This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

Active Learning Through Adaptive Heterogeneous Ensembling

Zhenyu Lu, Xindong Wu, *Fellow, IEEE,* and Josh C. Bongard

Abstract—An open question in ensemble-based active learning is how to choose one classifier type, or appropriate combinations of multiple classifier types, to construct ensembles for a given task. While existing approaches typically choose one classifier type, this paper presents a method that trains and adapts multiple instances of multiple classifier types toward an appropriate ensemble during active learning. The method is termed Adaptive Heterogeneous Ensembles (henceforth referred to as AHE). Experimental evaluations show that AHE constructs heterogeneous ensembles that outperform homogeneous ensembles composed of any one of the classifier types, as well as bagging, boosting and the random subspace method with random sampling. We also show in this paper that the advantage of AHE over other methods is increased if (1) the overall size of the ensemble also adapts during learning; and (2) the target data set is composed of more than two class labels. Through analysis we show that the AHE outperforms other methods because it automatically discovers complementary classifiers: for each data instance in the data set, instances of the classifier type best suited for that data point vote together, while instances of the other, inappropriate classifier types disagree, thereby producing a correct overall majority vote.

Index Terms—ensemble methods, active learning, heterogeneous ensembles.

1 Introduction

Classification techniques build predictive models from data instances represented by features and associated class labels. In many real applications, it is easy to collect the features but obtaining the class labels is a lengthy and expensive process. For example, in email spam filtering applications, the collection of features can be automated by using computer programs to scan the emails, but to label emails as "spam" or not requires the efforts of human experts. The problem of the limited amount of labeled data which is referred to as the data scarcity problem, is a major challenge for classification techniques. Given a hidden target data distribution, the set of data instances that are sampled from it contain differing amounts of information for learning. A small subset of data instances often contain sufficient information for learning. Based on this fact, active learning is an approach that tackles the data scarcity problem by only choosing a subset of unlabeled data instances for labeling and training.

One way of categorizing active learning approaches is to consider the number of classifiers used for learning: one classifier or a set of classifiers are involved. Active learning approaches in the first category typically train one classifier of a chosen type, and choose data instances that are expected to help the training of this classifier. Uncertainty sampling [2] is a well-known representative in this category, in which the data instances that the current classifier is least certain about are considered informative. Methods that followed this approach have worked not only with probabilistic classifiers [2][3], but also with non-probabilistic classifiers such as decision trees [4], nearest-neighbor classifiers [5], and support vector machines [6]. As popular as it is, uncertainty sampling has some intrinsic problems: 1) the most uncertain data points are often not good representatives of the whole distribution, which means that pursuing such data points might lead the algorithms to mistakenly focus on outliers or noise; and 2) the quality of the chosen data instances and the performance of uncertainty sampling methods are largely limited by the quality of the single classifier. Efforts to solve the first problem took into consideration not only the uncertainty, but also the representativeness of data instances. Settles and Craven [7] proposed an information density framework, where data instances are considered informative not only when they are uncertain, but also when they are representative of the distribution. Nguyen and Smeulders [8] proposed an approach that focuses on selecting the most representative data instances by first building a classifier on the set of cluster representatives and then propagating labels in the same cluster.

The second category of active learning approaches is represented by the query-by-committee (QBC) framework [9], in which a committee of classifiers works as a group to choose informative data instances rather than relying on one classifier. The QBC framework considers the data instances that cause the most disagreement among committee members to be informative. The key to the success of QBC approaches rests on generating an accurate yet diverse committee, which

A shorter and preliminary version that introduces the Adaptive Heterogeneous Ensembles (AHE) was published as a full paper in ICDM'2009 [1]. This version extends our work in that: 1) in the methods section (Section 2), a new variant of AHE is introduced that not only adapts the combination of classifier types, but also adapts the ensemble sizes; 2) in the results section (Section 3), in addition to the previous results that incorporated C4.5 and Naive Bayes classifiers into the ensembles, the k-nearest neighbor is incorporated as well, and AHE is tested when the adaptation set is incorporated as part of the training set; and 3) experimental analysis (Section 4) reveals when and why heterogeneous ensembles outperform homogeneous ensembles.

- *Zhenyu Lu, Xindong Wu and Josh Bongard are with the Department of Computer Science, The University of Vermont, Burlington, Vermont, US, 05405*
  E-mail: {zlu, xwu, jbongard}@uvm.edu

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

is also crucial to ensemble learning methods in general. Many ensemble methods [10][11][12][13][14][15] have shown better generalization abilities than single classifiers. Ensemble-based active learning is a promising approach that combines ensemble learning with the QBC framework. Along this direction, Abe and Mamitsuka [16] proposed query-by-bagging and query-by-boosting which combine bagging [10] and boosting [11] with QBC, and Melville and Mooney [17] combined DECORATE [18] with QBC, which constructs diverse ensembles using artificial training instances.

An open question in ensemble learning is how to choose the correct classifier type(s) for constructing ensembles. As is known, different classifier types have different intrinsic strengths and biases, which makes them suitable for different tasks. However, existing approaches typically choose one particular classifier type as the base classifier type: decision trees are usually considered as the best "off-the-shelf" base classifier. Compared with the large literature of ensemble learning, this challenging problem of choosing the right classifier type is understudied, especially when it comes to the construction of heterogeneous ensembles. Dietterich [19] explained that ensemble methods partly solved three problems faced by supervised learning: 1) the statistical problem, which arises when there is limited training data for the hypothesis space; 2) the computational problem, which arises when it is computationally intractable to find the best model for the hypothesis space and heuristic methods that could become stuck in local minima are employed to train the models; and 3) the representational problem, which arises when the chosen model type is incompatible with the target distribution. The first and the second problems can be alleviated by a group decision from a set of classifiers that are equally good at explaining the training data or in different local minima. While in some cases techniques such as weighted voting can help alleviate the third problem, a better solution to the representational problem is incorporating multiple classifier types in the same ensemble. An ensemble that contains more than one classifier type is called a heterogeneous ensemble. All existing ensemble-based active learning approaches [16][17] focus on employing ensemble methods to construct homogeneous ensembles for active learning. To the best of our knowledge, our work is the first comprehensive study on heterogeneous ensembles for active learning.

Dzeroski and Zenko [20] constructed heterogeneous ensembles by including one instance of each chosen type. Caruana et al. [21] proposed an approach that first builds a large library of classifiers of different types, and then constructs heterogeneous ensembles by selecting members from the library. The previous efforts however have had limited success in answering some fundamental questions about heterogeneous ensembles: 1) are heterogeneous ensembles really required; 2) if so, when can heterogeneous ensembles be expected to work better than homogeneous ensembles; 3) why might heterogeneous ensembles work better than homogeneous ensembles; and 4) is it possible to automatically construct appropriate heterogeneous ensembles for a given task? This paper addresses all of these questions.

This paper shows that active learning with heterogeneous ensembles outperforms homogeneous ensembles, and that different ratios of classifier types in an ensemble suit different data sets. The question then arises as to how to find a suitable ratio of classifier types for a given data set. An algorithm is introduced in this paper to adapt the ratio of different classifier types in a heterogeneous ensemble during training, which we term adaptive heterogeneous ensembles (AHE). The algorithm starts with a heterogeneous ensemble and proceeds through several training iterations. In each iteration, the current ratio of different classifier types in the ensemble is adapted toward better performance, and a new ensemble of classifiers is trained according to the new combination. Two variants of the AHE are introduced: Stably-sized AHE, in which the ensemble size remains the same during training; and Variably-sized AHE, in which the ensemble size is allowed to expand or shrink for better expected performance. Although ensemble size is an important factor, most existing ensemble methods predefine a stable ensemble size as Stably-sized AHE. Systematic approaches for finding appropriate sizes for ensembles remain understudied. In this paper, the Variably-sized AHE is introduced as a variant of AHE to study the performance of heterogeneous ensembles with appropriately adapted sizes.

In this paper, Stably-sized AHE with three classifier types — C4.5, Naive Bayes and kNN — are first tested on 18 UCI data sets. The results show that heterogeneous ensembles constructed by Stably-sized AHE with three classifier types outperform other Stably-sized AHE variants with two or one classifier types, and bagging, boosting, and the random subspace method with random sampling. Variably-sized AHE is then tested and the results show that Variably-sized AHE with three classifier types outperforms Variably-sized AHE variants with two or one classifier types, Stably-sized AHE with three classifier types, and bagging, boosting, and the random subspace method with random sampling, both when the adaptation set (data used to alter the ratio of classifier types during learning) is kept separate from, or incorporated into the training set. Experimental analysis shows that different classifier types in heterogeneous ensembles adapt to complement each other, and that the greater this complementarity, the better the AHE performs, thus proving the effectiveness of the introduced measure.

The random subspace method [12] is employed to create multiple instances of each classifier type. There are other ensemble creation techniques that sample the training instances, such as bagging and boosting, but the reason for choosing the random subspace method is that by sampling the feature space there is no requirement for a reasonably large number of data instances to start training. However by allowing a starting training set, AHE could also work with ensemble creation techniques that sample training instances. There are three scenarios for active learning: membership query synthesis [22][23][24], in which synthetic data instances are generated and queried; stream-based selective sampling [25][26], in which a sequence of data instances are drawn for an active learner to decide whether to query their labels or not; and pool-based sampling [2][7][27][28], in which a large number of unlabeled data instances are presented to the active learner for querying. Experiments in this work follow

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

the stream-based selective sampling scenario because it is more efficient to scan a window of data instances than the entire training pool to choose queries at each training iteration. However AHE could work in the pool-based sampling scenario if the whole training set is deemed as the window. For a detailed introduction to these scenarios, Settles [29] has written a survey of the active learning literature.

In its present form, AHE is an iterative algorithm that constructs a new ensemble of classifiers at each iteration. If the active learning component in AHE is turned off, AHE can serve as a general ensemble method. Note that most common ensemble methods do not require an iterative training process of generating multiple ensembles, which makes AHE a better fit for ensemble-based active learning, where a new ensemble is required to be constructed with the update to the training set at each iteration. In this paper, AHE is introduced, studied and tested as an ensemble-based active learning approach.

