multiplying training error and confidence did not improve the performance of the ensemble. Notably, though, all of the weighting schemes outperformed the accuracy of a single BBN classifier.

3.5 Parameters for Random Forests

We decided to test how the prediction accuracy of random forests changed as we added more features and classifiers. We varied the number of features from 5 to 40 and varied the number of base classifiers from 5 to 60 using ten-fold cross validation on the Leukemia dataset. The results show that the prediction accuracy ranges from 70% to 82.86% with no definitive trend when either or both the number of features or base classifiers increase. Therefore, we conclude that increasing the number of features or base classifiers has little effect on the performance of the random forest ensemble.

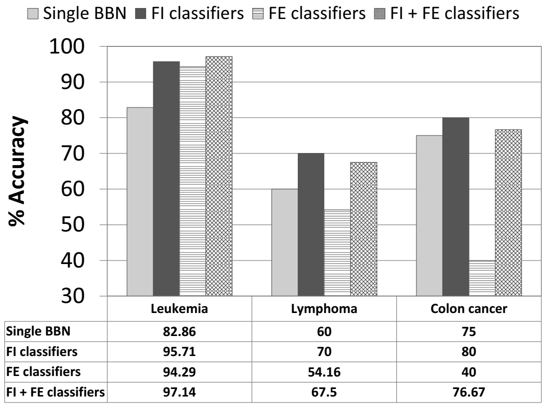
3.6 FI and FE Classifiers

BENCH creates two different types of classifiers during training, Feature Inclusion (FI) classifiers and Feature Exclusion (FE) classifiers. BENCH arrives at its final prediction by combining predictions from both types of classifiers. Note that BENCH will stop if both the FI and FE datasets are empty. In this case, the user would need to adjust the α and β parameters, as discussed in Section 2.4. We decided to test the predictive power of each type of classifier to see how much each contributed to the final prediction using the confidence-based weighting scheme. In Figure 4, we present the results of running a single BBN, only FI classifiers, only FE classifiers, and the combined prediction of FI and FE classifiers for all three datasets. As one can see, using only the FE classifiers typically leads to a decreased performance overall; however, using just the FI classifiers leads to a comparable performance as compared to the combined classifiers and even improves performance for the Colon Cancer dataset. Thus, we can also achieve accurate predictions using just the FI classifiers.

3.7 Learning Time

As discussed in Section 3.6, we can achieve accurate predictions using only the FI classifiers. We can use this observation to improve the training time of methods that train classifiers on the entire feature set, like bagging or a single BBN. If we train classifiers only on the FI datasets, it is possible to see a significant reduction in training time.

Our experiments on the Leukemia dataset shows that the training time for the FI classifier (0.5 seconds) is three orders of magnitude smaller than the training time for the single BBN classifier (1453.4 seconds) or the bagging method (1134.2 seconds). Even though random forests and BENCH



	Leukemia	Lymphoma	Colon cancer
Single BBN	82.86	60	75
FI classifiers	95.71	70	80
FE classifiers	94.29	54.16	40
FI + FE classifiers	97.14	67.5	76.67

Figure 4: Comparison of prediction accuracy between FI and FE classifiers using confidence weighting on Leukemia, Lymphoma, and Colon Cancer datasets

have similar training times (0.5 seconds), using the FI classifiers results in almost 10% higher accuracy. (Random forest prediction accuracy is 82.85%, from Figure 3, and the FI ensemble classification accuracy is given as 95.71%, from Figure 4.) Even though this result does not include the time required to produce the biclusters, biclustering is performed in less than 25 seconds. We feel that the similar training time for random forest to the training time of an ensemble of FI classifiers is offset by the decrease in prediction accuracy. Additionally, we observed (Section 3.5) that increasing the number of classifiers or features does not necessarily improve the prediction accuracy of the random forest technique.

The reason for the small training time of FI classifiers is that the number of features used for training an FI classifier is small relative to the original feature space. For example, in the Leukemia dataset, the average number of features in an FI dataset is 226, which is much smaller than the 7,129 features in the original dataset. It is interesting to observe that by carefully selecting groups of data points and features together, we can build an ensemble of classifiers capable of achieving higher accuracy than randomly selecting data points (in bagging) or features (random forest), individually.

