Chapter 12
Embedding Random Projections in Regularized Gradient Boosting Machines

Pierluigi Casale, Oriol Pujol, and Petia Radeva

Abstract. Random Projections are a suitable technique for dimensionality reduction in Machine Learning. In this work, we propose a novel Boosting technique that is based on embedding Random Projections in a regularized gradient boosting ensemble. Random Projections are studied from different points of view: pure Random Projections, normalized and uniform binary. Furthermore, we study the effect to keep or change the dimensionality of the data space. Experimental results performed on synthetic and UCI datasets show that Boosting methods with embedded random data projections are competitive to AdaBoost and Regularized Boosting.

12.1 Introduction

Random Projections (RPs) have been widely employed as dimensionality reduction technique. RPs are based on the idea that high dimensional data can be projected into a lower dimensional space without significantly losing the structure of the data. RPs can also be viewed as a tool for generating diversity in the creation of an ensemble of classifiers. The underlying idea is the same used in various new ensemble methods such as Rotation Forest [11] or Rotboost [13]. Using RPs, the different embeddings of the original feature space provide multiple view of the original features space. This kind of diversity can be generated while projecting data into subspaces, the space having the dimensionality of the original feature space or even spaces of higher dimensionality than the original one. Here, different Random Projections are studied and applied in the construction of a Boosting ensemble.

Pierluigi Casale
Computer Vision Center, Barcelona, Spain
E-mail: pierluigi@cvc.uab.es

Oriol Pujol · Petia Radeva
Dept. of Applied Mathematics and Analysis, University of Barcelona, Barcelona, Spain
Computer Vision Center, Barcelona, Spain
E-mail: {oriol,petia}@maia.ub.es

O. Okun et al. (Eds.): Ensembles in Machine Learning Applications, SCI 373, pp. 201–216.
springerlink.com

From the point of view of incremental optimization, AdaBoost can be viewed as an additive model fitting procedure that approximates the optimization of an exponential loss function. Changing the exponential loss function with a least square loss function yields to a new model of Boosting, known as LsBoost [7]. Gradient Boosting Machines (GBMs) generalize this idea for any arbitrary loss function. In this study, RPs have been integrated into the LsBoost algorithm. The stepwise approximation is obtained by projecting data onto random spaces at each step of the optimization process and searching for the classifier that best fits the data in the new space. Nevertheless, fitting the data too closely yields to poor results. Regularization methods attempt to prevent the over-fitting by constraining the fitting procedure. For that reason, a study on the effect of the L2 penalization term has also been conducted.

The approach has been evaluated on synthetic and real problems datasets. The new method has been compared with AdaBoost and LsBoost. Results show that the new method is competitive with respect to AdaBoost and LsBoost and, for some specific problems, using RPs significantly improves the classification accuracy. Additionally, results show that the use of the L2 regularization parameter is specially justified as a way to avoid overfitting and model noise ensuring smoothness in the solutions.

This chapter is organized as follows. In the next section, previous works about RPs in the Machine Learning are briefly reported. In Sect. 12.3, GBMs and RPs are formally introduced and the proposed method for embedding RPs into GMBs is described. In Sect. 12.4, results are reported and finally, in Sect. 12.5, we give conclusions.

12.2 Related Works on RPs

Arriaga and Vempala [1] propose an algorithmic theory of learning based on RPs and robust concepts. They show how RPs are a suitable procedure for reducing dimensionality while preserving the structure of the problem. In their work, they proposed a very simple learning algorithm based on RPs mainly consisting in two steps: randomly projecting the data into a random subspace and running the algorithm in that space, taking advantage of working with a lower dimensionality. Blum [2] reports this basic algorithm showing how, if a learning problem is separable with a large margin, the problem still remains separable in a reduced random space. Moreover, even picking a random separator on data projected down to a line, provides a reasonable change to get a weak hypothesis as well. Dasgupta [3] used RPs with Gaussian mixture models for classification of both synthetic and real data. In his work, data are projected into a randomly chosen d-dimensional subspace and the learning algorithm works in this new smaller space, achieving highly accurate classification results. In the context of supervised learning, Fradkin and Madigan [5] compare the performances of C4.5, Nearest Neighbours and SVM, using both PCA and RPs as dimensionality reduction technique. The results of their experiments are always favorable to PCA. More recently, Rahimi and Recht [10] use also Random

