Hybrid Random Forests: Advantages of Mixed Trees in Classifying Text Data

Baoxun Xu[1], Joshua Zhexue Huang[2], Graham Williams[2], Mark Junjie Li[2], and Yunming Ye[1]

[1] Department of Computer Science, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 518055, China
[2] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China
amusing002@gmail.com, zx.huang@siat.ac.cn, Graham.Williams@togaware.com, jj.li@siat.ac.cn, yeyunming@hit.edu.cn

Abstract. Random forests are a popular classification method based on an ensemble of a single type of decision tree. In the literature, there are many different types of decision tree algorithms, including C4.5, CART and CHAID. Each type of decision tree algorithms may capture different information and structures. In this paper, we propose a novel random forest algorithm, called a hybrid random forest. We ensemble multiple types of decision trees into a random forest, and exploit diversity of the trees to enhance the resulting model. We conducted a series of experiments on six text classification datasets to compare our method with traditional random forest methods and some other text categorization methods. The results show that our method consistently outperforms these compared methods.

Keywords: Random Forests, Hybrid Random Forest, Classification, Decision Tree.

1 Introduction

Random forests [1,2] are a popular classification method which builds an ensemble of a single type of decision tree. The decision trees are often either built using C4.5 [3] or CART [4], but only one type was exploited within a single random forest. In recent years, random forests have attracted increasing attention due to (1) its competitive performance compared with other classification methods, especially for high-dimensional data, (2) algorithmic intuitiveness and simplicity, and (3) its most important capability - "ensemble" using bagging [5] and stochastic discrimination [2].

The most popular forest construction procedure was proposed by Breiman [1], novelly using bagging to generate training data subsets for building individual trees. A subspace of features is then randomly selected at each node to grow branches of a decision tree. The trees are then combined as an ensemble into a random forest [1].

P.-N. Tan et al. (Eds.): PAKDD 2012, Part I, LNAI 7301, pp. 147–158, 2012.

In the literature, different types of decision trees algorithms have been proposed, including C4.5, CART and CHAID [6]. Each type of decision tree algorithms employs a different tree building process and captures different discriminative information.

Random forests gain some of their performance advantage through the diversity of the trees in the resulting ensemble. We can add another kind of diversity to the random forest framework by removing any potential bias from using a single type of decision tree. We propose to use several different types of decision trees for each training data subset, and select the best tree as the individual tree classifier in the random forest model.

Our method is motivated by the experiences of foresters in dealing with the development and care of hybrid forests. An important concept in Forestry is that of a "hybrid forest." Such a forest uses multiple tree species as a mixed planting in accordance with soil structure (moisture, nutrients, acidity). This method has demonstrated highly economic, ecological and practical value in forestry research. Mimicing this idea, we have developed a hybrid random forest method to explore whether we can further enhance the classification performance of a random forest ensemble classifier. Specifically, we build three different types of tree classifiers (C4.5, CART and CHAID) for each training data subset. We then evaluate the performance of the three classifiers and select the best tree. In this way, we build a hybrid random forest which may include different types of decision trees in the ensemble. The added diversity of the decision trees can effectively improve the accuracy of each tree in the forest, and hence the accuracy of the ensemble.

To demonstrate the effectiveness of our proposed method, we apply it to the popular application of text classification. With the ever-increasing volume of text data from the Internet, databases, and archives, text categorization has become a key technique for handling and organizing text data. It has received growing attention in recent years. A set of popular and mature machine learning approaches have been deployed for categorizing text documents, including random forests [8], support vector machines (SVM) [9], naive Bayes (NB) [10], k-nearest neighbors (KNN) [11], and decision trees. Due to algorithmic simplicity and prominent classification performance for high dimensional data, random forests have become a preferred method.

In this paper, we compare the performance of our random forest with that of other three random forest methods, i.e., C4.5 random forest, CART random forest and CHAID random forest, and other three mainstream text categorization methods, i.e., support vector machines, naive Bayes and k-nearest neighbors, on six datasets. The experimental results show that our hybrid random forest achieves improved classification performance over these six compared methods.

