

Accuracy–diversity based pruning of classifier ensembles

Vasudha Bhatnagar · Manju Bhardwaj ·
Shivam Sharma · Sufyan Haroon

Received: 21 January 2013 / Accepted: 19 January 2014 / Published online: 27 February 2014
© Springer-Verlag Berlin Heidelberg 2014

Abstract Classification ensemble methods have recently drawn serious attention due to their ability to appreciably pull up prediction performance. Since smaller ensembles are preferred because of storage and efficiency reasons, ensemble pruning is an important step for construction of classifier ensembles. In this paper, we propose a heuristic method to obtain an optimal ensemble from a given pool of classifiers. The proposed accuracy–diversity based pruning algorithm takes into account the accuracy of individual classifiers as well as the pairwise diversity amongst these classifiers. The algorithm performs a systematic bottom-up search and conditionally grows sub-ensembles by adding diverse pairs of classifiers to the candidates with relatively higher accuracies. The ultimate aim is to deliver the smallest ensemble with highest achievable accuracy in the pool. The performance study on UCI datasets demonstrates that the proposed algorithm rarely misses the optimal ensemble, thus establishing confidence in the quality of heuristics employed by the algorithm.

Keywords Classifier ensemble · Ensemble pruning · Diversity · Accuracy · Heuristic · Brute force search · Optimal ensemble

1 Introduction

Classifier ensembles, also known as multiple classifier systems, have been recognized as better performers than their individual constituent classifiers [12, 22, 24, 27, 38, 42]. The main idea underlying classification ensembles is to combine a set of predictive models (henceforth called classifiers), each of which performs the same prediction task. The aim is to obtain a composite global model capable of more accurate and reliable decisions than the best model in the set. Dietterich has justified the superiority of ensembles over individual classifiers giving computational, statistical and representational reasons [12]. Several empirical studies have demonstrated the validity of the idea [2, 13, 14, 17, 34]. Recently, ensemble techniques have been successfully implemented for evolving/streaming datasets as well [9, 43].

An ensemble E is constructed by producing a pool of classifiers from the given training set using either same or different induction algorithms, and a pre-specified generation strategy. When an unseen instance is to be predicted, each classifier in the ensemble is asked for its prediction, and finally all predictions are combined using a combiner function. Variations in the “generation methods” and “combiner functions” give rise to a variety of ensemble methods [6, 22, 27, 38].

Research in the area of ensemble methods has focused on strategies to generate pools of classifiers [2, 3, 14] and design of combiner functions [22, 27]. During the last decade, ensemble pruning has attained importance in the research

V. Bhatnagar
Department of Computer Science, Delhi University,
Delhi 110007, India
e-mail: vbhatnagar@cs.du.ac.in

M. Bhardwaj (✉)
Department of Computer Science, Maitreyi College,
Chanakya Puri, New Delhi 110021, India
e-mail: mbhard@hotmail.com; mbhardwaj@maitreyi.du.ac.in

S. Sharma
Nagarro Software Pvt. Ltd., Gurgaon, India
e-mail: shivam.sharma@nagarro.com

S. Haroon
Adobe Systems India Private Limited, Noida (U.P.), India
e-mail: sharoon@adobe.com

related to multiple classifier systems, since smaller ensembles with comparative performance have lower overheads with respect to storage and prediction [16,21,29,30,32,33,35,36,39,41,44,45].

Accuracy and size of an ensemble are two important issues in ensemble technology. Theoretically, the accuracy of an ensemble of independent classifiers asymptotically approaches 1 as its size increases [27]. Practically speaking, this notion fails when classifiers with predictions identical to previous classifier members are added to the ensemble [7,32]. Further, Margineantu and Dietterich [30] have shown that the size of an ensemble can be substantially reduced if a slight loss of performance is acceptable. Studies have empirically verified that a systematically selected subset of classifiers in an ensemble often gives better performance than the complete set [29,32,33].

Ensemble pruning has also been studied as ensemble selection [3,21,45]/thinning [1]. The goal is to remove bulk forming classifiers, which have little or no contribution in improving the performance of the ensemble. This problem has been shown to be NP complete [39]. Several heuristic methods have been proposed to solve the problem [33,35,36] in addition to clustering approach [16], probabilistic approach [8], genetic programming [23,45] and semi-definite programming [44]. Tsoumakas et al. [41] give an elaborate account of these approaches in a comprehensive survey.

Diversity amongst classifiers (both pairwise and non-pairwise) is a well-studied aspect of multiple classifier systems [5,24,26]. Kuncheva proved that diverse classifiers lead to uncorrelated errors, which in turn improve ensemble accuracy [25]. Bagging [2], random space method [20] and Boosting [14] induce diversity implicitly as the members of ensembles are produced. Several other algorithms explicitly use diversity to prune the ensemble [25,30].

Though the concept of diversity in ensembles has been studied for over a decade, there is no single definition for it [26,40]. Further, there are evidences that diversity is not necessarily strongly correlated with ensemble accuracy [23]. Accuracy–diversity trade off has been well studied [26,40] and it is opined that accuracy need not be sacrificed for diversity in ensembles. However, recent works that study the theoretical aspects of diversity, and argue in favor of diversity for ensemble pruning have rekindled interest in diversity [11,23,28].

Ko et al. [23] have exploited accuracy and diversity together to prune an ensemble. They propose a compound function that takes into consideration both accuracies of individual classifiers and pairwise diversities among them. During the course of current study, we found that pairwise diversity and ensemble accuracy can be integrated in a *simplistic manner* to discover minimum-size, maximum-accuracy ensemble, i.e., an optimal ensemble.

In this paper, we propose a novel and *conceptually simple* method for ensemble pruning, which integrates the use of accuracy and diversity. The proposed accuracy–diversity pruning (ADP) algorithm takes as input a pool of classifiers ($\mathcal{E} = \{h_1 \dots h_T\}$) and aims to find a minimum-size, maximum-accuracy ensemble. Essentially, all the classifiers in the pool must have error rate <0.5 for the method to work well. ADP algorithm is independent of the method of ensemble generation, and uses majority voting with 0–1 loss function. *Simplicity* and *effectiveness* are attractive features of the algorithm. Other salient features of the proposed algorithm are:

1. It makes an explicit use of accuracy and diversity while growing the ensemble bottom-up. The search is purely data driven and *parameterless*, which is the main strength of the algorithm.
2. Instead of greedy hill climb, the method performs multi-pronged hill climb search while ordering the ensembles according to their accuracies, thus avoiding getting stuck in *local minima* (Sect. 3).
3. The algorithm *guarantees* that the resulting ensemble has accuracy greater than or equal to the best classifier in the pool (Sect. 3.2). To the best of our knowledge, this guarantee is not furnished by any existing ensemble selection method.
4. Unlike the published works on ensemble pruning, ADP algorithm strives to achieve *optimality*. Experimental evaluation in Sect. 4 demonstrates that ADP discovers optimal ensemble most of the times. Results of none of the previous algorithms have been assessed for optimality.

2 Related work

Margineantu and Dietterich [30] pointed out the drawbacks of including all the classifiers in the ensemble. There have been several research studies that aim to select classifiers from large pools to train small ensembles with high generalization accuracy [16,21,29,30,32,33,35,36,39,41,44,45]. An elaborate account of various approaches for ensemble pruning is presented in [41]. We present below the works that are closely related to the two strategies employed in the proposed algorithm. Firstly, we describe works that have used *Ordered aggregation* approach to prune ensemble, followed by those which have demonstrated the effectiveness of incorporating *diversity* in ensembles.

2.1 Ordered aggregation approach for ensemble pruning

Ordered aggregation of classifiers in an ensemble has been found to be an efficient approach for ensemble pruning [33].

This method ranks the classifiers based on some performance criteria to select either a pre-specified number of classifiers [1] or select all classifiers that satisfy a pre-specified threshold [29, 32]. This strategy avoids scanning the massive search space of $\mathcal{O}(2^T)$, where T is the number of classifiers under consideration. Rokach [37] proposed an ensemble pruning algorithm based on ranking method that takes into account agreement level among the members in addition to agreement level between each member's prediction and the real label. Recently, Guo and Boukir [18] proposed ordering-based ensemble pruning method which relies on measuring the margin-based information quantity of each base classifier in the ensemble. The algorithm uses a validation set V (called the pruning set in the paper) to calculate the margin of each instance in V . Subsequently, the margin-based criterion is computed for each classifier to order them and include the best in ensemble.