Projections for building a weighted sum of linear separators. In their work, authors show that using Random Projections is equivalent to use the kernel trick. At the same time, Random Projections provide a faster decaying of the testing error rate with respect to the standard AdaBoost.

12.3 Methods

In this section, the formulation of both GBMs and RPs is presented. Starting from LsBoost algorithm, a particular definition of GBMs, a new method for embedding RPs into the GBMs is described. The method is a slight modification of the original LsBoost algorithm. In the modified version, training data are projected onto random spaces where the best classifier fitting the data is found.

12.3.1 Gradient Boosting Machines

In regression and classification problems, given a set of training sample $\{y_i,\mathbf{x}_i\}_1^N$, we look for a function $F^*(\mathbf{x})$ that maps $\mathbf{x}$ to y such that, over the joint distribution of all $(y,\mathbf{x})$-values, the expected value of some specified loss function $\Psi(y,F(x))$ is minimized. Usually, the function $F(\mathbf{x})$ is member of parameterized class of functions $F(\mathbf{x};\mathbf{P})$:

$$F(\mathbf{x};\mathbf{P}) = \sum_{m=0}^{M} \beta_m h(\mathbf{x};\mathbf{a}_m) \,, \tag{12.1}$$

where $\mathbf{P} = \{\beta_m, \mathbf{a}_m\}_0^M$ is a set of parameters. Nevertheless, we can consider $F(\mathbf{x})$ evaluated at each point $\mathbf{x}$ to be a parameter and minimize:

$$\Phi(F(\mathbf{x})) = E_y[\Psi(y,F(\mathbf{x})|\mathbf{x})], \tag{12.2}$$

at each individual $\mathbf{x}$, directly with respect to $F(\mathbf{x})$. The solution is of the type:

$$F^*(\mathbf{x}) = \sum_{m=0}^{M} f_m(\mathbf{x}) \,, \tag{12.3}$$

where $f_0(\mathbf{x})$ is an initial guess, and $\{f_m\}_1^M$ are incremental functions, known as "steps" or "boosts". Using steepest-descent, we get :

$$f_m(\mathbf{x}) = -\rho_m g_m(\mathbf{x}) \,, \tag{12.4}$$

where, assuming that differentiation and integration can be interchanged,

$$g_m(\mathbf{x}) = E_y\left[\frac{\partial \Psi(y,F(\mathbf{x}))}{\partial F(\mathbf{x})}|\mathbf{x}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})} \tag{12.5}$$

and

$$F_{m-1}(\mathbf{x}) = \sum_{i=0}^{m-1} f_i(\mathbf{x}). \tag{12.6}$$

When the joint distribution of $(y,\mathbf{x})$ is represented by a finite data sample, $E_y[\cdot|\mathbf{x}]$ cannot be evaluated accurately at each $\mathbf{x}_i$ and, if we could perform parameter optimization, the solution is difficult to obtain. In this case, given the current approximation $F_{m-1}(\mathbf{x})$ at the m-th iteration, the function $\beta_m h(\mathbf{x};\mathbf{a})$ is the best greedy step towards the minimizing solution $F^*(\mathbf{x})$, under the constraint that the step direction $h(\mathbf{x},\mathbf{a}_m)$ be a member of the parameterized class of functions $h(\mathbf{x},\mathbf{a})$. One possibility is to choose the member of the parameterized class $h(\mathbf{x};\mathbf{a})$ that is most parallel in the N-dimensional data space with the unconstrained negative gradient $\{-g_m(\mathbf{x}_i)\}_1^N$. In this case, it is possible to use $h(\mathbf{x},\mathbf{a}_m)$ instead of the unconstrained negative gradient $-g_m(\mathbf{x})$. Weights ρ_m are given by the following line search:

$$\rho_m = argmin_\rho \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i;\mathbf{a}_m)) \tag{12.7}$$

and the approximation updated in the following way:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x};\mathbf{a}_m). \tag{12.8}$$

