Adaptive Informative Sampling for Active Learning

Zhenyu Lu, Xindong Wu, Josh Bongard

University of Vermont, Department of Computer Science, Burlington, VT 05401

zlu@uvm.edu; xwu@cems.uvm.edu; jbongard@uvm.edu

October 8, 2009

Abstract

Many approaches to active learning involve periodically training one classifier and choosing data points with the lowest confidence, but designing a confidence measure is nontrivial. An alternative approach is to periodically choose data instances that maximize disagreement among the label predictions across an ensemble of classifiers. Many classifiers with different underlying structures could fit this framework, but some ensembles are more suitable for some data sets than others. The question then arises as to how to find the most suitable ensemble for a given data set. In this work we introduce a method that begins with a heterogeneous ensemble composed of multiple instances of different classifier types, which we call adaptive informative sampling. The algorithm periodically adds data points to the training set, adapts the ratio of classifier types in the heterogeneous ensemble in favor of the better classifier type, and optimizes the classifiers in the ensemble using stochastic methods. Experimental results show that the proposed method performs consistently better than homogeneous ensembles. Comparison with random sampling and uncertainty sampling shows that the algorithm effectively draws informative data points for training.

1 Introduction

Classification is a technique that uses both features and class label information to build predictive models from data sets. In many real world applications, data sets are presented with only feature information and acquiring the labels is expensive. For example, an expensive or time-consuming test may have to be conducted on a physical object to determine which class it belongs to. Active learning is a technique that selects a subset of data points for labeling and training. The subset of data points needs to be chosen carefully so that it is informative enough for learning, but small enough to keep the labeling cost manageable.

A common approach to active learning is to iteratively train one classifier and select the data points using some confidence measure. Uncertainty sampling [12] iteratively trains one classifier, and chooses the data points that the classifier is most uncertain about. This approach is intuitive, but designing confidence measures without bias is nontrivial. For example, several criteria for choosing data points [18] [6] have been suggested for support vector machines (SVM) [8], with each performing well on different applications.

Another approach is to use an ensemble of classifiers and choose data points according to the uncertainty of the ensemble. Query by committee [19] suggests that given a set of diverse but partially accurate classifiers, the best data points are those that cause maximal disagreement among the class predictions of the classifiers when supplied without labels. However, the method for finding such classifiers or such disagreement-inducing data points is not specified. The estimation-exploration algorithm [2](EEA) uses multiple stochastic optimization algorithms to build on query by committee. A stochastic optimization algorithm is used to optimize a set of diverse models, and another stochastic optimization algorithm is used to induce desired data points that maximize disagreement among the models. As a result of the interaction between these two optimization algorithms, the EEA actively requests useful data points and thereby accelerates modeling.

Informative sampling [13] applied the EEA as an active learning method for classification. The algorithm works by iteratively optimizing an ensemble of classifiers based on the current training set and scanning a portion of the whole data set to select the data point that causes maximal disagreement among the label predictions of the current classifiers. It has been shown to outperform random sampling [21] and balanced sampling [11] for a large unbalanced data set called the National Trauma Data Bank (NTDB) [1]. Although artificial neural networks (ANN) [10] were chosen as the classifier type in [13], informative sampling is a general algorithm that could work with any classifier. Many types of classi-

---

[1] http://www.facs.org/trauma/ntdb.html

fiers exist in the literature, such as decision trees (DT) [16], ANNs [10], and SVMs. Different types of classifiers have different underlying structures, making them differentially suitable for different data sets. For example, the decision boundary formed by DTs consists of lines or planes that are orthogonal to one of the features, which makes DTs more suitable to a data set with decision boundaries that are orthogonal to the features than another data set that has highly nonlinear and continuous boundaries. The question then arises as to how to find the suitable classifier type for a given data set.

Many approaches have been suggested in the field of ensemble learning [9][20][7] to construct ensembles of heterogeneous classifiers, in which existing methods such as C4.5 and SVM were used to train base classifiers. These ensembles typically contain one instance of each type, or obtain multiple instances from each type by exploiting different parameters with existing training methods. One problem is that some methods such as C4.5 are deterministic, which means that only one classifier can be built on a given data set. In our case, different individuals from the same classifier type are expected to be developed such that disagreement among them indicates model uncertainty. Stochastic optimization techniques are used here to develop multiple instances from each classifier type. Therefore the ensemble consists of diverse heterogeneous classifiers. In this paper, an algorithm that builds on top of informative sampling is introduced: adaptive informative sampling (henceforth referred to as AIS). The major difference between AIS and informative sampling is that informative sampling only works with ensembles with static combinations of classifier types, whereas AIS adapts the ratio of classifier types towards better accuracy. The algorithm starts with multiple classifiers drawn from different classifier types and proceeds through two stages. In the first stage, classifiers are trained and replaced based not only on their classification error, but also on their type, and data points are chosen and added to the training set. The replacement aims to adapt the ratio of classifier types in the ensemble so that the more suitable model type will eventually saturate the ensemble. In the second stage, the classifiers in the ensemble are optimized for a certain number of iterations.

The major contributions of this paper are: 1) empirically proving that different classifier types are suitable for different data sets with active learning; 2) introducing an algorithm framework that adapts the classifier ratio for heterogeneous ensembles; 3) introducing an adaptation strategy for AIS that outperforms random sampling and uncertainty sampling.

In this work, ANNs and DTs were used as example classifier types. At this stage, AIS only deals with
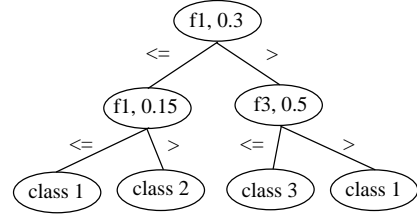

Figure 1: General form of the decision trees in this paper.

numeric data sets. Two sets of experiments were conducted on two synthetic data sets and three data sets from the UCI Machine Learning Repository [1]. The first set of experiments shows the effectiveness of AIS to find the better classifier type on a given data set. For all tested data sets, the suggested algorithm performs better than or comparable to using a homogeneous ensemble of the better classifier type, because the better classifier type always saturates the ensemble of classifiers automatically. The second set of experiments compares AIS to uncertainty sampling and random sampling, which shows that AIS draws informative data points into the training set.