4 BENCH Generalization

Thus far, we have shown that biclustering-driven ensemble methodology is effective for building an ensemble of BBNs classifiers; however, this method can be generalized to other classification methods. To verify this, we tested BENCH using SVMs and decision trees as a base classifier. For decision trees, we chose the J48 decision tree model that implements C4.5 algorithm in WEKA. For SVMs, we used the basic C-SVC (C-support Vector Classifiers) in WEKA-Libretos, since all class attributes in our datasets are two-class. The results of these tests appear in Table 1.

5 Conclusion

We have presented BENCH, a hybrid method for ensemble classification. We have shown that BENCH improves prediction accuracy by about 15% relative to individual classifiers and other ensemble methods, such as bagging and random forests. By sampling simultaneously along the feature space and the data points, we achieve a higher prediction accuracy than traditional ensemble methods that sample along only one dimension. For future work, we plan to study the theoretical aspects of BENCH.

Table 1: Comparison of the prediction accuracy of single and BENCH ensemble classifiers using BBNs, decision trees, and SVMs as base classifiers on the Leukemia dataset

Classifier	Single classifier	BENCH ensemble
BBN	82.86	97.14
Decision Tree	82.86	95.71
SVM	91.42	97.14

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. 1029711 and by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy. Oak Ridge National Laboratory is managed by UT-Battelle for the LLC U.S. DOE under contract no. DEAC05-00OR22725.

References

[Aljamaan and Elish, 2009] H.I. Aljamaan and M.O. Elish. An empirical study of bagging and boosting ensembles for identifying faulty classes in object-oriented software. In *Proc. of the CIDM IEEE Symposium on*, pages 187–194, 2009.

[Bauer and Kohavi, 1999] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Machine Learning*, 36:105–139, 1999.

[Ben-dor *et al.*, 2002] A. Ben-dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. In *Proc. of the 6th Annual ICCB*, pages 49–57, 2002.

[Bonneau *et al.*, 2006] R. Bonneau, D. Reiss, P. Shannon, M. Facciotti, L. Hood, N. Baliga, and V. Thorsson. The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biology*, 7(5):R36, 2006.

[Braga-Neto and Dougherty, 2004] Ulisses M. Braga-Neto and Edward R. Dougherty. Is cross-validation valid for small-sample microarray classification? *Bioinformatics*, 20(3):374–380, 2004.

[Braga-Neto, 2007] Ulisses Braga-Neto. Fads and fallacies in the name of small-sample microarray classification - a highlight of misunderstanding and erroneous usage in the applications of genomic signal processing. *Signal Processing Magazine, IEEE*, 24(1):91–99, 2007.

[Breiman, 1996] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[Cheng and Church, 2000] Y. Cheng and G.M. Church. Biclustering of expression data. In *Proc. of the Eighth ISMB*, pages 93–103, 2000.

[Cooper and Herskovits, 1992] G.F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[Davidson, 2004] I. Davidson. An ensemble technique for stable learners with performance bounds. In *Proc. of the 19th AAAI*, pages 330–335, 2004.

[Diaz-Uriarte and de Andres, 2006] R. Diaz-Uriarte and S. Alvarez de Andres. Gene selection and classification of microarray data using random forest. *BMC*, 7(1):3, 2006.

[Harrington *et al.*, 2000] C.A. Harrington, C. Rosenow, and J. Retief. Monitoring gene expression using DNA microarrays. *Current Opinion in Microbiology*, 3(3):285–291, 2000.

[Ho, 1998] T.K. Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, 1998.

[Ihmels *et al.*, 2004] J. Ihmels, S. Bergmann, and N. Barkai. Defining transcription modules using large-scale gene expression data. *BMC*, 20(13):1993–2003, 2004.

[Jing *et al.*, 2008] Y. Jing, V. Pavlovic, and J. Rehg. Boosted Bayesian network classifiers. *Machine Learning*, 73:155–184, 2008.

[Kim and Cho, 2006] K. Kim and S. Cho. Ensemble classifiers based on correlation analysis for DNA microarray classification. *Neurocomputing*, 70(1-3):187–199, 2006.

