Cocktail Ensemble for Regression

Yang Yu[1] Zhi-Hua Zhou[1] Kai Ming Ting[2]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
emails: {yuy,zhouzh}@lamda.nju.edu.cn
[2]Gippsland School of Information Technology, Monash University, Australia
email: kaiming.ting@infotech.monash.edu.au

Abstract

This paper is motivated to improve the performance of individual ensembles using a hybrid mechanism in the regression setting. Based on an error-ambiguity decomposition, we formally analyze the optimal linear combination of two base ensembles, which is then extended to multiple individual ensembles via pairwise combinations. The Cocktail ensemble approach is proposed based on this analysis. Experiments over a broad range of data sets show that the proposed approach outperforms the individual ensembles, two other methods of ensemble combination, and two state-of-the-art regression approaches.

1. Introduction

Ensemble learning uses multiple individual learners to solve a problem. Since it can significantly improve the generalization ability of a single learner, it has attracted a lot of attention, and many ensemble approaches have been developed. According to the styles of training individual learners, current ensemble approaches can be roughly categorized into two classes: (i) approaches where individual learners are trained in parallel, and (ii) approaches where individual learners must be trained sequentially. Representative parallel ensemble approaches include Bagging [4] and Random Forests [5]; and examples of sequential ensemble approaches include AdaBoost [8] and Stochastic Gradient Boosting [9]. However, despite the emergence of many ensemble approaches, it has been shown that none is significantly superior to another over a range of data sets [10].

We are motivated to further improve the performance of a single ensemble by combining more than one individual ensemble. The simplest approach in the regression setting is to average the outputs of its base ensembles. However, is there a better combination approach? Does the hybrid approach perform better than its base ensembles?

This paper investigates these issues. Through analysis of an error-ambiguity decomposition on linear combination of individual ensembles, we derive the optimal combination for two individual ensembles. Moreover, we find this optimal combination can be estimated effectively and efficiently from data, by estimating the performance of individual ensembles it combines. This finding leads to a simple yet effective solution for combining multiple ensembles based on the optimal pairwise combination, while acknowledging that obtaining the global optimal is intractable. Based on the analysis, we propose the *Cocktail ensemble* approach to combine different ensembles. Empirical evaluation over a broad range of data sets shows that the proposed approach is superior to simple averaging, selecting the best ensemble, its individual ensembles, Stochastic Gradient Boosting [9] and Iterated Bagging [6].

2. Related Work

MultiBoosting [18, 19] combines Bagging/Wagging [1] with Boosting, by taking Boosting as individual learner for Bagging/Wagging, in order to reduce both variance and bias at the same time. Stochastic Gradient Boosting [9] uses a bootstrap sample of the training set to train a regressor in each iteration, which can also be viewed as a hybrid of Boosting and Bagging [9]. Iterated Bagging [6] is motivated by iteratively reducing the bias of Bagging, which can be regarded as a sequential ensemble approach taking Bagging as the base learner.

In contrast to the above-mentioned approaches, which combine a parallel ensemble approach and a sequential ensemble approach, our approach combines any kinds of individual ensembles, and it has a strong theoretical basis.

E-GASEN [22] also combines ensembles. It trains multiple ensembles using the same ensemble approach GASEN [23] and the same base learning algorithm, and then uses simple averaging to combine these ensembles. In contrast, our work attempts to combine individual ensembles generated by different ensemble approaches and different base learning algorithms.

Our work uses a weighted averaging approach to combine individual ensembles. Various weighted averaging combination approaches have been employed in the literatures, e.g., [14, 12, 16, 23]. All of these methods set weights solely based on empirical estimation on training set, or by 10-fold cross validation or some other estimation methods that are computationally expensive, while we show in this paper that the solution of the optimal weights for combining two individual ensembles can be effectively estimated from data with a very small computational cost.

In [13], *error-ambiguity* decomposition is proposed to explain and exploit performance of ensembles. In this paper, the *error-ambiguity* decomposition is our tool to find the optimal combination of two individual ensembles.

