Self-Adaptive Heterogeneous Random Forest

Mohamed Bader-El-Den
School of Computing, University of Portsmouth
Buckingham Building, PO1 3HE, UK
Email: Mohamed.Bader@port.ac.uk

***Abstract*—Random Forest RF is an ensemble learning approach that utilises a number of classifiers to contribute though voting to predicting the class label of any unlabelled instances. Parameters such as the size of the forest N and the number of features used at each split M, has significant impact on the performance of the RF especially on instances with very large number of attributes. In a previous work Genetic Algorithms has been used to dynamically optimize the size of *RF*. This study extends this genetic algorithm approach to further enhance the accuracy of *Random Forests* by building the forest out of heterogeneous decision trees, heterogeneous here means trees with different M values. The approach is termed as Heterogeneous Genetic Algorithm based Random Forests (*HGARF*). As Random Forests generates a typical large number of decision trees with randomisation over the feature space when splitting at each node for all the trees, this has motivated the development of a genetic algorithm based optimisation. Typically, *HGARF* accepts as an input a forest $\overrightarrow{RF}$ of N trees, the initial population is randomly generated from $\overrightarrow{RF}$ as a number of smaller random forests $\overrightarrow{rf_i}$ where each one has a number $n_i \leq N$ of trees. This population of forests is then evolved through a number of generations using genetic algorithms. Our extensive experimental study has proved that Random Forests performance could be boosted using the genetic algorithm approach.**

I. Introduction

Random Forest (RF) is an ensemble classification method aims to boost the performance of classification techniques. It is based on the process of building a number of classifiers, and then collectively use them all to identify unlabelled instances. Two widely used ensemble approaches could be identified, namely, *boosting* and *bagging*. Boosting is an incremental process of building a sequence of classifiers, where each classifier works on the incorrectly classified instances of the previous one in the sequence. AdaBoost [7] is the representative of this class of techniques. However, AdaBoost is pruned to overfitting. The other class of ensemble approaches is the Bootstrap Aggregating (*Bagging*) [4]. Bagging involves building each classifier in the ensemble using a randomly drawn sample of the data, having each classifier giving an equal vote when labelling unlabelled instances. Bagging is known to be more robust than boosting against model overfitting. The main representative of bagging is *Random Forests* [5]. In random forests, a number of trees are generated, having each tree built using a randomly drawn instances from the data set. Randomisation is also applied when selecting the best node to split on for all the trees. Typically this is an input parameter which is equal to $\sqrt{F}$, where F is the number of features in the data set. More details about random forests are presented in Section II.

This paper proposes a novel approach to optimising random forests boosting their performance. The approach is termed Heterogeneous Genetic Algorithm based Random Forests (*HGARF*). The *HGARF* approach starts by generating a large heterogeneous random forest of N decision trees, forming a vector $\overrightarrow{RF}$. In classical RF, all trees are built using the same M value which is the the number of features used at each split M which is the number of features used at each split M, normally M is set to square root of the number of features F. However, there is no defined way to select the value of M. In this paper, heterogeneous RF means that the forest contains trees built with different M values, ranging from 2 to $F-1$. Drawing randomly from $\overrightarrow{RF}$ a number of vectors each denoted as $\overrightarrow{rf_i}$, where the number of trees in $\overrightarrow{rf_i}$ denoted as $n_i \leq N$, $i = 1..S$, and S in the number of random forests. In genetic algorithms terminology S is the size of the population. This initial population is then evolved through a number of generations, with the fitness function for each individual being its classification accuracy.

The following chapters are organised as follows. Section II a background about RA and genetic algorithm GA. The proposed approach *HGARF* is detailed in Section III. Extensive experimental study validating *HGARF* is presented in Section IV. A discussion of related work is given in Section VI. Finally, the paper is concluded with a short summary and pointer to future developments in Section VII.

II. Background

A. Random Forests

In classification which is considered as a supervised learning approach, the target is to identify the value of an attribute, known as the *class attribute*, based on the values of the other attributes in the same instance or record of data. This identification is based on learning from historical data. The attributes other than the class are known as *predictors*. Thus, if the value of the class label is y, and the values of the predictors form the vector $\vec{x}$, then $y = f(\vec{x})$. Any classification technique attempts to find $\hat{f}(\vec{x})$ that approximates the function $f(\vec{x})$.