[Kim and Cho, 2008] Kyung-Joong Kim and Sung-Bae Cho. An evolutionary algorithm approach to optimal ensemble classifiers for dna microarray data analysis. *Evolutionary Computation, IEEE Transactions on*, 12(3):377–388, 2008.

[M. Hall, 2009] G. Holmes B. Pfahringer P. Reutemann I. H. Witten M. Hall, E. Frank. The WEKA data mining software: An update; sigkdd explorations, 2009.

[Melville and Mooney, 2003] P. Melville and R.J. Mooney. Constructing diverse classifier ensembles using artificial training examples. In *Proc. of the Eighteenth IJCAI*, pages 505–510, 2003.

[Murali and Kasif, 2003] T.M. Murali and S. Kasif. Extracting conserved gene expression motifs from gene expression data. *Pacific Symposium on Biocomputing*, pages 77–88, 2003.

[Murphy, 2002] K P. Murphy. Dynamic Bayesian Networks: Representation, Inference and Learning, 2002.

[Opitz and Maclin, 1999] D. Opitz and R. Maclin. Popular ensemble methods: an empirical study, 1999.

[Peng, 2006] Yonghong Peng. A novel ensemble machine learning for robust microarray data classification. *Computers in Biology and Medicine*, 36(6):553–573, June 2006.

[Preli *et al.*, 2006] A. Preli, S. Bleuler, P. Zimmermann, A. Wille, P. Bhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *BMC*, 22(9):1122–1129, 2006.

[Reiss *et al.*, 2006] D. Reiss, N. Baliga, and R. Bonneau. Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. *BMC*, 7(1):280, 2006.

[Schapire, 1990] R.E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[Sohn and Dagli, 2003] S. Sohn and C.H. Dagli. Combining evolving neural network classifiers using bagging. In *Proc. IJCNN*, volume 4, pages 3218–3222, 2003.

[Tanay *et al.*, 2002] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *BMC*, 18(suppl_1):136–144, 2002.

[Tanay *et al.*, 2005] A. Tanay, R. Sharan, and R. Shamir. Biclustering algorithms: a survey. In *Handbook of Computational Molecular Biology Edited by: Aluru S. Chapman & Hall/CRC Computer and Information Science Series*, 2005.

[Tao *et al.*, 2006] D. Tao, X. Tang, X. Li, and X. Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE TPAMI*, 28(7):1088–1099, 2006. PMID: 16792098.

[Tsymbal *et al.*, 2003] A. Tsymbal, S. Puuronen, and D.W. Patterson. Ensemble feature selection with the simple Bayesian classification. *Information Fusion*, 4(2):87–100, 2003.

[Tu *et al.*, 2009] M. Tu, D. Shin, and D. Shin. A comparative study of medical data classification methods based on decision tree and bagging algorithms. In *Proc. Eighth IEEE International Conference on DASC*, pages 183–187, 2009.

[Ulitsky *et al.*, 2010] I. Ulitsky, A. Maron-Katz, S. Shavit, D. Sagir, C. Linhart, R. Elkon, A. Tanay, R. Sharan, Y. Shiloh, and R. Shamir. Expander: from expression microarrays to networks and functions. *Nat. Protocols*, 5(2):303–322, 2010.

[Wang *et al.*, 2003] H. Wang, W. Fan, P.S. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *Proc. of the ninth ACM SIGKDD*, pages 226–235, 2003.

[Waske *et al.*, 2010] B. Waske, S. van der Linden, J.A. Benediktsson, A. Rabe, and P. Hostert. Sensitivity of support vector machines to random feature selection in classification of hyperspectral data. *IEEE Transactions on GRSS*, 48(7):2880–2889, 2010.

[Yousefi *et al.*, 2010] Mohammadmahdi R. Yousefi, Jianping Hua, Chao Sima, and Edward R. Dougherty. Reporting bias when using real data sets to analyze classification performance. *Bioinformatics*, 26(1):68–76, 2010.

[Zhang *et al.*, 2008] D. Zhang, S. Chen, Z. Zhou, and Q. Yang. Constraint projections for ensemble learning. In *Proceedings of the 23rd AAAI*, pages 758–763, 2008.