The rest of the paper is structured as follows. Section 2 introduces AHE and its two variants. Section 3 presents the experimental results. Section 4 analyzes why heterogeneous ensembles work and Section 5 concludes the paper.

TABLE 1
Adaptive Heterogeneous Ensembles

Input: the training pool D_{pool}, the testing set D_{te}, the adaptation set D_{ad}, an initial training set D_{in}, a list of classifier types $T = \{t^{(1)}, t^{(2)}, ..., t^{(N)}\}$, an initial ensemble size M, a vector $V_{\text{in}} = \{v_{in}^{(1)}, v_{in}^{(2)}, ..., v_{in}^{(N)}\}$, where $v_{in}^{(i)}$ is the initial number of instances of classifier type $t^{(i)}$ in the ensemble, a window size W, and a stopping criterion S.
Initialize: the training set $D_{\text{tr}} = D_{\text{in}}$; the ensemble size $m = M$; $V = V_{\text{in}}$; train the initial ensemble $C = \{c^{(1)}, c^{(2)}, ..., c^{(M)}\}$ on D_{tr}, each classifier type $t^{(i)}$ has $v^{(i)}$ instances in C.
Algorithm:
while not S:
1 for each data instance d_{i} from D_{pool} in the current window:
record the vote entropy of C as VE_{i};
$t = \arg\max_i VE_{\text{i}}$;
query the label of d_{t};
add d_{t} with the acquired label to D_{tr} ;
* 2 through adaptation, update parameters of the ensemble, such as V or m;
3 train a new ensemble C' on D_{tr} according to the current V;
$C = C'$;
end while;
Output: The ensemble C trained according to the latest V.

2 Algorithm Description

Given a set of classifier types, the problem of using them to construct the optimal ensemble is nontrivial. In this paper, an optimal ensemble is such an ensemble that is constructed with the best ratio of classifier types with the smallest size. When the ensemble size is fixed, the time for an exhaustive search is polynomial to the size of the ensemble, which is inefficient because for regular ensemble methods, the ensemble size is usually large, and for ensemble-based active learning, such exhaustive searches need to be conducted in each iteration. When we allow the ensemble size to be changeable, the search space becomes infinite. We need efficient search methods both when the ensemble size is fixed and when the ensemble size is allowed to be changed. Towards this goal, this paper introduces the Adaptive Heterogeneous Ensembles (AHE) framework, which is the first attempt to tackle this problem for active learning, and two methods Stably-sized AHE and Variably-sized AHE which are based on random search that tackles this problem for when the ensemble size is fixed and changeable.

AHE is an iterative algorithm that starts with a heterogeneous ensemble. In each iteration, one data instance is chosen for labeling and added to the training set, the properties of the ensemble are adapted toward better performance, the current classifiers are discarded and a new ensemble of classifiers is trained according to the new properties. This section first introduces the framework of the AHE algorithm, and then are introduced.

2.1 Adaptive Heterogeneous Ensembles

AHE is an algorithm that constructs heterogeneous ensembles. A heterogeneous ensemble contains multiple classifier instances of multiple classifier types. The inputs to AHE are a training pool, a testing set, an adaptation set which is used for adapting the configuration of the ensemble , an initial training set, a list of classifier types, an initial ensemble size, a vector where each element represents the initial number of instances of a classifier type, a window size and a stopping criterion.

In this work, the pool of training data is made available to the algorithm in a streaming manner similar to [30]. The whole potential training set is partitioned into chunks of equal size, and in each iteration one chunk of data serves as the pool for choosing one new data point. There are two reasons to run the algorithm in a streaming manner. The first is for the efficiency of the algorithm. Active learning is especially desirable when the potential unlabeled training set is large. Therefore, scanning the whole pool of potential training data in each iteration to choose one data point is prohibitively inefficient. The second reason is that in this way the algorithm works on both static and streaming data.

AHE starts with an initial heterogeneous ensemble, where each classifier type started with the same number of instances because we assume an appropriate ratio of classifier types is not known *a priori*.. Each classifier instance is built on a randomly sampled feature subset with half of the total number of features for the initial training set, which is similar to the choice of the random subspace method [12].

The algorithm then proceeds through several training iterations. In each iteration there are three phases.

In the first phase of a training iteration, each classifier in the initial ensemble makes predictions on each of the first chunk of data instances in the training pool. The data instance that causes maximal disagreement among the ensemble members is chosen for label querying. This data instance, with its label, is then added to the training set. In this paper, the prediction disagreement among ensemble members for one data instance is evaluated by their vote entropy [31]:

$$VE = -\sum_j \frac{L_j}{M} \log \frac{L_j}{M},$$

where L_j represents the number of votes for the jth output

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

label, and M is the committee size. Although other disagreement measures such as the average Kullback-Leibler (KL) divergence [32] have also been proposed, vote entropy remains to be the most popular because there is no known study that shows the advantage of other methods over vote entropy. In this paper, AHE is introduced as a general ensemble-based active learning framework, rather than a specific algorithm implementation. For this reason, vote entropy was chosen as the disagreement measure to study AHE's performance in the general case.

The second phase of a training iteration is the adaptation phase, in which the number of instances of each classifier type is updated towards better performance. The two variants of the AHE introduced in this paper differ from each other in how they accomplish this.

In the third phase of a training iteration, the current classifiers are discarded and a new ensemble is trained on the updated training set with the updated ratio of classifier types. Each new classifier instance of the ensemble is trained on a randomly sampled feature subset with half of the total number of features. The reason for resampling the feature space in each iteration is that the training set is updated in each iteration. Then a new iteration begins.

The algorithm stops when a pre-defined stopping criterion S is met. In the conducted experiments, the stopping criteria are predefined numbers of iterations. This choice allows for convenient iteration to iteration comparisons to other peer methods, which were tested in similar settings. In practice, stopping criteria are often determined by economic or other external factors [29], rather than internal factors such as a threshold for accuracy changes over iterations. For example, the number of label queries is often determined by the labeling budget. The output of the algorithm is the ensemble trained during the last iteration. Table 1 reports the Adaptive Heterogeneous Ensembles algorithm.

The time complexity of AHE is $O(TMWF(T) + TMA(S)))$, where T is the number of iterations, M is the ensemble size, W is the size of the training pool in each iteration, $F(T)$ is the average complexity of building a model given T training instances, and $A(S)$ is the average complexity of make predictions on S data instances in the adaptation set. For most existing classifiers, it is much more time consuming to build the classifiers than making predictions using built classifiers. The exceptions are the lazy classifiers. Thus, in most cases, the time complexity of AHE is in the order of $O(TMWF(T)$, which is the same as Query-by-Bagging.

2.2 Stably-sized Adaptive Heterogeneous Ensembles

In this section, Stably-sized Adaptive Heterogeneous Ensembles is introduced as the first variant of the AHE framework in the context of active learning. In this variant, the ratio of instances of each classifier type is adapted. The Stably-sized Adaptive Heterogeneous Ensembles is the same as the general AHE framework except for the adaptation phase.

In the second phase of a training iteration, variants of the current ensemble are created. For each classifier type, one member of its type is randomly chosen to be taken out of the whole ensemble. All reduced ensembles, as well as the original ensemble, are then tested on the adaptation set for their accuracies. If one variant achieves the highest accuracy, then the number of classifiers of its corresponding type is decreased by one, because it is expected to increase the accuracy of the ensemble. If one variant achieves the lowest accuracy, its corresponding type is increased by one. If no variant is more accurate than the original ensemble, then the current configuration is retained. For example, if the original ensemble contains six C4.5 classifiers and four Naive Bayes classifiers, the algorithm generates two variants of the original ensemble. The first variant contains five C4.5 classifiers and four Naive Bayes classifiers, with one C4.5 classifier randomly taken out. The second variant contains six C4.5 classifiers and three Naive Bayes classifiers, with one Naive Bayes classifier randomly taken out. If the first variant achieves the best accuracy and the second variant achieves the worst accuracy on the adaptation set, the number of C4.5 classifiers is decreased by one and the number of Naive Bayes classifiers is increased by one. The ensemble in the next iteration will contain five C4.5 classifiers and five Naive Bayes classifiers.

The choice of searching all the variants of the current ensemble that has one less classifier is to keep the search effort manageable. For example, considering all the variants of two less classifiers will bring the search complexity to polynomial rather than linear, and considering all the subensembles makes the search problem exponential. Furthermore, It is accepted that larger ensemble sizes generally provides better predictive accuracies, which means that it is unlikely that we are losing much search space by not considering other smaller ensemble variants. Table 2 reports the algorithm for Stably-sized Adaptive Heterogeneous Ensembles.

TABLE 2
Stably-sized Adaptive Heterogeneous Ensembles

* 2 for each $t^{(i)}$ in T:
  $c^{(r)}$ is a random classifier of type $t^{(i)}$;
  record the accuracy of $C \setminus c^{(r)}$ on D_{ad} as ACC_{i};
end for;
record the accuracy of C on D_{ad} as ACC_{c};
$p = \arg\max_i ACC_{\text{i}}$;
$q = \arg\min_i ACC_{\text{i}}$;
if $ACC_{\text{p}} > ACC_{\text{c}}$:
  $v^p = t^p - 1$;
  $v^q = t^q + 1$;
else: remain the current V;
end if;

2.3 Variably-sized Adaptive Heterogeneous Ensembles

The Variably-sized Adaptive Heterogeneous Ensembles is different from the Stably-sized Adaptive Heterogeneous Ensembles because in its adaptation phase the overall ensemble size is changed along with the number of instances of each type.