3. Cocktail Ensemble

Given a training set D consisted of N examples, i.e., $D = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$, where $\boldsymbol{x}_i$ is an instance and $\boldsymbol{y}_i$ is a single variant regression. The task of regression is to learn a function $f : \boldsymbol{X} \rightarrow \mathcal{R}$ from D that approximates an underlying function f^0 by optimizing certain criterion, e.g. minimization of the mean squared error.

Combining Two Ensembles.
Denote two ensembles f_1 and f_2, which aggregate their individual learners in sets $\{f_{1,i}\}_{i=1}^{S_1}$ and $\{f_{2,j}\}_{j=1}^{S_2}$ with size S_1 and S_2, respectively. The functions of the ensembles are:

$$f_1(\boldsymbol{x}) = \mathbb{E}_i[f_{1,i}(\boldsymbol{x})] \ , \ f_2(\boldsymbol{x}) = \mathbb{E}_j[f_{2,j}(\boldsymbol{x})]$$

where the expectation can be implemented, taking averaging as an example, as $\mathbb{E}_i[f_{1,i}(\boldsymbol{x})] = \sum_i f_{1,i}(\boldsymbol{x})/S_1$ and $\mathbb{E}_j[f_{2,j}(\boldsymbol{x})] = \sum_j f_{2,j}(\boldsymbol{x})/S_2$. We explores the linear combination, i.e. the weighted averaging of ensembles f_1 and f_2, since linear combination is a more generic combination than simple averaging. Through a linear combination of ensembles f_1 and f_2, a new ensemble is formed:

$$f^c(\boldsymbol{x}) = pf_1(\boldsymbol{x}) + (1-p)f_2(\boldsymbol{x}) \quad \text{w.r.t.} \quad p \in [0,1]$$

where p is the weight for f_1, and $1-p$ is the weight for f_2. Here the restriction of p is to ensure that the generalization error of f^c is upper bounded by the worse of f_1 and f_2. By this restriction, f^c can be viewed as an interpolation between f_1 and f_2, which may have lower risk of making errors than an extrapolation.

Following the *error-ambiguity* decomposition [13], given E_1 and E_2 as generalization errors of f_1 and f_2, respectively, we derive[1] that the optimal weight of f_1 is:

$$p^* = \frac{E_2 - E_1}{2\Delta} + 0.5 \tag{1}$$

[1]Derivation details will be presented in a longer version of the paper.

where, we define

$$\Delta = \mathbb{E}_{\boldsymbol{x}}[(f_1(\boldsymbol{x}) - f_2(\boldsymbol{x}))^2]$$

as the *squared output difference* of the two ensembles. The optimal weight leads to the minimum generalization error of f^c:

$$E^{c*} = \frac{E_1 + E_2}{2} - \frac{\Delta}{4} - \frac{(E_1 - E_2)^2}{4\Delta} \tag{2}$$

where E_1, E_2 and Δ can be estimated from data efficiently and effectively (to be shown in Section 4).

Combining Multiple Ensembles
Optimal combination of multiple ensembles is intractable in general [23]. Since we have proved that the optimal combination of two ensembles is easy to solve, we propose a simple approach to combine multiple ensembles based on the optimal pair-wise combination, as follows.

Let $\mathcal{F} = \{f_i\}_{i=1}^N$ be the set of N base ensembles. Select the first base ensemble as the one with lowest error,

$$f_1^c = \arg\min_{f \in \mathcal{F}} E(f) \ ,$$

then each subsequent selected base ensemble is the one which reduces the combined estimated error the most,

$$f_2^c = p_2 f_1^c + (1-p_2) \arg\min_{\substack{f \in \mathcal{F} \\ p \in [0,1]}} E(pf_1^c + (1-p)f) \ ,$$

$$\ldots$$

$$f_N^c = p_N f_{N-1}^c + (1-p_N) \arg\min_{\substack{f \in \mathcal{F} \\ p \in [0,1]}} E(pf_{N-1}^c + (1-p)f) \ .$$

where $E(f)$ is the estimated error of f, and weights $p_2, \ldots, p_N$ are determined by Eq.1. f_N^c is the final combination of the given base ensembles. Note that it is possible that a base ensemble is selected more than once.