Instead of using single classifier, ensemble learning uses a set of classifiers to identify unlabelled instances . Boosting and bagging are the two known successful approaches to ensemble learning. Random forests belongs to the bagging approach. *Bagging* (Bootstrap Aggregating) has been proposed by Breiman in [4]. It is based on generating a number replicas from the training data by uniformly sampling the instances with replacement. This sampling approach is known as *Bootstrap*. It allows duplicate instances to appear in the same replica, and also allows some instances to be left out. Statistically for a large replica that has the number of instances equal to the size of the data set, 63.2% of the instances do

Symbol		Definition
N	:	Number of trees in the random forest
F	:	Total number of features in the data set
M	:	Number of features to split on at each node
$\overrightarrow{F_M}$	:	The vector of M features to split on
T_i	:	The i^{th} tree in the random forest
$B(\overrightarrow{F_M})$	:	Best feature to split on
$\overrightarrow{RF}$	:	The vector of all trees in the random forest

TABLE I. RANDOM FORESTS ALGORITHM NOTATION

appear at least once in the replica. Having a number of replicas, each denoted as r out of the training data, a classifier $c(r)$ is built using the sampled instances in r. The classification is done via voting among a vector of classifiers $\overrightarrow{c(r)}$ that have been built using the corresponding vector of replicas $\overrightarrow{r}$.

Bagging has been applied successfully to an ensemble technique, termed *Random Forests*. In addition to the Bootstrap sampling, randomisation over the feature space is also used. The technique is based on building a number of decision tree classifiers, having each tree built from one replica out of the training data. However, when splitting the nodes of the decision tree, only a subset of all the features is used. Assuming that the number of features in the data set is F, the standard setting for the random features to be used at each split is $M = \sqrt{F}$.

The *Random Forests* algorithm is depicted in Algorithm 1. Notation used in the algorithm is listed in Table I. It has been also established empirically that setting the number of trees N in the forest to 100 will yield the best results. However, increasing N beyond 100 mostly will not have much effect on the accuracy neither positively nor negatively.

Algorithm 1 Random Forests Algorithm

{User Settings}
input N, M
{Process}
Create an empty vector $\overrightarrow{RF}$
for $i = 1 \rightarrow NoTrees$ **do**
  Create an empty tree T_i
  repeat
    Sample M out of all features F
    Create a vector of the M features $\overrightarrow{F_M}$
    Find Best Split Feature $B(\overrightarrow{F_M})$
    Create A New Node using $B(\overrightarrow{F_M})$ in T_i
  until No More Instances To Split On
  Add T_i to the $\overrightarrow{RF}$
end for
{Output}
A vector of trees $\overrightarrow{RF}$

B. Genetic Algorithm

GA is a well established evolutionary approach. Basic details about GA could be found in [10] In an ordinary GA, the chromosome represents an encoded solution. For some problems, the direct encoding of a solution in a GA's chromosome results in complex and large chromosomes that may need complex repairs after the application of the GA's operators. In contrast, [13], [18] introduced what could be called as indirect encoding or *Indirect GAs* (IGAs), where each gene in the chromosome represents a heuristic – could be seen as rule of thumb, an educated guess or small rules – instead of representing part of solution. In an *indirect GA*, the chromosome may be much more compact and robust, since it represents the heuristics that will be used in order to get a solution. A chromosome that represents heuristics instead of a is know as a *Heuristic Chromosome* (HC).

The most common form of HC, whether, is one where the HC consists of a number of genes and each of these genes represents the ID of a heuristic. In order to build a solution, the heuristics in an HC are called one after the other or in parallel based on the problem and what exactly the chromosome represents. One of the main differences between different HC approaches lies in structure of the HC and what exactly each gene represents.

The approach adopted in this paper is similar the IGA approach, a single random tree could be considered as a heuristic. Each gene in the chromosome represent a pointer to a random tree classifier, and the chromosome as a whole represent an ensemble classifier (forest). In order to get a solution (classification) of a given instance, the genes in the chromosome are used to evaluate the instance as detailed in section III.

III. HGARF

HGARF is based on *GARF* which was first introduced in [3]. *HGARF* aims to investigate the development of a more diverse forest by generating the trees using different M values. In classical RF, all trees are built using the same M value which is the the number of features used at each split M which is the number of features used at each split M, normally M is set to square root of the number of features F. However, there is no defined way to select the value of M. In this paper, heterogeneous RF means that the forest contains trees built with different M values, ranging from 2 to $F - 1$. This range is chosen as M equals 1 means that the tree is built completely randomly, such a tree could mislead the forest. On the other hand, $M = F$ means that the it is a full tree with no random component which is against the concept of RF and excluded to avoid the risk of over-fitting.