When $y \in \{-1,1\}$ and the loss function $\Psi(y,F)$ depends on y and F only through their product $\Psi(y,F) = \Psi(yF)$, the algorithm reduces to Boosting. If the loss function is $\Psi(y,F) = \frac{(y-F)^2}{2}$, gradient boosting produces the stagewise approach of iteratively fitting the current residuals. The algorithm, shown in Table 12.1, is called LsBoost.

Table 12.1 LsBoost Algorithm

1.	$F_0(\mathbf{x}) = argmin_\rho \sum_{i=1}^N \Psi(y_i,\rho)$
2.	For $m = 1$ to M or meanwhile $error > \varepsilon$ do:
3.	$\tilde{y}_i^m = y_i - F_{m-1}, i = 1,N$
4.	$(\rho_m,\mathbf{a}_m) = argmin_{\mathbf{a},\rho} \sum_{i=1}^N [\tilde{y}_i^m - \rho h(\mathbf{x}_i;\mathbf{a})]^2$
5.	$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}_i;\mathbf{a}_m)$
6.	End

12.3.2 Random Projections

RPs are techniques that allow to reduce the dimensionality of a problem while still retaining a significant degree of the structure of the data. The Johnson-Lindenstrauss Lemma [8] states that, given m points in $\Re^n$, it is possible to project these points into a d-dimensional subspace, with $d = O(\frac{1}{\gamma^2} log(m))$. In this space, relative distances and angles between all pairs of points are approximately preserved up to $1 \pm \gamma$, with high probability.

Formally, given $0 < \gamma < 1$, a set X of m points in $\Re^N$, and a number $n > n_0 = O(\frac{1}{\gamma^2} log(m))$, there is a Lipschitz function $f : \Re^N \rightarrow \Re^n$ such that

$$(1-\gamma)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\gamma)\|u-v\|^2. \tag{12.9}$$

If m data points in a feature space are considered row-vectors of length N, the projection can be performed by multiplying all the m points by a randomly generated $N \times n$ matrix. The random matrix should be one of the following types:

- P with columns to be d pure random orthonormal vectors;
- $U_{-1,1}$ with each entry to be 1 or -1 drawn independently at random;
- $N_{0,1}$ with each entry drawn independently from a standard Normal Distribution $N(0,1)$.

While using these types of projections ensures that relative distances and angles are approximately preserved, there is no guarantee that using other types of matrices could preserve the structure of the data. Though data might be projected down to much lower dimensions or into the same space [2], projecting data onto spaces of higher dimension does not rely on any theoretical results. Here, the possibility to project data in a superspace is also taken into account.

12.3.3 Random Projections in Boosting Machine

The original algorithm of LsBoost has been slightly modified to be adapted to RPs. In the line 4 of the algorithm in Table 12.1, it is possible first searching analytically for the optimal set of weighting values for each candidate classifier and select the classifier that best approximates the negative gradient with the corresponding precomputed weight [9]. It is possible to find the optimal weighting value for each candidate classifier $h(x;a)$ by:

$$\frac{\partial[(\tilde{\mathbf{y}}^m - \rho_a^T h(x;a))^T(\tilde{\mathbf{y}}^m - \rho_a^T h(x;a))]}{\partial \rho_a} = 0 \tag{12.10}$$