In its adaptation phase, two subsets of ensemble variants are created. In the first subset, each classifier is taken out of the original ensemble. In the second subset, a new instance is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 3
Variably-sized Adaptive Heterogeneous Ensembles

$type(i)$ returns the corresponding classifier type for the ith classifier;
* 2 for each $c^{(i)}$ in C:
  record the accuracy of $C \setminus c^{(i)}$ on D_{ad} as $ACC_{\text{i}}^{(-)}$;
end for;
for each $t^{(i)}$ in T:
  c^r is a new instance of $t^{(i)}$ trained on D_{tr};
  $C^{'}$ = c^r + C;
  record the accuracy of $C^{'}$ as $ACC_{\text{i}}^{(+)}$
end for;
record the accuracy of C on D_{ad} as ACC_{c};
$p = type(\arg\max_i ACC_{\text{i}}^{(-)})$;
$q = \arg\max_i ACC_{\text{i}}^{(+)}$;
if $ACC_{\text{p}} > ACC_{\text{c}}$ and $ACC_{\text{p}} > ACC_{\text{q}}$:
  $t^p = t^p - 1$, $m = m - 1$;
else if $ACC_{\text{q}} > ACC_{\text{c}}$ and $ACC_{\text{q}} > ACC_{\text{p}}$:
  $t^q = t^q + 1$, $m = m + 1$;
else: remain the current V;
end if;

created for each classifier type, and then added to the current ensemble. All variants and the current ensemble are tested on the adaptation set for accuracy. If one variant with reduced size achieves the best accuracy, then the number of classifiers of its corresponding type is decreased by one. The numbers of instances of other types remains the same, thereby leading to a reduction of one in the size of the ensemble. If one variant with increased size achieves the best accuracy, then the number of classifiers of its corresponding type is increased by one. The numbers of instances of other types remains the same, thereby leading to an addition of one in the size of the ensemble. If the current ensemble achieves the highest accuracy, the size and internal ratio of classifier types remain unchanged. For example, if an ensemble contains six C4.5 classifiers and four Naive Bayes classifiers, the algorithm generates 12 variants of the original ensemble. Ten reduced ensemble variants of size nine are generated and placed in the first subset. The remaining two expanded variants are generated and placed in the second subset. The first variant is constructed by randomly generating a new C4.5 classifier and combining it with the original ensemble. The second variant is constructed by randomly generating a new Naive Bayes classifier and combining it with the original ensemble. If for example the ensemble variant with one more C4.5 classifier achieves the best accuracy on the adaptation set, then the number of C4.5 classifiers is increased by one, and the ensemble size is increased by one. The ensemble in the next iteration will contain seven C4.5 classifiers and four Naive Bayes classifiers. The reason for limiting the search space to ensemble variants with one more or one less classifier is similar to what has been discussed in Section 2.2. Table 3 reports the algorithm for Variably-sized Adaptive Heterogeneous Ensembles.

2.4 Analysis of Heterogeneous Ensembles

This section analyzes the performance of heterogenous ensembles to investigate why heterogeneous ensembles could achieve better accuracies. Assume there is a trained ensemble of size M, which contains equal numbers of classifiers from two classifier types $t^{(1)}$ and $t^{(2)}$. Denote the data set as D, and the number of class labels as N. Given the testing set D_{te}, the testing instances can be decomposed as:

$$Pred(D_{te}) = Pred_{\text{same}}(D_{te}) + Pred_{\text{diff}}(D_{te}),$$

where $Pred_{\text{same}}(D_{te})$ represents the set of data instances in D_{te} that the two subensembles of all $t^{(1)}$ classifiers and all $t^{(2)}$ classifiers are either both correct or wrong in predictions, and $Pred_{\text{diff}}(D_{te})$ represents the set of data instances that only one of the classifier types are recognizing. For data instances in $Pred_{\text{same}}(D_{te})$, having an extra classifier type makes no difference.

For d_i in $Pred_{\text{diff}}(D_{te})$, assume the ensemble of $t^{(1)}$ classifiers makes correct prediction, and the ensemble of $t^{(2)}$ classifiers makes incorrect prediction. Denote $p_{t^{(1)}}$ as the probability of a $t^{(1)}$ classifier to make the correct prediction, under our assumption, $p_{t^{(1)}}$ is greater than $1/N$. Denote $p_{t^{(2)}}$ as the probability of a $t^{(2)}$ classifier making the correct prediction. The expected number of correct predictions of the ensemble on d_i is then:

$$V_c = p_{t^{(1)}} * \frac{M}{2} + p_{t^{(2)}} * \frac{M}{2}.$$

Assume a classifier makes a random guess on any of the rest of the class labels, the expected number of predictions on any incorrect label is then:

$$V_o = \frac{(1 - p_{t^{(1)}}) * \frac{M}{2}}{N-1} + \frac{(1 - p_{t^{(2)}}) * \frac{M}{2}}{N-1}.$$

The difference between V_c and V_o is:

$$V_c - V_o = \frac{N * (p_{t^{(1)}} + p_{t^{(2)}}) - 2}{N-1} * \frac{M}{2}.$$

The following observations can be made from the above equation: 1) if the classifiers of $t^{(2)}$ make a random guess for d_i, the heterogeneous ensemble makes correct prediction independent of $p_{t^{(1)}}$ on d_i ($V_c - V_o > 0$), because $N * p_{t^{(1)}} > 1$ and $N * p_{t^{(2)}} = 1$. Note that $p_{t^{(1)}}$ is greater than $1/N$ from our assumption; and 2) when $p_{t^{(1)}} > 2/N$, the heterogeneous ensemble makes correct prediction independent of $p_{t^{(2)}}$ on d_i, because $N * p_{t^{(1)}} > 2$ and $N * p_{t^{(2)}} >= 0$. Note that the constraint on $p_{t^{(1)}}$ decreases as N grows larger. For example, when $N = 10$, $p_{t^{(1)}}$ only needs to be larger than 0.2. In conclusion, heterogeneous ensembles can outperform homogeneous ensembles because when two classifier types disagree, the classifier type that makes correct predictions can "correct" the whole ensemble by dominating the decision of the ensemble, and heterogeneous ensembles are expected to perform better when there are more class labels.

3 Experimental Results

This section introduces the nomenclature, the experimental settings, the data sets and then reports the experimental results in three subsections. In subsection 3.3, variants of the Stably-sized Adaptive Heterogeneous Ensembles are compared with bagging, boosting, the random subspace method with random sampling, and query-by-bagging (henceforth referred to as

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 4
Characteristics of data sets used in the experiments

Data sets	Size	#Features	#Classes	#Iterations	Window size
isolet	7797	617	26	200	25
letter	20000	16	26	500	28
madelon	4400	500	2	364	5
magic	19020	10	2	1000	13
mfeat-factors	2000	216	10	280	5
mfeat-fourier	2000	76	10	280	5
mfeat-karhunen	2000	64	10	280	5
mfeat-pixel	2000	240	10	280	5
mfeat-zernike	2000	47	10	280	5
mushroom	8124	22	2	1000	5
optdigits	5620	64	10	786	5
page	5473	10	5	766	5
pendigits	10992	16	10	1000	7
landsat	6435	36	6	900	5
segment	2310	19	7	323	5
spambase	4601	57	2	644	5
splice	3190	60	3	446	5
waveform(2)	5000	40	3	700	5

QBB). In subsection 3.4, variants of the Variably-sized Adaptive Heterogeneous Ensembles are compared with bagging, boosting, the random subspace method with random sampling, query-by-bagging (henceforth referred to as QBB), and the best variant of Stably-sized Adaptive Heterogeneous Ensembles. In this paper the adaptation set is an independent subset of labeled data that is required by AHE variants. In subsection 3.5, after the training instances were chosen either by random sampling or AHE, the adaptation set was combined with the chosen data instances to form the training set, and then ensembles were trained on the training set, thereby equalizing the labeled data points required by AHE and the comparing methods.

3.1 Nomenclature

Three classifier types — C4.5 decision tree, Naive Bayes, and k-nearest neighbors — and two algorithm variants — Stably-sized AHE and Variably-sized AHE — were studied in this paper. Therefore there are 14 possible AHE variants with at least one classifier type. This section introduces the nomenclature for these variants. Each variant is represented by a string "AHE_" concatenated with two other strings. The first string indicates how many classifier types are involved. The second string indicates whether the variant is Stably-sized AHE or Variably-sized AHE. For example, "AHE_CNK_Var" represents the Variably-sized variant that uses all three classifier types, and "AHE_C_St" represents the Stably-sized variant that has only the C4.5 classifier type.

3.2 Experimental Settings and Data Sets

For each AHE variant, the random subspace method was employed to create multiple instances from the same classifier type. The size of the feature subspace was chosen to be half of the size of the feature set, which is similar to the setting in the original paper of the random subspace method [12]. The three base learners were the corresponding implementations by Weka [33] with default settings, except for the k-nearest neighbors classifiers where k was chosen to be 5 instead of using the default setting 1. Eighteen data sets were chosen from the UCI machine learning repository [34] to evaluate the performance of AHE. The smallest chosen data sets contain 2000 data instances because there should be a relatively large number of data instances in the stream-based selective sampling scenario [see section 1]. The data sets were chosen to cover a variety of domains and properties such as the number of features and output labels. For each data set, 70% of the data instances were randomly placed in the training pool, 15% of the data instances were randomly placed in the testing set, and the remaining 15% were placed in the adaptation set. The window sizes and number of iterations were chosen to balance the running time and performance. For most data sets, approximately 20% of the data instances in the pool was placed in the training set after the active learning processes were finished, therefore the window sizes were 5. This is because we are choosing one data instance for labeling in each iteration out of the current window. For some data sets such as "isolet", the window sizes were set to make sure that the experimental results could come back in a timely manner. The number of iterations, which serves as the stopping criteria, is determined by the following formula:

$$\#\text{Iterations} = \lceil \frac{\#\text{of data instances}}{\text{Window size}} \rceil.$$

We are interested in small ensembles in this paper because in our scenario creating a large new ensemble in each iteration is inefficient. For all experiments, the initial ensemble size was chosen to be 12 because it is divisible by 2 and 3, which makes it easier to test variants of AHE with both two and three classifier types. For each AHE variant, the starting ensemble contained an even number of instances of each classifier type involved. Each initial training set contained one randomly chosen data instance with the label. Table 4 reports the characteristics of each data set used.