Cocktail Ensemble
The result of the above analysis forms the basis of the Cocktail ensemble approach. The pseudo-code is provided in Table 1. The accuracy of the error estimation method used

Table 1. Cocktail Ensemble

Cocktail($\mathcal{F}$)
Input: $\mathcal{F}$ = a set of N trained base ensembles
Process:
  f_1^c = the ensemble in $\mathcal{F}$ with the smallest estimated error
  $e_{\min} = +\infty$
  for $i = 2, \ldots, N$
    f_i = **null**
    for each $f \in \mathcal{F}$
      e = estimated error of combining f and f_{i-1}^c by Eq.2
      if $e < e_{\min}$ **then** let $f_i = f$ and $e_{\min} = e$
    end for
    if f_i is **null then** $f_N^c = f_{i-1}^c$ and **break**
    $f_i^c = p_i f_{i-1}^c + (1-p_i) f_i$, where p_i is obtained by Eq.1
  end for
  return f_N^c

Table 2. Normalized mean squared errors of Cocktail, PickBest, Avg, RF, LR and NN.

Dataset	Cocktail_3	Avg_3	PickBest_3	Cocktail_2	Avg_2	PickBest_2	RF	LR	NN
abalone	**.610±.00**	.673±.00	.619±.00	**.610±.00**	.613±.00	.619±.00	**.619±.00**	.662±.00	1.00±.00
analcat.g	.943±.02	**.927±.02**	1.00±.04	.945±.02	**.931±.02**	.965±.04	.951±.03	**.950±.02**	.951±.01
analcat.w	.707±.03	.786±.02	**.701±.03**	.707±.03	.745±.02	**.701±.03**	**.701±.03**	.856±.02	1.00±.01
autoHorse	**.164±.01**	.333±.01	.196±.02	**.164±.01**	.203±.01	.196±.02	.503±.02	**.196±.02**	1.00±.02
autoMpg	**.394±.01**	.486±.01	.413±.02	**.394±.01**	.396±.01	.413±.02	.477±.01	**.409±.01**	1.00±.01
autoPrice	.418±.02	.553±.01	**.414±.03**	.418±.02	.452±.02	**.414±.03**	**.414±.03**	.604±.03	1.00±.01
bank8FM	**.043±.00**	.162±.00	.046±.00	**.043±.00**	.047±.00	.046±.00	**.046±.00**	.074±.00	1.00±.00
bodyfat	**.044±.00**	.195±.00	**.044±.00**	**.044±.00**	.067±.00	**.044±.00**	.133±.00	**.044±.00**	1.00±.01
breastTumor	**.925±.01**	.930±.01	1.00±.03	**.932±.01**	.951±.01	.967±.02	.982±.01	**.962±.02**	.980±.00
chatfield.4	.436±.01	.517±.01	**.431±.01**	.434±.01	.447±.01	**.431±.01**	.548±.01	**.431±.01**	1.00±.02
cholesterol	.942±.01	**.931±.01**	1.00±.02	.943±.01	**.935±.01**	.982±.02	**.954±.01**	.958±.01	.965±.00
chscase.c6	**.977±.00**	.979±.00	**.977±.01**	**.976±.00**	.985±.00	1.00±.01	.989±.01	.992±.00	**.973±.00**
chscase.d	**.128±.00**	.573±.01	**.128±.00**	**.128±.00**	.970±.01	.942±.01	1.00±.01	.942±.01	**.128±.00**
chscase.w	**.000±.00**	.123±.01	**.000±.00**	**.000±.00**	.003±.00	**.000±.00**	.011±.00	**.000±.00**	1.00±.07
cleveland	**.858±.01**	.872±.01	.870±.01	**.860±.01**	.883±.01	.870±.01	1.00±.02	**.870±.01**	.937±.01
cloud	.217±.01	.438±.02	**.212±.01**	.217±.02	.303±.02	**.212±.01**	.553±.04	**.212±.01**	1.00±.04
cpu	**.136±.05**	.443±.05	**.136±.05**	**.136±.05**	.277±.06	**.136±.05**	.580±.08	**.136±.05**	1.00±.03
csb.ch15	.873±.01	.897±.00	**.872±.01**	.873±.01	.886±.00	**.872±.01**	**.872±.01**	.951±.00	1.00±.00
csb.ch21a	**.317±.01**	.429±.00	.319±.01	**.317±.01**	.341±.00	.319±.01	**.319±.01**	.448±.00	1.00±.01