The following section introduces the adaptive informative sampling algorithm. Section 3 gives the experimental results. Section 4 concludes the paper.

2 Methods

In this section, the structures and optimizing methods for the ANNs and DTs are described, then the informative sampling algorithm is reviewed, and finally the adaptive informative sampling algorithm is introduced.

2.1 Decision Trees The DTs optimized in this paper take the form of binary trees, and has similar structure as the DTs described in GaTree [15]. An illustration is given in Figure 1. Each internal node is associated with a certain feature and a real number called the split point. Each leaf node is associated with a class label. Given a trained decision tree and a data point for prediction, the classification process works as follows: 1) starting from the root node, check the value of the data point on the feature that corresponds to the current node; 2) if the value is greater than the split point on the current node, follow the right link to the next node; otherwise, follow the left link; 3) repeat 1 and 2 until a leaf node is reached, and output the class label on the leaf node. The class label on each leaf node is set to be the majority label of the training subset that reaches that node. To cover the case where splitting a feature into more than two sub-ranges is necessary for accurate classifications, a feature is allowed to be associated with
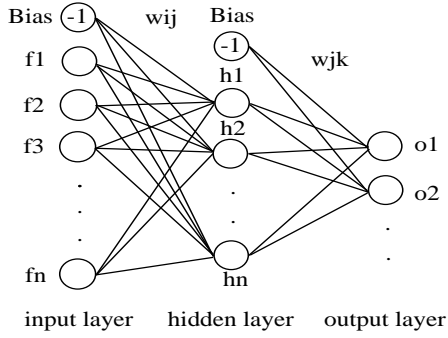
Figure 2: General form of the ANNs in this paper.

multiple nodes. In this way, the binary decision tree has the same expressive power as any N-ary decision tree.

The optimization of a decision tree involves three mutation operators: add, delete and change. When an add mutation operator is applied, an internal node is randomly selected, a new node with a randomly generated feature and split point is inserted into the tree at the position of the selected node, the subtree with the selected node as the root is pushed down as the left or right child of the new node, and a leaf node is added as the other child. When a delete operator is applied, an internal node is randomly selected, all its children are deleted, and the node is changed into a leaf node. When a change operator is applied, an internal node is randomly selected, the feature associated with it is changed into a random new one, and the split point is changed to a random value. At the beginning of the optimization for a decision tree, a random tree is generated, with each internal node containing a randomly chosen feature and split point. The number of internal nodes for the random tree is set to be half of the number of features for a given data set. This choice is relatively arbitrary, as optimization may add or remove nodes to obtain a better structure. However this initial choice was made such that the initial DTs have similar structural complexity compared to the ANNs. At each optimization step, a copy of each DT is made, and one of the three mutation operators is chosen randomly and applied to each tree. A parallel hill climber [17] is used to optimize the DTs: for each tree in the ensemble, if the child tree is more fit than its parent in terms of accuracy, the parent tree is replaced with it. Otherwise, the parent is retained.

2.2 Artificial Neural Networks The ANNs used in this paper are standard feed-forward neural networks with three layers (input, hidden, output) [10]. The activation function is the sigmoid function. An illustration is given in Figure 2. Nodes in two adjacent layers are

associated with links. The weight for each link has a real value between -1 and 1. Each node in the input layer corresponds to a feature that has been normalized into a real value between 0 and 1, except for a bias node with -1 as the value. The number of hidden nodes is set to be half of the number of features, plus a bias node with -1 as the value. For a two-class classification task, there is one node in the output layer, which outputs a real value between 0 and 1. For an output value that is greater than 0.5, the ANN outputs 1 as the output; otherwise, the output is 0. For a multi-class classification task, the number of nodes in the output layer equals the number of classes, and the index of the output node with the largest value is used as the output.

Mutation of the ANNs involves randomly selecting a weight and changing it to a random real value between -1 and 1. The stochastic optimization algorithm is the same as the DTs. At the beginning, random ANNs are generated such that each weight of the ANN is a random real value between -1 and 1. At each optimization step, a copy of each ANN is made, and the mutation operator is applied. If the child ANN is better, the parent is replaced with it. Otherwise, the parent is retained and the child is discarded.

Since the two classifier types work with each other when both exist in an optimizing ensemble, a crucial question arises as to how to exert a similar training speed on both types, to ensure that one classifier type does not die out simply because it learned more slowly than the other type. In our framework, the number of the hidden nodes in an ANN is set to be half of the number of the inputs. For a data set that has 16 features and 2 classes, an ANN has a total of 17 * 9 + 9 * 1 = 162 weights. One mutation of an ANN therefore has a structural impact of 1/162 on the classifier. For the same data set, a randomly generated decision tree has 8 internal nodes. It is hard to determine the size of the DTs without knowledge of the actual data set. We assume the same probability for each mutation operator to take effect. Under this assumption, the mean number of internal nodes of a set of randomly generated DTs is close to 8. Each node has two components. So one mutation of a decision tree has a structural impact of 1/16 on the classifier. In order to make the impact of mutation on the two classifier types more similar, in this paper, when the ANN is evolved, a certain number of consecutive iterations of optimizations are applied. The number is set to be half of the number of input nodes. While it is quite challenging to compare the two classifier types in a perfectly fair manner because of the different structures they have, the increased mutation rate narrows the difference. For the data set mentioned above, each new ANN experiences a 8/162 =

Table 1: The informative sampling algorithm

---

1 Initialization

    Create a random ensemble of classifiers.

2 Stage 1

  1) Exploration Phase

    a) Pass a portion of the data set to the classifiers to find a single training data point.

    b) Fitness of a candidate data point is the amount of disagreement it causes among classifiers.

    c) Data point that causes the most disagreement is added to the training set.

  2) Estimation Phase

    a) Use a stochastic optimization algorithm to train each candidate classifier; a parent classifier is replaced if its child is more fit.

    b) Fitness of a classifier is its predictive accuracy on the current training set.

  3) Repeat steps 2 and 3 for a certain number of iterations to obtain a subset as the training set.

3 Stage 2

  Optimize the ensemble for a certain number of iterations using the obtained training set until the classifiers achieve some termination criteria.