2.2 Diversity for ensemble pruning

Tang et al. [40] analyzed various diversity measures and concluded that seeking diversity in an ensemble could be viewed as an implicit way to maximize the minimum margin of the ensemble. Heuristic method based on diversity-based ensemble pruning proposed in [30] shows that 60–80 % ensemble members can be pruned out without substantial degradation in performance. Ensemble pruning methods that perform state space search using hill climbing based on diversity measures have been elaborated in [41]. Recently, diversity has been studied theoretically, and it has been shown that it is closely related to the hypothesis space complexity. It can be regarded as regularization factor on ensemble methods [28].

Ko et al. [23] examined correlation between ensemble accuracy and diversity, and established theoretically and empirically that the two are indeed correlated. They proposed compound measures based on accuracy of classifiers and different types of diversities among them, to empirically substantiate their hypothesis, which is somewhat contrary to earlier studies [26]. This study asserts that diversity *should* be taken into account along with accuracy of individual classifiers for the improved performance of pruned ensemble. Experimental results demonstrate that genetic algorithm (GA)-based ensemble selection using the compound measure as objective function yields more accurate ensembles than those obtained using the diversity measures alone.

The present proposal takes cue from the conclusion of work by Ko et al. [23] and integrates accuracy and pairwise diversity for ensemble pruning. We hypothesize that classifier accuracies and pairwise diversities can be combined in more *simplicistic and efficient manner* by embodying them in search process. This approach eliminates expensive computation of compound measures to be used as objective function during ensemble selection by GA. The proposed ADP

heuristically prunes the search space while systematically including accurate and diverse classifiers in the ensemble. *The search is guaranteed to yield an ensemble with accuracy not lower than the best classifier in the pool.* No such guarantee is attainable for an ensemble selected using evolutionary algorithms. Thus, *ADP algorithm is highly likely to discover optimal ensemble.* In case the optimal is missed, the discovered sub-optimal ensemble is empirically found to be close to the optimal ensemble in terms of both accuracy and size.

3 Accu-div heuristic

As mentioned earlier, ADP algorithm strives towards an optimal ensemble by integrating accuracy and diversity while pruning a pool of classifiers. Unlike greedy hill-climbing approach, which is likely to lead to *locally maximum* accuracy ensemble, the proposed algorithm does a multi-pronged hill climbing and examines multiple candidate solutions (sub-ensembles) for optimality in breadth-first fashion. We are hesitant to call it beam search with variable beam width, because of its purely data-driven nature. Ensembles in the proposed approach are grown by incrementally adding a pair of high diversity classifiers. Based on the performance, at each stage it is ascertained if an ensemble is worth growing further.

3.1 Search strategy

The brute force search necessitates examining $\overline{2^T - 1}$ subsets of \mathcal{E} to discover the optimal ensemble E . The search space can be modeled as a lattice with subset relation (\subset) as the ordering operator. Figure 1 shows the complete search space for a pool of five classifiers. The null ensemble is the infimum, and the complete pool is the supremum in the lattice. The optimal ensemble(s), i.e., minimum-size, maximum-accuracy ensemble, is (are) hidden in the lattice. Each node in the lattice is a potential solution (optimal ensemble) that needs to be examined. Thus in a pool of size T , the brute force search for optimal ensemble has computational complexity of $\mathcal{O}(2^T)$.

ADP algorithm heuristically reduces the search space by growing the ensembles incrementally and conditionally. Since the algorithm uses majority voting as combining function, ensembles are grown by adding a pair of classifiers at a time. Thus, the search is limited to examining nodes at odd-numbered levels in lattice.¹ The heuristic discipline of growing the ensemble based on accuracy and pairwise diversity, systematically discards the portions of the lattice that are not likely to contain the optimal ensemble. Though theoretical

¹ Please note that this does not reduce the complexity of the problem.

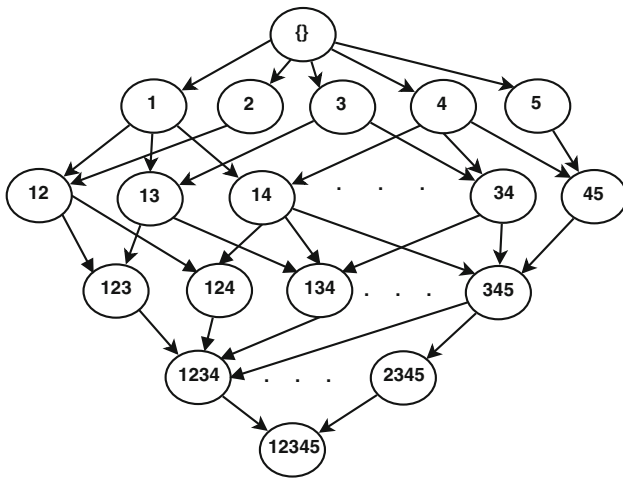


Fig. 1 Complete search space for a pool of five classifiers

complexity of the search is still $O(2^T)$, empirical evaluation shows that the search space reduction for the experimented datasets varies from 70 to 99 %, with minor average loss of accuracy (Table 2).

Given a pool of T classifiers, the algorithm performs a systematic bottom-up search starting with top 50 % accurate classifiers. Thus, the better half of the sub-ensembles of size 1 is retained and the rest are discarded. The most accurate classifier is the best accuracy sub-ensemble so far. Pairwise diversity of T classifiers is computed, and top 50 % diverse pairs are retained for growing sub-ensembles. At each level of the lattice, these pairs are incrementally added to each of the retained sub-ensembles and the grown sub-ensembles are examined for improved performance. If none of the grown sub-ensembles exhibits better performance than the best accuracy sub-ensembles so far, the algorithm stops and reports the latter as the optimal ensemble. Otherwise, on an optimistic note the algorithm proceeds to the next level by retaining top 50 % sub-ensembles as potential candidates for growing.

Apparently, due to 50 % reduction at level 1, portions of lattice below these classifiers are completely missed. However, a close examination of the algorithm reveals that the discarded classifiers get a second chance to get introduced in the search space when the existing sub-ensembles are grown by addition of diverse classifier pairs. But this takes place in a purely data-driven manner.

In order to understand this aspect better, we randomly selected ten datasets from Table 1. For each of the ten datasets, we first constructed a library of 101 classifiers using bagging and J48 algorithm from Weka [19]. Then we drew random samples of sizes 11, 21 and 31, on which we executed Steps 5–8 of ADP algorithm. In a pool of size $T (= 2k + 1)$, let C_1 denote the top 50 % accurate classifiers, and $\overline{C_1}$ denote the remaining classifiers in the pool. We computed diversity (Q-statistic) for all $\binom{T}{2}$ classifier pairs, and selected top

Table 1 UCI datasets used for experimentation

S. no	Dataset	#Att	#Rec	#Cl
1	Anneal (AN)	39	798	5
2	Balance scale (BS)	4	625	3
3	Br Cancer (BC)	11	699	2
4	Car (CR)	6	1,728	4
5	Contraceptive (CM)	9	1,473	3
6	Cr approval (CA)	16	690	2
7	Cyl band (CB)	40	852	2
8	Dermat (DM)	35	366	4
9	Ecoli (EC)	9	336	9
10	Flags (FL)	30	194	4
11	Glass (GL)	10	214	7
12	Heart (HH)	13	270	2
13	Hepatitis (HP)	21	155	2
14	Hyperthyroid (HT)	21	7,200	3
15	Image (IM)	19	2310	7
16	Ionosphere (IN)	34	351	2
17	Kr-vs.-Kp (KR)	36	3196	2
18	M factor (MF)	76	2,000	10
19	Sonar (SR)	61	208	2
20	Soy bean (SB)	35	307	19
21	Teaching assistant (TA)	5	151	3
22	Trans fusion (TF)	5	748	2
23	Vehicle (VH)	18	846	4
24	Wine (WN)	14	178	3

Att attributes, Rec records, Cl classes

50 % in set DP. Next, set DP was partitioned into three mutually exclusive partitions: pairs consisting of classifiers from C_1 (Set A), pairs consisting of classifiers from $\overline{C_1}$ (Set B) and pairs consisting of one classifier each from C_1 and $\overline{C_1}$ (Set C). Sizes of these sets are:

$$|A| = \frac{(k + 1)k}{2}, \quad |B| = \frac{k(k - 1)}{2}, \quad |C| = k(k + 1), \quad (1)$$

In each experiment, we counted the percentage pairs in set DP belonging to sets A, B, C. After repeating the exercise 30 times for each dataset, we recorded the results of 900 ($30 \times 3 \times 10$) experiments. Figure 2 shows that points belonging to set C constitute a compact cluster at higher percentage level. This establishes that the top 50 % diverse classifier pairs largely belong to set C, and the classifiers rejected in Step 5 are highly likely to be used in ensemble growing.