For the random sampling methods we compared the AHE to — bagging, boosting and the random subspace method — we used the implementations by Weka [33] with default settings. For existing ensemble-based active learning approaches, we compared the AHE to query-by-bagging (QBB) [16]. Query-by-boosting was not compared because: 1) it achieved similar performance to QBB; and 2) the original query-by-boosting can only deal with two-class problems, and there is no proven way of extending it to multi-class problems. The base classifier for the four methods was the C4.5 classifier implemented by Weka [33]. The choice of C4.5 as the base classifier for the three methods is because: 1) decision trees are regarded as the best "off-the-shelf" base classifier, and C4.5 is one of the most popular training methods for decision trees; and 2) the original forms of the three methods used decision trees as the base classifier. Due to space limits, the other two base classifiers — kNN and Naive Bayes — were not chosen to be tested with the three methods when using random sampling. Instead AHE variants of all combinations of the three classifier types were tested with the random subspace method when using active learning. In each iteration for the three benchmark methods, one data instance was randomly chosen to be added to the training set. For each method, 30 independent trials were conducted.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

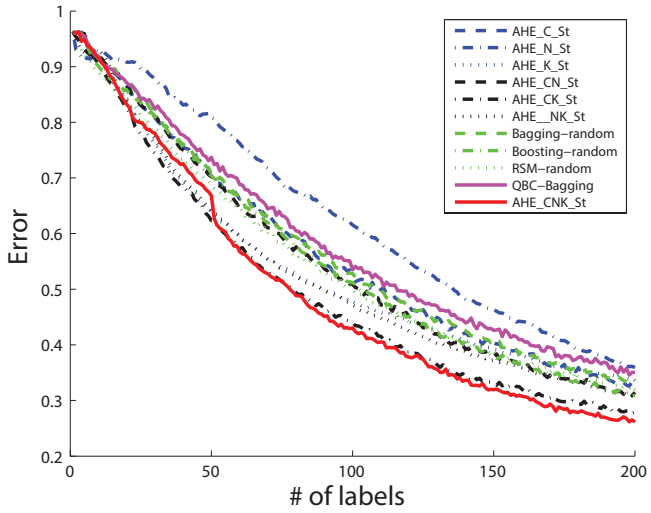


Fig. 1. Error curves of AHE_CNK_St and the other peer methods on the "isolet" data set. When all methods have 200 data points in the training set, AHE_CNK_St ties with AHE_CK_St and is statistically significantly better than all other tested methods(statistical t-test, significance level 95%).

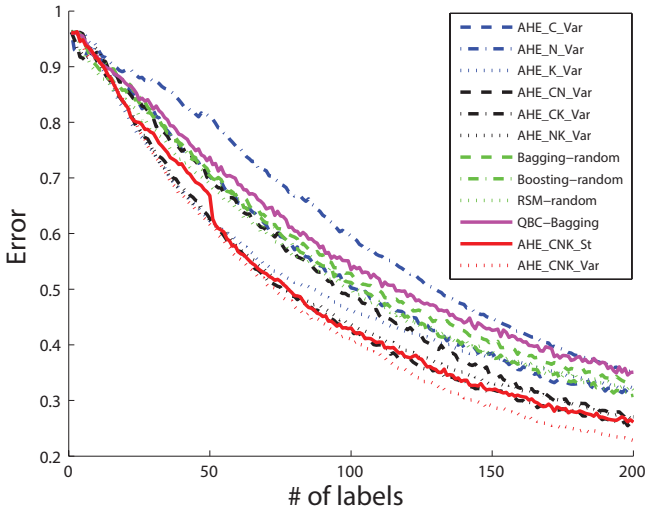


Fig. 2. Error curves of AHE_CNK_Var and the other peer methods on the "isolet" data set. When all methods have 200 data points in the training set, AHE_CNK_Var is statistically significantly better than all other tested methods(statistical t-test, significance level 95%).

TABLE 5
Accuracy comparison of AHE_CNK_St and other methods. Each cell represents the win-tie-loss count (statistical t-test, significance level 95%) when the compared methods have chosen a certain percentage of training points for labeling on 18 data sets.

#labels	20%	40%	60%	80%	100%
AHE_C_St	13-5-0	12-6-0	11-7-0	10-8-0	9-8-1
AHE_N_St	15-3-0	14-4-0	13-5-0	15-3-0	16-2-0
AHE_K_St	9-8-1	9-9-0	9-9-0	7-11-0	7-11-0
AHE_CN_St	8-10-0	8-10-0	8-10-0	10-8-0	9-9-0
AHE_CK_St	4-13-1	5-13-0	3-15-0	5-13-0	3-15-0
AHE_NK_St	8-9-1	9-9-0	9-9-0	9-9-0	9-9-0
Bagging	17-0-1	16-0-2	15-1-2	14-2-2	15-2-1
Boosting	15-2-1	14-2-2	14-2-2	14-1-3	15-1-2
Random Subspace Method	17-1-0	16-1-1	14-3-1	13-4-1	14-2-2

3.3 Stably-sized Adaptive Heterogeneous Ensembles

The experimental results of Stably-sized AHE are reported first in Figure 1 on one example data set, and then Table 5 on all 18 data sets. Figure 1 reports the error curves of variants of Stably-sized AHE, as well as bagging, boosting, and the random subspace method with random sampling on the "isolet" data set. At the end of the 200 iterations, it is shown that Stably-sized AHE with all three classifiers achieves lower mean errors than all the tested methods. Statistical t-tests (significance level 95%) indicate that Stably-sized AHE with all three classifiers ties with Stably-sized AHE with C4.5 and k-nearest neighbors classifiers, and significantly outperforms all the other methods.

Each cell of Table 5 summarizes the win-tie-loss count (statistical t-test, significance level 95%) on 18 data sets between Stably-sized AHE with all three classifiers and the other tested methods at several stages of training. For example, in the first cell of the first column, "13-5-0" represents that at the end of 20% of the training iterations, Stably-sized AHE with all three classifiers significantly outperforms Stably-sized AHE with only C4.5 on 13 data sets and ties on five data sets with no losses. It is shown that Stably-sized AHE with all three classifiers outperforms other variants of Stably-sized AHE and bagging, boosting, the random subspace method with random sampling.

3.4 Variably-sized Adaptive Heterogeneous Ensembles

The experimental results obtained with Variably-sized AHE are divided into three parts: errors, ensemble sizes and combinations of classifier types found in the trained ensembles. The results are first reported as figures for an example data set, then as tables that summarize the results on all 18 data sets. In the adaptation phase of Variably-sized AHE, for each classifier type, a new classifier instance is created, and the variant that combines the new classifier instance with the original ensemble is denoted as $e_{t^{(i)}+1}$. This subset of variants , denoted as E_+ are then compared with E_- (the set of variants with each existing classifier taken out) and the original ensemble to determine the combination of classifier types and the ensemble size in the next iteration. For Variably-sized AHE with homogeneous ensembles, E_+ contains one instance, while for AHE_CNK_Var, E_+ contains three instances. If AHE_CNK_Var outperforms Variably-sized AHE with homogeneous ensembles, there are then two possible reasons: 1) the heterogeneity helps; and 2) heterogeneous ensembles tend to construct larger ensembles because they have more chances to increase the sizes of the ensembles. In our experiments, to eliminate the concern about the second reason, in the adaptation phase of Variably-sized AHE with homogeneous ensembles, three new classifier instances from the same type were created and combined with the original ensemble to compare with E_- and the original ensemble. Therefore Variably-sized AHE with homogeneous ensembles had equal chances of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 6
Accuracy comparison of AHE_CNK_Var and other methods. Each cell represents the win-tie-loss count (statistical t-test, significance level 95%) on 18 data sets.

#labels	20%	40%	60%	80%	100%
AHE_C_Var	13-5-0	14-3-1	11-5-2	12-5-1	12-6-0
AHE_N_Var	18-0-0	16-2-0	15-3-0	16-2-0	17-1-0
AHE_K_Var	13-4-1	13-5-0	12-6-0	13-4-1	13-4-1
AHE_CN_Var	13-5-0	13-5-0	14-4-0	14-4-0	13-5-0
AHE_CK_Var	11-7-0	14-4-0	10-8-0	7-11-0	8-10-0
AHE_NK_Var	10-8-0	13-5-0	14-4-0	14-4-0	16-2-0
Bagging	16-1-1	17-1-0	17-0-1	17-0-1	17-0-1
Boosting	15-3-0	16-2-0	16-2-0	17-1-0	18-0-0
Random Subspace Method	17-1-0	18-0-0	16-2-0	17-1-0	17-0-1
AHE_CNK_St	5-12-1	8-9-1	10-7-1	8-10-0	12-6-0

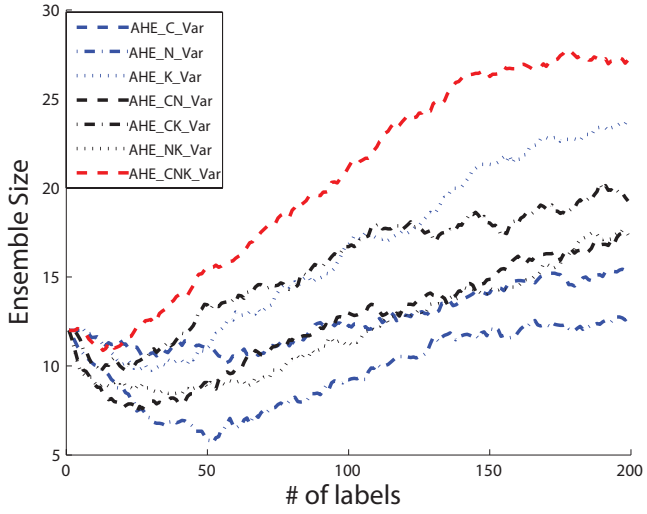


Fig. 3. Ensemble size changes of variants of Variably-sized AHE on the "isolet" data set.

increasing the ensemble sizes as AHE_CNK_Var that uses all three classifier types. This makes sure that if AHE_CNK_Var outperforms Variably-sized AHE with homogeneous ensembles, it is because of the heterogeneity.