However, how can a RF generated with variable M value? and what is the best M distribution?. To overcome this problem, GA is used to evolve the forest instead of directly generating it.*HGARF* uses variable size chromosome. Each chromosome (individual) in the population represents a forest. Each of the the genes in the chromosome represents a random tree. Traditional genetic operators are employed by the proposed HGARF For the crossover, a standard single point crossover operators is adopted, two modes of operation for the crossover operator have been developed and tested. In the first mode all the repeated genes that could occur because of the crossover in the new individuals are removed, this to make sure that the each evolved forest has no repeated trees. The second mode does not make this extra check and allows the repetition of the trees in the offspring. For the mutation, a standard uniform mutation operator is employed, the operator replaces a randomly chosen tree/gene with another randomly

selected tree from the input trees forest that does not already exist in the forest/individual.

Each dataset is divided into three sets, training, validation (for GA training) and testing. The training set is used for building the random tress (input random forest) with M ranging from 2 to $F-1$. The accuracy of the trees during the training is very high, reaching in most cases above 99% accuracy. By the accuracy here we mean the ability of correctly classifying a given instance. This because these instances have been seen before during the building stage and in random trees does not use burning. As a result, is not possible to use these instances (training set) for training the HGARF as well, and another indebtedness set of instances (Optimization set) is need for training the GA. In this paper we may refer to the validation set as GA-training set.

A. Fitness

Before HGRAF starts the evolution process, each tree in the input forest is used to classify each of the instances in the GA-training set, all the classification results for all the trees on all the instances are stored in a buffer. This is done to speed up the evolution process and especially the fitness evaluation. So in the fitness evaluation of each individual, instead of evaluating the performance of all the trees in the individual against all the instances in the GA-training set, the classification results are collected directly from the buffer.

A given instance is considered as correctly classified, if the number of trees in the individual that has correctly classified it is grater than the number of trees that has given incorrect classification. In contrast, a given instance is considered as incorrectly classified, if the number of trees in the individual that has correctly classified it is less than or equal to the number of trees that has given incorrect classification. We call it a tie, if the number of trees that has correctly classified the instance is equal to the number of the instances that has been incorrectly classified,

The fitness of the individual is based on the number instances he has correctly classified.

$$f(v) = \sum_{i}^{K} c(v,i) + \frac{s(v,i)}{K} \tag{1}$$

where K is the number of instances in the validation set. $c(v,i)$ return 1 if individual v has correctly classified instance number i 0 otherwise. $s(v,i)$ return 1 if it is a tie 0 otherwise. If it is a tie we consider it as an incorrect classification. However, this could mean that the performance of the individual could be improved by small change in the trees combination, and may benefit more from the genetic operators. Therefore, we slightly increase the fitness of the individual by $1/K$ for each tie.

B. HGARF Algorithm

This section provides details about the *HGARF* algorithm. Using the same notation in Table I with the addition of NG representing the number of generations in genetic algorithm and S denoting the size of the population (number of individual random forests), the algorithm is depicted in Algorithm 2.

Algorithm 2 HGARF Algorithm

{User Settings}
input N, S, NG
{Process}
for $i = 2 \rightarrow F-1$ **do**
  $\overrightarrow{RF}$ = $\overrightarrow{RF}$ + Call Random Forest($N/(F-2)$, i) {This will generate initial RF with equally distributed M value }
end for
for $i = 1 \rightarrow S$ **do**
  $x = Randomise(1 \rightarrow N)$
  Add tree T_x to $\overrightarrow{rf_i}$
end for
for $j = 1 \rightarrow NG$ **do**
  Apply GA
  $b \leftarrow$ index of best $\overrightarrow{rf_i}$
end for
{Output}
A vector of trees $\overrightarrow{rf_b}$

IV. Experimental Study

A series of experiments has been conducted to evaluate the performance of *HGARF* against the *GARF* and the state of the art classification techniques. For the experiments, Waikato Environment for Knowledge Analysis *(WEKA)* [20] is used. The performance of *HGARF* is compared against the state of the art classification techniques; C4.5 decision tree [14], Support Vector Machines *(SVM)* [19] and AdaBoost [7]. We have used also WEKA to build the random forest, we denote this as *RFweka*.