The rest of this paper is organized as follows. Section 2 introduces the framework for building a hybrid Random Forest, and gives a brief analysis of the method. The evaluation methods are presented in Section 3, we present experimental results in Section 4. Our conclusions and future work are presented in Section 5.

2 Hybrid Random Forests

In this section, we introduce a general framework for building hybrid random forests. We then briefly review three types of decision tree algorithms, i.e., C4.5, CART and CHAID. We also present our hybrid random forest algorithm that integrates the best trees from the different types of decision tree algorithms.

2.1 Framework for Building Hybrid Random Forest

As an ensemble learner, the performance of a random forest is highly dependent on two factors: the diversity among the trees and the accuracy of each tree [12]. Diversity is commonly obtained by using bagging and random subspace sampling. We introduce a further element of diversity by using different types of trees.

Continuing our analogy with forestry, the different data subsets from bagging represents the "soil structures." Different decision tree algorithms represent "different tree species". Our approach has two key aspects: one is to use three types of decision tree algorithms to generate three different tree classifiers for each training data subset; the other is to evaluate the accuracy of each tree as the measure of tree importance. In this paper, we use the out-of-bag accuracy to assess the importance of a tree.

Following Breiman, we use bagging to generate a series of training data subsets from which we build trees. For each tree, the data subset used to grow the tree is called the "in-of-bag" (IOB) data and the remaining data subset is called the "out-of-bag" (OOB) data. Since OOB data is not used for building trees we can use this data to objectively evaluate each tree's accuracy and importance. The OOB accuracy gives an unbiased estimate of the true accuracy of a model.

Given n instances in a training dataset D and a tree classifier $h_k(IOB_k)$ built from the k'th training data subset IOB_k, we define the OOB accuracy of the tree $h_k(IOB_k)$ for each $di \in D$ as:

$$OOBAcc_k = \frac{\sum_{i=1}^{n} I(h_k(d_i) = y_i; d_i \notin IOB_k)}{\sum_{i=1}^{n} I(d_i \notin IOB_k)} \quad (1)$$

where I(.) is an indicator function. The larger the $OOBAcc_k$, the better classification quality a tree has.

We use the out-of-bag data subset OOB_i to calculate the out-of-bag accuracies of the three types of trees (C4.5, CART and CHAID) with evaluation values A_1, A_2 and A_3 respectively.

Fig. 1 illustrates the procedure for building a hybrid random forest model. Firstly, a series of IOB/OOB datasets are generated from the entire training dataset by bagging. Then, three types of tree classifiers (C4.5, CART and CHAID) are built using each IOB dataset. The corresponding OOB dataset is used to calculate the OOB accuracies of the three tree classifiers. Finally, we select the tree with the highest OOB accuracy as the final tree classifier, which is included in the hybrid random forest.

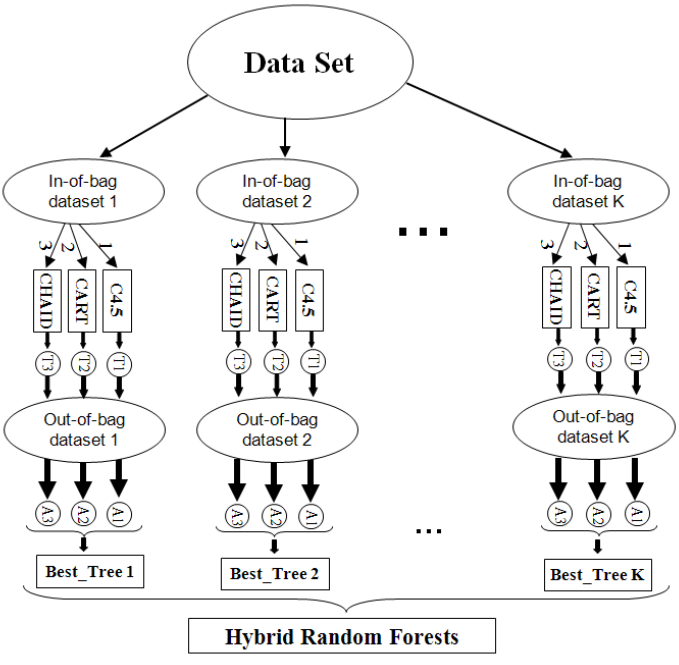

Fig. 1. The Hybrid Random Forests framework

Building a hybrid random forest model in this way will increase the diversity among the trees. The classification performance of each individual tree classifier is also maximized.