Figure 2 reports the error curves for all variants of Variably-sized AHE (henceforth denote as AHE_Var), AHE_CNK_St, and bagging, boosting, and the random subspace method with random sampling. It is shown that AHE_CNK_Var achieves lower errors than the other methods after 200 iterations. Statistical t-tests (significance level 95%) show that the differences are significant.

Table 6 summarizes the win-tie-loss count between AHE_CNK_Var and the other methods on all 18 data sets. It is shown that AHE_CNK_Var not only outperforms other variants of Variably-sized AHE, bagging, boosting, and the random subspace method with random sampling, but also outperforms AHE_CNK_St.

Figure 3 reports the ensemble size changes of AHE_Vars during training on the "isolet" data set. It is shown that AHE_CNK_Var grows larger ensembles than other variants. Note that Variably-sized AHE with homogeneous ensembles has an equal chance of increasing the ensemble sizes.

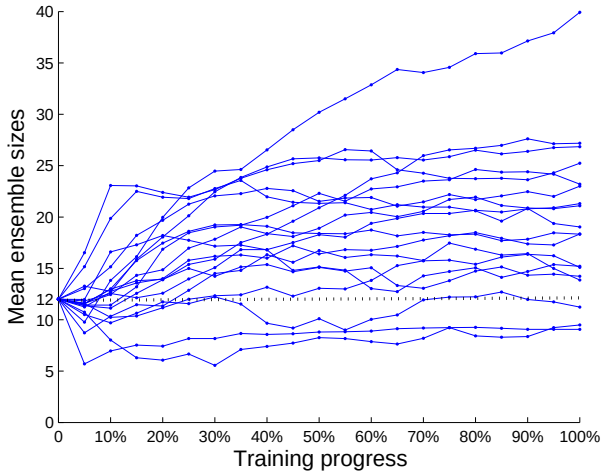


Fig. 4. Ensemble size changes of AHE_CNK_Var on all 18 data sets.

Table 7 summarizes the differences in the final ensemble sizes between AHE_CNK_Var and all the other variants. Each cell represents the percentage ratio of the mean final ensemble sizes between the corresponding method and AHE_CNK_Var. For example, in the first cell of the first column, "87.84%" means that the average sizes of the ensembles constructed by AHE_C_Var are 87.84% of the sizes of the ensembles constructed by AHE_CNK_Var. Percentages in bold indicate that the mean final ensemble sizes of the corresponding variant is statistically significantly larger than the mean final ensemble sizes of AHE_CNK_Var. It is shown that AHE_CNK_Var tends to construct larger ensembles than most other variants. The only exception is that AHE_C_Var generally constructs larger ensembles than AHE_CNK_Var. Note however that AHE_CNK_Var outperforms AHE_C_Var (Table 6, top row).

Figure 4 reports the mean ensemble sizes of AHE_CNK_Var on all 18 data sets. Each dot represents the mean ensemble size of AHE_CNK_Var on one data set when the corresponding percentage of training instances has been chosen into the training set. It is shown here that for most data sets, the mean ensemble sizes tend to change drastically at early stages, but plateaus towards the later stages.

Figure 5 reports the mean numbers of instances of each classifier type of the ensembles constructed AHE_CNK_Var over training iterations on the "isolet" data set. It is shown that the constructed ensembles contain more C4.5 classifiers in the ensemble than the other two classifier types. Although the number of classifiers from the other two classifier types are small, their existence accounts for the improved accuracy of AHE_CNK_Var over AHE_C_Var (see Figure 2).

Table 8 summarizes the average percentage of each classifier type in the ensembles constructed by AHE_CNK_Var on all 18 data sets. For each data set and each classifier type, the mean and standard deviation of the number of instances in the final ensembles are reported. It is shown in the last row that, on average over all data sets, the ensembles constructed by AHE_CNK_Var contain 47.4% of C4.5 classifiers, 33.2%

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 7
Comparison of the sizes of the ensembles constructed by AHE_CNK_Var and other variants of Variably-sized AHE. Each cell shows the percentage of the corresponding method to AHE_CNK_Var. Numbers are shown in bold when the ensemble sizes of the corresponding method are statistically larger (t-test, significance level 95%) than the ensemble sizes produced by AHE_CNK_Var.

Data Sets	AHE_C_Var	AHE_N_Var	AHE_K_Var	AHE_CN_Var	AHE_CK_Var	AHE_NK_Var
isolet	87.84%	46.91%	56.7%	63.9%	70.89%	64.88%
letter	**117.53%**	42.65%	63.02%	68.86%	79.55%	49.67%
madelon	92.98%	61.75%	87.72%	60.35%	62.46%	59.3%
magic	78.85%	84.62%	**120.67%**	53.61%	81.97%	88.94%
mfeat-factors	**128.36%**	37.82%	36.0%	63.09%	60.55%	44.55%
mfeat-fourier	100.43%	56.03%	65.52%	66.24%	66.24%	50.0%
mfeat-karhunen	**123.59%**	73.91%	88.02%	66.42%	71.51%	52.14%
mfeat-pixel	78.37%	47.39%	62.1%	48.02%	56.47%	48.49%
mfeat-zernike	110.86%	70.66%	78.59%	67.62%	60.67%	52.93%
mus	97.43%	112.87%	100.0%	76.47%	80.88%	73.16%
optdigits	**134.48%**	77.28%	89.04%	76.75%	61.96%	57.73%
page	81.01%	55.79%	97.33%	61.13%	54.01%	62.91%
pendigits	**136.43%**	66.73%	**116.99%**	79.16%	89.84%	74.26%
landsat	**143.2%**	39.91%	67.32%	73.46%	82.24%	46.05%
segment	104.89%	40.62%	**117.44%**	60.26%	67.11%	56.51%
spambase	89.93%	47.78%	105.85%	70.02%	79.16%	88.76%
splice	105.47%	70.49%	**117.11%**	64.92%	87.1%	48.26%
waveform(2)	109.02%	48.31%	100.12%	62.32%	82.86%	64.14%

TABLE 8
Percentages of classifier types in the ensembles constructed by AHE_CNK_Var on 18 Data Sets. For each classifier type, the means and standard deviations of its percentages in the final ensembles are reported. The last column reports the improvement (mean accuracy of AHE_CNK_Var over average mean accuracy of other methods). The 18 data sets are reordered decreasingly according to their improvements. The last row reports the average of mean percentages across 18 data sets for each classifier type, and the Spearman's rank correlation efficient between the standard deviations of corresponding classifier type and the last column.

Classifier Types	C45		NB		KNN		Improvement
Data Sets	mean	std	mean	std	mean	std	
letter	0.71	0.05	0.11	0.04	0.19	0.04	1.068
mfeat-pixel	0.37	0.07	0.2	0.05	0.43	0.07	1.056
isolet	0.58	0.07	0.18	0.04	0.24	0.05	1.05
mfeat-karhunen	0.43	0.08	0.2	0.06	0.37	0.07	1.05
mfeat-zernike	0.31	0.1	0.23	0.08	0.46	0.09	1.047
mfeat-fourier	0.53	0.09	0.25	0.05	0.22	0.06	1.041
segment	0.54	0.09	0.12	0.08	0.34	0.1	1.032
spambase	0.57	0.12	0.15	0.06	0.28	0.1	1.031
mfeat-factors	0.53	0.08	0.16	0.09	0.31	0.09	1.024
pendigits	0.37	0.1	0.06	0.05	0.57	0.09	1.023
optdigits	0.32	0.09	0.15	0.05	0.53	0.08	1.02
splice	0.44	0.09	0.41	0.1	0.15	0.07	1.019
landsat	0.49	0.1	0.08	0.05	0.44	0.09	1.019
page	0.61	0.12	0.2	0.11	0.18	0.11	1.012
waveform(2)	0.34	0.09	0.34	0.07	0.32	0.09	1.009
mus	0.66	0.15	0.15	0.1	0.2	0.16	1.004
madelon	0.34	0.17	0.31	0.13	0.36	0.19	0.996
magic	0.4	0.11	0.21	0.08	0.39	0.11	0.993
average percentage — rank correlation	0.474	-0.75	0.195	-0.65	0.332	-0.75	

of k-nearest neighbor classifiers, and 19.5% of Naive Bayes classifiers. Although on average the ensembles contain more C4.5 classifiers and less Naive Bayes classifiers, the case is different for each data set. For example, for the "Pendigit" and "Optdigit" data sets, the ensembles contain more k-nearest neighbor classifiers. Ensembles constructed for the "Splice" and "Waveform" data sets contain over 33% of Naive Bayes classifiers.