delta.a	**.636±.00**	.710±.00	.638±.00	**.636±.00**	.654±.00	.638±.00	**.638±.00**	.730±.00	1.00±.01
diggle.t.a2	**.012±.00**	.131±.00	.014±.00	**.012±.00**	.013±.00	.014±.00	.023±.00	**.013±.00**	1.00±.01
fruitfly	.928±.01	.958±.01	**.927±.01**	**.928±.01**	.979±.02	.975±.02	1.00±.02	.971±.02	**.927±.01**
housing	**.244±.00**	.366±.00	.263±.01	**.244±.00**	.249±.00	.263±.01	**.263±.01**	.342±.01	1.00±.01
kin8nm	**.363±.00**	.564±.00	**.363±.00**	**.363±.00**	.470±.00	**.363±.00**	**.363±.00**	.643±.00	1.00±.01
lowbwt	.475±.01	.519±.01	**.470±.01**	.475±.01	.474±.01	**.470±.01**	**.470±.01**	.500±.01	1.00±.01
meta	**.963±.02**	**.963±.02**	.970±.02	.963±.02	**.961±.02**	**.961±.02**	**.961±.02**	.987±.02	1.00±.02
mu284	**.009±.00**	.125±.00	.011±.00	**.009±.00**	**.009±.00**	.011±.00	.013±.00	**.011±.00**	1.00±.03
no2	.841±.01	.873±.00	**.840±.01**	.841±.01	.850±.01	**.840±.01**	**.840±.01**	.920±.01	1.00±.00
pbc	**.855±.01**	.865±.01	.895±.02	.854±.01	**.846±.01**	.895±.02	.909±.01	**.878±.01**	1.00±.01
pharynx	**.766±.02**	.869±.02	**.766±.02**	**.766±.02**	.830±.02	**.766±.02**	**.766±.02**	.972±.04	1.00±.01
plasma.r	.931±.02	**.901±.01**	.983±.05	.926±.02	**.916±.01**	.980±.05	**.894±.01**	1.00±.02	.920±.00
pollen	.975±.00	.977±.00	**.974±.00**	**.974±.00**	.981±.00	.976±.00	1.00±.00	.976±.00	**.974±.00**
puma8NH	**.364±.00**	.549±.00	**.364±.00**	**.364±.00**	.455±.00	**.364±.00**	**.364±.00**	.683±.00	1.00±.01
pwLinear	**.348±.00**	.460±.00	.427±.02	.348±.00	**.345±.00**	.427±.02	**.410±.01**	.451±.01	1.00±.02
quake	**.962±.00**	.965±.00	.976±.01	**.963±.00**	.967±.00	.967±.00	**.967±.00**	1.00±.00	.974±.00
rmftsa.l	**.522±.01**	.595±.01	.562±.01	**.522±.01**	**.522±.01**	.562±.01	**.562±.01**	.611±.00	1.00±.02
sensory	**.743±.01**	.804±.01	.744±.01	**.743±.01**	.763±.01	.744±.01	**.744±.01**	.873±.01	1.00±.00
servo	**.223±.01**	.345±.01	.235±.02	**.223±.01**	**.223±.01**	.235±.02	**.225±.01**	.264±.01	1.00±.01
socmob	**.492±.02**	.590±.01	.594±.07	.492±.02	**.484±.01**	.594±.07	**.500±.01**	.651±.02	1.00±.01
space.ga	**.447±.00**	.593±.00	**.447±.00**	**.447±.00**	.511±.00	**.447±.00**	**.447±.00**	.668±.00	1.00±.00
stock	**.024±.00**	.188±.00	**.024±.00**	**.024±.00**	.064±.00	**.024±.00**	**.024±.00**	.176±.00	1.00±.00
strike	**.855±.01**	.864±.00	.868±.01	.855±.01	**.850±.01**	.868±.01	**.858±.01**	.860±.00	1.00±.00
tecator	**.054±.00**	.264±.01	.055±.00	**.054±.00**	.103±.00	.055±.00	.282±.01	**.055±.00**	1.00±.00
veteran	.850±.02	.877±.02	**.838±.02**	.851±.02	.861±.02	**.838±.02**	.926±.03	**.838±.02**	1.00±.01
visual.g	**.055±.00**	.186±.00	**.055±.00**	**.055±.00**	.072±.00	**.055±.00**	**.055±.00**	.134±.00	1.00±.02
water.treat	.250±.12	.666±.06	**.244±.11**	.250±.12	.534±.08	**.244±.11**	.977±.02	**.244±.11**	1.00±.03
wisconsin	.886±.02	**.881±.01**	1.00±.03	.893±.01	**.887±.02**	.968±.03	.938±.02	**.930±.02**	.941±.01
witmer.1987	**.008±.00**	.164±.00	**.008±.00**	**.008±.00**	.025±.00	**.008±.00**	.089±.01	**.008±.00**	1.00±.02