The experiments are conducted on 15 real standard data sets from UCI repository [1]. Description of the used data sets is given in Table III. A variety of data sets with diversity in the number of instances, number of classes and number of attributes is used.

The data sets is divided into three equal parts; one third for training, one third for optimisation (validation), and one third for testing, as shown in Figure 1. In *HGARF*, the validation part is used to evolve initial random forests. To conduct fair experiments, the training and validation parts of the data sets are combined and used as the training for the other methods. The same testing set is used to calculate the performance of all the classifiers.

The main results of the the experiments are shown in Table III. Due to limited size, experiments with different *HGARF* settings. The table shows our *HGARF* technique is very competitive with the other methods.

To assess the robustness of *HGARF* with varying the experimental settings of genetic algorithm, we have conducted as set of experiments with different settings. These settings are available in Table IV. The corresponding results are presented in Table V. It can be noted that *HGARF* has proved to be robust with the various setting of parameters, achieving consistently good accuracy over all the data sets used in our experimental study. Also, Table IV shows the performance of *HGARF* using the same experimental settings.

Demonstrating the evolved RF confidence over the initial

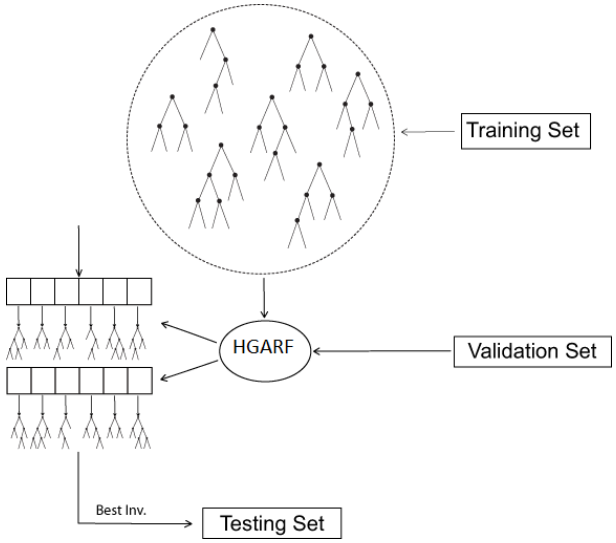

Fig. 1. Experimental Setup

DS Name (No. Ins.)	HGARF	GARF	RFweka	AdaBoost	C 4.5	SVM
diabetes (256)	79.83	78.52	75.39	80.08	76.56	79.30
glass (71)	77.10	71.83	76.06	33.80	59.16	47.89
ionosphere (116)	97.24	96.55	92.24	91.38	93.10	91.38
iris (50)	95.08	96.00	94.00	96.00	96.00	90.00
labor (18)	96.11	94.44	77.78	83.33	77.78	83.33
soybean (227)	87.55	85.46	87.23	32.60	83.70	N/A
vote (145)	99.30	96.55	98.62	99.31	97.24	7.24
credit-g (333)	74.14	73.87	72.67	68.47	69.67	71.47
ecoli (113)	70.51	71.68	69.03	24.78	68.14	61.06
letter (6667)	86.64	84.01	N/A	N/A	N/A	N/A
liver-disorders (116)	71.30	69.83	68.97	59.49	61.21	57.76
sonar (69)	87.97	88.41	81.16	76.81	76.81	84.06
vehicle (282)	75.31	73.76	74.82	38.65	65.96	66.31
vowel (330)	79.78	74.55	80.00	15.76	65.15	51.52
waveform-500 (1667)	85.98	85.18	85.00	73.00	74.15	86.08

TABLE III. PERFORMANCE OF *HGARF* AGAINST *GARF* AND THE STATE-OF-THE-ART TECHNIQUES

Name	Ins	Class.	Train.	Valid.	Test.
diabetes	768	2	256	256	256
glass	214	7	71	72	71
ionosphere	351	2	118	117	116
iris	150	3	50	50	50
labor	57	2	20	19	18
soybean	683	19	228	228	227
vote	435	2	145	145	145
credit-g	1000	2	333	334	333
ecoli	336	8	110	113	113
letter	20000	17	6666	6667	6667
liver-disorders	345	7	113	116	116
sonar	208	61	69	70	69
vehicle	846	19	282	282	282
vowel	990	14	330	330	330
waveform-500	5000	41	1665	1668	1667