3.2 Algorithmic details

In this subsection, we present a step by step explanation of the ADP algorithm (Algorithm 1). The algorithm takes an

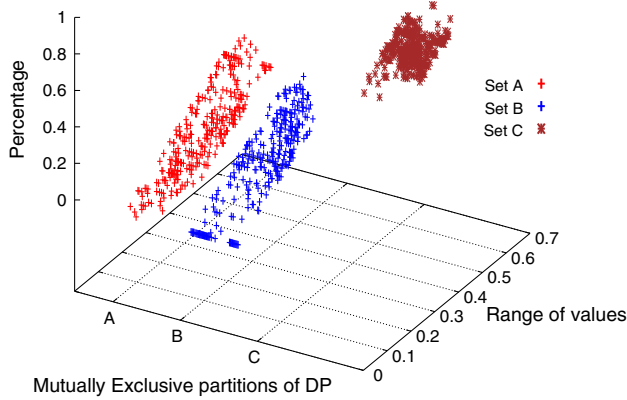


Fig. 2 Percentage composition of sets A, B and C

odd number of classifiers with their respective accuracies on validation set as input and sorts the classifiers in decreasing order of their accuracies (Step 1). In Steps 2–5, the top 50 % performing classifiers are retained in set C_1 and rest are discarded. They constitute the set of candidate ensembles at level 1. In addition, the highest accuracy classifier h_{max} is noted which constitutes the highest accuracy ensemble E discovered so far. In Step 6, $2k+1$ C_2 diversity pairs of $2k+1$ classifiers are generated and their diversities are calculated. In Steps 7 and 8, the classifier pairs constructed in Step 6 are sorted in decreasing order of their diversity, and top 50 % pairs are retained in set DP and rest are discarded. Steps 10–31 constitute the loop for growing the ensembles from level $2l-1$ to level $2l+1$ ($l = 1, 2, \dots, k$).

In Steps 12–20, each candidate ensemble E_i from set C_{2l-1} is combined with all such pairs of classifiers in set D , which are disjoint to E_i , to form candidate ensembles X of size $2l+1$ and stored in set S . Accuracy of each ensemble X in set S is also calculated. Step 21 sorts the ensembles in set S in decreasing order of their accuracies. In Step 23, the accuracy of best ensemble (e_{max}) in set S is compared with the maximum accuracy achieved so far a_{max} . If its accuracy is greater than the accuracy of E , then top 50 % performing ensembles in set S are retained in set C_{2l+1} for the next level, and E is updated to e_{max} . Otherwise, the loop is terminated, and ensemble E with accuracy a_{max} is reported as the output.

Being a heuristic algorithm, it cannot be guaranteed that the ADP algorithm will always find the optimal ensemble. However, the algorithm guarantees that on the pessimistic side the performance of the discovered ensemble will not be lower than that of the best classifier in the pool. On the optimistic side, the discovered ensemble is same as the optimal ensemble.

Claim 1 Accuracy (on validation set) of the ensemble delivered by Algorithm 1 is not lower than the accuracy of the best classifier in the pool.

Algorithm 1 Accuracy–Diversity Pruning (ADP) Algorithm

```

Require:  $\mathcal{E}$  : Set of  $T$  ( $=2k+1$ ) classifiers  $\mathcal{E} = \{h_1, \dots, h_T\}$  with their accuracies on validation set.
1: Sort the classifiers in  $\mathcal{E}$  in decreasing order of their accuracies in set  $C_{sorted}$ 
2: Let  $h_{max}$  be the maximum accuracy sub-ensemble of size 1.
3:  $E = h_{max}$  {Maximum accuracy ensemble seen so far}
4:  $a_{max} = acc(h_{max})$ 
5:  $C_1$  = set of top 50 % candidates in  $C_{sorted}$  {Candidate sub-ensembles of size 1}
6: Generate all pairs of classifiers in  $\mathcal{E}$  and compute their diversities.
7: Sort the diversity pairs in decreasing order of their diversity.
8: DP = set of top 50 % diverse pairs.
9:  $l = 1$ 
10: while  $C_{2l-1} \neq \phi$  &&  $l \leq k$  &&  $a_{max} \neq 1$  do
11:    $S = \phi$ 
   {Generate all candidate ensembles of size  $2l+1$ .}
12: for each candidate  $E_i$  in  $C_{2l-1}$  do
13:   for each pair  $P$  in DP do
14:     if  $E_i \cap P = \phi$  then
15:        $X = E_i \cup P$ 
16:       Compute accuracy of  $X$ 
17:        $S = S \cup X$ 
18:     end if
19:   end for
20: end for
21: Sort the ensembles in  $S$  in decreasing order of their accuracies
22:  $e_{max}$  = Maximum accuracy ensemble in set  $S$ 
23: if  $acc(e_{max}) > a_{max}$  then
24:    $a_{max} = acc(e_{max})$ 
25:    $E = e_{max}$ 
26:    $C_{2l+1}$  = Top 50 % ensembles in  $S$ 
27: else
28:    $C_{2l+1} = \phi$ 
29: end if
30:  $l = l + 1$ 
31: end while
32: return  $E, a_{max}$ 

```

Proof Given the pool \mathcal{E} of T classifiers, $\mathcal{E} = \{h_1, \dots, h_T\}$. Let h_{max} be the best performing classifier on validation set, and E_{adp} be the ensemble discovered by ADP algorithm. It is required to prove that $acc(h_{max}) \leq acc(E_{adp})$.

In Step 3 of Algorithm 1, $E_{adp} = h_{max}$. As the control enters the while loop, Step 23 ensures that the best accuracy ensemble at the current level is better than the previous best. Thus, $acc(h_{max}) \leq acc(E_{adp})$. \square

Since the classifiers are ordered on validation set accuracies, *similar claim of optimality on the test set is unreasonable*. Extensive experimental analysis presented in Table 3 offers evidence that ADP performs competitively on test set when compared to the optimal ensemble obtained by brute force search.

4 Experimental analysis

In this section, experimental evaluation of the proposed approach is presented. The proposed algorithm (ADP) and

Brute force (BF) search were implemented using C++ and bash shell scripts, and executed on quad core machine using OpenMP shared memory model. ADP program is straight forward parallel implementation with no optimizations. Twenty-four UCI [15] datasets shown in Table 1 were used for experiments. The experimental studies were designed to answer the following research questions:

1. How good is the quality of heuristics employed in ADP algorithm? (Sect. 4.2)
2. How does ADP approach compare with other ensemble pruning methods? (Sect. 4.3)
3. How do mismatches and execution timings scale with increasing pool size? (Sect. 4.4)
4. How do the *good* and *bad* diversity components [26] vary while ADP grows the ensemble? (Sect. 4.5)

4.1 Methodology

We employed tenfold cross validation (CV) for experimental evaluation of ADP algorithm. Conforming to tenfold CV method, in each of the ten iterations, nine folds constituted the training set and the tenth fold was used as test set. To further the confidence in results, CV was carried out three times for each dataset.

In each iteration, one-third data from the training set were retained as validation set and remaining two-thirds data were used to train classifiers. Bagging was used to generate a library L of 51 classifiers for each iteration. Each classifier was trained using decision tree² implementation (J48) in Weka [19], with default parameters ($C = 0.25$, $M = 2$).

Each experiment consisted of executing the compared pruning algorithms on the same set of classifiers. To add randomization in experimental evaluation, classifiers were picked up randomly from L to construct pools of sizes 11 and 21 for pruning. Since cross-validation was carried out three times, 30 experiments were carried out for each dataset and pool size. All reported results are averaged over 30 experiments.

4.2 Evaluating quality of heuristics

To answer Q.1 above, we (i) observe how often ADP algorithm discovers the optimal ensemble (obtained by BF search), (ii) compare ADP ensemble accuracies with optimal ensemble accuracies on validation set, (iii) inspect the trend for ADP missing the optimal ensemble with increasing pool size, (iv) compare ADP ensemble accuracies with optimal ensemble accuracies on test set.

² Recall that the choice of base classifier is incidental, since ADP is independent of the method of ensemble generation.

ADP algorithm and BF search were executed on pool sizes 11 and 21. Our decision to use BF search as benchmark limited the dataset size as well as pool size, because of its exponential time complexity. For each dataset and pool size, classifiers were drawn randomly from L . Note that during random selection, if a classifier is duplicated in a pool, its pairwise diversity with the identical classifier will be minimum and the pair will be discarded by ADP algorithm. By carrying out 30 experiments each on 24 datasets for 2 pool sizes, we performed 1,440 comparisons of ADP ensemble and optimal ensemble.

