Dynamic Integration with Random Forests

Alexey Tsymbal
Dept of Computer Science,
Trinity College Dublin, Dublin 2, Ireland
tsymbalo@cs.tcd.ie

Mykola Pechenizkiy
Dept of Math IT, University of Jyväskylä,
P.O.Box 35, 40351 Finland
mpechen@it.jyu.fi

Pádraig Cunningham
Dept of Computer Science,
Trinity College Dublin, Dublin 2, Ireland
cnnnghmp@cs.tcd.ie

ABSTRACT

Random Forests are a successful ensemble prediction technique that combines two sources of randomness to generate base decision trees; bootstrapping instances for each tree and considering a random subset of features at each node. Breiman in his introductory paper on Random Forests claims that they are more robust than boosting with respect to overfitting noise, and are able to compete with boosting in terms of predictive performance. Multiple recently published empirical studies conducted in various application domains confirm these claims. Random Forests use simple majority voting to combine the predictions of the trees. However, it is clear that each decision tree in a random forest may have different contribution in classifying a certain instance. In this paper, we demonstrate that the prediction performance of Random Forests may still be improved in some domains by replacing the combination function. Dynamic integration, which is based on local performance estimates of base predictors, can be used instead of majority voting. We conduct experiments on a selection of classification datasets, analysing the resulting accuracy, the margin and the bias and variance components of error. The experiments demonstrate that dynamic integration increases accuracy on some datasets. Even if the accuracy remains the same, dynamic integration always increases the margin. A bias/variance decomposition demonstrates that dynamic integration decreases the error by significantly decreasing the bias component while leaving the same or insignificantly increasing the variance. The experiments also demonstrate that the intrinsic similarity measure of Random Forests is better than the commonly used Heterogeneous Euclidean/Overlap Metric in finding a neighbourhood for local estimates in this context.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Management – *Database Applications, Data Mining*

I.2.6 [**Computing Methodologies**]: Artificial Intelligence – *Learning*

I.5.1 [**Computing Methodologies**]: Pattern Recognition – *Models*

General Terms

Algorithms, Performance, Design, Experimentation

Keywords

Ensemble, Random Forests, Dynamic Integration, Locally Weighted Learning

1. INTRODUCTION

Ensemble learning is one of the most important current directions in machine learning and data mining. Predictive performance demonstrated by ensembles of models is usually comparable or even better than the performance that can be achieved by the best single sophisticated models in many application domains [8].

Random Forests are a relatively young (they were introduced in 2001), but effective and popular ensemble learning technique [6]. It is based on combining two sources of randomness when generating decision trees: (1) each tree is constructed on a bootstrap replicate of the original dataset as in bagging, and (2) a random feature subset of a fixed predefined size is considered for each node at tree construction. Random Forests were demonstrated to compare favourably with boosting in terms of predictive performance and to be more robust than boosting with respect to overfitting noisy instances in various classification and regression domains. In classification, Random Forests often achieve error rates comparable to the Bayes rate [6].

In the standard Random Forests algorithm [6] simple majority voting or averaging are used to combine predictions of the base decision trees. In classification, each decision tree votes for the most popular class by casting a unit vote. In regression, outputs of the base decision trees are simply averaged. However, obviously, base decision trees may have different strengths at processing different instances. Due to the two sources of randomness used, base models in Random Forests are usually relatively weak, and significantly vary in their predictive performance. It was demonstrated with various ensemble designs that ensemble performance may often be improved by replacing the simple majority voting procedure or averaging with a more sophisticated combination function [7, 15].

A natural possible extension to Random Forests is to improve the combination of trees by taking into account their local predictive performance. One such combination technique, which could be used here, is dynamic integration [15]. In dynamic integration, local predictive performance is first estimated for each base model based on the performance on similar instances, and then it is used to calculate a corresponding weight for combining predictions with locally weighted voting (Dynamic Voting), or simply a model with the best local performance is selected (Dynamic Selection).

Random Forests provide us with an intrinsic similarity metric, which could be used in dynamic integration. The proportion of the base trees where two instances appear together in the same leaves can be used as a measure of similarity between the instances [6]. In this paper we evaluate the two alternative combination functions. We find that dynamic integration does improve the performance of Random Forests. We also find that this intrinsic similarity metric is very effective; this is not surprising as it is *in tune* with the dynamics of the ensemble.

This paper is organized as follows: in Section 2 we review the Random Forests ensemble prediction technique, in Section 3 we consider how it can be augmented with dynamic integration of predictive models, in Section 4 we present the results of our experiments comparing majority voting with dynamic integration of classifiers as combination functions in Random Forests on a selection of benchmark datasets, and in Section 5 we conclude with a brief summary and further research directions.

2. RANDOM FORESTS

Breiman in his paper [6] demonstrated that optimal ensemble performance could be achieved by injecting randomness in order to minimize correlation between base models while maintaining accuracy. In terms of the bias/variance decomposition of error, this means that the base models in an ensemble should have maximal variance while keeping bias low.

In Random Forests this is achieved by combining two sources of randomness. First, instances used to grow each tree are sampled randomly without replacement from the original training set. Second, Random Forests consist of using randomly selected features at each node to grow each tree [6]. Using the Strong Law of Large Numbers, Breiman demonstrated that Random Forests always converge so that overfitting is not a problem, that is Random Forests never overfit as more trees are added. Multiple recent empirical studies demonstrate Random Forests to be competitive in accuracy with the best classification and regression algorithms in a number of application domains.

Random Forests have a set of desirable properties [6]:

(1) their predictive performance is as good as boosting and sometimes better (especially on noisy datasets);

(2) they are relatively robust to outliers and noise;

(3) they are faster than many other ensembles, bagging and boosting in particular;

(4) due to the use of bootstrapping as in bagging, they give useful internal (so-called out-of-bag) estimates of error, strength (margin), correlation and feature importance;

(5) they are simple and easily parallelized.

Random Forests were demonstrated to produce error rates not far above the Bayes error rate [6]. However, in some domains their accuracy still can be improved. For example, Robnik-Šikonja in [11] considered two ways of improving Random Forests: (1) a combination of several feature selection criteria such as Gini index, Gain ratio, MDL, ReliefF and *j*-measure in order to reduce correlation and increase variance in the forests, and (2) replacement of majority voting with locally weighted voting as a new combination function in Random Forests. It was demonstrated that both of the ways might lead to some improvement in performance and the improvement due to the replacement of combination function was statistically significant for several datasets. In this paper our focus is on the second way – the replacement of combination function.