is important, which directly affects the performance of the Cocktail ensemble. We have employed the out-of-bag estimation method [21] in our evaluation for convenience (the reason will be clear in the next section). There is no reason that other equally effective methods cannot be used.

4. Empirical Study

Experimental Settings

To empirically evaluate the performance of the Cocktail ensemble, we use three types of base ensemble learning ap-

proaches, denoted by *RF*, *LR* and *NN*. *RF* is Random Forests [5], whose base learners are REPTrees [20] without post-pruning which are C4.5-like regression trees; *LR* is Bagging [4] of logistic regressors [11]; *NN* is Bagging of RBF neural networks [2]. Since RF, LR and NN all use bootstrap sampling, it is convenient to use the out-of-bag estimation [21] method for error estimation. Algorithm implementations are based on WEKA [20] with its default settings.

In addition to each of the three base ensembles, Cocktail is compared with *Avg* and *PickBest*. *Avg* is simple averaging, which uses the same base ensembles of Cocktail but aggregates their outputs with equal weights. *PickBest* chooses to use the best among the base ensembles according to their estimated generalization errors. The ensemble sizes of RF, LR and NN are set to 100, which makes the ensemble sizes of Cocktail, Avg and PickBest 200 or 300, depending on whether two or three base ensembles are used in the combination.

The empirical study uses 48 regression data sets from UCI Repository [3] and StatLib [17]. For each data set, we conduct a 10 times 10-fold cross-validation to estimate the performance of each approach. Then several statistical significance tests are employed. The first one is *Friedman test* [7], which is a non-parametric test based on *rank* [7] applied to comparing multiple learners on multiple data sets. The Friedman test is performed in conjunction with the *Bonferroni-Dunn test* [7] with significance level 0.05. We also conduct a t-*test* with significance level 0.05 to compare two approaches on each data set. The win/tie/loss counts for the t-test are recorded, where a win (or loss) is counted when Cocktail is significantly better (or worse) than the compared approach on a data set in the t-test; otherwise a tie is recorded. A *sign-test* with significance level 0.05 is then conducted on the win/tie/loss counts of two approaches to determine whether the null hypothesis, i.e. the two approaches have no difference, should be accepted or rejected.

Results

Table 2 presents the results, where A_3 denotes that the approach A takes RF, LR, and NN as base ensembles, while A_2 denotes that the approach A takes RF and LR as base ensembles. Normalized mean squared errors are reported. That is, we divide the mean squared error of a specific algorithm on a specific data set by the maximum mean squared error at the same row, and then reports the ratio. There are three groups of algorithms, separated by vertical lines in the table. Algorithms in different groups employ different number of base ensembles. For every group, the best performance on each data set is bolded.

Figure 1 shows the result of the Friedman test of these approaches. Cocktail is shown to be significantly better than all other approaches. Note that, when combining two ensembles, neither Avg_2 nor PickBest_2 is significantly better than both of their base ensembles.