Performance comparison of ADP algorithm with BF search on validation set is presented in Table 2. Column “DS” displays the dataset, and “PS” indicates the pool size. Column “LA” reports the library accuracy. For the columns “Brute Force” and “ADP”, sub-columns “Acc” and “Size” show, respectively, the average accuracies and average sizes of discovered ensembles. It is observed from the table that average accuracy for BF ensembles is always better than the corresponding ADP ensembles, as expected. However, the sizes of ADP ensembles are generally smaller than BF ensembles hinting that loss in accuracy is often compensated by smaller-sized ensembles discovered by ADP algorithm. Further, for all datasets, average library accuracy values are much smaller than BF and ADP ensemble accuracies. This corroborates the well-accepted and understood fact that pruning leads to smaller-sized high accuracy ensembles.

4.2.1 Cost-benefit analysis

In this subsection, we analyze the difference in quality of ADP and BF ensembles on validation set. In Table 2, column “MM” indicates the number of times ADP ensemble discovers sub-optimal ensemble, “MAL” shows the maximum loss of accuracy among the sub-optimal ADP ensembles, and “AR” shows average (%) reduction in search space while discovering an ADP ensemble. Average reduction in search space is computed as % age ensembles pruned from lattice.

Column “MM” indicates that ADP algorithm discovers optimal ensemble 1,310/1,440 (91 %) times. In the remaining cases, when it discovers sub-optimal ensembles, a close study of accuracy loss is warranted. We choose to report the “MAL” among all the sub-optimal ensembles discovered by ADP algorithm to give an idea of the upper bound on error incurred by ADP heuristic. Optimal ensemble was missed 13 times for pool size 21, for MF dataset. However, “MAL” among all the 13 sub-optimal is only 0.45 %. Close analysis of SR dataset shows maximum accuracy loss of 6.38 %, when it misses optimal only 2/30 times. Despite this heavy accuracy loss, the average performance is marginally lower than that of BF (optimal) ensembles.

Table 2 Comparison of performance of BF search and ADP algorithm on validation set

S. no	DS	LA	PS	Brute force		ADP		Cost-benefit analysis			ToH
				Acc (%)	Size	Acc (%)	Size	MM	MAL (%)	RS (%)	
1	AN	91.70	11	94.00	3.73	94.00	3.73	0	0.00	79.71	NA
			21	95.35	4.47	95.30	4.13	4	0.55	97.47	NR
2	BS	81.15	11	83.69	4.00	83.66	3.93	1	0.71	75.26	NR
			21	86.03	6.73	85.74	5.07	11	1.42	93.63	R (0.0003)
3	BC	96.35	11	97.49	3.33	97.47	3.20	1	0.64	80.00	NR
			21	97.97	3.47	97.97	3.47	0	0.00	98.50	NA
4	CR	90.57	11	92.46	4.80	92.45	4.67	1	0.26	74.17	NR
			21	93.26	6.07	93.24	5.80	2	0.26	92.52	NR
5	CM	47.28	11	53.59	1.73	53.58	1.60	1	0.30	90.45	NR
			21	54.61	4.27	54.35	3.27	10	2.11	96.09	R (0.0061)
6	CA	88.50	11	91.37	3.53	91.30	3.33	4	0.65	78.59	NR
			21	92.56	5.60	92.39	4.80	6	1.28	95.12	R (0.0091)
7	CB	65.30	11	68.96	1.93	68.88	1.80	3	0.82	90.12	NR
			21	69.89	2.87	69.81	2.67	3	0.82	99.39	NR
8	DM	97.59	11	98.59	2.13	98.59	2.13	0	0.00	87.16	NA
			21	98.84	2.00	98.84	2.00	0	0.00	99.72	NA
9	EC	82.89	11	88.07	3.07	88.07	3.07	0	0.00	81.62	NA
			21	89.52	3.60	89.43	3.33	2	1.32	98.58	NR
10	FL	66.06	11	74.85	2.53	74.55	2.07	4	2.28	87.60	NR
			21	77.50	3.13	77.42	3.07	2	2.28	98.90	NR
11	GL	96.10	11	96.99	1.07	96.99	1.07	0	0.00	96.31	NA
			21	97.13	1.13	97.13	1.13	0	0.00	99.96	NA
12	HH	82.18	11	88.03	4.27	87.81	3.93	5	1.64	76.76	NR
			21	89.62	4.73	89.40	4.20	5	1.64	96.85	NR
13	HP	82.57	11	90.67	2.40	90.67	2.40	0	0.00	85.81	NA
			21	92.57	2.87	92.48	2.73	1	2.86	99.50	NR
14	HT	99.28	11	99.38	1.47	99.38	1.47	0	0.00	90.72	NA
			21	99.41	1.40	99.41	1.40	0	0.00	99.85	NA
15	IM	96.48	11	97.21	4.60	97.19	4.67	3	0.19	72.63	NR
			21	97.90	6.20	97.88	5.80	3	0.19	91.58	NR
16	IN	91.64	11	94.73	3.07	94.68	2.93	1	1.26	81.42	NR
			21	95.57	3.73	95.53	3.60	1	1.27	98.22	NR
17	KR	99.08	11	99.41	2.27	99.41	2.13	1	0.14	88.11	NR
			21	99.53	2.87	99.53	2.87	0	0.00	99.18	NA
18	MF	90.74	11	91.34	5.47	91.31	5.20	2	0.45	70.75	NR
			21	92.68	8.13	92.56	6.27	13	0.45	89.85	R (0.0001)
19	SR	77.16	11	86.17	4.33	85.89	4.00	2	6.38	74.99	NR
			21	90.14	5.93	89.79	5.27	5	2.13	93.30	NR
20	SB	82.53	11	87.88	2.80	87.88	2.80	0	0.00	82.70	NA
			21	90.38	4.80	90.04	3.67	7	1.45	97.57	R (0.0030)
21	TA	44.92	11	57.15	2.53	56.94	2.33	2	3.03	85.23	NR
			21	62.04	4.13	61.13	3.13	9	6.06	98.30	R (0.0023)
22	TF	78.60	11	81.04	3.40	81.00	3.20	2	0.60	80.22	NR
			21	82.61	5.07	82.55	4.73	3	0.60	95.31	NR

Table 2 continued

S. no	DS	LA	PS	Brute force		ADP		Cost-benefit analysis			ToH
				Acc (%)	Size	Acc (%)	Size	MM	MAL (%)	RS (%)	
23	VH	70.43	11	76.28	4.13	76.14	3.87	3	1.57	77.68	NR
			21	78.22	5.73	78.05	5.07	7	1.57	94.10	NR
24	WN	94.52	11	97.93	2.80	97.93	2.80	0	0.00	88.51	NA
			21	99.25	3.00	99.25	3.00	0	0.00	99.87	NA

DS dataset, *PS* pool size, *LA* library accuracy, *MM* mismatches in 30 repetitions, *MAL* maximum loss of accuracy by an ADP ensemble as compared to the corresponding BF ensemble, *RS* reduction in search space, *ToH* test of hypothesis (*NA* not applicable, *NR* not rejected, *R* rejected with corresponding *p*-value)

It is further observed that percentage average reduction in search space varies from 70.75 to 99.87 % over all experiments. This reduction in search space translates to speedier discovery of near-optimal ensembles compared to BF search.

4.2.2 Comparison of accuracies of ADP ensembles and BF ensembles

Given a library L of T classifiers, let L_P denote the set of $T C_P$ pools, each pool containing P classifiers. When BF search and ADP algorithm are run on all the pools in L_P , two distinct distributions of accuracies are obtained. Let P^O denote the distribution of accuracies obtained by BF search, and P^A denote the same obtained by ADP algorithm. Paired observations arise when BF search and ADP algorithm report accuracy values on the same pool. We hypothesize that the mean difference in the paired accuracy values of P^O and P^A is zero.

To test the hypothesis, we take a sample S of size 30 from L_P . Each element of S is a pool of P classifiers on which both algorithms are executed. Let $O = \{a_1^O, \dots, a_{30}^O\}$ and $A = \{a_1^A, \dots, a_{30}^A\}$, respectively, denote the sample accuracies of optimal ensembles and ADP ensembles. We apply paired t test to show that O and A values do not differ significantly. The paired t test investigates the differences between paired values in two samples, to determine how likely it is that the two samples belong to the same population.

For each dataset and pool size, let μ_D denote the mean of differences between the paired values, i.e., $\mu_D = \frac{1}{n} \sum_{i=1}^n (a_i^O - a_i^A)$. In consonance with our claim, the null hypothesis $H_0: \mu_D = 0$ is tested against the alternate hypothesis $H_1: \mu_D > 0$. Since accuracy values reported by BF search on validation set are never less than accuracy values reported by ADP algorithm, a right-tailed paired t test is carried out. MATLAB function is used to compute p value for those cases which have $MM > 0$.