3. IMPROVING RANDOM FORESTS

3.1 Dynamic Integration of Predictive Models

In this sub-section we consider the basic idea of dynamic integration of predictive models in an ensemble and give a brief review of related work.

Brodley and Lane [7] have shown that simply increasing diversity of an ensemble is not enough to ensure increased predictive performance. If an integration method does not utilize diversity, then no benefit arises from the integration. The task of ensemble integration is to decide which of the models to select or how to combine the results produced by the base models for each particular instance. A number of *selection* and *combination* approaches to integration have been proposed [7, 8, 12, 13, 15].

One of the most popular and simplest techniques used to combine the results of base models in a classification ensemble, which is also used in Random Forests, is simple voting (also called majority voting) [2]. In voting, the output of each base classifier is considered as a vote for that particular class value. The class value that receives the biggest number of votes is selected as the final classification. Weighted Voting (WV), where each vote has a weight proportional to the estimated generalization performance of the corresponding classifier, usually has better predictive performance than simple voting [2]. An analogue of voting in regression is the simple averaging of numeric outputs.

A number of selection techniques have also been proposed to address the task of integration. One of the most popular and simplest selection techniques is Cross-Validation Majority (CVM) [13]. In CVM, cross-validation accuracy for each base classifier is first estimated, and then the classifier with the highest accuracy is selected. The same principle can be used in regression.

The approaches described above are *static*. They select one model for the whole data space or combine the models uniformly. In *dynamic* integration information about each new instance to be processed is taken into account. Experimental studies in many application domains demonstrate that better results can often be achieved if integration is dynamic.

Three dynamic integration techniques based on the same local performance estimates; Dynamic Selection (DS), Dynamic Voting (DV), and Dynamic Voting with Selection (DVS), variants of which we also use in our experiments in this paper, were considered in [15]. These dynamic techniques have the same training phase. In classification, the local errors of each base classifier for each instance of the training set are estimated according to the 1/0 loss function using cross validation. The training phase finishes with training the base classifiers on the whole training set. The application phase begins with determining *k*-nearest neighbours for a new instance. Then, weighted nearest neighbour learning is used to predict the local errors of each base classifier for the new instance.

Then, DS simply selects a classifier with the least predicted local error. In DV, each base classifier receives a weight that is proportional to its estimated local accuracy, and the final classification is produced using weighted voting. In DVS, the base classifiers with the highest local errors are discarded (the classifiers with the errors that fall into the upper half of the error interval) and locally weighted voting (DV) is applied to the remaining classifiers.

In fact, the basic idea in the dynamic integration approach suggested consists in learning (meta-level learning) the predictive performance of base models. Lazy learning is used in order to predict the local performance of the base models in an ensemble.

Dynamic integration was successfully applied in a number of contexts, outperforming other integration methods. In [15] the three dynamic integration techniques were used to combine the base classifiers generated with bagging and boosting, improving the predictive performance of the two most popular ensemble techniques. In [14] dynamic integration was considered in the context of ensembles with base classifiers generated on different feature subsets (using so-called ensemble feature selection). In [12] an adaptation of the three dynamic integration techniques to regression is considered and applied for ensembles generated using the random subspace method.

3.2 Dynamic Integration in Random Forests

In this sub-section we consider the details of how dynamic integration can be implemented with Random Forests.

Random Forests have the very appealing property that each tree is built on a bootstrap replicate of the original training set. The remaining (out-of-bag) instances are useful for evaluating Random Forests and their component trees. The out-of-bag instances could be used to estimate the base trees' accuracy, margin, correlation, and even feature importance in the domain [6]. This property can be used in dynamic integration as well. With out-of-bag instances there is no need for cross-validation or for a separate validation set in order to get the estimates of the base models' performance. Instead, the performance of base models on out-of-bag instances can be simply recorded at the end of training phase and later used for local performance prediction on similar out-of-bag instances.

Different distance functions can be used for determining the neighbourhood of the current test instance in dynamic integration. The simplest and most common way is to use the heterogeneous Euclidean/overlap metric (HEOM)[17] in the instance space as in [15]. In HEOM, the Euclidean distance is used with numeric features, and the overlap distance with categorical features in order to find a distance between two instances $\mathbf{x}_1$ and $\mathbf{x}_2$ as shown in (1) and (2), where m is the number of features. For a numeric feature a the distance is normalised by the width of the range of values of the corresponding feature on the training set $range_a$:

$$d_{heom}(\mathbf{x}_1,\mathbf{x}_2) = \sqrt{\sum\nolimits_{a=1}^{m} heom_a^2(\mathbf{x}_1,\mathbf{x}_2)} \tag{1}$$

$$heom_a(\mathbf{x}_1,\mathbf{x}_2) = \begin{cases} \text{if } a \text{ is categorical,} \begin{cases} 0, \text{if } x_{1a} = x_{2a} \\ 1, \text{otherwise} \end{cases} \\ \text{else, } \dfrac{|x_{1a} - x_{2a}|}{range_a} \end{cases} \tag{2}$$

The Euclidean distance for numeric features and the overlap distance for categorical were demonstrated to be robust and difficult to compete with in many applications, see for example [17]. However, Random Forests provide us with an intrinsic instance similarity metric, which could be used in dynamic integration as well. Breiman [6] suggested the proportion of the component trees where two instances appear together in the same leaves as a measure of similarity between the two instances.

This distance was demonstrated to be useful in different tasks such as clustering, outlier removal, visualisation and finding prototypes in many application domains such as bioinformatics. It is important to note that two instances that are quite close together in the Euclidean (or HEOM) space might have relatively small intrinsic similarity if they are near the classification boundary. The intrinsic random forest similarity demands $O(nK)$ additional space for saving information about n training instances in the leaves of the K trees.