TABLE II. DATA SETS USED

Experiment	PSize	NG	CR	MR	ChLength	Var.
Experiment 1	100	50	0.9	0.1	100	YES
Experiment 2	100	50	0.9	0.1	200	NO
Experiment 3	500	50	0.9	0.1	100	YES
Experiment 4	400	50	0.9	0.1	400	YES

TABLE IV. THE PARAMETERS USED IN THE EXPERIMENTS. PSIZE: POPULATION SIZE, NG: NUMBER OF GENERATIONS, CR: CROSSOVER RATE, MR: MUTATION RATE, CHLENGTH: CHROMOSOME LENGTH, VAR: YES IS THE SIZE OF THE CHROMOSOME IS VARIABLE AND NO IF THE SIZE OF THE CHROMOSOME IS FIXED.

random forest (*RFin*), we have conducted a series of experiments reporting the percentage of trees voted in the random forest, contributing to the correct classification. In Figures 2 and 3, the x-axis represents the instances in the testing data, and the y-axis represents the percentage of trees voted to the correct class. It can be easily seen that for the *Glass* and *Sonar* data sets, The evolved RF has had in most cases higher confidence in finding the correct class labels.

V. DISCUSSION

Given the empirically validated robustness of random forests against noise, it is suitable to address the problem of changing data, known as concept drift. *HGARF* can address

Name	EXP1 pop 100 size var 100 GARF	HGARF	Size	Std.	EXP2 pop 100 inv200f GARF	HGARF	Size	Std.
diabetes	78.52	79.83	85	3.14	77.73	78.31	200	3.34
glass	71.83	77.1	82	3.8	73.24	74.56	200	4.26
ionosphere	96.55	97.24	82	4.45	94.83	95.42	200	2.89
iris	96	95.08	98	6.25	96	97.51	200	4.73
labor	94.44	96.11	98	4.22	94.44	97.95	200	3.16
soybean	85.46	87.55	92	4.21	84.14	82.2	200	4.09
vote	96.55	99.3	90	1.2	96.55	98.41	200	4.33
credit-g	73.87	74.14	97	2.06	73.57	78.03	200	3.78
ecoli	71.68	70.51	94	4.28	70.8	72.86	200	2.94
letter	84.01	86.64	95	4.96	83.77	87.67	200	2.63
liver-disorders	69.83	71.3	88	3.7	68.97	70.95	200	2.99
sonar	88.41	87.97	92	5.24	88.41	86.53	200	3.78
vehicle	73.76	75.31	86	3.37	73.76	74.75	200	2.76
vowel	74.55	79.78	100	5.8	74.24	77.57	200	3.35
waveform-500	85.18	85.98	94	5.02	84.82	81.75	200	3.69

Name	EXP3 pop 500 GARF	HGARF	Size	Std.	EXP4 pop 500 inv 400 GARF	HGARF	Size	Std.
diabetes	78.52	77.96	89	3.27	78.13	80.43	159	3.94
glass	70.42	71.03	91	3.05	70.42	74.3	187	3.48
ionosphere	96.55	97.82	174	4.55	95.69	96.27	237	4.35
iris	96	98.33	184	3.05	96	95.27	173	3.47
labor	94.44	92.78	93	2.81	94.44	95.64	159	2.79
soybean	88.55	88.86	74	3.12	85.9	88.79	172	4.42
vote	96.55	98.53	83	4.04	97.24	93.35	176	4.69
credit-g	74.77	75.97	76	2.91	73.87	75.48	162	3.4
ecoli	69.03	70.82	193	3.59	70.8	71.93	261	2.12
letter	84.36	85.61	70	3.58	83.97	84.25	170	3.74
liver-disorders	69.83	68.38	94	3.12	69.83	70.33	187	3.75
sonar	88.41	88.92	90	4.57	88.41	91.94	129	3.67
vehicle	73.76	74.83	66	3	73.76	75.45	173	3.78
vowel	75.15	75.62	179	3.19	74.55	71.08	261	2.91
waveform-500	85.42	82.57	80	4.36	85.24	88.44	157	3.97

TABLE V. PERFORMANCE OF *HGARF* WITH VARYING EXPERIMENTAL SETTINGS. THE COLUMNS *GARF* AND *HGARF* SHOW THE BEST RESULTS FOR *GARF* AND *HGARF* RESPECTIVELY. THE FOLLOWING COLUMNS SHOW THE SIZE OF THE FOREST AND THE STANDARD DEVIATION FOR *HGARF* . RESULT FOR *GARF*,

this issue, because of the natural evolution of genetic algorithm. However, an important issue is required to be addressed. Sudden and strong concept drift require new trees to be added to the forest, on one hand. On the other hand, gradual and weak concept drift can easily utilise genetic algorithm to use existing trees in the random forest. Extensions to *HGARF* to address this issue would also have a great potentil in the data stream mining area [9], [8].