For 28 out of 34 populations, the null hypothesis is accepted at 1 % significance level. It can be seen from Table 2 that low p values (< 0.01) have been obtained for the datasets

showing large number of mismatches. Thus in 28/34 cases, there was no statistical evidence for rejecting H_0 in favor of H_1 .

In 6/34 experiments, the difference between the average ADP accuracies and BF accuracies is significant, and the probability of the two samples from same population showing such difference is very small. In other words, for these six datasets, ADP is not performing as well as BF, which is also evident from “MM” column in the table for these datasets.

4.2.3 Comparison of test accuracies of BF search and ADP ensembles

Performance of a classifier on the test set indicates the true capability to generalize. Accordingly, we compare the performance of BF and ADP ensembles on test set and report the results in Table 3. For 8/48 cases, all ADP ensembles exactly match the corresponding BF ensembles. Average ADP test accuracies for the remaining cases are also comparable with the corresponding average BF test accuracies.

We hypothesize that BF test accuracies are almost same as ADP test accuracies for all datasets. Since the possibility of ADP ensemble reporting higher test accuracies than BF ensemble cannot be ruled out, the null hypothesis $H_0: \mu_D = 0$ is tested against the alternate hypothesis $H_1: \mu_D \neq 0$. Two-tailed paired t test is applied for drawing statistical inference. The null hypothesis was tested at 1 % significance level. The last column in Table 3 shows that null hypothesis is accepted for 39/40 cases, indicating that both the approaches perform equally well. The hypothesis is rejected for one case (dataset WN pool size 11) where ADP ensembles perform much better than BF ensembles.

Thus, we conclude that generalization accuracy of ADP matches that of BF search. On the basis of this result, it is reasonable to prescribe ADP algorithm for the evaluation of any novel ensemble method with respect to optimal ensemble.

Table 3 Comparison of test accuracies for BF search and ADP algorithm

S. no	DS	PS	BF Acc (%)	ADP Acc (%)	ToH
1	AN	11	92.65	92.56	NR
		21	94.61	94.74	NR
2	BS	11	79.99	80.05	NR
		21	81.11	81.00	NR
3	BC	11	96.23	96.32	NR
		21	96.66	96.61	NR
4	CR	11	89.87	89.85	NR
		21	90.95	90.95	NA
5	CM	11	50.26	50.42	NR
		21	48.92	49.69	NR
6	CA	11	86.47	86.52	NR
		21	87.20	87.25	NR
7	CB	11	67.41	67.04	NR
		21	67.22	67.22	NR
8	DM	11	96.83	96.92	NR
		21	96.82	96.91	NR
9	EC	11	83.04	83.04	NA
		21	83.02	83.22	NR
10	FL	11	64.22	64.76	NR
		21	65.77	65.94	NR
11	GL	11	95.64	95.79	NR
		21	97.08	97.38	NR
12	HH	11	80.00	80.12	NR
		21	79.38	79.75	NR
13	HP	11	81.17	81.17	NA
		21	84.58	84.58	NA
14	HT	11	99.25	99.25	NA
		21	99.25	99.25	NA
15	IM	11	96.42	96.45	NR
		21	96.81	96.71	NR
16	IN	11	91.44	91.54	NR
		21	92.39	92.3	NR
17	KR	11	99.34	99.33	NR
		21	99.42	99.42	NA
18	MF	11	88.92	88.72	NR
		21	90.00	89.75	NR
19	SR	11	75.70	75.37	NR
		21	75.63	74.56	NR
20	SB	11	80.70	80.7	NA
		21	81.86	81.54	NR
21	TA	11	47.06	46.84	NR
		21	44.59	45.54	NR
22	TF	11	78.78	78.60	NR
		21	79.01	79.01	NR

Table 3 continued

S. no	DS	PS	BF Acc (%)	ADP Acc (%)	ToH
23	VH	11	71.44	71.60	NR
		21	71.44	71.60	NR
24	WN	11	94.39	95.14	NR
		21	93.50	96.85	R (0.0006)

DS dataset, PS pool size, ToH test of hypothesis (NA not applicable, NR not rejected, R rejected with corresponding p-value)

4.3 Comparison of ADP algorithm with other ensemble techniques

In this subsection, we compare the performance of ADP algorithm with complete (unpruned) ensemble (CE) and three other ensemble pruning techniques. There are two criteria on the basis of which it can be compared with other ensemble pruning methods:

- (i) Search strategy: ADP has been compared with forward selection (FS) and backward elimination (BE) methods, since these two pruning techniques search the lattice to locate the optimal ensemble, just like ADP. In FS (BE) method, a pair of classifiers is progressively added to (removed from) the ensemble and monitored for improvement in accuracy [31]. The selection (elimination) of classifiers is continued till the accuracy improves.
- (ii) Use of accuracy and diversity in heuristics: We also compare ADP algorithm with the one proposed by Ko et al. [23] which uses a compound diversity function (CDF) measure for pruning an ensemble. This measure takes into account the accuracy of each classifier member (a_i), as well as pairwise diversity (d_{ij}) between classifier members C_i & C_j in the ensemble. We use Q-statistic for diversity calculation in ADP approach and computation of CDF measure. For an ensemble of size P , the CDF measure is calculated as follows [23]:

$$CDF = \frac{P}{P-1} \left(\prod_{i=1}^P (1 - a_i) \right)^{\frac{1}{P}} \left(\prod_{i,j=1, i \neq j}^P d_{ij} \right)^{\frac{1}{P(P-1)}} \tag{2}$$

We computed CDF values for all ensembles in the search lattice, and the ensemble(s) with minimum CDF value are selected as optimal ensemble(s). This is unlike the proposal in [23] which uses GA to select the optimal ensemble. To avoid randomization involved in GA, we carried out exhaustive search to locate the optimal

ensemble. The obtained optimal ensemble is then used to evaluate the test sets for the corresponding datasets.

Results of the experiments are shown in Table 4. For each dataset and pool size, the highest accuracies reported on the test set have been displayed in bold. It can be observed from the table that for 42/48 cases, pruning yields better results than the complete ensemble. Comparing the sizes of ensembles obtained by all the methods, FS reports smallest ensembles and BE the largest, although average accuracy values reported by FS ensembles are not better than ADP algorithm for majority datasets. The CDF ensembles are considerably larger than ADP ensembles with 8/48 wins.

To further strengthen our conclusion, we applied Friedman test for multiple comparisons. This non-parametric test is commonly used to test the null hypothesis (H_0) that all classifier performances are equivalent, against the alternate (H_1) that at least one classifier exhibits performance which is significantly different from others [10]. The test returns p value, which is examined to draw an inference about the hypothesis. If the p value is close to zero, the null hypothesis is rejected indicating that the performance of at least one classifier is significantly different from others.

Friedman test ranks the methods on the basis of ensemble accuracy for each dataset, with lower values of ranks indicating lower performance. Mean rank for a method is obtained by averaging its ranks over all datasets. It was found that mean rank for ADP algorithm was the maximum (84.11) followed by CDF (75.86) and FS (75.13). Lower ranks are reported for CE (73.47) and BE (68.93). At 95 % confidence level, very low p value of $8.99886e-19$ was obtained. Consequently, the null hypothesis was not accepted, implying that sample mean accuracy of at least one method was significantly different from that of the others.

Next, Nemenyi test is employed for pairwise comparison of classifiers to detect the classifier(s) significantly different from the others. Figure 3 shows the outcome of the Nemenyi test. Mean rank of each classifier is indicated by a circle, and horizontal bars across each circle denote the intervals about these mean values. The extent of overlapping of intervals indicates the difference between two estimates being compared. It is evident from the figure that ADP algorithm is significantly different from the other methods, since it shows a smallest interval overlap with other methods. ADP scores over other algorithms because of highest mean rank (indicated by circle on ADP bar). The intervals for CDF and FS show substantial overlapping indicating similar performance, the fact is vindicated by almost equal mean ranks obtained by Friedman's test. We attribute improved ranking of CDF to an exhaustive search for CDF ensemble (against GA-based search in [23]).

4.4 Scalability of ADP algorithm

We also investigated if there exists a trend in execution timings and discovery of sub-optimals with increasing pool sizes. We executed BF search and ADP algorithm for pool sizes varying from 11 to 25 with increment of 2. The experiments were done for three datasets—one with heavy losses (BS), one with moderate losses (CA) and the last one with no losses (DM). For each dataset and pool size, 30 experiments were carried out (as described in Sect. 4.2).