0.05 impact assuming independency between mutations, compared to a structural impact of $1/16 = 0.06$ that one mutation has on a decision tree. This gives the two classifier types a similar structural impact for each iteration of optimization.

2.3 Informative Sampling The algorithm starts by creating an ensemble of randomly created classifiers, then it proceeds in two stages. The data set is divided into subsets of size p using the original ordering of the data set.

During each iteration of the first stage, two phases are conducted: the exploration phase and the estimation phase. The exploration phase runs before the estimation phase. At the outset of the algorithm, the first data point to the pth data point are supplied to the current set of initially random classifiers in a sequential manner. For each data point, class predictions are made by the classifiers, and the variance across the class predictions are calculated. A high variance indicates that the numbers of predictions for each class are similar, and that there is therefore disagreement among the classifiers as to the true class of that data point. The data point with the highest such variance is chosen and added to the training set, along with its corresponding label.

The exploration phase is followed by the estimation phase. In this phase, the classifiers are trained by stochastic optimization as described above. Each classifier is evaluated and potentially replaced once using a hill climbing algorithm [17]. The fitness of a classifier is defined as the accuracy on the current training set as described in Section 3.1.

The algorithm then returns to the exploration phase. This time, the $(p+1)$th data point to the $2p$th data point are supplied to the models that were evolved in the estimation phase. One data point again is selected based on model disagreement and added to the training set (which now contains two data points). The estimation phase is run again after the exploration phase on this updated training set. One iteration of the algorithm consists of a single run of the exploration phase followed by a single run of the estimation phase. The algorithm executes several iterations to obtain a subset of the whole data set as the training set.

In the second stage, the algorithm uses a parallel hill climbing algorithm to optimize the set of classifiers in the ensemble until certain termination criteria is met.

Several changes are made to the originally-proposed EEA in order to apply it as a selective sampling method for unbalanced data sets. In previous applications [5][3][4], virtual data points are created as candidate tests. This is legitimate for systems for which training data can be generated on the fly and then labeled by the system under study, but not for pre-existing data sets. The solution to this is to use a portion of the data set without their labels as candidate tests. Table 1 is an outline of the informative sampling algorithm.

2.4 Adaptive Informative Sampling The adaptive informative sampling algorithm starts by creating an ensemble of classifiers with equal numbers of ANNs and DTs. In this paper, the ensemble size is 50, so there are 25 ANNs and 25 DTs in the ensemble at the outset. The algorithm then randomly selects 50 data points and adds them to the training set. The reason for this is that DTs need a certain number of data points to initialize the class labels on each leaf (see section 2.1). The algorithm has two stages as the informative sampling algorithm. The first stage of AIS also consists of the exploration phase and the estimation phase. The exploration phase is exactly the same as in the informa-

Table 2: The adaptive informative sampling algorithm

---

1 Initialization

    a) Create a random ensemble of classifiers with
       equal numbers of ANNs and decision trees.

    b) Randomly select some data points, query their
       labels, and add them to training set.

2 Stage 1

  1) Exploration Phase

    a) Pass a portion of the data set to the classifiers
       to find a single training data point.

    b) Fitness of a candidate data point is the amount
       of disagreement it causes among classifiers.

    c) Data point that causes the most disagreement is
       added to the training set with the queried label.

  2) Adaptation Phase

      Update the ensemble of classifiers according to
      a replacement strategy to adapt the ratio of two
      classifier types in the ensemble.

  3) Estimation Phase

    a) Use a stochastic optimization algorithm to train
       each candidate classifier; a parent classifier is
       replaced if its child is more fit.

    b) Fitness of a classifier is its predictive accuracy
       on current training set.

  4) Repeat steps 2 and 3 for a certain number of
    iterations to obtain a subset as the training set.

3 Stage 2

  Optimize the ensemble for a certain number of
  iterationusing the obtained training set until the
  classifiers achieve some termination criteria.

---

tive sampling algorithm.

In the adaptation phase, before each classifier is updated using the hill climbing algorithm, a replacement strategy is applied to update the relative number of the two classifier types.

Two replacement strategies are explored in this paper. The first replacement strategy does not consider classifier type. Each replacement simply replaces the worst classifier with a copy of the best one in the ensemble, without considering whether they are of the same type or not.

The second replacement strategy locates the worst classifier in the ensemble and records its type (ANN or DT). Then this type is punished by replacing it with a copy of the best classifier of the other type. For example, if in the current ensemble a decision tree has the highest classification error, then a copy of the best ANN in the ensemble is used to replace it. When all classifiers in the ensemble have the same type, the worst classifier is replaced by a new classifier of the other type. This new classifier is generated by creating a random classifier of the required type and performing the training as was done for the rest of the classifiers in the ensemble. For example, at the outset of the second pass through the estimation phase, a new classifier is generated by creating a random classifier and optimizing it once on the 51 data points in the current training set (50 random data points + 1 data point chosen in the exploration phase). This is to cover the case that one classifier type might have the potential to provide a better classifier than the other type, but evolves slower during early optimization and thus all instances of this type might be prematurely removed from the ensemble. This strategy explicitly considers both fitness and classifier type when performing replacement.

The two replacement strategies were developed to study if classifier type should be explicitly considered when conducting replacement.

Then AIS proceeds the same ways as informative sampling. Table 2 is an outline of the adaptive informative sampling algorithm.

3   Results

Two sets of experiments were conducted. The first aims to study the effectiveness of AIS to adapt the ensemble in favor of the better classifier type. The second compares AIS to uncertainty sampling and random sampling. In this section, the details of the data sets and experimental settings are first introduced, then the results are given.

3.1   Data sets and experimental settings Experiments were conducted on five data sets, among which two are synthetic, and the other three are drawn from the UCI Machine Learning Repository [1].