Moreover, *GARF HGARF* adopted the accuracy of the individual random forests as the fitness function. Exploring the use of ensemble diversity [11] as another fitness function may lead to further boosting the performance of *HGARF*. The argument is that both the accuracy of the ensemble and its diversity when used together could lead to the optimal ensemble classification. Further investigations is this area are planned.

VI. RELATED WORK

Genetic algorithm has been applied in machine learning and data mining extensively. The main application is the use of genetic algorithm in the feature selection problem. An early survey on this topic could be found in [12]. However, the relevant work to the research reported in this paper is detailed in the following.

Robnik-Sikonja [15] has proposed possible extensions to random forests that have proved to boost the accuracy of the original techniques presented in Section II. The motivation behind these extensions is to decrease the correlation among the trees in the random forest. As the original technique proposed by [5] uses *Gini index* for finding the best split among the randomised vector of attributes $\overrightarrow{F_M}$. In an attempt to decrease the dependency among attributes, Robnik-Sikonja has used ReliefF [16] as a measure of the quality of the attributes. This extension has not proved to have a good peformance on real data sets. A combination of measures for the quality of attributes has been used to decide the split, having each fifth tree in the forest uses a different measure. This method has proved to boost the performance of the random forest, but not significantly. The other approach proposed by Robnik-Sikonja was the use of weighted voting among the trees using similarity of the instances with regards to their performance on the individual trees. This method has proven to always boost the performance, or at least being as good as the performance of the original the random forests technique over a number of real data sets.