4.4.1 Discovery of sub-optimals

We observed the number of sub-optimals discovered by ADP algorithm. The number of mismatches corresponding to each dataset and pool size is plotted in Fig. 4. Visual inspection of the graph reveals that there is no monotonic trend for losses with increasing pool size.

4.4.2 Execution timings

Current ADP implementation is a straight-forward parallel implementation, with tremendous scope for optimizations. For each dataset and pool size, we recorded execution timings and plotted the graph (Fig. 5). It is clear that the execution time rises exponentially with the pool size. However, this is deceptive considering that the brute force timing could not be plotted in the same graph because of massive scale difference. For DM dataset, one ADP experiment for pool size 25 on an average took 3 s to complete, while the average time for brute force search was more than 3 min. The general improvement in timings is also indicated by column 'AR' of Table 2, which shows massive reduction in the search space by ADP algorithm. As the pool size increases further, the difference in execution times of BF and ADP also increases exponentially.

We admit that the current implementation is suitable for medium pool sizes so much so that for pool sizes >21 , it is slower than FS and BE.

Closer observation of execution timings reveals the two tasks that slow the algorithm. These are (i) writing of all intermediate ensembles in file and (ii) their sorting, at each level. For larger pool sizes, this problem is too acute to be tolerated. Design of smart indexing is a potential solution for the problem.

4.5 Good and bad diversity

Kuncheva and Brown established that overall diversity in majority voting ensembles can be decomposed into *Good* and *Bad* diversities [4]. The good diversity pulls the performance of the ensemble up, while the bad diversity pulls it down. Using the notations in [4], these two are related as follows:

Table 4 Comparison of ADP algorithm with other ensemble techniques on test set

S. no	DS	PS	CE	CDF		ADP		FS		BE	
				Acc	Size	Acc	Size	Acc	Size	Acc	Size
1	AN	11	91.36	6.73	92.15	3.73	92.56	3.67	92.57	8.13	91.57
		21	92.57	7.40	94.15	4.13	94.74	4.2	94.45	17.87	92.90
2	BS	11	79.30	8.80	79.14	3.93	80.05	3.73	79.83	8.13	78.23
		21	79.67	12.73	79.93	5.07	81.00	4.60	80.74	17.33	79.30
3	BC	11	95.52	6.47	95.85	3.20	96.32	3.13	95.51	9.20	95.37
		21	95.94	8.27	96.09	3.47	96.61	3.27	95.61	19.07	95.75
4	CR	11	89.27	8.67	89.51	4.67	89.85	4.40	89.62	8.07	88.79
		21	89.85	11.20	89.83	5.80	90.95	5.20	90.6	16.20	90.14
5	CM	11	47.25	10.60	47.27	1.60	50.42	1.60	50.42	5.40	43.81
		21	46.46	15.67	46.52	3.27	49.69	2.60	49.51	14.53	46.12
6	CA	11	85.85	7.27	85.89	3.33	86.52	3.53	86.52	8.27	85.07
		21	86.43	9.67	86.43	4.80	87.25	4.47	86.38	17.93	86.04
7	CB	11	65.68	8.40	66.23	1.80	67.04	1.73	66.91	8.53	65.80
		21	65.99	12.27	67.10	2.67	67.22	2.67	66.98	19.53	66.36
8	DM	11	96.55	5.00	96.55	2.13	96.92	2.07	96.28	10.07	96.46
		21	96.64	7.60	96.63	2.00	96.91	2.00	96.09	20.27	96.64
9	EC	11	82.42	7.00	82.64	3.07	83.04	3.13	82.64	8.60	81.44
		21	81.85	8.47	81.75	3.33	83.22	3.13	82.43	19.00	81.05
10	FL	11	64.05	5.87	64.40	2.07	64.76	2.13	64.24	7.40	62.16
		21	63.89	7.13	64.26	3.07	65.94	2.67	62.88	19.00	61.97
11	GL	11	96.93	5.73	97.86	1.07	95.79	1.07	95.03	10.33	97.55
		21	97.24	9.87	97.84	1.13	97.38	1.13	96.14	20.60	97.23
12	HH	11	80.25	5.93	79.63	3.93	80.12	3.93	78.77	8.60	78.52
		21	79.51	7.47	78.52	4.20	79.75	3.87	78.39	18.2	79.01
13	HP	11	80.87	5.40	82.46	2.40	81.17	2.40	79.21	8.93	79.42
		21	81.85	8.13	86.11	2.73	84.58	2.93	82.44	19.47	82.03
14	HT	11	99.16	6.00	99.23	1.47	99.25	1.47	99.12	10.13	99.16
		21	99.15	9.87	99.19	1.40	99.25	1.40	99.14	20.53	99.15
15	IM	11	96.33	7.07	96.32	4.67	96.45	4.60	96.29	8.47	96.06
		21	96.22	8.73	96.26	5.80	96.71	5.53	96.36	16.67	96.28
16	IN	11	90.30	5.47	90.87	2.93	91.54	2.80	90.3	8.80	89.63
		21	92.01	7.33	90.78	3.60	92.30	3.47	90.20	18.60	91.63
17	KR	11	99.17	5.80	99.34	2.13	99.33	2.07	99.25	9.40	99.17
		21	99.07	7.80	99.34	2.87	99.42	2.73	99.32	19.60	99.07
18	MF	11	89.17	9.20	89.33	5.20	88.72	4.80	88.88	8.73	88.32
		21	90.18	12.40	89.95	6.27	89.75	6.53	90.17	17.00	90.07
19	SR	11	76.64	3.87	74.21	4.00	75.37	3.87	74.89	8.33	73.43
		21	77.44	6.73	73.60	5.27	74.56	4.60	71.63	18.13	75.21
20	SB	11	79.16	6.40	80.58	2.80	80.70	2.67	79.17	8.20	77.98
		21	79.27	7.33	80.58	3.67	81.54	3.60	80.03	18.93	78.40
21	TA	11	47.89	3.93	45.17	2.33	46.84	2.27	45.03	8.13	43.32
		21	47.89	4.93	42.62	3.13	45.54	2.80	43.70	18.53	47.67
22	TF	11	78.24	9.40	78.42	3.20	78.60	3.13	78.47	8.20	77.43
		21	78.51	13.33	79.23	4.73	79.01	4.73	78.33	18.07	78.47

Table 4 continued

S. no	DS	PS	CE	CDF		ADP		FS		BE	
				Size	Acc	Size	Acc	Size	Acc	Size	Acc
23	VH	11	70.53	8.73	71.09	3.87	71.60	3.47	71.32	7.47	68.09
		21	70.62	11.27	70.10	5.07	71.60	4.47	71.13	14.93	70.38
24	WN	11	94.59	5.55	97.88	2.80	95.14	2.80	93.47	9.07	94.23
		21	94.40	10.47	98.07	3.00	96.85	2.93	93.67	19.73	94.41
W/L/T			6/42/0		8/40/0		31/15/2		1/45/2		0/48/0

DS dataset, PS pool size, CE complete ensemble, CDF compound diversity function approach, FS forward search, BE backward elimination, W/L/T win/loss/ties

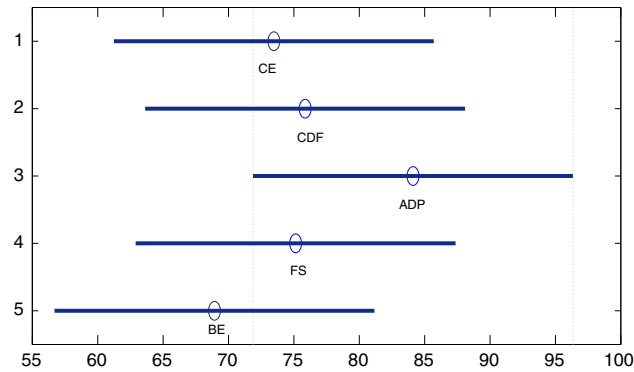


Fig. 3 Comparison of CE, CDF, ADP, FS and BE algorithms using Nemenyi test

$$E_{\text{maj}} = \int_x e_{\text{ind}}(x) - \underbrace{\int_{x^+} \frac{1}{T} \sum_{t=1}^T \delta_t(x)}_{\text{good diversity (G)}} + \underbrace{\int_{x^-} \frac{1}{T} \sum_{t=1}^T \delta_t(x)}_{\text{bad diversity (B)}} \quad (3)$$

Here, $e_{\text{ind}}(x)$ is the average individual loss for classifiers in the pool. E_{maj} is majority voting loss, $\delta_t(x)$ is the disagreement between t th classifier and the ensemble, and x^- and x^+ denote the data subspaces where the ensemble is correct and incorrect, respectively.