It can be seen in Table 9 that for some data sets, the ratio of base classifier types is consistent across independent runs (e.g. note the standard deviation for the "letter" data set), and for other data sets the ensembles across runs converge to very different ratios (e.g. note the standard deviation for the "madelon" data set). We hypothesized that for these latter data sets, different base classifiers are interchangeable: different ratios of classifier types can reach similar low prediction errors. For the former data sets however, different classifier types are required to explain different subsets of the data, so separate runs converge to the same ratio of classifier types. This hypothesis produces the following prediction: the greater the consistency of classifier type ratios across runs, the greater the improvement margin of the AHE_CNK_Var over the tested homogeneous ensemble methods (AHE_C_Var, AHE_N_Var, AHE_K_Var, bagging, boosting and the random subspace method with random sampling). An improvement metric is introduced to indicate how much AHE_CNK_Var

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

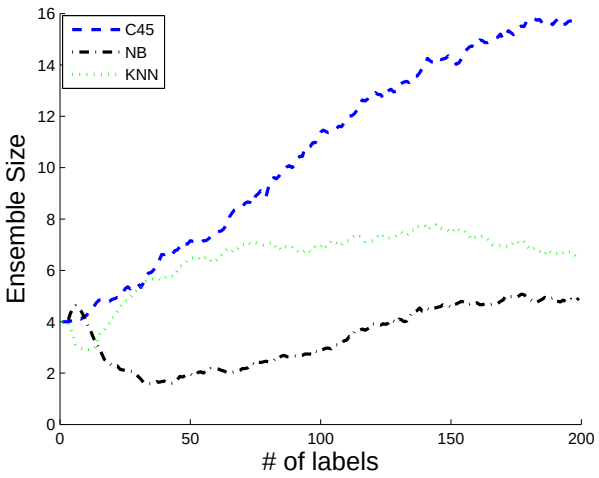


Fig. 5. Ensemble combinations of AHE_CNK_Var on "isolet" data set.

outperforms other homogeneous methods. For each data set, the improvement is defined as:

$$\text{improvement} = \frac{\text{mean accuracy of AHE\_CNK\_Var}}{\text{mean accuracies of homogeneous methods}}. \quad (1)$$

If the prediction is correct, the standard deviations of base classifier memberships should be inversely correlated with the improvement. The improvements are reported in the last column of Table 8, and the Spearman's rank correlation coefficients between the standard deviation of each classifier type and improvement are reported in the last row. The hypothesis is supported in that the standard deviations are inversely correlated to the improvements with correlation coefficients -0.75, -0.65 and -0.75 for C4.5, Naive Bayes, and kNN classifiers respectively.

3.5 Adaptation Set Incorporated into Training Set

In the previous two sections, the AHE requires additional labeled data points to determine how to change the ratio of base classifier types, even though these additional data points are not used to training the ensemble. It could be argued that this provides an unfair advantage for the AHE over the other methods tested. To demonstrate that this is not the case, 30 independent trials of all 11 ensemble variants (bagging, boosting, RSM and the eight variants of the AHE) were re-run. The AHE variants were trained as before, but after training a new ensemble with the discovered ratio of classifier types was trained on the training and adaptation sets. Bagging, boosting, and the random subspace methods were trained on a random training set and the adaptation set. Table 9 reports the error percentages of bagging, boosting, and RSM with random sampling, AHE_Vars and AHE_CNK_St. Each cell reports the mean and standard deviation of the error percentages of the corresponding method on the corresponding data set. It can be seen that in most cases AHE_CNK_Var outperforms the other compared methods.

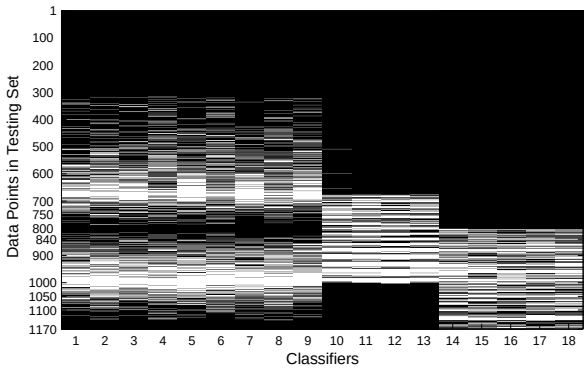


Fig. 6. Complementarity on "isolet" data set. A black square represents the classifier made correct prediction on the corresponding data point, a white square represents an incorrect prediction.

4 Analysis and Discussion

AHE_CNK_Var was shown in Section 3 to outperform bagging, boosting and the random subspace method with random sampling. There are three possible explanations for the AHE_CNK_Var's superior performance: 1) the ensemble size is adapted during training in AHE_CNK_Var; 2) the quality of the data instances chosen by the active learning process of AHE_CNK_Var are better than those chosen by random sampling; and 3) AHE_CNK_Var allows for the combination of different classifier types. The first factor can not be the only contributing factor because AHE_CNK_St, which does not change the size of the ensemble during training, was shown in Table 3.3 to also outperform random sampling methods. If only the first and the second factors are required to explain AHE_CNK_Var's superior performance, then AHE_CNK_Var should not outperform its homogeneous variants, such as AHE_C_Var (Variably-sized AHE with only C4.5 classifiers in the ensemble). Since Table 6 shows that AHE_CNK_Var outperforms all of its homogeneous variants, the heterogeneity of AHE_CNK_Var must also be a contributing factor to its performance.

Section 3 addressed the first of the four questions about heterogenous ensembles discussed in Section 1: do ensembles containing multiple classifier types outperform ensembles of only a single type?. The fourth of the four questions — is it possible to automatically construct appropriate heterogeneous ensembles that outperform homogeneous ensembles? — is also addressed because heterogeneous ensembles constructed by AHE was shown to outperform homogeneous ensembles. This section addresses the second and the third question: when are heterogeneous ensembles expected to outperform homogeneous ensembles, and when they do, why?

We will first focus on the third question because the understanding of this question will shed light on the second question. To understand how heterogeneous ensembles work, a single trial from the application of AHE_CNK_Var to the "isolet" data set was chosen and predictions of the constructed ensemble were plotted in Figure 6. Rows of Figure 6 represent data instances in the testing set, and columns represent classifiers in the constructed ensemble. Each line segment in Figure

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 9
Error percentage of AHE_CNK_Var and other methods on 18 Data Sets when the adaptation is incorporated into the training set. Each cell reports the mean and standard deviation of a method on one data set. For each data set, the top three accurate methods based on average error percentage are bolded. The last row summarizes the win/tie/loss situation of AHE_CNK_Var compared with the corresponding method in pair-wise t-tests at 95% significance level.

Data Sets	Non-AHE Approaches				AHE Variants							
	Bagging	Boosting	RSM	QBB	C_Var	N_Var	K_Var	CN_Var	CK_Var	NK_Var	CNK_St	CNK_Var
isolet	16.96± 0.94	12.13± 0.77	14.46± 0.81	15.95± 0.88	12.97± 0.76	16.68± 0.77	15.03± 0.59	11.58± 0.66	**10.05**± **0.69**	11.99± 0.75	**10.57**± **0.76**	**8.91**± **0.54**
letter	15.33± 0.57	11.39± 0.49	14.48± 0.96	14.31± 0.6	**10.2**± **0.36**	42.52± 2.02	12.84± 1.26	12.48± 0.68	**10.07**± **0.67**	17.42± 2.14	14.19± 1.09	**10.38**± **0.65**
madelon	41.92± 2.91	44.07± 2.65	**33.8**± **3.29**	44.17± 3.61	**36.97**± **4.72**	43.03± 1.64	44.82± 3.29	**39.45**± **4.23**	40.76± 4.67	42.72± 2.77	41.5± 3.43	41.1± 2.92
magic	**13.58**± **0.36**	**14.94**± **0.51**	15.73± 0.98	**13.52**± **0.44**	16.74± 1.45	26.13± 1.0	17.02± 1.25	19.13± 2.32	16.59± 1.28	20.33± 1.6	19.89± 3.3	18.62± 1.89
mfeat-factors	10.59± 1.36	7.24± 1.25	9.29± 1.38	7.59± 1.16	6.86± 1.0	7.67± 0.82	5.8± 0.58	6.88± 1.01	**5.12**± 0.74	5.44± **0.74**	**5.27**± **0.92**	**4.76**± **0.59**
mfeat-fourier	25.91± 1.63	24.91± 1.72	24.12± 2.23	24.16± 1.55	21.43± 1.53	25.93± 1.3	**21.14**± **1.43**	22.94± 1.53	**19.69**± **1.82**	21.88± 1.48	22.26± 2.09	**19.91**± **1.17**
mfeat-karhunen	17.93± 1.8	12.7± 1.67	13.61± 2.03	14.5± 1.82	8.71± 1.29	7.71± 0.82	5.43± 0.74	7.22± 1.04	**5.16**± **0.84**	**5.43**± **0.67**	5.47± 0.95	**4.4**± **0.83**
mfeat-pixel	21.6± 1.98	10.74± 1.52	15.78± 2.38	18.19± 2.42	16.04± 2.57	8.42± 0.59	**5.61**± **0.79**	7.92± 0.71	**5.44**± **0.76**	5.76± 0.58	6.52± 1.43	**5.36**± **0.64**
mfeat-zernike	29.92± 2.79	27.82± 1.87	27.81± 1.47	29.49± 1.8	26.6± 1.7	27.45± 1.25	22.95± 1.76	25.54± 1.66	23.18± 1.89	**23.08**± **1.66**	**22.96**± **1.58**	**22.18**± **1.4**
mushroom	0.11± 0.08	0.04± 0.07	0.09± 0.08	**0.01**± **0.03**	0.01± **0.03**	3.49± 2.38	0.04± 0.07	0.01± 0.04	0.01± 0.04	0.12± 0.14	**0.0**± **0.0**	0.02± 0.08
optdigits	6.41± 0.94	3.84± 0.51	4.87± 0.74	4.37± 0.53	2.48± 0.32	8.99± 0.64	**2.09**± **0.32**	3.19± 0.64	**2.02**± **0.35**	2.31± 0.3	2.25± 0.24	**1.97**± **0.29**
page	3.54± 0.46	3.87± 0.44	3.69± 0.33	**3.07**± **0.26**	**3.23**± **0.32**	12.24± 8.35	4.38± 0.32	3.54± 0.45	3.59± 0.4	4.91± 0.73	**3.3**± **0.76**	3.5± 0.45
pendigits	3.9± 0.48	1.84± 0.28	2.49± 0.38	2.77± 0.24	1.36± 0.24	16.66± 1.65	0.8± 0.19	1.78± 0.33	**0.93**± **0.18**	**1.08**± **0.23**	1.14± 0.22	**0.97**± **0.25**
satImg	12.25± 0.59	11.49± 0.65	11.58± 0.54	10.76± 0.48	**9.35**± **0.46**	20.12± 0.39	10.45± 0.45	10.41± 0.75	**9.72**± **0.57**	10.61± 0.48	10.34± 0.43	**10.0**± **0.51**
segment	5.25± 1.01	**4.23**± **0.92**	5.37± 1.41	**3.43**± **0.69**	4.41± 1.36	23.84± 3.76	5.48± 1.53	4.85± 1.56	4.38± 1.65	6.62± 2.05	4.28± 1.06	**3.89**± **1.14**
spambase	7.18± 0.48	6.48± 0.7	6.82± 0.64	**5.68**± **0.63**	**6.12**± **0.54**	27.67± 3.37	8.01± 0.66	6.55± 0.65	6.31± 0.64	8.96± 1.2	6.23± 0.48	**6.17**± **0.74**
splice	7.91± 0.83	6.83± 0.8	7.86± 1.49	6.82± 1.03	5.95± 1.02	6.26± 0.86	11.31± 1.18	**5.75**± **1.35**	**5.85**± **0.76**	7.52± 1.38	6.56± 1.26	**5.42**± **0.88**
waveform	20.04± 1.11	19.37± 1.2	19.34± 1.15	19.07± 1.15	**18.37**± **1.1**	22.66± 1.33	**18.82**± **1.1**	20.79± 1.72	**18.94**± **1.39**	21.0± 1.49	22.07± 1.61	19.39± 1.1
win/tie/loss	15/2/1	13/4/1	14/2/2	12/3/3	9/4/5	18/0/0	11/5/2	13/5/0	5/12/1	17/1/0	11/7/0	-