The first synthetic data set, henceforth referred to as S1, is an unbalanced data set with a highly nonlinear decision boundary. It contains 20000 data points, of which 2000 were randomly chosen as the testing set, and the remaining 18000 were used as the pool for selecting training data points. 50 data points were randomly drawn into the training set at first, then 1 out of p = 30 candidate training data points were chosen out

of the pool sequentially and added to the training set during the first stage of the algorithm. The amount of training data points supplied to the classifiers was chosen to ensure reasonable performance for each data set. For S1 the first stage of each run executed 550 such iterations, such that there were 550 data points chosen by the algorithm and 50 randomly chosen data points in the training set. Each data point has 8 real-valued features between 0 and 1 and one binary class label. The class label of each data point is determined using the following function:

$$(3.1) \qquad o = \sum_j f_j^2$$

where f_j is the value of the jth feature. If o is less than 2, the data point is assigned a label of 1, otherwise a label of 0 is assigned. This highly nonlinear function severs as the decision boundary for this data set.

The second synthetic data set is referred to as S2. It is the same in every aspect as S1 described above except for the decision boundary. The class label of a data point in S2 is decided by the value of its 4th feature. If it is less than 0.3, the data point is assigned a label of 0; otherwise a label of 1 is assigned. Thus, its decision boundary is a plane orthogonal to the 4th feature.

The first data set from UCI to was the Spambase Data Set[2], henceforth referred to as Spam. The data set was first shuffled, then of the 4601 data points, 601 were randomly selected as the testing set and the remaining 4000 data points served as the pool for potential training data points. 50 data points were randomly drawn into the training set at first, then 1 out of $p = 10$ data points were chosen by the algorithm and added to the training set in each iteration. There were 350 such iterations, such that the final training set contains 400 data points. Each data point has 57 features that have been normalized to real values between 0 and 1. The class label has two possible values: 1 represents "Spam" and 0 represents "Non-Spam". Class "Spam" has 1814 instances, the other 2787 instances have "Non-Spam" labels.

The second data set from UCI to was the Pen-Based Recognition of Handwritten Digits Data Set[3](henceforth referred to as Pendigit). Of the 10992 data points in the data set, 3498 were given as the testing set and the remaining 7494 in the given training set served as the pool for selecting training data points. 50 data points were randomly drawn into the training set at first, then 1 out of $p = 10$ data points were chosen

and added to the training set in each iteration of the first stage. There were 650 such iterations. The final training set therefore contains 700 data points. Each data point has 16 features that have been normalized to real values between 0 and 1. There are 10 possible values for the class label, with each corresponding to one of the 10 digits.

The third data set from UCI was the Landsat Satellite Data Set[4] (henceforth referred to as Landsat). The data set has a total of 6435 data points, of which 2000 were given as the testing set, and the remaining 4435 served as the pool for choosing data points. There were fifty randomly chosen data points in the training set at first, and then 1 out of $p = 5$ data points were chosen and added to the training set in each iteration of the first stage. There were 750 such iterations. The final training set contains 800 data points. Each data point has 36 features, each of which has been normalized to real values between 0 and 1. There are 6 possible values for the class label.

A fitness function for computing the performance of a classifier was suggested in [13] as:

$$(3.2) \qquad f = \frac{c_0}{t_0} \times \frac{c_1}{t_1}$$

where c_0 is the number of correct predictions of class 0 on the training set, t_0 is the total number of class 0 data points in the training set, c_1 is the number of correct predictions of class 1, and t_1 is the total number of class 1 data points. This fitness function was suggested to handle both balanced and unbalanced data sets for 2-class classification tasks. It makes sense for 2-class classification tasks because it forces the classifiers to learn equally on both classes, but for multi-class classification tasks, it might not be suitable because it unfairly penalizes classifiers that are accurate for most classes, but inaccurate for a few sparsely populated classes. In this paper, equation (3) is used to calculate the fitness of classifiers for the S1, S2 and Spam data sets. For the Pendigit and Landsat data sets, the fitness function is defined as the number of misclassifications on the training set.

For each experiment, 200 iterations of optimization were conducted in the second stage after the process of choosing data points has finished, in order to provide the ensemble more chances to learn from the whole chosen data points, especially for the data points that have been added to the training set during later iterations in the first stage. 30 independent runs were conducted on each of the five data sets. At the end of each run, the most accurate classifier was evaluated on the

[2]http://archive.ics.uci.edu/ml/datasets/Spambase.
[3]http://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits

[4]http://archive.ics.uci.edu/ml/datasets/Statlog + (Landsat + Satellite)
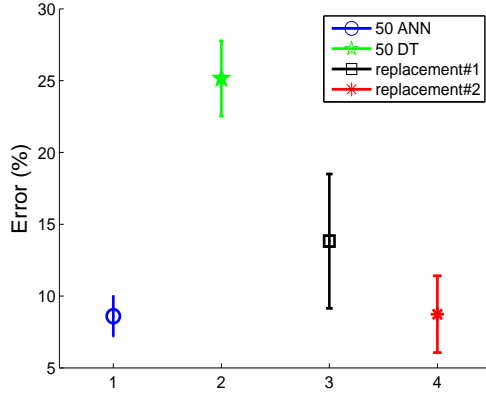
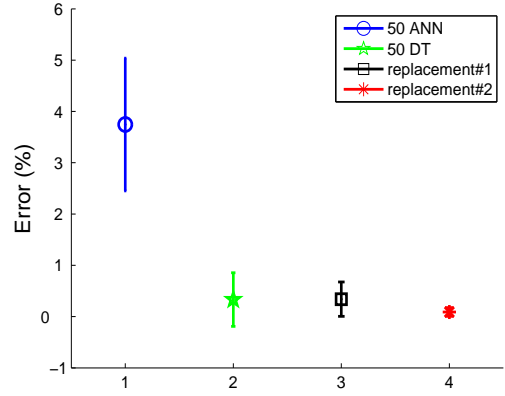Figure 3: Comparison on S1 with homogeneous ensembles

Figure 4: Comparison on S2 with homogeneous ensembles

testing set by counting the number of misclassifications on the testing set. In other words, the ensemble makes predictions by selecting the best classifier and using it as the prediction model. The mean and standard deviation of the misclassifications across 30 runs were reported as performance measure.

3.2 Comparison with Homogeneous Ensembles

This section reports the comparison between the heterogeneous and homogeneous ensembles. Three sets of comparisons were made across each data set: a comparison between informative sampling with 50 ANNs only and the same algorithm with 50 DTs only; a comparison between adaptive informative sampling with the two replacement strategies; and a comparison between AIS and informative sampling. The experiments are to show that with the right replacement strategy, AIS performs consistently well through all five data sets. A homogeneous ensemble with all ANNs or DTs might do comparably well on some data sets, but significantly worse on others.