To strengthen our hypothesis of effectiveness of accuracy–diversity integration, we examined *Good* (G) and *Bad* (B) diversities at each step of growing the ensembles. In this experiment, we used pool of size 31 because ADP algorithm executed reasonably fast on this pool size. We found that the biggest ensemble of size 7 was obtained five times in BC dataset. These ensembles are denoted as E_1, \dots, E_5 . We selected these ensembles for closer investigation and studied their growth at each step. For each of these five cases, the algorithm explored up to size 9 ensembles and reported size 7 ensemble as optimal. G and B were computed at each step as per Eq. 3 and plotted (Fig. 6). It can be observed that good diversity increases and bad diversity decreases till the optimal ensemble is located. Extending the ensemble beyond this point, the good diversity decreases and the bad diversity increases. Thus, the maximum value of G and minimum value of B are attained when the “optimal” ensemble is obtained.

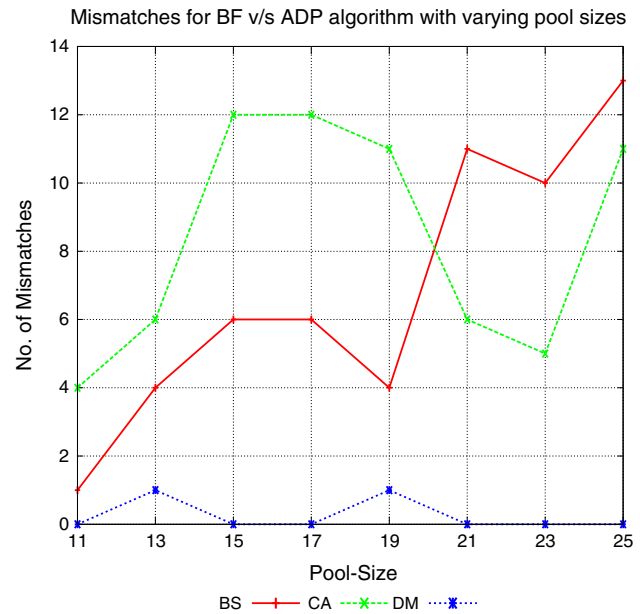


Fig. 4 Number of mismatches with optimal ensemble for three datasets BS, CA, DM (pool sizes 11–25)

4.6 Discussion

ADP algorithm is a heuristic algorithm and we do not expect it to always discover an optimal ensemble. Experimental evaluation, however, demonstrates that the quality of heuristics is reasonably good. Intuition underlying the heuristics in ADP algorithm is as follows:

1. Low accuracy classifiers are not likely to participate in optimal ensemble, unless paired with a high accuracy classifier. This motivates (a) rejection of 50 % low accuracy classifiers (Step 5), (b) rejection of 50 % low diversity classifier pairs (Step 8) and (c) extending sub-ensemble by adding relatively high diversity classifier pair (Steps 12–20).
2. If at the current level, the best accuracy sub-ensemble has accuracy less than or equal to that of previous level best accuracy sub-ensemble, then this cannot be the optimal ensemble. Further, it is not likely to be found at higher

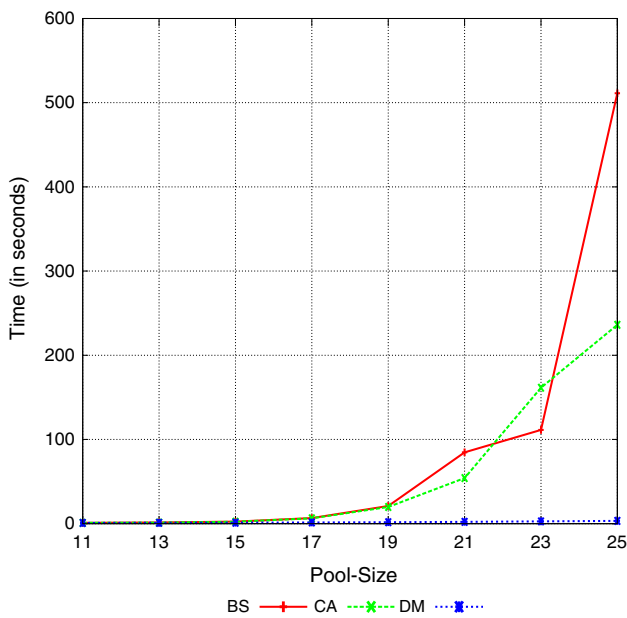


Fig. 5 Execution times for three datasets BS, CA, DM (pool sizes 11–25)

levels.

This motivates the stopping criteria at Step 28. Close investigation reveals that this is a frequent cause of failure when a sub-optimal ensemble is discovered by ADP.

3. Only relatively accurate ensembles are likely to grow into an optimal ensemble.

This motivates rejection of 50 % low-performing candidate sub-ensembles in the while loop (Step 26).

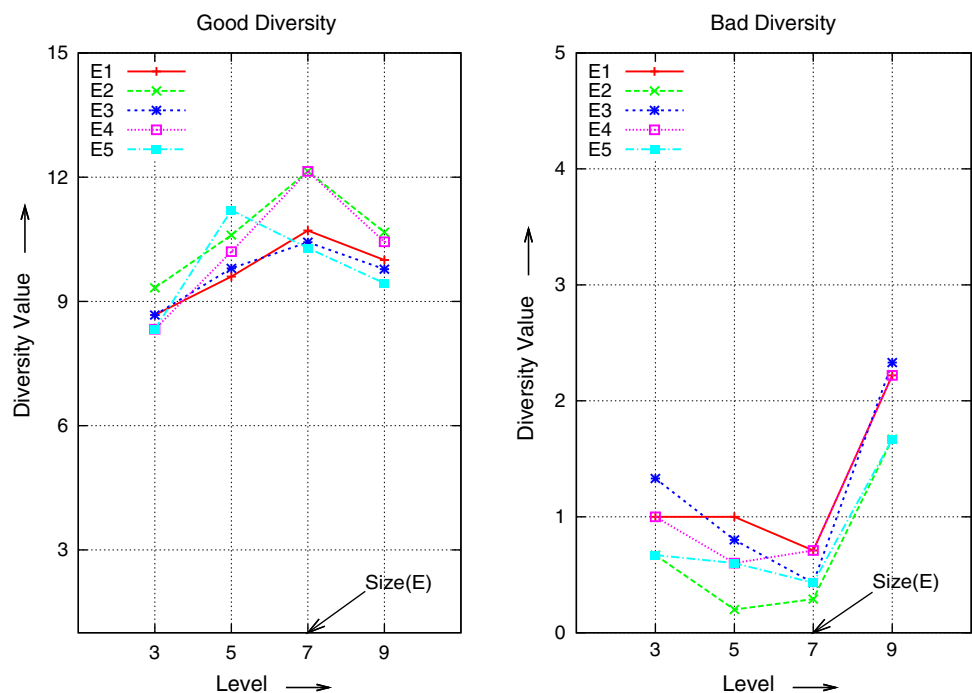
There are certain limitations of the experimentation, which were beyond our control and which we would like to present here.

Firstly, despite our best intention to use datasets that have been commonly used in other studies [29,32], we had to restrict to relatively smaller datasets so that the BF search on pool size 21 and above, could be executed in reasonable time. For instance, a single experiment for pool size 31 on Magic dataset took almost 2 days to complete. With the available hardware, it was not possible to complete 30 experiments in time. Hence, the pool size was restricted.

Secondly, scalability of the algorithm is a serious issue that needs to be addressed for it to be of practical utility. The main factor that dampens the scalability is the generation of duplicate ensembles while combining candidate sub-ensembles of size $2l - 1$ with disjoint classifier pairs in set DP to generate ensembles of size $2l + 1$. For example, while classifier {1} can be combined with classifier pair {2, 3} to generate size 3 ensemble {1, 2, 3}. Similarly, classifiers {2} and {3} can be combined with pairs {1, 3} and {1, 2}, respectively, to generate the ensemble {1, 2, 3}. This leads to duplication of effort as an ensemble generated once need not to be generated again.

In the current implementation of ADP algorithm, duplicates are removed by sorting. This sorting is responsible for exponential rise in execution times for higher pool sizes. Therefore, there is a need to devise an efficient strategy/data structure to avoid generation of duplicates, which will lead to a substantial reduction in execution time.

Fig. 6 Variation in good/bad diversities for five biggest ensembles in BC dataset



5 Conclusion and future work

In this paper, we present a heuristic algorithm ADP which makes use of both accuracy and diversity for ensemble pruning. The algorithm is conceptually simple and operationally efficient in search space reduction. Using multi-pronged hill climb, ADP algorithm systematically grows ensembles by adding a pair of diverse classifiers. An ensemble is grown further if its performance has improved after addition of the pair.