We use both HEOM and the intrinsic random forest similarity as metrics for finding a neighbourhood for local performance estimates in our experiments with dynamic integration in this paper. In order to calculate the weight for model i in dynamic integration for a new instance **x**, we suggest to use:

$$w_i(\mathbf{x}) = \frac{\sum_{j=1}^{k} I(\mathbf{x}_j \in OOB_i) \cdot \sigma(\mathbf{x},\mathbf{x}_j) \cdot mr_i(\mathbf{x}_j)}{\sum_{j=1}^{k} I(\mathbf{x}_j \in OOB_i) \cdot \sigma(\mathbf{x},\mathbf{x}_j)} \tag{3}$$

where k is the size of the neighbourhood, OOB_i is the set of out-of-bag instances for model i, $I()$ is an indicator function, $\sigma(\mathbf{x},\mathbf{x}_j)$ is a distance-based relevance coefficient and $mr_i(\mathbf{x}_j)$ is the margin (4) of model i on jth nearest neighbour of **x**. Margin is defined as usual for a classifier with crisp outputs (1 for a correct prediction, and –1 for a wrong one):

$$mr_i(\mathbf{x}) = \begin{cases} 1, h_i(\mathbf{x}) = y(\mathbf{x}) \\ -1, h_i(\mathbf{x}) \neq y(\mathbf{x}) \end{cases} \tag{4}$$

The denominator in (3) is necessary for normalizing the relevance coefficients $\sigma(\mathbf{x},\mathbf{x}_j)$ to sum to one on the corresponding out-of-bag instances. In fact, weight (3) represents the expected margin of model i at instance **x**. We normalize weights (3) to be non-negative and to sum to one in order to apply them in (locally) weighted voting in dynamic integration. After weights (3) are calculated, the three dynamic integration functions (DS, DV and DVS) are applied as described in Section 3.1.

The distance-based weight coefficient $\sigma(\mathbf{x},\mathbf{x}_j)$ should reflect similarity between the two instances. In our experiments with two distance metrics we use the inverse HEOM distance and the cube of the intrinsic random forest similarity as the corresponding distance-based weight coefficients:

$$\sigma(\mathbf{x},\mathbf{x}_j) = 1/d_{heom}(\mathbf{x},\mathbf{x}_j) \tag{5}$$

$$\sigma(\mathbf{x},\mathbf{x}_j) = rf\_sim^3(\mathbf{x},\mathbf{x}_j) \tag{6}$$

A simple weighting function that just raises the distance to a negative power is perhaps the most commonly used weighting function in locally weighted learning, and is often used for the Euclidean and overlap metrics [1]. It is also used in most previous experiments with dynamic integration [14,15]. The cube in (6) was selected after a small series of experiments on a few separate benchmark datasets (not included in the experiments presented in Section 4) comparing different natural number powers of the intrinsic similarity. It is important to notice that (5) and (6) are not necessarily optimal and perhaps better weighting functions could be suggested for particular datasets. However, it is enough to use these simple functions in our experiments to demonstrate the potential of locally weighted learning in dynamic integration. In our experiments we also consider a non-weighted

($\forall(\mathbf{x},\mathbf{x}_j):\sigma(\mathbf{x},\mathbf{x}_j)=1$) variant of (3), demonstrating that the use of weights is usually superior for both of the distance metrics.

Now we can categorize the dynamic integration technique for Random Forests previously considered in [11] according to the introduced notation. The proposed technique used the intrinsic random forest similarity without locally weighting margins. The technique itself is almost identical to DVS – it just skips the negative weights in (3). In this study, comparing with [11], we analyse two distance metrics (HEOM and intrinsic similarity), compare the locally weighted and non-weighted cases, and different sizes of neighbourhood with respect to the three dynamic integration functions (DS, DV, and DVS). Besides analysing accuracy for ensembles of different sizes, we consider the influence of dynamic integration on the ensemble margin and provide the bias/variance decomposition of error.

4. EXPERIMENTAL STUDIES

4.1 Experimental Setup

In our experimental studies we used an implementation based on the machine learning library WEKA 3.4.2 (available at http://www.cs.waikato.ac.nz/~ml/weka/), which is currently perhaps the most popular library of machine learning algorithms [18]. In this implementation, Information Gain is used as the splitting criterion for growing each (unpruned) tree, the number of randomly selected features in each node is the integer part of $\log_2 M+1$, where M is the number of features in the dataset.

We experiment with random forests including 10, 25, 50 and 100 trees. A forest of 100 trees is constructed each time at the training phase, and the first 10, 25 and 50 trees out of these 100 are considered for the ensembles of smaller sizes. At the application phase, we experiment with 4 different integration strategies to combine the predictions of the trees; plain static majority voting (SV) and the three dynamic techniques; DS, DV and DVS. In order to determine the neighbourhood for the dynamic techniques, two distance metrics are used; HEOM and intrinsic similarity. We consider both a locally weighted and non-weighted calculation of local performance estimates (3). We experiment with 4 different sizes of neighbourhood; 15, 31, 63, and 127. We estimate the accuracy and the margin for the four integration strategies using 30 runs of hold-out cross validation with 70%/30% train/test split of the dataset.

In our experiments we use 27 benchmark datasets. 24 of these datasets are from the UCI machine learning repository [4]. The Parity2 and Parity3 datasets were considered in [11]. They have 2 and 3 binary parity attributes respectively. Each of these parity problems also contains 10 random irrelevant binary attributes. The Images dataset consists of 1000 image windows drawn from 2 monochrome images of natural scenes. The sizes of the original images were 512 x 256 pixels, and windows of size 50 x 50 were randomly drawn from the images. Each image window was presented as a one *d*-dimensional column vector (*d* = 2500). These images were previously considered by Bingham and Mannila in [3] (here we consider the 2 first classes only instead of all 13 considered in their paper). The characteristics of the datasets are collected in Table 1, including the number of instances (*Size*), categorical features (*Cat.Feats*), numeric features (*Num.Feats*) and class values (*Classes*).