6 represents the correctness of that ensemble member on the corresponding testing data instance. A black line represents a correct prediction, and a white line represents an incorrect prediction. In this particular trial, the final ensemble converged on nine C4.5 classifiers, four Naive Bayes classifiers, and five kNN classifiers. Since the testing instances are unordered, the rows of Figure 6 were swapped so that incorrect predictions for each classifier type could be clustered together.

It is shown in the resulting figure that the three classifier types excel on different subsets of the testing data, and thus complement each other when predicting. For example, most C4.5 and Naive Bayes classifiers made incorrect predictions on testing instances between 700 and 750 in the figure, but the kNN classifiers vote unanimously correctly on these instances, thus ensuring correct overall majority votes for the data instances. More specifically, although there are only five correct votes from the kNN classifiers, since there are 26 possible output labels for the "isolet" data set, these data instances have a high probability of being correctly classified by the ensemble because it is unlikely that there will be more than five C4.5 and Naive Bayes classifiers voting together on any of the other 25 incorrect output labels. The kNN classifiers complement the ensemble in that without their existence, data instances, such as testing instances between 700 and 750, would be wrongly classified by the ensemble. A similar situation holds for the testing instances between 1000 and 1050 in Figure 6 for the Naive Bayes classifiers. Most C4.5 and kNN classifiers make wrong predictions for testing instances in this region, but the four Naive Bayes classifier vote unanimously correctly on these instances. For testing instances betwenn 800 and 840, most correct predictions are contributed by the C4.5 classifier type. It is also noticeable that C4.5 classifiers are more diverse amongst themselves, because instances of the C4.5 classifier type disagree more with each other than the other two types, which contributes to the fact that there are more C4.5 classifiers in the ensemble.

A hypothesis is derived from the observations above that heterogeneous ensembles perform well because different classifier types complement each other: errors made by one classifier type are masked by instances of the classifier type best suited to predict that data point. If the hypothesis is true, then there should be a positive correlation between how much the classifier types complement each other and how well heterogeneous ensembles perform. To test this hypothesis, a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

notion of complementarity is introduced to gauge the degree to which classifier types complement each other. The higher the complementarity, the more the classifier types complement each other. The improvement (Equation 1) introduced in Section 3 is employed to represent how well heterogeneous ensembles perform.

Complementarity is defined as

$$\text{c} = \min(\text{rC}, \text{cC}), \tag{2}$$

which is the minimum value of the row complementarity and the column complementarity of the purity matrix. The purity matrix is defined as a $K \times N$ matrix, where each row represents each data instance in the testing set, and each column represents each classifier type. Therefore K is equal to the number of data instances in the testing set, and N is equal to the number of classifier types. Given a trained ensemble of classifiers and their predictions on the testing set, each element of the purity matrix is computed as

$$\text{PM}(i,j) = \frac{\#\text{correct predictions of classifier type } j \text{ on data } i}{\#\text{classifiers of type } j},$$

which is then the ratio of how many instances of the ith classifier type predicted the correct label of the jth testing instance.

The row complementarity of the purity matrix is then defined as

$$\text{rC} = \frac{1}{K}\sum_{i=1}^{K}(\max(\text{PM}(i,:)) - \text{mean}_{-max}(\text{PM}(i,:))),$$

which is the average of the difference between the highest value and the mean value of the remaining elements of each row. PM(i, :) is the ith row vector of PM, max(PM(i, :)) is the maximum value of the ith row, and $\text{mean}_{-max}(\text{PM}(i,:))$ represents

$$\frac{\sum_{j=1}^{N}\text{PM}(i,j) - \max(\text{PM}(i,:))}{N-1}.$$

The column complementarity of the purity matrix is defined similarly as

$$\text{cC} = \frac{1}{N}\sum_{j=1}^{N}(\max(\text{PM}(:,j), \frac{1}{N}) - \text{mean}(\text{PM}(:,j), -\frac{2}{N})),$$

which is the mean of the difference between the average of the largest $\frac{1}{N}$th of the elements and the mean of the remaining elements of each column. PM(:, j) is the jth column vector of PM, $\max(\text{PM}(:,j), \frac{1}{N})$ is the mean of the largest $\frac{1}{N}$th of the elements of PM(:, j), and $\text{mean}(\text{PM}(:,j), -\frac{2}{N})$ represents

$$\frac{\sum_{i=1}^{K}\text{PM}(i,j) - \frac{K}{N}\max(\text{PM}(:,j), \frac{1}{N})}{\frac{(N-1)K}{N}}.$$

Given both the complementarity (Equation 2) and the improvement (Equation 1) measures, to test our hypothesis that they should be positively correlated, for each independent trial on each data set the improvement and the complementarity were plotted against one another as shown in Figure 7. Figure 7(a) reports the results for all data sets, and Figure 7(b) reports the results for data sets with more than four output labels. For each figure, the result of the linear regression is also shown in solid lines. The dotted line shows when the improvement is equal to one.

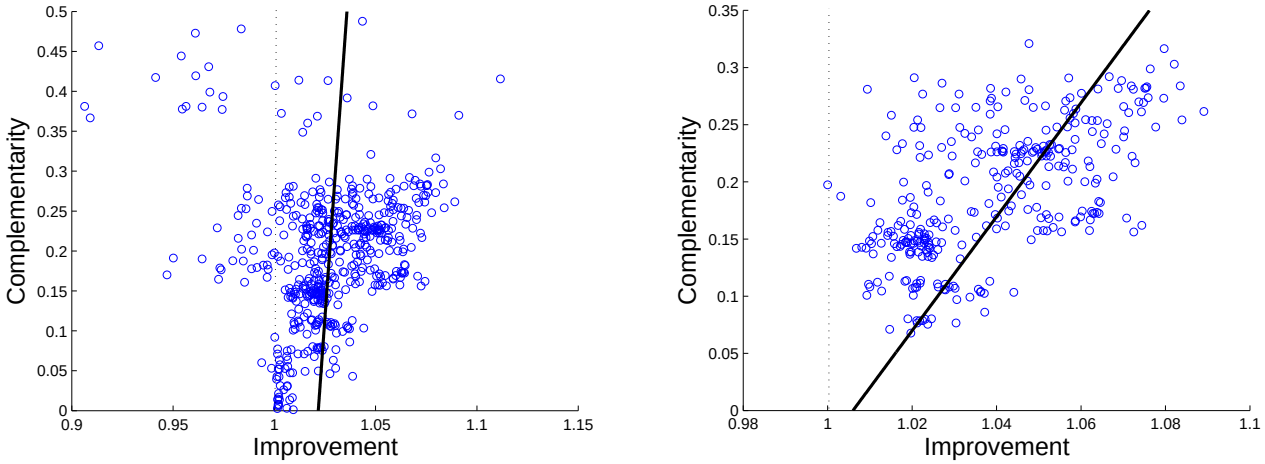
With a correlation coefficient of 0.0884, there is no clear linear correlation between improvement and complementarity across all data sets (Figure 7(a)). Figure 7(b) however show a strong positive linear correlations with correlation coefficients of 0.5923. Two conclusions can be drawn from Figure 7: 1) there is a positive linear correlation between improvements and complementarity when there are more than two class labels in a data set; and 2) the positive linear correlation grow stronger as the minimum number of output labels increases from two to three. This result is consistent with our analysis in Section 2.4. The two observations support our hypothesis that the AHE outperforms the other ensemble methods tested when different classifier types are better suited to explain different subsets of the data set, especially when the data set has more than three output labels. Thus the second question —- when are heterogeneous ensembles expected to outperform homogeneous ensembles? — is also answered in that heterogeneous ensembles are expected to work better for multi-class problems, and the third question — why do heterogeneous ensembles work better? — is answered in that heterogeneous ensembles perform well when the classifier types involved complement each other.

5 Conclusions and Future Work

This paper addresses four fundamental questions about heterogeneous ensembles: 1) heterogeneous ensembles are needed because appropriately constructed heterogeneous ensembles outperform homogeneous ensembles; 2) heterogeneous ensembles are expected to outperform homogeneous ensembles when the target task has more than two class labels; 3) heterogeneous ensembles outperform homogeneous ensembles because the classifier types complement each other; and 4) AHE is introduced as a framework that automatically adapts heterogeneous ensembles towards appropriate combinations of classifier types.