Empirical evaluation of ADP algorithm shows that most of the times it discovers an optimal ensemble from the pool. In case it discovers sub-optimal ensemble, the loss of accuracy is insignificant. The growth strategy indicates that at each step, *good* diversity of the ensemble increases and the *bad* diversity decreases. Since the algorithm delivers near optimal ensembles, it can be used as a benchmark for evaluating other ensemble-pruning algorithms. However, the current implementation of the algorithm makes it unscalable. In future, we intend to overcome this limitation so that bigger pools can be pruned to obtain optimal ensembles.

Acknowledgments Authors gratefully acknowledge the constructive comments and suggestions forwarded by the anonymous reviewers, which significantly improved the experimentation.

References

- Banfield, R.E., Hall, L.O., Bowyer, K.W., Kegelmeyer, W.P.: Ensemble diversity measures and their applications to thinning. *Inf. Fusion* **6**(1), 49–62 (2005)
- Breiman, L.: Bagging predictors. *Mach. Learn.* **24**, 123–140 (1996)
- Breiman, L.: Random forests. *Mach. Learn.* **45**, 5–32 (2001)
- Brown, G., Kuncheva, L.I.: “Good” and “bad” diversity in majority vote ensembles. In: *MCS 2010*, pp. 124–133 (2010)
- Brown, G., Wyatt, J.L., Harris, R., Yao, X.: Diversity creation methods: a survey and categorisation. *Inf. Fusion* **6**(1), 5–20 (2005)
- Caruana, R., Munson, A., Niculescu-Mizil, A.: Getting the most out of ensemble selection. In: *Proceedings of the Sixth International Conference on Data Mining, ICDM '06*, pp. 828–833. IEEE Computer Society (2006)
- Caruana, R., Niculescu-Mizil, A., Crew, G., Ksikes, A.: Ensemble selection from libraries of models. In: *Proceedings of the twenty-first international conference on Machine learning, ICML '04*. ACM (2004)
- Chen, H., Tiho, P., Yao, X.: Predictive ensemble pruning by expectation propagation. *IEEE Trans. Knowl. Data Eng.* **21**(7), 999–1013 (2009)
- Dai, Q.: A competitive ensemble pruning approach based on cross-validation technique. *Knowl. Based Syst.* **37**, 394–414 (2013)
- Demsar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006)
- Didaci, L., Fumera, G., Roli, F.: Diversity in classifier ensembles: fertile concept or dead end? In: Zhou, Z.H., Roli, F., Kittler, J. (eds.) *Multiple classifier systems*. Lecture Notes in Computer Science, vol. 7872, pp. 37–48. Springer, Berlin (2013)
- Dietterich, T.: Ensemble methods in machine learning. In: *International workshop on Multiple Classifier Systems*, pp. 1–10 (2000)
- Dietterich, T.G.: An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Mach. Learn.* **40**, 139–157 (2000)
- Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern classification*, 2nd edn. Wiley (2001)
- Frank, A., Asuncion, A.: UCI machine learning repository (2010). <http://archive.ics.uci.edu/ml>
- Giaccinto, G., Roli, F., Fumera, G.: Design of effective multiple classifier systems by clustering of classifiers. In: *Proceedings of ICPR2000, 15th International Conference on Pattern Recognition*, pp. 3–8 (2000)
- Giaccinto, G., Roli, F.: Design of effective neural network ensembles for image classification purposes. *Image Vis. Comput. J.* **19**, 699–707 (2001)
- Guo, L., Boukir, S.: Margin-based ordered aggregation for ensemble pruning. *Pattern Recognit. Lett.* **34**(6), 603–609 (2013)
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. *SIGKDD Explor.* **11**(1), 10–18 (2009)
- Ho, T.K.: The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**, 832–844 (1998)
- Partalas, G.T., Vlahavas, I.: Focussed ensemble selection: a diversity based method for greedy ensemble selection. *Front. Artif. Intell. Appl.* pp. 117–121 (2008)
- Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(3), 226–239 (1998)
- Ko, A.H.R., Sabourin, R., de Souza Britto Jr., A.: Compound diversity functions for ensemble selection. *IJPRAI* **23**(4), 659–686 (2009)
- Hoboken, L.I.: *Combining pattern classifiers: methods and algorithms*. Wiley-Interscience, Hoboken, New Jersey (2004)
- Kuncheva, L.I.: Using diversity measures for generating error-correcting output codes in classifier ensembles. *Pattern Recognit. Lett.* **26**(1), 83–90 (2005)
- Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with ensemble accuracy. *Mach. Learn.* **51**, 181–207 (2003)
- Lam, L.: Classifier combinations: Implementations and theoretical issues. In: *Multiple Classifier Systems*, pp. 77–86 (2000)
- Li, N., Yu, Y., Zhou, Z.H.: Diversity regularized ensemble pruning. In: Flach, P., Bie, T., Cristianini, N. (eds.) *Machine Learning and Knowledge Discovery in Databases*. Lecture Notes in Computer Science, vol. 7523, pp. 330–345. Springer, Berlin (2012)
- Lu, Z., Wu, X., Zhu, X., Bongard, J.: Ensemble pruning via individual contribution ordering. In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '10*, pp. 871–880. ACM (2010)
- Margineantu, D.D., Dietterich, T.G.: Pruning adaptive boosting. In: *Proceedings of the Fourteenth International Conference on Machine Learning, ICML '97*, pp. 211–218. Morgan Kaufmann Publishers Inc. (1997)
- Marill, T., Green, D.: On the effectiveness of receptors in recognition systems. *IEEE Trans. Inf. Theor.* **9**(1), 11–17 (2006)
- Martínez-Muñoz, G., Suárez, A.: Pruning in ordered bagging ensembles. In: *Proceedings of the 23rd international conference on Machine learning, ICML '06*, pp. 609–616. ACM (2006)
- Martínez-Munoz, G., Suarez, A.: Aggregation ordering in bagging. In: *International Conference on Artificial intelligence and Applications (IASTED)* pp. 258–263 (2004)
- Opitz, D.W.: Feature selection for ensembles. In: *Proceedings of the 16th national conference on Artificial intelligence, AAAI '99/IAAI '99*, pp. 379–384. American Association for Artificial Intelligence (1999)

35. Partalas, I., Tsoumakas, G., Vlahavas, I.: Focused ensemble selection: A diversity-based method for greedy ensemble selection. In: Proceeding of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence, pp. 117–121. IOS Press (2008)
36. Partalas, I., Tsoumakas, G., Vlahavas, I.: Pruning an ensemble of classifiers via reinforcement learning. *Neurocomputing* **72**, 1900–1909 (2008)
37. Rokach, L.: Collective-agreement-based pruning of ensembles. *Comput. Stat. Data Anal.* **53**(4), 1015–1026 (2009)
38. Rokach, L., Maimon, O.: *Data mining with decision trees: theory and applications*. World Scientific Publishing Co., Inc. (2008)
39. Tamon, C., Xiang, J.: On the boosting pruning problem. In: Proceedings of the 11th European Conference on Machine Learning, ECML '00, pp. 404–412. Springer (2000)
40. Tang, E.K., Suganthan, P.N., Yao, X.: An analysis of diversity measures. *Mach. Learn.* **65**, 247–271 (2006)
41. Tsoumakas, G., Partalas, I., Vlahavas, I.: Ensemble pruning primer. In: O. Okun, G. Valentini (eds.) *Applications of supervised and unsupervised ensemble methods*, vol. 245, pp. 1–13 Springer, Heidelberg (2009)
42. Webb, G.I., Zheng, Z.: Multistrategy ensemble learning: reducing error by combining ensemble learning techniques. *IEEE Trans. Knowl. Data Eng.* **16**, 980–991 (2004)
43. Zhang, P., Zhu, X., Shi, Y., Wu, X.: An aggregate ensemble for mining concept drifting data streams with noise. *Advances in Knowledge Discovery and Data Mining. Lecture Notes in Computer Science*, vol. 5476, pp. 1021–1029. Springer, Berlin (2009)
44. Zhang, Y., Burer, S., Street, W.N.: Ensemble pruning via semi-definite programming. *J. Mach. Learn. Res.* **7**, 1315–1338 (2006)
45. Zhou, Z.H., Tang, W.: Selective ensemble of decision trees. In: *Proceedings of the 9th international conference on Rough sets, fuzzy sets, data mining, and granular computing*, pp. 476–483. Springer, Berlin (2003)