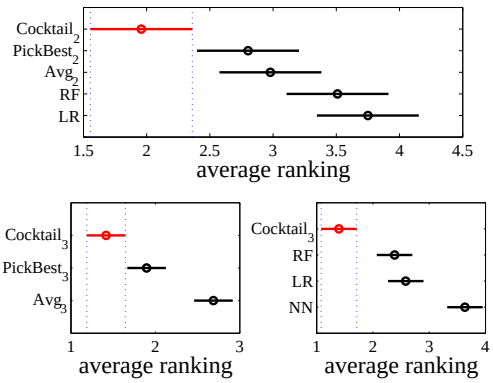

Figure 1. The result of the Freidman test for Cocktail, PickBest and Avg with base ensembles RF and LR over 48 data sets. The circles indicate the average rankings, and the bars indicate the critical values of the Bonferroni-Dunn test for a two-tailed test at significance level 0.05. Approaches having non-overlapped bars are significantly different.

Table 3. Win/Tie/Loss counts of t-test for Cocktail against PickBest, Avg, RF, LR and NN.

	Cocktail_2 against			
	PickBest_2	Avg_2	RF	LR
w/t/l	**25/17/6**	**34/4/10**	**31/14/3**	**36/7/5**

	Cocktail_3 against				
	PickBest_3	Avg_3	RF	LR	NN
w/t/l	**24/18/6**	**42/3/3**	**31/14/3**	**40/5/3**	**39/7/2**

Table 3 summarizes the results of t-tests of Cocktail against PickBest, Avg, RF, LR and NN. It reveals that Cocktail is the best, since it is significantly better than all other methods in the sign test on the win/tie/loss counts, where Cocktail has more than three times the number of wins than the number of losses, compared to any of the other methods.

It can be observed that NN is worse than RF and LR on many data sets, because we make no attempt to tune its many parameters but only use the default configuration in WEKA [20]. Interestingly, we find that adding a bad base ensemble has little negative impact to Cocktail.

To investigate how well Cocktail estimates weight p according to Eq. 1, we plot the performance of Cocktail_2 with $p \in \{0, 0.01, 0.02, \ldots, 1\}$ for each data set from the first fold of the cross-validation experiment. Figure 2 shows 12 of the data sets.[2] The abscissa of each plot indicates weight p; $p = 0$ in the error curve is the error for RF, and $p = 1$ is the error for LR. To provide a clear visualization, the axis has been normalized into the range $[0, 1]$. The estimated weight and the optimal weight are marked by two lines parallel to the axis, while the weight for PickBest is indicated

[2]Details of this experiment and the following experiments will be presented in a longer version of the paper.

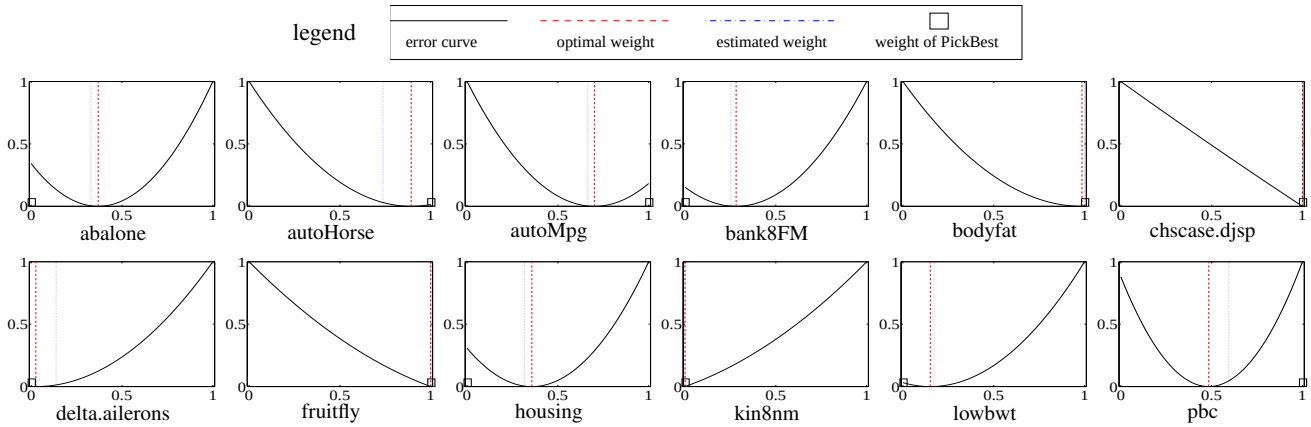

Figure 2. The optimal weight, the estimated weights by Cocktail and PickBest, and the performance of Cocktail in a single fold of 10-fold cross-validation.