Table 1. Dataset summary

Name	Size	Cat.Feats	Num.Feats	Classes
Audiology	226	69	0	24
Balance	625	0	4	3
Breast cancer	286	9	0	2
Car	1728	6	0	4
Pima diabetes	768	0	8	2
DNAp	106	57	0	2
Glass recognition	214	0	9	6
Heart disease	270	0	13	2
Images	1000	0	2500	2
Ionosphere	351	0	34	2
Iris Plants	150	0	4	3
LED	300	7	0	10
LED17	300	24	0	10
Liver disorders	345	0	6	2
Lymphography	148	15	3	4
MONK-1	432	6	0	2
MONK-2	432	6	0	2
MONK-3	432	6	0	2
Parity2	200	12	0	2
Parity3	200	13	0	2
Sonar	208	0	60	2
Soybean	47	0	35	4
Thyroid	215	0	5	3
Tic-tac-toe	958	9	0	2
Vehicle	846	0	18	4
Voting	435	16	0	2
Zoo	101	16	0	7

4.2 Analysis of Classification Accuracy

As was mentioned in Section 2, Random Forests often give an error rate comparable to the Bayes rate. This is especially so for many relatively simple datasets from the UCI repository. Thus, it is no surprise that their accuracy is very difficult to beat for any technique, including Random Forests with dynamic integration. In our experiments, on 12 out of the 27 datasets considered there was a statistically significant accuracy improvement due to the use of dynamic integration with some configuration. With the remaining datasets accuracy achieved was similar to plain Random Forests, or the difference was insignificant. We continue the analysis of experimental results focusing on the 12 datasets on which some form of dynamic integration helped to improve accuracy.

The first surprising tendency which we could see from the experimental results was that the accuracy of DS was very poor. On most datasets DS significantly decreased the accuracy of Random Forests with any local learning scheme and the size of neighbourhood. Only with 2 datasets was its accuracy better; MONK-2 and Parity2. These datasets represent artificial concepts “well suitable” for dynamic integration. Such a poor behaviour of DS is surprising, because much research in the area of ensembles is concentrated on (dynamic) classifier selection, and this is justified by its good performance in many application domains. However, in the context of Random Forests, the base models are usually weak and diverse, which makes the task of classifier selection difficult. This is in line with [11], where the poor performance of classifier selection in the context of Random Forests was also mentioned. We continue the analysis of experimental results with the other two dynamic integration techniques; DV and DVS. Fortunately, their behaviour is much better and they are able to compete with majority voting in terms of accuracy. We shall come back to DS in Section 4.4, where its behaviour is explained in terms of bias/variance error decomposition.

DVS and DV give very close results, with DVS being a little better on average. The focus of our further analysis of experimental results is on DVS. However, all the dependencies presented hold true both for DVS and DV (unless otherwise stated).

An important question with any lazy learning technique is the size of neighbourhood used. In Figure 1 the average accuracy is presented for DVS with ensembles of 4 sizes (10, 25, 50 and 100) for 4 different sizes of neighbourhood (15, 31, 63, 127). The results are averaged over the 12 datasets and 4 different local learning schemes (2 distances with and without local weighting each).

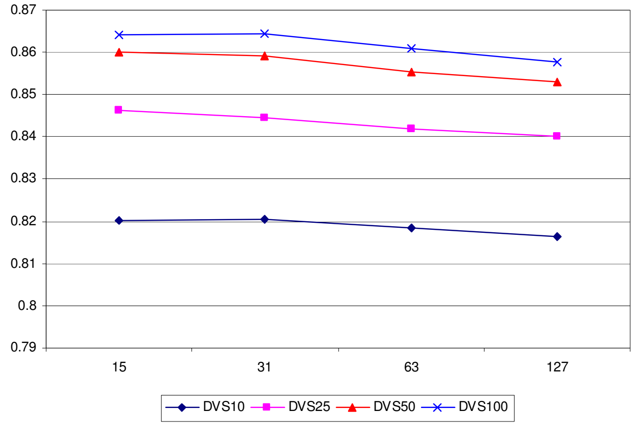

Figure 1. Classification accuracy of DVS Random Forests for different sizes of neighbourhood and ensemble sizes

Naturally, accuracy decreases with the increase in the size of neighbourhood, becoming closer to simple static majority voting. Dynamic integration is not sensitive to the size of neighbourhood. The neighbourhoods of 15 and 31 instances give very close results. This supports a similar claim made in [11].

These results are averaged over the datasets and different local learning schemes. However, the same tendency holds true in every case. For the locally weighted cases the lines presented are a little flatter (they are closer to horizontal lines), while for the non-weighted cases the size of neighbourhood is a little more important. We continue our analysis of experimental results focusing on the size of neighbourhood equal to 15, as it gives the best improvement due to dynamic integration on average.

Now let us consider different local learning schemes for dynamic integration. In Figure 2 the accuracy of plain Random Forests with static voting (*SV*) is compared with Random Forests with DVS for the 4 different local learning schemes; HEOM, equally-weighted (*DVSheom*) and locally weighted (*DVSheomw*), and intrinsic random forest similarity, equally-weighted (*DVSrf*) and locally weighted (*DVSrfw*) for the 4 different ensemble sizes with 15 instances in the neighbourhood, averaged over the 12 datasets.

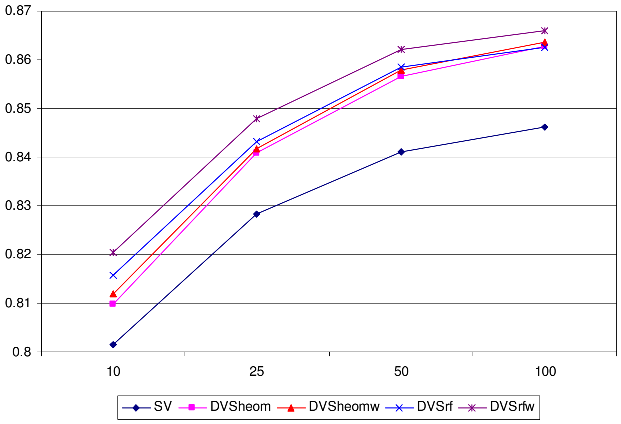

Figure 2. Classification accuracy of plain and DVS Random Forests for different local learning schemes and ensemble sizes

This figure reveals a few interesting tendencies. First, it shows the average improvement due to dynamic integration, which is more than 1.5% with any local learning scheme for the ensembles with 100 trees. Second, all the schemes give close results. However, it can be seen that the locally weighed schemes out-perform their equally weighted counterparts, and the intrinsic similarity results out-perform the HEOM results in general. An interesting result is that the locally weighted intrinsic similarity scheme clearly stands out on the figure. As we shall later see, this superiority will also be supported by tests for statistical significance and the analysis of margin for each dataset.

In Table 2 accuracy results are given for the ensembles of 100 trees for plain Random Forests with static voting (*SV*) and for the four local learning schemes used in combination with DVS (*DVSheom*, *DVSheomw*, *DVSrf* and *DVSrfw*) for the 12 datasets. The neighbourhood size of 15 is used in the dynamic integration strategies.