Figure 3 reports the algorithm's performances on S1. S1 was constructed to have a highly continuous and non-linear decision boundary, with the intuition that it should be difficult for DTs to perform well. The result confirms this intuition. For the first comparison (compare line 1 to 2), informative sampling with 50 ANNs has a mean of 171.97 misclassifications, which corresponds to a predictive accuracy of 91.4%. The same algorithm with 50 DTs has a mean of 503 misclassifications, which is an accuracy of 74.85%. When working with informative sampling on S1, it is therefore shown that ANNs produce significantly more accurate and generalized classifiers than DTs. For the second comparison (compare line 3 to 4), AIS with the first replacement strategy has a mean of 276.57 misclassifications,
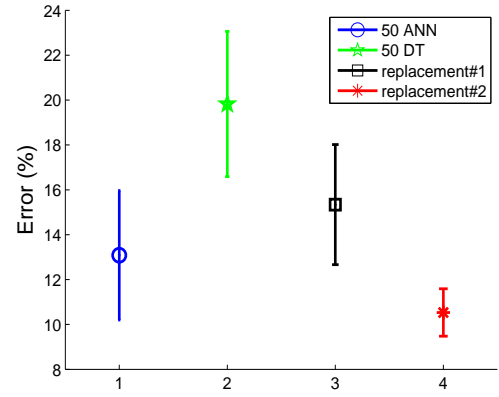

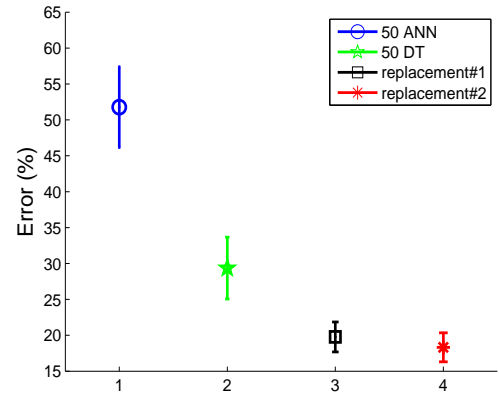Figure 5: Comparison on 'Spam' with homogeneous ensembles

Figure 6: Comparison on 'Pendigit' with homogeneous ensembles

which corresponds to an accuracy of 86.17%. The same algorithm with the second replacement strategy has a mean of 174.77 misclassifications, which is an accuracy
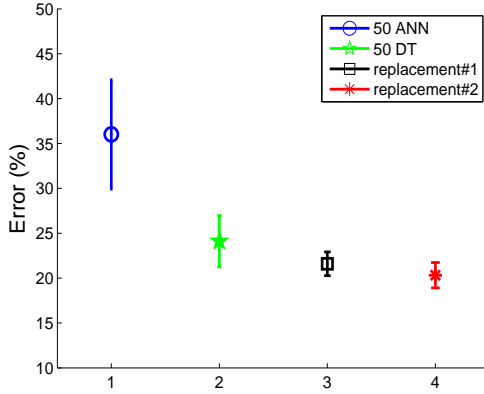
Figure 7: Comparison on 'Landsat' with homogeneous ensembles

of 91.3%. The second replacement strategy shows a smaller standard deviation. It is therefore shown that adaptive sampling with the second replacement strategy works better than the same algorithm with the first replacement strategy on S1. For the third comparison, adaptive sampling with the second replacement strategy is marginally worse than informative sampling with all ANNs (compare line 4 to 1) but comparable, and the first replacement strategy is worse than the same algorithm (compare line 3 to 1).

Figure 4 reports the results on S2. This data set was constructed with a simple decision boundary that is perpendicular to one of the features, which DTs should handle well. For the first comparison (compare line 1 to 2), informative sampling with 50 ANNs has a mean of 74.9 misclassifications, which corresponds to an accuracy of 96.26%. The same algorithm with 50 DTs has a mean of 6.63 misclassifications, which is an accuracy of 99.67%. Using all ANNs is not only noticeably worse, but also much less consistent, as evidenced by the larger error bar. It is therefore shown that using all DTs achieves a better performance in this context, which is again consistent with our intuition. For the second comparison (compare line 3 to 4), AIS with the first replacement strategy has a mean of 6.8 misclassifications, which corresponds to an accuracy of 99.66%. The same algorithm with the second replacement strategy has a mean of 1.8 misclassifications, which is an accuracy of 99.91%. The second strategy achieves a noticeably smaller standard deviation. It is shown that the two algorithms perform almost perfectly, and comparably, on S2, which is not surprising for such a simple data set. For the third comparison (compare line 2 to 3 and 4), AIS with either strategy is slightly better than informative sampling with all DTs.

Figure 5 gives the performance comparison on the Spambase data set. For the first comparison (compare line 1 to 2), informative sampling with 50 ANNs has a mean of 78.63 misclassifications, which corresponds to an accuracy of 86.92%. The same algorithm with 50 DTs has a mean of 119.13 misclassifications, which is an accuracy of 80.18%. It is shown that for this data set, using all ANNs achieves a better performance than using all DTs. For the second comparison (compare line 3 to 4), AIS with the first replacement strategy has a mean of 92.19 misclassifications, which corresponds to an accuracy of 84.66%. The same algorithm with the second replacement strategy has a mean of 63.3 misclassifications, which is an accuracy of 89.5%. The second strategy has a noticeably smaller standard deviation. It is therefore shown that the second strategy is better than the first one. For the third comparison, AIS with the second strategy is slightly better than informative sampling using all ANNs (compare line 4 to 1).

Figure 6 reports the results on Pendigit. For the first comparison (compare line 1 to 2), informative sampling with 50 ANNs has a mean of 1810.9 misclassifications, which corresponds to an accuracy of 48.23%. By comparison, notice that this data set has a total of 10 classes, so that a random guess has a predictive accuracy of 10%. The same algorithm with 50 DTs has a mean of 1026.8 misclassifications, which is an accuracy of 70.65%. On this data set, using all DTs achieves a better performance than using all ANNs. For the second comparison (compare line 3 to 4), AIS with the first replacement strategy has a mean of 684.27 misclassifications, which corresponds to an accuracy of 80.44%. The same algorithm with the second replacement strategy has a mean of 636.97 misclassifications, which is an accuracy of 81.79%. In this case, the second strategy is slightly better than the first strategy. For the third comparison, AIS with either replacement strategy is better than informative sampling using all decision trees (compare lines 3 and 4 to 2).