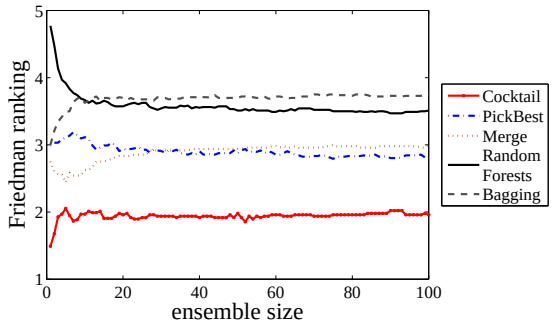

Figure 3. The influence of ensemble size on Friedman ranking

by a box at either 0 or 1 of the abscissa.

It can be observed that for the 12 data sets shown in Figure 2, the estimated weights are close to the optimal weights. Even in those cases in which the estimated weights might appear to have a big difference to the optimal weights, they are usually in the relatively flat region of the error curve. Thus, there is only a small real difference in terms of performance. It also discloses that the optimal weight on different data sets could appear at different p values.

To investigate the influence of the ensemble size on the performance of Cocktail$_2$, the predictive performance of ensembles with different sizes $(1, 2, \ldots, 100)$ are obtained. The Friedman rankings are depicted in Figure 3, where the Friedman rankings are obtained for each ensemble size. It can be observed that Cocktail$_2$ is consistently the best approach in term of the Friedman ranking.

Finally, in terms of efficiency, averaged over all data sets in the 10 times 10-fold cross-validation, Cocktail$_3$ only costs 1.2% more time than using Avg$_3$ which does not involve the error estimation.

Compare to Sequential Ensemble Approaches

We compare Cocktail$_2$ to two state-of-the-art sequential approaches for regression, i.e., Stochastic Gradient Boosting [9] and Iterated Bagging [6].

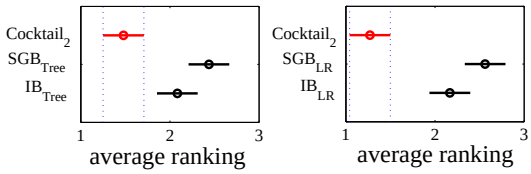

Figure 4. The result of the Freidman test for Cocktail, Stochastic Gradient Boosting (SGB) and Iterated Bagging (IB) with base regressors REPTree (Tree) and logistic regression (LR) over 48 data sets.

Table 4. Win/Tie/Loss counts of t-test for Cocktail against Stochastic Gradient Boosting (SGB) and Iterated Bagging (IB) with base regressors REPTree (Tree) and logistic regression (LR).

	Cocktail$_2$ against			
	SGB$_{\text{Tree}}$	IB$_{\text{Tree}}$	SGB$_{\text{LR}}$	IB$_{\text{LR}}$
w/t/l	**37/4/7**	**30/7/11**	**41/1/6**	**33/13/2**

Denote SGB_{Tree} and SGB_{LR} as the Stochastic Gradient Boosting approachs with base learners REPTree [20] and logistic regression [11], respectively. Denote IB_{Tree} and IB_{LR} as the Iterated Bagging approachs with the two base learners, respectively. The ensemble size is set to 200 for all these approaches, which is the same as that of Cocktail$_2$. For Stochastic Gradient Boosting, the shrinkage parameter is set to 0.005 for small data sets (having less than 500 instances) and 0.05 for large ones (having more than 500 instances), which is the same as in [9]; the fraction parameter is set to 0.5, which leads to the best performance according to [9]. For Iterated Bagging, the threshold multiplier is set to 1.1 and 50 base learners per stage, as did in [6].

Figure 4 shows the result of the Friedman test of these approaches. Cocktail is shown to be significantly better than the two sequential approaches.