2.2 Decision Tree Algorithms

The core of our approach is the diversity of decision tree algorithms in our random forest. Different decision tree algorithms grow structurally different trees from the same training data. Selecting a good decision tree algorithm to grow trees for a random forest is critical for the performance of the random forest. Few studies have considered how different decision tree algorithms affect a random forest. We do so in this paper.

The common decision tree algorithms are as follows:

Classification Trees 4.5. (C4.5) is a supervised learning classification algorithm used to construct decision trees. Given a set of pre-classified objects, each described by a vector of attribute values, we construct a mapping from attribute values to classes. C4.5 uses a divide-and-conquer approach, which is similar to recursive partitioning, to grow decision trees. C4.5 selects the test that maximizes the information gain ratio (IGR) [3].

Classification and Regression Tree. (CART) is a recursive partitioning method that can be used for both regression and classification. Beginning with the entire dataset, a tree is constructed by splitting subsets of the dataset by

considering all predictor variables for splitting. The best predictor is chosen at each node using a variety of impurity or diversity measures. The goal is to produce subsets of the data which are homogeneous with respect to the target variable [4]. The main difference between C4.5 and CART is the test selection and evaluation process.

Chi-squared Automatic Interaction Detector. (CHAID) method is based on the chi-square test of association. A CHAID decision tree is constructed by repeatedly splitting subsets of the space into two or more nodes. To determine the best split at any node, any allowable pair of categories of the predictor variables is merged until there is no statistically significant difference within the pair with respect to the target variable [6,7].

From these decision tree algorithms, we can see that the difference lies in the way to split a node, such as the split functions and binary branches or multi-branches. In this work we use these different decision tree algorithms to build a hybrid random forest.

2.3 Algorithm

In this subsection, we present our hybrid random forest algorithm which integrates the three types of tree classifiers. The detailed steps are introduced in

Algorithm 1.

In Algorithm 1, lines 10-17 loop to build K decision trees. In the loop, Line 11 samples the training data D by sampling with replacement to generate an in-of-bag data subset IOB_i for building a decision tree. Lines 12-15 build three types of tree classifiers (C4.5, CART and CHAID). In this procedure, Line 13 calls the function $createTree_j()$ to build a tree classifier. Line 14 calculates the out-of-bag accuracy of the tree classifier. After this procedure, Line 16 selects the tree classifier with the maximum out-of-bag accuracy. K decision trees are thus generated to form a hybrid random forest model $\mathscr{M}$.

Generically, function $createTree_j()$ first creates a new node. Then, it tests the stopping criteria to decide whether to return to the upper node or to split this node. If we chose to split this node, then we randomly select m features as a subspace for node splitting. These features are used as candidates to generate the best split to partition the node. For each subset of the partition, $createTree_j()$ is called again to create a new node under the current node. If a leaf node is created, it returns to the parent node. This recursive process continues until a full tree is generated.

3 Evaluation Methods

We use two measures to evaluate the classification performance of the hybrid random forest, the test accuracy and the F1 metric. The test accuracy measures the performance of a random forest on a separate test dataset. The F1 metric is a commonly used measure of classification performance.

Algorithm 1. Hybrid Random Forest Algorithm

Input:
- D : the training dataset,
- A : the features space $\{A_1, A_2, ..., A_M\}$,
- Y : the class features space $\{y_1, y_2, ..., y_q\}$,
- K : the number of trees,
- m : the size of subspaces.