In this paper, Stably-sized AHE was introduced as a variant of AHE that adapts the combinations of classifier types in stably-sized ensembles using active learning. Experimental results show that Stably-sized AHE with C4.5, Naive Bayes and kNN classifiers outperforms all other variants of Stably-sized AHE with a subset of these classifier types, and bagging, boosting and the random subspace method with random sampling. Variably-sized AHE was introduced as another variant of the AHE framework that adapts not only the combinations of classifier types, but also the sizes of the ensembles. Variably-sized AHE with C4.5, Naive Bayes and kNN classifiers is shown to outperform the best variant of Stably-sized AHE, any other variant of Variably-sized AHE with a subset of these classifier types, and bagging, boosting and the random subspace method with random sampling. These experimental results show that, with our settings, heterogeneous ensembles outperform homogeneous ensembles, and the introduced AHE framework effectively finds appropriate combinations of classifier types. Through observation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering



(a) "All data sets" (b) "Data sets that have more than 4 output labels"

Fig. 7. Improvements and complementarities of individual runs. Each point represents the complementarity and improvement of one individual run. The solid black lines represent the linear regression models. The correlation coefficients are 0.0884, and 0.5923, respectively.

on one trial of AHE_CNK_Var on one example data set, a hypothesis was formulated that heterogeneous ensembles outperform homogeneous ensembles because different classifier types complement each other. Measures of improvement and complementarity were introduced to gauge how much heterogeneous ensembles outperform other homogeneous methods and how much different classifier types complement each other. Experimental analysis showed that when there are more than three class labels in data sets, there is a positive linear correlation between improvement and complementarity, thus supporting the hypothesis. In this paper heterogeneous ensembles were tested with small ensembles. However our conclusion in this paper is dependent on the fact that the inherent different inherent biases of classifier types makes them complement each other, which is independent of the ensemble sizes.

One interesting aspect to investigate is how AHE would perform with increasing number of classifier types. Table 10 shows some preliminary results of the comparison between AHE with three classifier types (AHE_CNK_Var) and AHE with four classifier types (AHE_CNKL_Var), where the additional classifier was Logistic Regression. Each cell in Table 10 reports the mean and standard deviation of error percentages after training finished over 30 independent runs. Both AHE_CNK_Var and AHE_CNKL_Var start with equal numbers of classifiers for the involved classifier types. Note that for this set of experiments the three tested data sets were re-partitioned into the training, validation and testing set randomly using the same 70%, 15%, 15% setting. All other experimental settings were the same. For each data set, the cell with lower average error percentage is bolded. It is shown that with the additional Logistic Regression classifier, AHE reaches significantly lower error percentages with smaller number of training instances. But as the number of classifier types increase, it is expected that the increase of accuracies decrease. Because it is harder to find a new classifier type that complements a large number of existing classifier types. In future work, we would like to perform extensive experiments to investigate this speculation.

Although the AHE was tested using active learning, in future work, we would like to test the AHE using passive learning. To adapt a heterogeneous ensemble towards better combinations of classifier types, a crucial question is how to effectively evaluate ensemble members' importance to the ensemble. AHE introduced in this paper employed trial-and-error as the way to find appropriate combinations of classifier types; we would like to explore more effective adaptation methods in future.

REFERENCES

[1] 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, pp. 327–336.

[2] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of Research and Development in Information Retrieval*, 1994, pp. 3–12.

[3] A. Culotta and A. McCallum, "Reducing labeling effort for structured prediction tasks," in *Proceedings of the National Conference on Artificial Intelligence(AAAI)*, 2005, pp. 746–751.

[4] D. D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," in *Proceedings of the International Conference on Machine Learning(ICML)*, 1994, pp. 148–156.

[5] M. Lindenbaum, S. Markovitch, and D. Rusakov, "Selective sampling for nearest neighbor classifiers," *Machine Learning*, vol. 54(2), pp. 125–152, 2004.

[6] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[7] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP)*, 2008, pp. 1069–1078.

[8] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proceedings of the International Conference on Machine Learning(ICML)*, 2004, pp. 79–86.

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

[10] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24(2), pp. 123–140, 1996.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2014.2304474, IEEE Transactions on Knowledge and Data Engineering

TABLE 10
Comparison between AHE_CNK_Var and AHE_CNKL_Var (L represents the additional Logistic Regression classifier type). The last row summarizes the win/tie/loss situation of AHE_CNKL_Var compared with AHE_CNKL_Var according to pair-wise t-tests at 95% significance level.

#labels	33%		67%		100%	
Data Sets	AHE_CNK_Var	AHE_CNKL_Var	AHE_CNK_Var	AHE_CNKL_Var	AHE_CNK_Var	AHE_CNKL_Var
mfeat-karhunen	12.89±1.5	12.28±1.42	7.5±1.33	7.55±1.37	5.24±1.29	5.62±1.17
page	4.35±0.79	**3.25±0.41**	3.99±0.6	**3.25±0.41**	3.17±0.6	2.93±0.45
segment	10.2±2.9	**7.37±1.8**	7.09±2.03	**5.56±1.61**	5.31±1.36	4.73±0.87
win/tie/loss	2-1-0	-	2-1-0	-	0-3-0	-

[11] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55(1), pp. 119–139, 1997.

[12] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20(8), pp. 832–844, 1998.

[13] L. I. Kuncheva and J. J. Rodriguez, "Classifier ensembles with a random linear oracle," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19(4), pp. 500 – 508, 2007.

[14] H. Chen and X. Yao, "Multiobjective neural network ensembles based on regularized negative correlation learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22(12), pp. 1738 – 1751, 2010.

[15] L. Xu, B. Li, and E. Chen, "Ensemble pruning via constrained eigen-optimization," in *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM)*, 2012, pp. 715–724.

[16] N. Abe and H. Mamitsuka, "Query learning strategies using boosting and bagging," in *Proceedings of the International Conference on Machine Learning(ICML)*, 1998, pp. 1–9.

[17] P. Melville and R. Mooney, "Diverse ensembles for active learning," in *Proceedings of the International Conference on Machine Learning(ICML)*, 2004, pp. 584–591.

[18] P. Melville and R. Mooney, "Constructing diverse classifier ensembles using artificial training examples," in *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*, 2003, pp. 505–510.

[19] T. G. Dietterich, "Ensemble methods in machine learning," in *Proceedings of the 1st International Workshop on Multiple Classifier Systems*, 2000, pp. 1–15.

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

[21] R. Caruana, A. Munson, and A. Niculescu-Mizil, "Getting the most out of ensemble selection," in *Proceedings of the International Conference on Data Mining(ICDM)*, 2006, pp. 828–833.

[22] D. Angluin, "Queries and concept learning," *Machine Learning*, vol. 2, pp. 319–342, 1988.

[23] D. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *Journal of Artificial Intelligence Research*, vol. 4, pp. 129–145, 1996.

[24] J. Du and C. X. Ling, "Asking generalized queries to domain experts to improve learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22(6), pp. 812–825, 2010.

[25] D. Cohn, L. Atlas, R. Ladner, M. El-Sharkawi, R. M. II, M. Aggoune, and D. Park, "Training connectionist networks with queries and selective sampling," in *Advances in Neural Information Processing Systems (NIPS)*, 1990, pp. 566–573.

[26] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15(2), pp. 201–221, 1994.

[27] G. Túr, D. Hakkani-Túr, and R. E. Schapire, "Combining active and semi-supervised learning for spoken language understanding," *Speech Communication*, vol. 45(2), pp. 171–186, 2005.

[28] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Batch mode active learning with applications to text categorization and image retrieval," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21(9), pp. 178–191, 2009.

[29] B. Settles, "Active learning literature survey," *Computer Sciences Technical Report 1648, University of Wisconsin-Madison*, 2009.

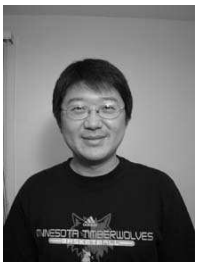
[30] A. McCallum and K. Nigam, "Employing em in pool-based active learning for text classification," in *Proceedings of the International Conference on Machine Learning (ICML)*, 1998, p. 359367.

[31] I. Dagan and S. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proceedings of the International Conference on Machine Learning(ICML)*, 1995, pp. 150–157.

[32] A. McCallum and K. Nigam, "Employing em in pool-based active learning for text classification," in *Proceedings of the International Conference on Machine Learning (ICML)*, 1998, p. 359367.

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

[34] A. Asuncion and D. J. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html



Zhenyu Lu Zhenyu Lu obtained his Ph.D. degree in Computer Science from the University of Vermont. His main research interests are active learning and ensemble learning. He has published in various data mining forums, including ACM SIGKDD, IEEE ICDM and SIAM SDM. He is currently working as a data scientist at Adometry Inc. with focuses on online and offline advertisement analysis.

Xindong Wu Xindong Wu is a Professor of Computer Science at the University of Vermont (USA), a Yangtze River Scholar in the School of Computer Science and Information Engineering at the Hefei University of Technology (China), and a Fellow of the IEEE. He holds a PhD in Artificial Intelligence from the University of Edinburgh, Britain. His research interests include data mining, knowledge-based systems, and Web information exploration. He has published over 200 refereed papers in these areas in various journals and conferences, including IEEE TKDE, TPAMI, ACM TOIS, DMKD, KAIS, IJCAI, AAAI, ICML, KDD, ICDM, and WWW, as well as 25 books and conference proceedings. His research has been supported by the U.S. National Science Foundation (NSF), the U.S. Department of Defense (DOD), the National Natural Science Foundation of China (NSFC), and the Chinese Academy of Sciences, as well as industrial companies including Microsoft Research, U.S. West Advanced Technologies and Empact Solutions.

Josh Bongard Josh Bongard obtained his Bachelors degree in Computer Science from McMaster University,Canada; his Masters degree from the University of Sussex, United Kingdom; his PhD from the University of Zurich, Switzerland; and served as a postdoctoral associate at Cornell University. He was named a Microsoft New Faculty Fellow, one of the top 35 innovators under the age of 35 by MIT's Technology Review Magazine, and is a National Science Foundation CAREER award recipient. He is currently an associate professor at the University of Vermont in the Computer Science Department.