The table includes the dataset name, the minimum, average and maximum accuracy of ensemble members, their agreement (the proportion of instances correctly classified by every ensemble member) and coverage (the proportion of instances correctly classified by at least one ensemble member), and the classification accuracies for the five integration strategies. Numbers given in bold represent the significant wins of corresponding DVS strategies over SV (according to the paired *t*-test with 0.95 level of significance).

This table demonstrates the fact that Random Forests ensembles contain weak and highly diverse base classifiers. In many domains Random Forests out-perform the best component decision tree (except the Glass, Zoo and Parity problems). Out of the four local learning strategies, locally weighted intrinsic similarity (DVSrfw) demonstrates the most robust behaviour with the best average accuracy and 9 wins (with 8 wins for DVSheom and 7 wins for DVSrf and DVSheomw). Dynamic integration (DVS strategy) gives always similar or better accuracy than SV in these domains. The same situation holds true with DVS and with the ensembles of other sizes (10, 25 and 50).

Table 2. Classification accuracy for plain Random Forests and for the four local learning schemes used with DVS

Dataset	Min	Aver	Max	Agr	Cov	SV	DVSheom	DVSheomw	DVSrf	DVSrfw
Audiology	0.316	0.507	0.707	0.010	0.907	0.727	**0.741**	**0.740**	**0.739**	**0.739**
Car	0.755	0.830	0.888	0.228	1.000	0.935	**0.938**	**0.937**	0.936	0.937
DNAp	0.385	0.636	0.872	0.000	1.000	0.908	0.914	0.911	0.908	0.913
Glass	0.495	0.637	0.770	0.047	0.991	0.762	0.765	0.764	0.770	**0.772**
Images	0.566	0.639	0.708	0	1	0.85	**0.859**	**0.86**	**0.857**	**0.859**
MONK-1	0.624	0.824	0.989	0.068	1.000	0.997	**0.999**	**1.000**	**1.000**	**1.000**
Parity2	0.397	0.658	0.999	0.000	1.000	0.925	**0.973**	**0.974**	**0.978**	**0.977**
Parity3	0.350	0.543	0.860	0.000	1.000	0.639	**0.716**	**0.724**	**0.713**	**0.724**
Sonar	0.524	0.689	0.835	0.001	1.000	0.830	**0.841**	0.840	**0.840**	**0.844**
Tic-tac-toe	0.673	0.765	0.845	0.013	1.000	0.936	**0.960**	**0.961**	**0.961**	**0.966**
Vehicle	0.605	0.675	0.738	0.061	1.000	0.746	0.748	0.748	0.749	0.749
Zoo	0.709	0.844	0.959	0.491	0.988	0.898	0.898	0.904	0.899	**0.912**
Average	*0.533*	*0.687*	*0.848*	*0.077*	*0.991*	*0.846*	*0.863*	*0.864*	*0.863*	*0.866*

An interesting tendency revealed by this table, concerns the behaviour of dynamic integration on datasets with different types of features (categorical and numeric). Dynamic integration (at least in its present implementation) is more robust and demonstrates bigger gains on datasets with categorical features. First, the original pool of datasets included 14 categorical and 12 numeric datasets (one, Lymphography, is heterogeneous). The selected 12 datasets with an increase of predictive performance due to dynamic integration include 8 categorical and only 4 numeric datasets. Second, the average gain in accuracy due to dynamic integration is considerably bigger for categorical data.

In Figure 3 average accuracy of plain and DVS Random Forests for different local learning schemes and ensemble sizes is shown averaged over the 4 numeric datasets only (Glass, Images, Sonar and Vehicle). This figure has the same format as Figure 2.

While it can be seen from the figure that all the discovered tendencies (the superiority of dynamic integration and the relative superiority of intrinsic similarity and locally weighted schemes) hold true with the numeric datasets as well, the gain due to dynamic integration is considerably less with numeric data than with the categorical data (0.9% vs 2.5% with the DVSrfw locally weighted learning strategy, which gives the best gain).

4.3 Analysis of Classification Margin

Besides the accuracy for the different integration techniques considered we also measured margin for static voting, DV and DVS. We did not measure margin for DS because in DS one classifier only is selected and the average margin for a classifier with crisp predictions can be simply derived from its accuracy.

The margin of a classifier h on instance $\mathbf{x}$ measures the extent to which the average vote for the right class $y(\mathbf{x})$ exceeds the maximal average vote for any other class:

$$mr(\mathbf{x}, y(\mathbf{x})) = P(h(\mathbf{x}) = y(\mathbf{x})) - \max_{y_i \neq y(\mathbf{x})} P(h(\mathbf{x}) = y_i) \qquad (7)$$

Average margin over the test instances represents an estimate of expected margin for the classification problem considered and is an important characteristic for any learning algorithm [6,11].

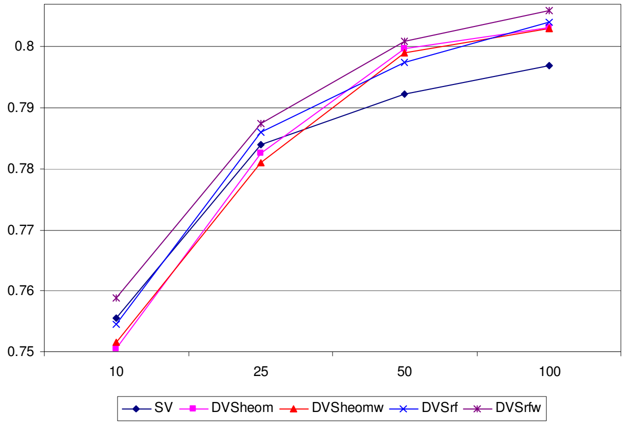

Figure 3. Classification accuracy of plain and DVS Random Forests for different local learning schemes and ensemble sizes for the 4 numeric datasets

In Table 3 the margin is given for the plain Random Forests and for the 4 local learning schemes used in combination with DV and DVS. From this table one can see that dynamic integration always increases the margin of plain Random Forests on these 12 datasets. Interestingly, this increase is always significant (according to the *t*-test). Besides, dynamic integration often increases the margin even when the accuracy of dynamic integration remains the same with static voting (on the rest of 27 datasets). This behaviour of dynamic integration is close to the behaviour of boosting which was proven in theory and demonstrated empirically to always increase (or at least not to decrease) the margin.