The results of the on the Landsat data set are reported in Figure 7. For the first comparison (compare line 1 to 2), using 50 ANNs with informative sampling has a mean of 720.2 misclassifications, which is an accuracy of 64%. 50 DTs with informative sampling has a mean of 482.33 misclassifications, which is a predictive accuracy of 75.88%. It is therefore shown that decision trees are more suitable for this data set than ANNs when working with informative sampling. For the second comparison (compare line 3 to 4), AIS with the first strategy has a mean of 431.86 misclassifications, which corresponds to an accuracy of 78.41%. And AIS with the second strategy has a mean of 406.56 misclassifications, which is an accuracy
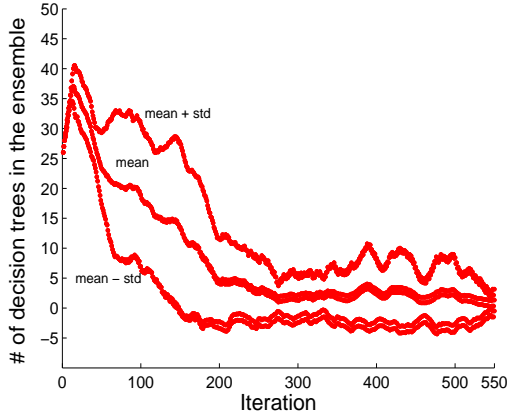
Figure 8: S1- Change in the number of decision trees in the ensemble over iterations for the second replacement strategy
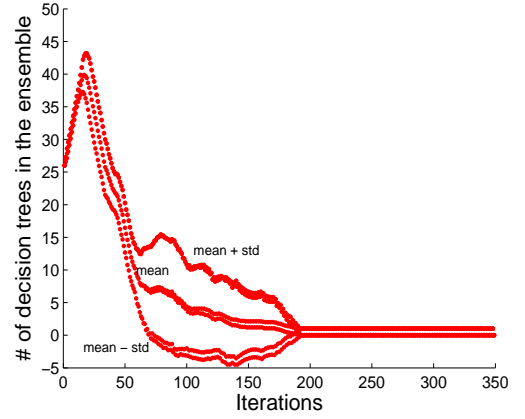

Figure 9: S2- Change in the number of decision trees in the ensemble over iterations for the second replacement strategy
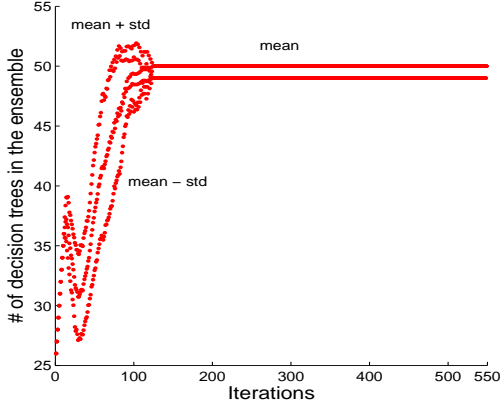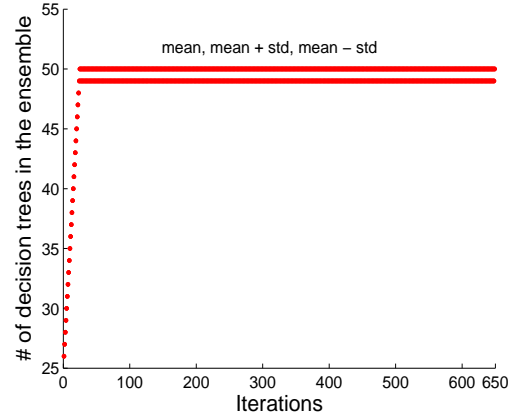
of 79.67%. On this data set, the second strategy is marginally better than the first strategy. For the third comparison (compare lines 3 and 4 to 2), AIS with either replacement strategy performs slightly better than using all decision trees.

Of the five data sets, using ANNs with informative sampling works better than using all decision trees on one synthetic data set and one data set from UCI, and using all decision trees performs better on the other three. In all cases, AIS with the second strategy performs either better than or comparable to informative sampling with the better classifier type. It has been shown that AIS with the second strategy, where both accuracy and classifier type information are considered, performs consistently well on all five data sets. The following results show the reason why the algorithm is always comparably to informative sampling using the better classifier type on a given data set.

Figure 10: Spam- Change in the number of decision trees in the ensemble over iterations for the second replacement strategy

Figure 11: Pendigit- Change in the number of decision trees in the ensemble over iterations for the second replacement strategy

On S1 and Spambase data sets, informative sampling using all ANNs is better than using all DTs. Figure 8 and Figure 10 show that ANNs saturate the ensemble with AIS and the second strategy. On S2, Pendigit and Landsat, DTs are better suited with informative sampling, Figure 9, Figure 11 and Figure 12 show that decision trees saturate the ensemble with AIS and the second replacement strategy. It is noticeable from the five figures that the more suitable classifier type does not always straightly saturate the ensemble. On Figure 8 and Figure 10, the number of decision trees in the ensemble goes up in the first several iterations, then ANNs start to perform better. On Figure 9, the ensemble first develops in favor of decision trees, then ANNs performs better for a few iterations, and then decision trees are better through the rest of the iterations.
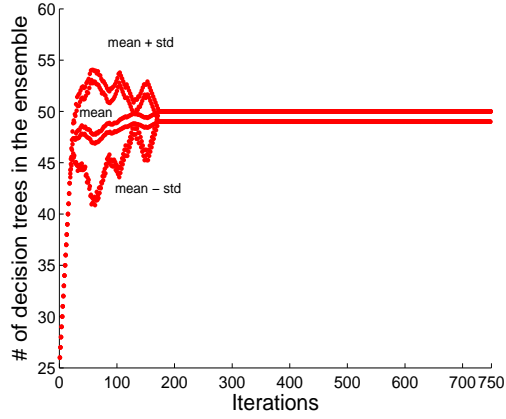
Figure 12: Landsat- Change in the number of decision trees in the ensemble over iterations for the second replacement strategy

Figure 13: S1- Comparison with Uncertainty Sampling and Random Sampling

Figure 14: S2- Comparison with Uncertainty Sampling and Random Sampling

While on Figure 11 and Figure 12, the ensemble is directly saturated with decision trees. These results show that each classifier type may exhibit different suitability at different stages of optimization. These results shed light on the reason why the second strategy does more consistently than the first strategy. When the classifier type information is not considered, the ensemble might be prematurely saturated with a classifier type that performs better at an early stage.