Table 4 summarizes the results of t-tests of Cocktail$_2$ against Stochastic Gradient Boosting and Iterated Bagging. It also reveals that Cocktail is significantly better than both

the sequential approaches. Cocktail$_2$ has more than two times the number of wins than the number of losses, compared to any of the other methods.

Note that our experiment shows that Iterated Bagging is better, but not significantly better, than Stochastic Gradient Boosting, while the empirical study in [15] concludes that Iterated Bagging is significantly better than Stochastic Gradient Boosting. We think that the difference in results is because the shrinkage of Stochastic Gradient Boosting was set to 1 in [15], which causes overfitting, while we set the shrinkage as recommended in [9].

5. Conclusions

This paper shows that it is possible to form a hybrid ensemble that often outperforms its base ensembles and two other forms of combination: Selecting the best and simple averaging. We also show that the proposed approach outperforms two state-of-the-art sequential ensemble approaches, Stochastic Gradient Boosting and Iterated Bagging with either REPTree or logistic regression as base learners.

We have provided a theoretical basis through an analysis of *error-ambiguity* decomposition. It shows that there is a simple closed form solution to the optimal weight for combining two base ensembles. When combining more than two ensembles, our proposed approach provides a simple yet effective solution based on the optimal pairwise combination. We also show that the out-of-bag estimation method works reasonably well for Cocktail.

Though we have only conducted experiments using three kinds of base ensembles, we believe that the same result can also be expected when other base ensemble approaches are used. The analysis reveals that the only condition to gain better performance is to use sufficiently different individual ensembles with a good error estimation method.

Acknowledgments: Supported by NSFC (60325207, 60635030) and 863 Program (2007AA01Z169). Part of the research was conducted when K. M. Ting was visiting the LAMDA group at Nanjing University. We thank the anonymous reviewers for their helpful suggestions.

References

[1] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[3] C. Blake, E. Keogh, and C. J. Merz. UCI repository of machine learning databases. [http://www.ics.uci.edu/∼mlearn /MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA, 1998.

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

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] L. Breiman. Using iterated bagging to debias regressions. *Machine Learning*, 45(3):261–277, 2001.

[7] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

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

[9] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[10] L. O. Hall, K. W. Bowyer, R. E. Banfield, D. Bhadoria, W. P. Kegelmeyer, and S. Eschrich. Comparing pure parallel ensemble creation techniques against bagging. In *ICDM'03*, pages 533–536, Melbourne, FL.

[11] F. E. Harrell. *Regression Modeling Strategies with Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer, New York, NY, 2001.

[12] R. Jacob. Methods for combining experts' probability assessments. *Neural Computation*, 7(5):867–888, 1995.

[13] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In *NIPS 7*, pages 231–238. MIT Press, Cambridge, MA, 1995.

[14] M. Perrone and L. Cooper. When networks disagree: Ensemble methods for hybrid neural networks. In R. Mammone, editor, *Artificial Neural Networks for Speech and Vision*, pages 126–142. Chapman & Hall, London, 1993.

[15] Y. L. Suen, P. Melville, and R. J. Mooney. Combining bias and variance reduction techniques for regression trees. In *ECML'05*, pages 741–749, Porto, Portugal.

[16] K. M. Ting and I. H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289, 1999.

[17] P. Vlachos and M. Meyer. Statlib data set archive. [http:// lib.stat.cmu.edu], Department of Statistics, Carnegie Mellon University, Pittsburgh, PA, 1998.

[18] G. I. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40(2):159–196, 2000.

[19] G. I. Webb and Z. Zheng. Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques. *IEEE Trans. Knowledge and Data Engineering*, 16(8):980–991, 2004.

[20] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed.* Morgan Kaufmann, San Francisco, 2005.

[21] D. H. Wolpert and W. G. Macready. An efficient method to estimate bagging's generalization error. *Machine Learning*, 35(1):41–55, 1999.

[22] J.-X. Wu, Z.-H. Zhou, and Z.-Q. Chen. Ensemble of GA based selective neural network ensembles. In *ICONIP'01*, pages 1477–1482, Shanghai, China.

[23] Z.-H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, 137(1-2):239–263, 2002.