Table 3. Classification margin for plain Random Forests and for the four local learning schemes with DV and DVS

Dataset	SV	DVheom	DVSheom	DVheomw	DVSheomw	DVrf	DVSrf	DVrfw	DVSrfw
Audiology	0.255	0.282	0.291	0.286	0.302	0.285	0.296	0.306	0.314
Car	0.701	0.724	0.729	0.726	0.733	0.729	0.735	0.746	0.754
DNAp	0.267	0.289	0.298	0.289	0.302	0.299	0.310	0.306	0.322
Glass	0.370	0.382	0.386	0.387	0.394	0.387	0.392	0.403	0.411
Images	0.276	0.284	0.287	0.286	0.289	0.286	0.289	0.29	0.295
MONK-1	0.614	0.677	0.689	0.680	0.700	0.696	0.716	0.728	0.754
Parity2	0.315	0.406	0.434	0.412	0.454	0.416	0.449	0.444	0.484
Parity3	0.092	0.138	0.160	0.140	0.172	0.143	0.165	0.165	0.187
Sonar	0.377	0.392	0.397	0.396	0.406	0.400	0.406	0.408	0.420
Tic-tac-toe	0.513	0.558	0.571	0.559	0.575	0.566	0.582	0.584	0.608
Vehicle	0.418	0.424	0.426	0.425	0.427	0.428	0.429	0.430	0.434
Zoo	0.752	0.759	0.761	0.771	0.775	0.761	0.763	0.779	0.782
Average	*0.413*	*0.443*	*0.452*	*0.446*	*0.461*	*0.450*	*0.461*	*0.466*	*0.480*

This behaviour is not so surprising, as the notion of a diverse ensemble is somewhat at odds with the concept of a high classification margin, i.e. diversity can be achieved by *squeezing* the margin. Naturally, the base classifiers in Random Forests are weak and diverse, and their margin can be simply improved.

Interestingly, the margins of weighted schemes are always greater than the corresponding margins of non-weighted schemes, and the margins using random forest similarity are always greater than the corresponding margins of HEOM. Another interesting and somewhat surprising tendency when one considers separate locally learning schemes is that while all the other three schemes give pretty close results, the margins with locally weighted intrinsic similarity (DVrfw and DVSrfw) usually clearly stand out and are always statistically significantly higher than all the other corresponding margins, supporting its relative superiority in accuracy shown in Figure 1 (this could be seen from the average values in Table 3 as well). The results in Table 3 clearly show the superiority of the intrinsic random forest similarity over HEOM in finding a neighbourhood for dynamic integration. The fact that the locally weighted intrinsic similarity always produces a statistically significantly greater margin, even though the corresponding accuracy may not always be significantly different in comparison with the other local learning schemes, demonstrates its greater strength in this context.

Another interesting tendency is that DVS always increases the margin of DV, and in the vast majority of cases this increase is statistically significant. This happens even when the accuracy of DVS is similar to or even less than that of DV. This behaviour can be explained by the fact that DVS tries to focus on more "likely to be correct here" classifiers and thus tends to decrease the contribution of wrong predictions, even if they do not change the final classification.

In Figure 4 the average margin is shown for the plain Random Forests and for the 4 local learning schemes with DVS for the ensembles of different sizes (10, 25, 50 and 100). The figure confirms previously described tendencies. It is interesting that the increase in margin with the addition of more ensemble members is not so obvious as the increase in accuracy. The increase in margin from 10 to 100 members is less than 0.01 for plain Random Forests and is about 0.015 with the dynamic integration techniques. The corresponding increase in accuracy is always about 4.5% (see Figure 2) and the lines are steeper. There is almost no increase in margin when the ensemble grows from 50 to 100 members. The reason for that might be that the margin reflects mostly the bias component of error, and small ensembles already have pretty good bias, and the increase in accuracy with the addition of new members is mostly due to the decrease in variance.

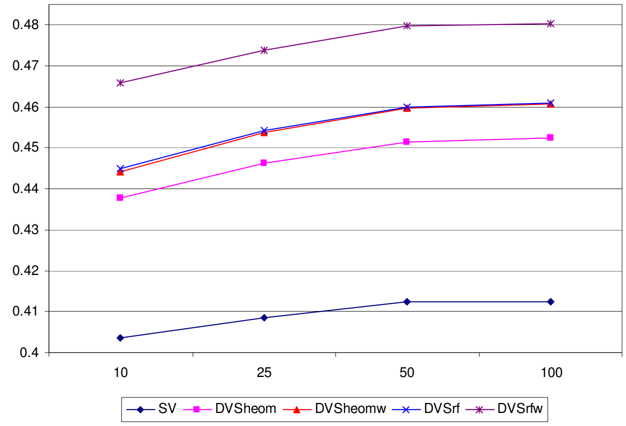

Figure 4. Classification margin of plain and DVS Random Forests for different local learning schemes and ensemble sizes

4.4 Bias/Variance Decomposition of Error

The bias/variance analysis is useful in focusing attention on two significant factors that govern the predictive performance of models trained by a learning system. If a learning system when provided different training data develops models that differ in their predictions, then the extent of such *variance* provides a lower limit on the expected error of those models when applied to any subsequent set of test data.

However, preventing such variance between the models will not guarantee the elimination of prediction error. This error is also governed by the accuracy of the learning *bias* and, in the case of classification, by the degree to which the correct classification for an instance can differ from those for other instances with identical descriptions (irreducible or Bayes error) [16].

Perhaps the most popular application domain for bias/variance analysis is ensemble learning [5,16]. The bias/variance decomposition of error helps to better understand ensemble's behaviour in reducing the generalization error of a single model. It was demonstrated that most ensemble techniques, such as bagging, reduce the variance component of error, while adaptive resampling-based (also called arcing and boosting) ensemble techniques such as AdaBoost are able to reduce both bias and variance [5,16].

While in regression there exists a well known and commonly used bias/variance decomposition based on squared (quadratic) loss, originating from the field of statistics called sampling theory, the situation with classification and zero-one loss is, unfortunately, not so clear. Recently, several decompositions have been proposed for zero-one loss, but many of them have a significant shortcoming in that they have only an intuitive relationship to the original purely additive squared-loss decomposition [9].