Output: A random forest $\mathscr{M}$;
Method:
for $i = 1$ *to* K **do**
  draw a bootstrap sample in-of-bag data subset IOB_i and out-of-bag data subset OOB_i from training dataset D;
  for $j = 1$ *to* 3 **do**
    $h_{i,j}(IOB_i) = createTree_j()$;
    use out-of-bag data subset OOB_i to calculate the out-of-bag accuracy $OOBAcc_{i,j}$ of the tree classifier $h_{i,j}(IOB_i)$ by Equation (1);
  end for
  select $h_i(IOB_i)$ with the highest out-of-bag accuracy $OOBAcc_i$ as best tree i;
end for
combine the K optimal tree classifiers $h_1(IOB_1), h_2(IOB_2), ..., h_K(IOB_K)$ into a random forest $\mathscr{M}$

Function createTree()
create a new node $\mathscr{N}$;
if stopping criteria is met **then**
  return $\mathscr{N}$ as a leaf node;
else
  randomly select m features as a subspace;
  use these m features as candidates to generate the best split for the node to be partitioned;
  call createTree() for each split;
end if
return $\mathscr{N}$;

Test Accuracy. Let D_t be a test dataset and Y_t be the class labels. Given $d_i \in D_t$, the number of votes for d_i on class j is

$$N(d_i, j) = \sum_{k=1}^{K} I(h_k(d_i) = j) \tag{2}$$

The test accuracy is calculated as

$$Acc = \frac{1}{n}\sum_{i=1}^{n} I(N(d_i, y_i) - \max_{j \neq y_i} N(d_i, j) > 0) \tag{3}$$

where n is the number of objects in D_t and y_i indicates the true class of d_i.

F1 Metric. To evaluate the performance of classification methods in dealing with an unbalanced class distribution, we use the F1 metric introduced by Yang and Liu [13]. This measure is equal to the harmonic mean of recall (α) and precision (β). The overall F1 score of the entire classification problem can be computed by a micro-average and a macro-average.

Micro-averaged F1. This is computed globally over all classes, and emphasizes the performance of a classifier on common classes. Define α and β as follows:

$$\alpha = \frac{\sum_{i=1}^{q} TP_i}{\sum_{i=1}^{q}(TP_i + FP_i)}, \quad \beta = \frac{\sum_{i=1}^{q} TP_i}{\sum_{i=1}^{q}(TP_i + FN_i)} \tag{4}$$

where q is the number of classes. TP_i (True Positives) is the number of objects correctly predicted as class i, FP_i (False Positives) is the number of objects that are predicted to belong to class i but do not. The micro-averaged F1 is computed as:

$$MicroF1 = \frac{2\alpha\beta}{\alpha + \beta} \tag{5}$$

Macro-averaged F1. This is first computed locally over each class, and then the average over all classes is taken. It emphasizes the performance of a classifier on rare categories. Define α and β as follows:

$$\alpha_i = \frac{TP_i}{(TP_i + FP_i)}, \quad \beta_i = \frac{TP_i}{(TP_i + FN_i)} \tag{6}$$

$F1$ for each category i and the macro-averaged F1 are computed as:

$$F1_i = \frac{2\alpha_i\beta_i}{\alpha_i + \beta_i}, \quad MacroF1 = \frac{\sum_{i=1}^{q} F1_i}{q} \tag{7}$$

The larger the MicroF1 and MacroF1 values are, the better the classification performance of the classifier.

4 Experiments

In this section, we conduct experiments to demonstrate the effectiveness of the hybrid random forest algorithm for classifying text data. Text datasets with various sizes and characteristics are used in the experiments. The experimental results show that the hybrid random forest algorithm not only outperforms single-tree type random forest algorithms, i.e., C4.5_RF, CART_RF and CHAID_RF, in classification accuracy, but also outperforms other three mainstream text categorization methods, i.e., SVM, NB and KNN.

4.1 Datasets

In the experiments, we used six real-world text datasets. These text datasets are selected due to their diversities in the number of terms or features, the number of documents, and the number of classes. Their dimensionalities vary from 2000 to 11,465, numbers of instances vary from 918 to 11,162 and the minority class rate varies from 0.32% to 6.43%. In each text dataset, we randomly select 70% of documents as the training dataset, and the remaining data as the test dataset. Detailed information of the six text datasets is listed in Table 1.

Table 1. Summary statistic of 6 text datasets

Dataset	#Terms	#Documents	#Classes	%Minority class
Fbis	2000	2463	17	1.54
Re0	2886	1504	13	0.73
Oh5	3012	918	10	6.43
Re1	3758	1657	25	0.6
Wap	8460	1560	20	0.32
Ohscal	11465	11162	10	6.35

These datasets are frequently used as text document classification benchmark data [14]. Dataset **Fbis** was compiled from Foreign Broadcast Information Service TREC-5 [15]. The datasets **Re0** and **Re1** were selected from Reuters-21578 text categorization test collection Distribution 1.0 [16]. Datasets **Oh5** and **Ohscal** are from the OHSUMED subset of the MEDLINE database [17]. **Wap** is from the WebACE project (WAP) [18].