3.3 Comparison with Uncertainty Sampling and Random Sampling

In this section, AIS with the second replacement strategy is compared against uncertainty sampling and random sampling. For uncertainty sampling, DTs, as one of the classifier types used in AIS, were chosen as the classifier. The confidence measure is the ratio between the number of data points belongs to the majority class label and the total number of data points in the leaf nodes of the trees. The training method for decision trees was J48, an implementation of C4.5 by Weka [22]. ANNs were not used with uncertainty sampling because designing a confidence measure for ANNs is a nontrivial task. Three training methods were combined with random sampling for comparison with AIS. In the first training method, random sampling was implanted into the AIS algorithm. The exploration phase of the algorithm was replaced by random sampling, which means that instead of choosing the data point that induces the maximal disagreement, a data point was chosen randomly out of the current pool of candidate data points. The other two methods working with random sampling were J48 and Back-propagation [10]. The Back-propagation algorithm was implemented with the generalized delta rule and incremental training,
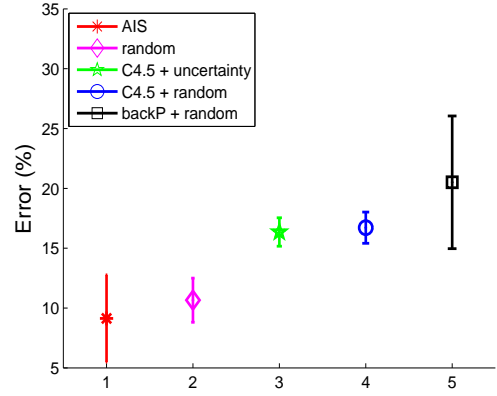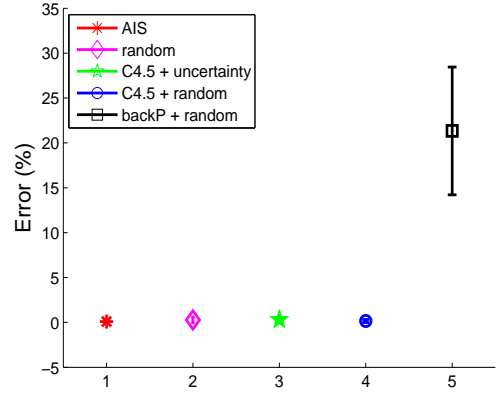
the activation function was the sigmoid function. The learning rate was 0.3 and the momentum was 0.9. The number of hidden nodes was set to be half of the number of the input features. It is worth pointing out that an increase of number of hidden nodes could sufficiently increase the accuracy of Back-propagation. The number was chosen because the ANNs used by AIS have the same number of hidden nodes. The Back-propagation algorithm was allowed to train the ANNs for up to 2000 iterations. The number was relatively arbitrary, but it is reasonable comparing with the number of iterations that ANNs in AIS have been trained for. At the end of 2000 iterations of training, the iteration with the best accuracy on the testing set was reported as the performance of Back-propagation. For each of the comparing method, the same amount of data points was chosen as was by AIS, and 30 independent runs were conducted to report the mean and standard deviation.
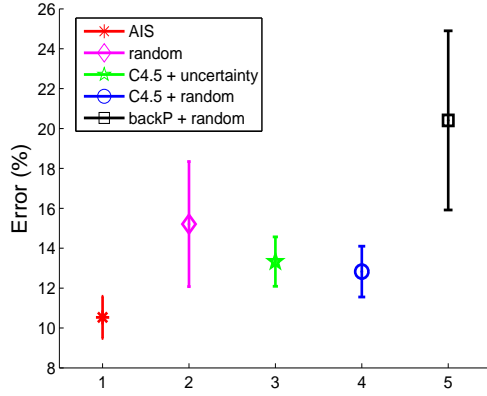
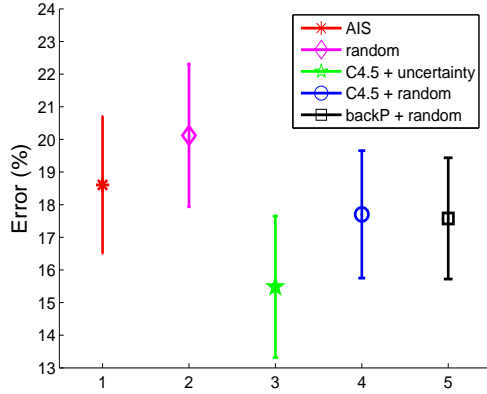Figure 15: Spam- Comparison with Uncertainty Sampling and Random Sampling

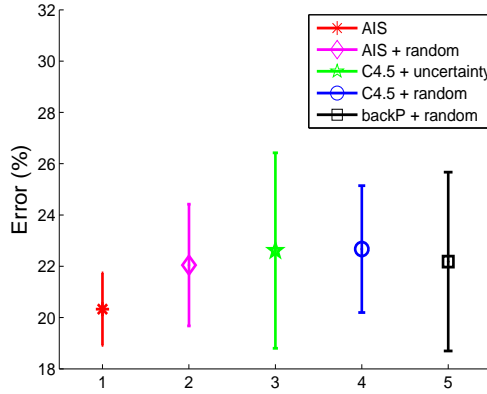Figure 16: Pendigit- Comparison with Uncertainty Sampling and Random Sampling