In our experiments, we consider two bias/variance decompositions; those of Kohavi and Wolpert [10] and Breiman [5]. They closely capture the original squared loss definitions and have a behaviour that corresponds with intuition. Both of these metrics decompose the error into three components; bias, variance and the irreducible Bayes error. While it is useful to take account of irreducible error in some cases, for the analysis of Random Forests, our interest centers on the manner in which bias and variance are affected by different ensemble integration strategies. The irreducible error is difficult if possible to estimate given only one dataset of limited size. Following the usual practice of bias/variance estimation (see for example [16]), we ignore the irreducible error and use modified formulae for the two decompositions in order to estimate the bias and variance components of error, distributing the irreducible error across the bias and/or variance terms.

For the Kohavi-Wolpert decomposition, we use their original formula of variance for a test instance $\mathbf{x}$ [10] and calculate bias as the difference between error and variance, thus combining the irreducible error with the original bias term:

$$variance_{KW}(\mathbf{x}) = \frac{1}{2}\left(1 - \sum_{y_i \in Y} P(h(\mathbf{x}) = y_i)^2\right) \qquad (8)$$

$$bias_{KW}(\mathbf{x}) = P(h(\mathbf{x}) \neq y(\mathbf{x})) - variance_{KW}(\mathbf{x}) \qquad (9)$$

In the simplified notation used, $h(\mathbf{x})$ is a model from a collection of classifiers generated with the same learning algorithm under investigation on a set of training samples. Y is the set of possible class values, and $y(\mathbf{x})$ is the ground truth class value.

The bias/variance decomposition of Breiman [5] is based on the concept of central tendency $h^{o}(\mathbf{x})$ and is calculated as follows:

$$bias_B(\mathbf{x}) = P((h(\mathbf{x}) \neq y(\mathbf{x})) \wedge (h(\mathbf{x}) = h^{o}(\mathbf{x}))) \qquad (10)$$

$$variance_B(\mathbf{x}) = P((h(\mathbf{x}) \neq y(\mathbf{x})) \wedge (h(\mathbf{x}) \neq h^{o}(\mathbf{x}))) \qquad (11)$$

The central tendency of a learning algorithm on instance $\mathbf{x}$ is the most probable (frequent) prediction made by classifiers generated with this algorithm on different random samples with this instance. The irreducible error is distributed across both bias and variance in (10)-(11). It is defined in [5] as the error that would be made by the Bayes Optimal Classifier as well, and can be both equal and different with respect to the central tendency.

In order to calculate the bias and variance estimates considered, we conduct a separate series of experiments in a similar way to [16]. The same test settings as before were used, but the values for the bias and variance metrics were estimated by ten runs of three-fold cross validation instead of the 30 Monte-Carlo cross validation runs used in the previous experiments. This was done in order to provide equal number of tests (10) for each instance. With this approach, every available instance is used the same number of times, both for training and for testing.

In Table 4 the results of bias/variance decomposition are presented for the usual static voting Random Forests (SV) and for the three dynamic integration techniques; DS, DV and DVS. Two decompositions were used; Kohavi-Wolpert (index *KW*) and Breiman's (index *B*) decomposition. Nine datasets are considered with the biggest gain in accuracy due to dynamic integration. Three datasets (Car, MONK-1, and Vehicle) were excluded in comparison with the previous experiments, as no interesting patterns in the behaviour of dynamic integration techniques could be observed when the error of static and dynamic integration is the same or the difference is subtle. Each cell in the table includes two numbers; bias and variance.

The values in Table 4 correspond to ensemble size 100, the size of neighbourhood 15 and the intrinsic similarity with locally weighted learning in dynamic integration. These parameters were demonstrated to be the best in the previous experiments. However, the same behaviour of bias and variance with dynamic integration could be observed with other parameters as well.

An interesting finding, which can be made from the numbers presented in Table 4 and from other not presented here numbers for bias/variance decomposition for other ensemble configurations is that although the two different bias/variance decompositions have different nature, they often give numbers close to each other and almost always capture the same behaviour of the dynamic integration techniques. This could be confirmed by the Pearson's correlation coefficient between the two decompositions over different datasets, which is always greater than 0.96 for different integration techniques in this context. An interesting question for further research is whether the two decompositions always give such well-coordinated values, or whether this is true for the present study only.

Analysing the behaviour of DS in Table 4, one can see that DS tries to reduce bias at the expense of the considerable increase in variance. The increase in variance is huge, and in some datasets bias is increased too (on DNAp, Images, Tic-tac-toe and Zoo with both decompositions, and Audiology, Parity 2 and Sonar with the Kohavi-Wolpert decomposition).

Table 4. Bias/variance decomposition for static voting and three dynamic integration techniques in Random Forests

Dataset	SV_{KW}	DS_{KW}	DV_{KW}	DVS_{KW}	SV_{B}	DS_{B}	DV_{B}	DVS_{B}
Audiology	0.182	0.186	0.169	0.167	0.163	0.123	0.154	0.153
	0.077	0.214	0.075	0.079	0.096	0.277	0.091	0.094
DNAp	0.056	0.138	0.047	0.048	0.064	0.111	0.04	0.04
	0.059	0.187	0.061	0.062	0.051	0.213	0.069	0.071
Glass	0.17	0.167	0.162	0.161	0.169	0.137	0.159	0.158
	0.07	0.157	0.07	0.072	0.07	0.187	0.073	0.074
Images	0.116	0.155	0.103	0.098	0.123	0.123	0.109	0.103
	0.043	0.19	0.046	0.049	0.036	0.222	0.04	0.044
Parity2	0.032	0.058	0.011	0.008	0.024	0.016	0.009	0.006
	0.056	0.137	0.03	0.022	0.064	0.179	0.032	0.024
Parity3	0.249	0.178	0.196	0.183	0.253	0.15	0.201	0.189
	0.136	0.197	0.13	0.131	0.133	0.226	0.125	0.126
Sonar	0.128	0.138	0.121	0.117	0.141	0.124	0.137	0.129
	0.061	0.161	0.062	0.062	0.048	0.174	0.046	0.05
Tic-tac-toe	0.031	0.047	0.016	0.012	0.03	0.031	0.011	0.008
	0.036	0.099	0.027	0.025	0.036	0.115	0.032	0.029
Zoo	0.033	0.043	0.026	0.026	0.029	0.032	0.016	0.021
	0.032	0.063	0.032	0.032	0.037	0.074	0.043	0.038
Average	*0.111*	*0.123*	*0.095*	*0.091*	*0.111*	*0.094*	*0.093*	*0.090*
	0.063	*0.156*	*0.059*	*0.059*	*0.063*	*0.185*	*0.061*	*0.061*

DV and DVS reduce error by reducing bias while trying to keep variance the same. DV and DVS, on these datasets, always decrease bias and this decrease is always significant. Sometimes this is accompanied by a usually insignificant increase in variance (the increase is significant for DNAp with Breiman's decomposition only). In some cases DV and DVS reduce both bias and variance (for example, for the Parity problems this decrease is significant). This behaviour can be observed from the average numbers as well.