4.2 Test Accuracy Improvement

The purpose of this experiment is to evaluate the effect of the hybrid random forest method on accuracy. The six text datasets were analyzed and results were compared with other three random forest methods (C4.5_RF, CART_RF and CHAID_RF). For each text dataset, we ran each random forest algorithm against different sizes of feature subspaces. Since the number of features in these datasets was very large, we started with a subspace of 15 features and increased the subspace with 5 more features each time. For a given subspace size, we built 100 trees for each random forest model. In order to obtain a stable result, we built 80 random forest models for each subspace size, each dataset and each algorithm, and computed the averages of the test accuracy as the final result for comparison.

Fig. 2 shows the plots of the average test accuracy of the four random forest models in different sizes generated with the four methods from the six text datasets. For the same number of features, the higher the accuracy, the better the result. From these figures, we can observe that the hybrid random forest

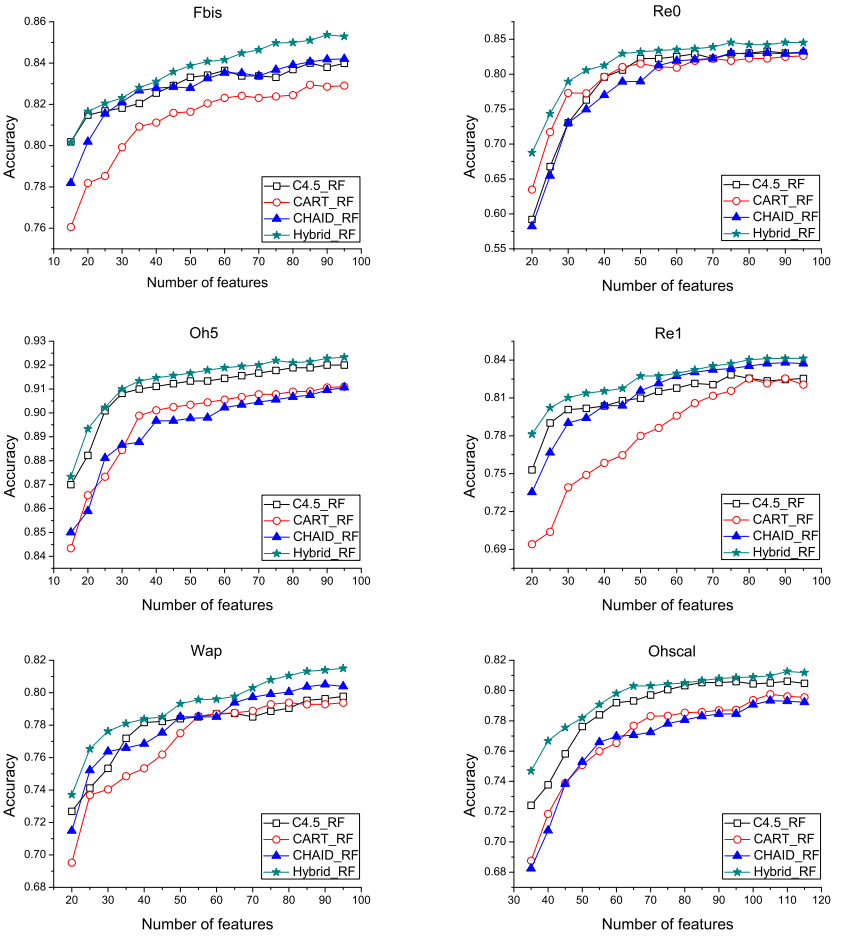

Fig. 2. Test accuracy changes against the number of features in the subspace on the 6 text datasets

algorithm consistently performs better than the other three random forest algorithms. The advantages are more obvious in the smaller subspaces. The hybrid random forest algorithm quickly achieves high accuracy as the subspace size increases. The other three random forest algorithms require larger subspaces to achieve a similar accuracy. These results illustrate that the hybrid random forest algorithm outperforms the other three random forest algorithms in the classification accuracy results on all the six text datasets.