Figure 17: Landsat- Comparison with Uncertainty Sampling and Random Sampling

Of the five tested data sets, AIS with the second strategy performs the best or among the best on four of them. Figure 13 reports the results on S1, where AIS achieves a mean of 91.3% accuracy as reported in Section 3.2, which significantly outperforms C4.5 with uncertainty sampling (mean accuracy 83.65%), C4.5 with random sampling (mean accuracy 83.28%), and Back-propagation with random sampling (mean accuracy 79.49%). AIS with random sampling is slightly worse than AIS in mean accuracy (89.34%), but is marginally better than AIS in standard deviation. On S2, as shown by Figure 14, AIS (mean accuracy 99.91%), AIS with random sampling (mean accuracy 99.72%), C4.5 with uncertainty sampling (mean accuracy 99.70%) and C4.5 with random sampling (mean accuracy 99.84%) all achieve near 100% accuracy, which is not surprising for such a simple data set. Back-propagation performs the worst with a mean accuracy of 78.67% and a noticeably larger standard deviation. Figure 15 gives the results on Spambase data set. It is shown that AIS outperforms all competing methods with a mean accuracy of 89.5%, which are AIS plus random sampling with a mean accuracy of 84.79%, C4.5 plus uncertainty sampling with a mean accuracy of 86.67%, C4.5 plus random sampling with a mean accuracy of 87.17%, and Back-propagation plus random sampling with a mean accuracy of 79.59%. As shown by Figure 16, C4.5 with uncertainty sampling achieves a mean accuracy of 84.52% and is slightly better than the competing methods on Pendigit. C4.5 with random sampling and Back-propagation with random sampling performs closely with mean accuracies of 82.42% and 82.3%. AIS (mean accuracy 81.79) performs slightly better than AIS with random sampling (mean accuracy 79.88%). Figure 17 reports the results of Landsat data set, on which AIS is slightly better than the other methods by achieving a mean accuracy of 79.67%, compared with AIS plus random sampling with 77.95%, C4.5 plus uncertainty sampling with 77.39%, C4.5 plus random sampling with 77.33%, and Back-propagation plus random sampling with 77.82%.

4 Conclusions and Future Work

In this paper, evidence was provided to demonstrate that different classifier types exhibit different performances when incorporated into the informative sampling algorithm, which indicates the requirement for a heterogeneous ensemble because no one classifier type does consistently well across the data sets. An extension of the informative sampling algorithm that adapts the ratio of classifier types in a heterogeneous ensemble of classifiers was introduced, which is referred to as the adaptive informative sampling algorithm. Stochastic optimization algorithms were used to develop mul-

tiple instances of each classifier type. This algorithm starts with a combination of multiple classifier types and updates the relative ratio of the classifier types in the ensemble during each iteration. Of the two tested replacement strategies, the one that takes not only the accuracy but also the classifier type into account when performing replacement is shown to have a better performance. This suggests that in addition to accuracy, active learning techniques that maintain an ensemble of multiple classifier types should explicitly consider classifier type when replacing members of the ensemble. AIS that made use of this strategy achieves performances better than the informative sampling algorithm when using the better classifier type on the five data sets studied in this paper. This allows the algorithm to perform consistently well across data sets, without having to determine *a priori* a suitable classifier type. Experiments were then conducted to show that AIS outperforms random sampling and uncertainty sampling, which indicates that the algorithm chooses informative data points.

Although ANNs and decision trees were employed here, AIS is a generalized algorithm that could work with a wide range of classifier types. In previous work [14], we introduced adaptive heterogeneous ensembles (AHE), which utilizes existing training methods such as Naive Bayes to form an ensemble. An important extension of this work would be combining AHE with AIS to construct ensembles consist of classifiers trained with both existing training methods and stochastic methods. Future work will also include applying the algorithm to more real world data sets.

References

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[2] J. Bongard and H. Lipson. Automating genetic network inference with minimal physical experimentation using coevolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 333–345, 2004.

[3] J. Bongard and H. Lipson. Active coevolutionary learning of deterministic finite automata. *Journal of Machine Learning Research*, 6:1651–1678, 2005.

[4] J. Bongard and H. Lipson. Nonlinear system identification using coevolution of models and tests. *IEEE Transactions on Evolutionary Computation*, 9:361–384, 2005.

[5] J. Bongard, V. Zykov, and H. Lipson. Resilient machines through continuous self-modeling. *Science*, 314:1118–1121, 2006.

[6] C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *Proceedings of the 17th International Conference on Machine Learning*, pages 111–118, 2000.

[7] R. Caruana, A. Munson, and A. Niculescu-Mizil. Getting the most out of ensemble selection. In *Proceedings of the Sixth International Conference on Data Mining*, pages 828–833, 2006.

[8] N. Cristianini and J. S. Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, 2000.

[9] S. Dzeroski and B. Zenko. Is combining classifiers with stacking better than selecting the best one? *Machine Learning*, 54(3):255–273, 2004.

[10] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, NJ, USA.

[11] N. Japkowicz. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI Workshop on learning from Imbalanced Data Sets*, pages 10–15, 2000.

[12] D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 148–156, 1994.

[13] Z. Lu, A. I. Rughani, B. I. Tranmer, and J. Bongard. Informative sampling for large unbalanced data sets. In *Proceedings of the Genetic and Evolutionary Computation Conference, Workshop on medical applications of genetic and evolutionary computation*, pages 2047–2054, 2008.

[14] Z. Lu, X. Wu, and J. Bongard. Active learning with adaptive heterogeneous ensembles. In *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM 2009)*, Miami, FL, USA, December 6-9, 2009.

[15] A. Papagelis and D. Kalles. Breeding decision trees using evolutionary techniques. In *Proceedings of the 18th International Conference on Machine Learning*, pages 393–400, 2001.

[16] J. R. Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann, San Francisco, CA, USA, 1993.

[17] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd ed.).* Prentice Hall, Upper Saddle River NJ, 2003.

[18] G. Schohn and D. Cohn. Less is more: active learning with support vector machines. In *Proceedings of the 17th International Conference on Machine Learning*, pages 839–846, 2000.

[19] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the Fifth Workshop on Computational Learning Theory*, pages 287–294, 1992.

[20] G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective voting of heterogeneous classifiers. In *Proceedings of the European Conference on Machine Learning*, pages 465–476, 2004.

[21] J. Waksberg. Sampling methods for random digit dialing. *American Statistical Association*, 73(361):40–46, March 1978.

[22] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, San Francisco, 2005.