When one compares DV and DVS, it is possible to see that DVS, as a technique involving classifier selection, tries to decrease bias. Interestingly, this is not always accompanied by the same increase in variance, and on average the variance terms of DV and DVS are the same.

5. CONCLUSIONS

Random Forests are one of the most successful ensemble learning techniques that generates component decision trees on the bootstrap replicates of the original dataset and considers random samples of features as the candidates for splitting at the nodes. Multiple recent empirical studies have demonstrated Random Forests to be competitive in accuracy with the best classification and regression algorithms such as boosting and support vector machines in a number of application domains. Although Random Forests often give accuracy close to the Bayes Optimal Classifier, in some domains their accuracy can still be improved.

One way for improving Random Forests is to replace majority voting with a more sophisticated combination function. In this paper we considered how Random Forests could be combined with dynamic classifier integration techniques. Our experimental study demonstrated that dynamic integration was able to improve the accuracy of Random Forests on 12 out of 27 datasets (with DV and DVS integration strategies).

More detailed experimental analysis revealed a few interesting tendencies with respect to dynamic integration in Random Forests. Dynamic integration techniques DV and DVS were demonstrated to always increase margin in comparison with the usual Random Forests – a characteristic that is similar to that of boosting. Bias/variance analysis demonstrated that DV and DVS tend to decrease bias while keeping variance the same. DS was proven to be inappropriate in this context, always significantly increasing variance.

Among the distance functions and local learning schemes considered, the best combination was the intrinsic similarity with locally weighted learning. Interestingly, this combination usually resulted in a significantly greater margin than all the other techniques, even when accuracy remained the same. In general, the intrinsic similarity metric demonstrated very promising behaviour, and it is an interesting question for further research whether this superiority will hold true in other data mining tasks and application domains.

Regression Random Forests are also competitive in terms of predictive performance as their classification counterparts. An important topic for further research is to apply dynamic integration to regression Random Forests and to analyze the resulting predictive performance checking whether the dependencies found in classification will hold true. Our preliminary experiments demonstrate that dynamic integration is often able to improve the predictive performance of regression random forests as well and that it has similar patterns in bias/variance decomposition to those presented here for classification random forests.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the Science Foundation Ireland under Grant No. S.F.I.-02/N.1/111. This research was partly supported by the Academy of Finland.

7. REFERENCES

[1] Atkeson C., Moore A., Schaal S. Locally weighted learning. *AI Review*, 11, 1997, 11-73.

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

[3] Bingham E., Mannila H. Random projection in dimensionality reduction: applications to image and text data. In: *Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD'2001*, ACM Press, 2001, 245-250.

[4] Blake C.L., E. Keogh, C.J. Merz. UCI repository of machine learning databases [http:// www.ics.uci.edu/ ~mlearn/ MLRepository.html], Dept. of Information and Computer Science, University of California, Irvine, CA, 1999.

[5] Breiman L. Bias, Variance, and Arcing Classifiers, Tech. Report 486, Statistics Dept., University of California, Berkeley, USA, 1996.

[6] Breiman L. Random Forests, *Machine Learning*, 45(1), 2001, 5-32.

[7] Brodley C., Lane T. Creating and exploiting coverage and diversity. In: *Proc. AAAI-96 Workshop on Integrating Multiple Learned Models*, Portland, OR, 1996, 8-14.

[8] Dietterich T.G. Ensemble methods in machine learning. In: *Proc. 2nd Int. Workshop on Multiple Classifier Systems MCS'2000,* LNCS 1857, Springer, 2000, 1-15.

[9] Domingos P. A unified bias-variance decomposition for zero-one and squared loss. In: *Proc. 17th National Conf. on Artificial Intelligence and 12th Conf. on Innovative Applications of Artificial Intelligence AAAI/IAAI'2000*, AAAI Press / MIT Press, 2000, 564-569.

[10] Kohavi R., Wolpert D. Bias plus variance decomposition for zero-one loss functions. In: *Proc. 13t Int. Conf. on Machine Learning*, Morgan Kaufmann, 1996, 275-283.

[11] Robnik-Šikonja M. Improving random forests. In: J.F. Boulicaut et al. (eds.), *Proc. 15th European Conf. on Machine Learning ECML'2004*, Springer, LNCS 3201, 2004, 359-370.

[12] Rooney N., Patterson D., Anand S., Tsymbal A. Dynamic integration of regression models. In: *Proc. 5th Int. Workshop on Multiple Classifier Systems MCS'2004,* LNCS 3181, Springer, 2004, 164-173.

[13] Schaffer C. Selecting a classification method by cross-validation, *Machine Learning*, 13, 1993, 135-143.

[14] Tsymbal A., Pechenizkiy M., Cunningham P. Sequential genetic search for ensemble feature selection. In: *Proc. 19th Int. Joint Conf. on Artificial Intelligence IJCAI'2005*, Morgan Kaufmann, 2005, 877-882.

[15] Tsymbal A., Puuronen S. Bagging and boosting with dynamic integration of classifiers. In: D.A. Zighed, J. Komorowski, J. Żytkow (eds.), Principles of Data Mining and Knowledge Discovery, Proceedings of PKDD'2000, Springer, LNAI 1910, 2000, 116-125.

[16] Webb G.I. MultiBoosting: a technique for combining boosting and wagging, *Machine Learning*, 40(2), 2000, 159-196.

[17] Wilson D.R., Martinez T.R. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research,* 6(1), 1997, 1-34.

[18] Witten I., Frank E. *Data Mining: Practical Machine Learning Tools With Java Implementations*, San Francisco: Morgan Kaufmann, 2000.