To further investigate the performance of the hybrid random forest, we computed the average accuracy of the trees in each single-type random forest. This is compared to the average accuracy of the trees of the same type within the one hybrid random forest. In all comparisons, the subspace size of $\sqrt{M}$ features

Table 2. A comparison of the average accuracy of trees within a single-type random forest and the average accuracy of trees of that same type within the hybrid random forest

Name	C4.5		CART		CHAID	
	C4.5_RF	Hybrid_RF	CART_RF	Hybrid_RF	CHAID_RF	Hybrid_RF
Fbis	0.6379	**0.6489**	0.6102	**0.6414**	0.6382	**0.6536**
Re0	0.6132	**0.6324**	0.6271	**0.6478**	0.6171	**0.6304**
Oh5	0.6516	**0.6664**	0.6134	**0.6515**	0.6245	**0.6607**
Re1	0.6611	**0.6793**	0.6058	**0.6608**	0.6516	**0.6718**
Wap	0.5267	**0.5343**	0.5086	**0.5396**	0.5195	**0.5297**
Ohscal	0.5391	**0.5448**	0.4732	**0.5004**	0.4665	**0.5204**

was used, where M is the total number of features in the dataset. The results are shown in Table 2. For example, for tree type C4.5 and dataset Fbis, the average accuracy of all trees from the random forest built using C4.5 (named as C4.5_RF) is 0.6379. The average accuracy of all C4.5 trees from the hybrid random forest (named as Hybrid_RF) is 0.6489. It is clearly seen in Table 2 that tree classifiers of any given type in our hybrid random forest always have higher average classification accuracy than those using only trees of the same one type.

4.3 Performance Comparisons of other Text Classification Method

We conduct a further experimental comparison against other three widely used text categorization methods, i.e., support vector machines (SVM), Naive Bayes (NB), and k-nearest neighbor (KNN). The SVM uses a linear Kernel with a regularization parameter of 0.03125, which is often used in text categorization. For Naive Bayes, we adopted the multi-variate Bernoulli event model that is frequently used in text classification [19]. For k-nearest neighbor (KNN), we set the number of neighbors as 13. In the experiments, we use WEKA's implementation for these three text classification methods [20]. We use a single subspace size of 90 features in all the six datasets to run the random forest algorithms, which provides a consistent result as shown in Fig. 2. In order to obtain stable results, we built 20 random forest models for each random forest algorithm and each dataset, and present the average results, we can see that the range of values are less than ± 0.005 and the hybrid trees are always more accurate.

The comparison results are listed in Fig. 3, 4 and 5. While the improvement is often quite small, there is always an improvement demonstrated. We observe that our proposed method always outperforms the compared mainstream text categorization methods.

5 Conclusion and Future Work

We have presented a new hybrid random forest algorithm which increases diversity amongst the ensemble of trees by choosing different tree algorithms. We