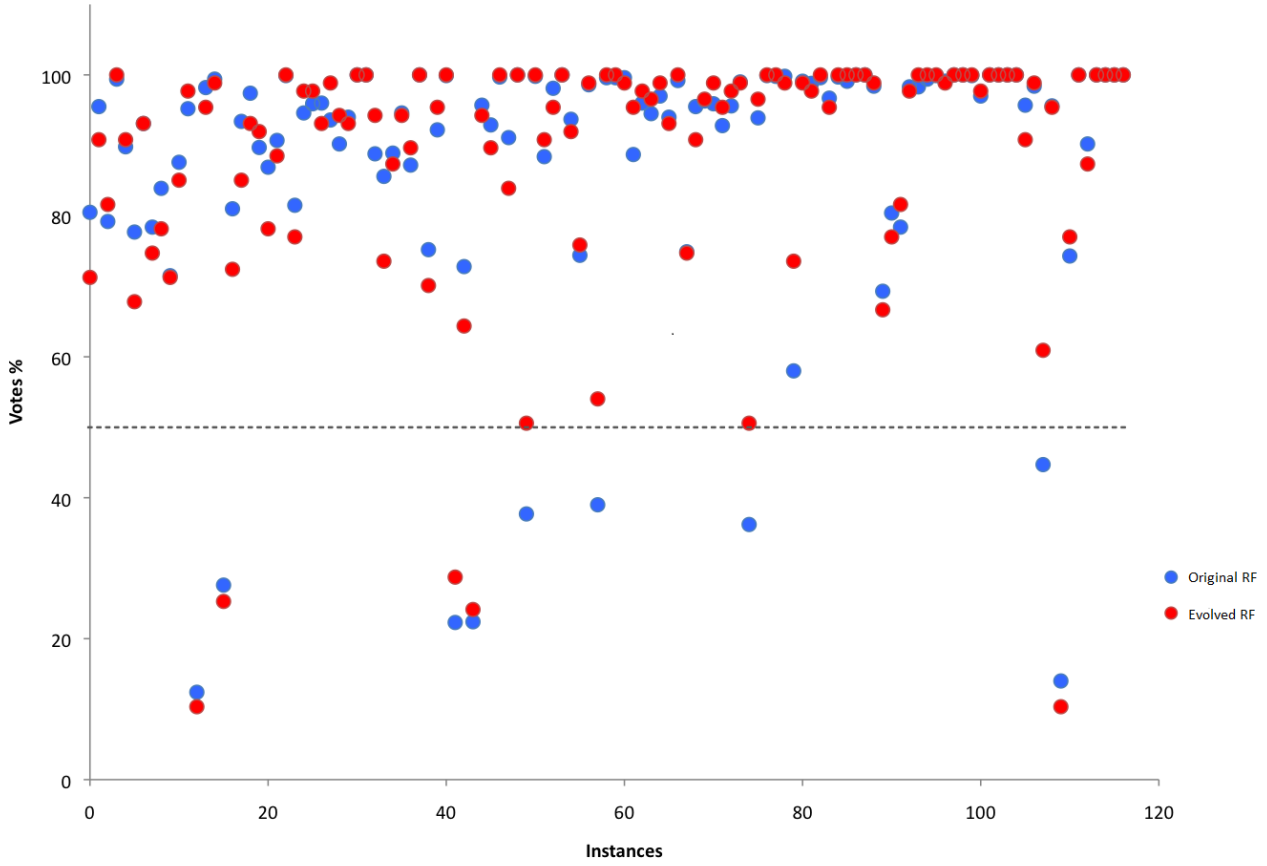

Fig. 2. The evolved RF versus the input RF in Voting Confidence: The Glass2 Data Se

Sylvester and Chawla [17] have proposed the *EVEN* (EVolutionary ENsembles). They have also attempted to use weighted voting among a set of homogeneous or heterogeneous classifiers. The *EVEN* system uses each of the classifier's performance over a validation set of data to weight the tree. Experimental validation has proved that *EVEN* can outperform the unweighted ensemble.

In a more recent work, Abdulsalam and Skillicorn [2] have used Hoeffding trees [6] to build a window of random forests to tackle the concept drift problem when mining streaming data.

VII. Conclusion and Future Work

This paper introduced a novel algorithm for evolving heterogeneous Random Forests, the forest is optimized using Genetic Algorithm. The algorithm is based on a previously introduced algorithm *GARF* that used GA for optimizing the size of the forest only. The approach is based on generating a large random forest using different number of attributes split for each tree M, that is later is decomposed into a number of smaller random forests. The smaller forests are composed of trees drawn randomly with replacement from the initial large random forest. Genetic algorithm is an optimisation technique is then applied to evolve this initial population of individual random forests with the fitness function being the classification of the forest.

References

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

[2] H. Abdulsalam, D. B. Skillicorn, and P. Martin. Classification using streaming random forests. *IEEE Trans. Knowl. Data Eng.*, 23(1):22–36, 2011.

[3] M. B. Bader-El-Den and M. M. Gaber. Garf: Towards self-optimised random forests. In *ICONIP (2)*, volume 7664 of *Lecture Notes in Computer Science*, pages 506–515. Springer, 2012.

[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] P. Domingos and G. Hulten. Mining high-speed data streams. In *KDD*, pages 71–80, 2000.

[7] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1995.

[8] M. M. Gaber, A. B. Zaslavsky, and S. Krishnaswamy. Mining data streams: a review. *SIGMOD Record*, 34(2):18–26, 2005.

[9] M. M. Gaber, A. B. Zaslavsky, and S. Krishnaswamy. A survey of classification methods in data streams. In C. C. Aggarwal, editor, *Data Streams - Models and Algorithms*, volume 31 of *Advances in Database Systems*, pages 39–59. Springer, 2007.

[10] R. L. Haupt and S. E. Haupt. *Practical Genetic Algorithms with CD-ROM*. Wiley-Interscience, 2004.

[11] L. I. Kuncheva. Using diversity measures for generating error-correcting output codes in classifier ensembles. *Pattern Recognition Letters*, 26(1):83–90, 2005.

[12] M. Martin-Bautista and M.-A. Vila. A survey of genetic feature selection in mining issues. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 2, pages 3 vol. (xxxvii+2348), 1999.