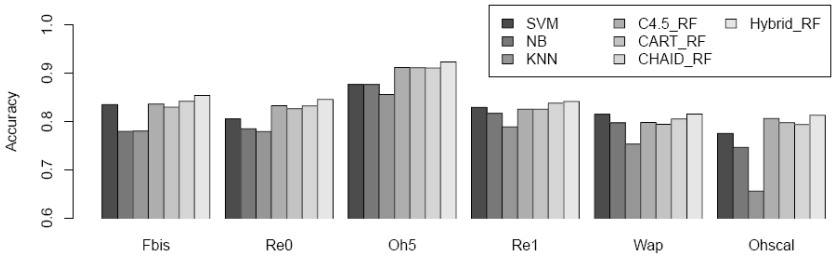

Fig. 3. Accuracy of the seven model builders

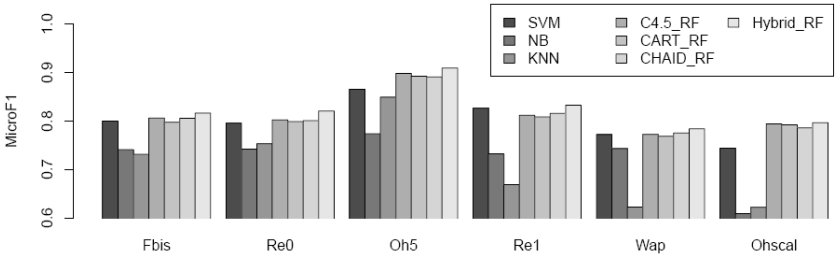

Fig. 4. MicroF1 for the seven model builders

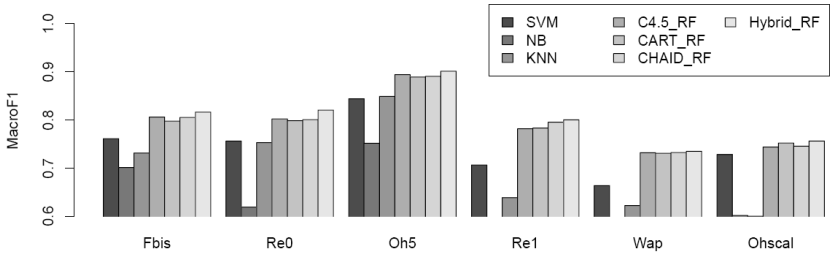

Fig. 5. MacroF1 for the seven model builders

demonstrate the advantage of our method in categorization. Our algorithm consistently improves classification performance. In the future work, we will consider alternative options for combining the three types of trees, rather than using the simple approach of keeping just one tree. For example, the results from the three trees might be combined into a single decision. Finally, alternative decision tree algorithms, or even other types of model builders, will be considered within this hybrid framework.

Acknowledgements. This research is supported in part by Shenzhen New Industry Development Fund under grant No.CXB201005250021A.

References

1. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
2. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
3. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo (1993)
4. Breiman, L., Friedman, J.H., Olshen R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth and Brooks/Cole Advanced Books and Software, Monterey, CA (1984)
5. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
6. Biggs, D., Suen, E.: A method of choosing multiway partitions for classification and decision trees. Journal of Applied Statistics 18(1), 49–62 (1991)
7. Ture, M., Kurt, I., Turhan Kurum, A., Ozdamar, K.: Comparing classification techniques for predicting essential hypertension. Expert Systems with Applications 29(3), 583–588 (2005)
8. Klema, J., Almonayyes, A.: Automatic categorization of fanatic texts using random forests. Kuwait Journal of Science and Engineering 33(2), 1–18 (2006)
9. Begum, N., Fattah, M.A., Ren, F.J.: Automatic text summarization using support vector machine. International Journal of Innovative Computing Information and Control 5(7), 1987–1996 (2009)
10. Chen, J.N., Huang, H.K., Tian, S.F., Qu, Y.L.: Feature selection for text classification with naive bayes. Expert Systems with Applications 36(3), 5432–5435 (2009)
11. Tan, S.: Neighbor-weighted K-nearest neighbor for unbalance text corpus. Expert Systems with Applications 28(4), 667–671 (2005)
12. Dietterich, T.G.: Machine learning research: Four current directions. AI Magazine 18(4), 97–136 (1997)
13. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: ACM SIGIR 1999, pp. 42–49 (1999)
14. Han, E.-H(S.), Karypis, G.: Centroid-based Document Classification: Analysis and Experimental Results. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 424–431. Springer, Heidelberg (2000)
15. TREC. Text retrieval conference, `http://trec.nist.gov`
16. Lewis, D.D.: Reuters-21578 text categorization test collection distribution 1.0 (2011), `http://www.research.att.com/~lewis`
17. Hersh, W., Buckley, C., Leone, T.J., Hickam, D.: OHSUMED: An interactive retrieval evaluation and new large test collection for research. In: SIGIR 1994, pp. 192–201 (1994)
18. Moore, J., Han, E., Boley, D., Gini, M., Gross, R., Hastings, K., Karypis, G., Kumar, V., Mobasher, B.: Web page categorization and feature selection using association rule and principal component clustering. In: WITS 1997 (1997)
19. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI Workshop 1998, pp. 41–48 (1998)
20. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd edn. Morgan Kaufmann, Burlington (2011)