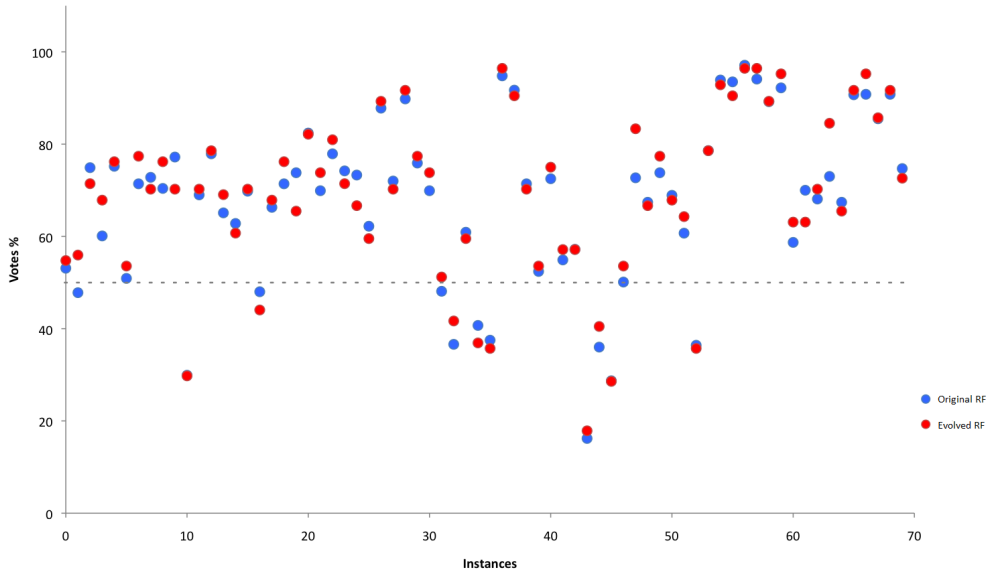

Fig. 3. The evolved RF versus the input RF in Voting Confidence: The Sonar Data Set

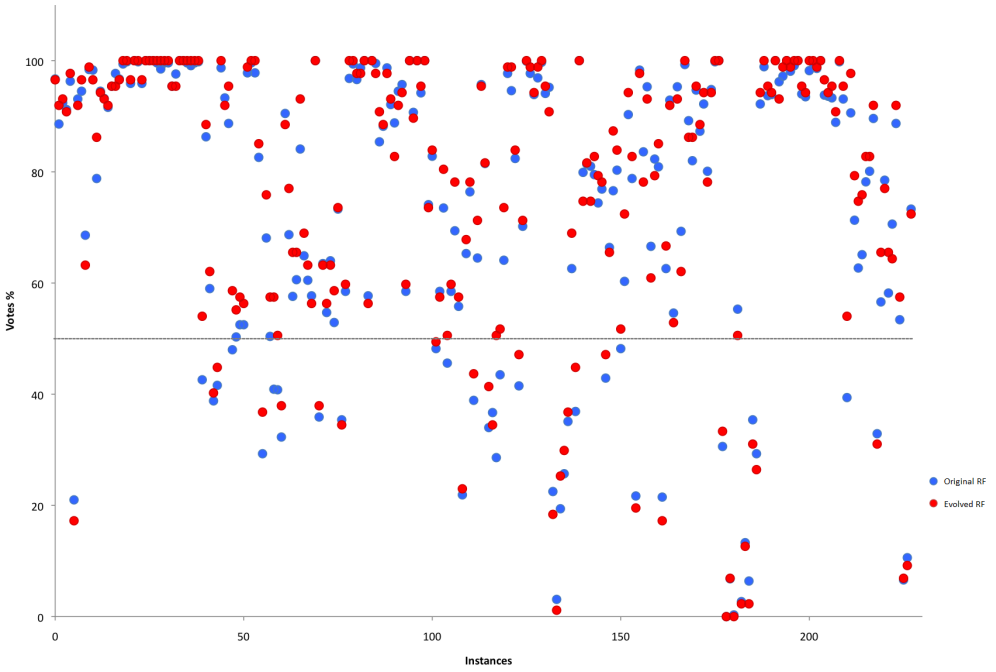

Fig. 4. The evolved RF versus the input RF in Voting Confidence: The Soybean Data Set

[13] I. Norenkov. Scheduling and allocation for simulation and synthesis of cad system hardware. In *In Proceedings EWITD 94, East-West International Conference, ICSTI*, pages 20–24, Moscow, 1994.

[14] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[15] M. Robnik-Sikonja. Improving random forests. In *ECML*, pages 359–370, 2004.

[16] M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Mach. Learn.*, 53:23–69, October 2003.

[17] J. Sylvester and N. Chawla. Evolutionary ensemble creation and thinning. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 5148–5155. IEEE, 2006.

[18] H. Terashima-Marin, P. Ross, and M. Valenzuela-Rendon. Evolution of constraint satisfaction strategies in examination timetabling. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 635–642, Orlando, Florida, USA, 13-17 July 1999. Morgan Kaufmann.

[19] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[20] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers, San Francisco, CA, 2nd edition, 2005.