solved by

$$\tilde{\mathbf{y}^m}^T h(x;a) = \rho_a^T h(x;a)^T h(x;a) = \rho_a^T N. \tag{12.11}$$

where, since $h(x;a) \in \{+1,-1\}$, the dot product $h(x;a)^T h(x;a)$ is just the number of training examples and the regularization parameter might simply be added. Therefore, the optimal set of weights is given by Eq.(12.12)

$$\rho_m = \frac{\tilde{\mathbf{y}}^m A}{N+\lambda}, \tag{12.12}$$

where $\tilde{\mathbf{y}}^m$ denotes the vector of residuals at step m, A is the matrix of training examples, N is the number of training examples and λ represents the L2 penalization

term. For $\lambda = 0$, regularization is not taken into account. Once the optimal set of values is found, a simple selection of the classifier that best approximates the negative gradient can be performed. The described procedure can be performed on training data projected onto random spaces.

Therefore, *RpBoost* is defined as Regularized Gradient Boosting where before considering the data by the weak classifiers, they are projected by a transformation represented by a specific RPs technique. Table 12.2 defines the RpBoost algorithm. In addition, the following is defined :

Definition 12.1. *Rpboost.sub* as RpBoost working of data projected into a random subspace;

Definition 12.2. *Rpboost.same* as RpBoost working of data projected into a random space of the same dimension than the original feature space;

Definition 12.3. *Rpboost.super* as RpBoost working of data projected into a random superspace.

Table 12.2 RpBoost Algorithm

Select the type of projection in $\{P, N_{0,1}, U_{-1,1}\}$
Set the dimension of the random space
Set the random matrix R_p
1. $F_0(\mathbf{x}) = argmin_\rho \sum_{i=1}^{N} \Psi(y_i, \rho)$
2. For $m = 1$ to M do:
3. $\tilde{y}_i^m = y_i - F_{m-1}, i = 1, N$
4. Set a new R_p
5. $A_r = AR_p$
6. $\rho_m = \frac{\tilde{y}_m A_r}{N+\lambda}$
7. $\mathbf{a}_m = argmin_{\mathbf{a}} \sum_{i=1}^{N} [\tilde{y}_i^m - \rho_m h(\mathbf{x}_i; \mathbf{a})]^2$
8. $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}_i; \mathbf{a}_m)$
9. End

12.4 Experiments and Results

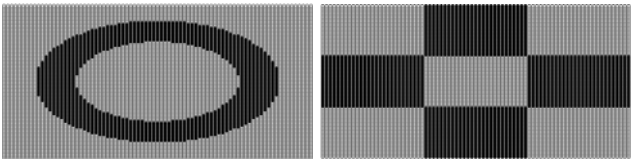
In order to test the performance of RpBoost, several tests on synthetic and real problems have been performed. RpBoost is compared with AdaBoost and LsBoost. In this setting, decision stumps are used as weak classifiers. The maximum dimension of the ensemble has been set to 500 classifiers. In order to have a straightforward comparison, in all the experiments, the dimension of the subspace and of the superspace have been set equal to the half and the double of the dimension of the feature space, respectively RPs are performed using a matrix $M \in \{P, U_{-1,1}, N_{0,1}\}$. Finally, results related to the effect of regularization in the creation of the ensemble are shown in the last section.

12.4.1 Test Patterns

Test patterns are synthetic bidimensional datasets proposed by Fawcett [4] for comparing classifiers. The patterns are randomly generated on a 2-D grid of points, between [0:4] x [0:4] with a resolution of 0.05, yielding 6561 total points. The points are labeled based on where they fall in the pattern. In Table 12.3, the formal description of patterns is reported. In Fig. 12.1, examples of the patterns are presented.

Table 12.3 Test Patterns

Test Pattern	Description
Sine	$Y = 0.84sin(1.78X)$
Linear	$Y = 1.87X \pm 1.74$
Parity	9 parity circles
Annulus	Annulus at (2.00, 2.00)
Parabolic	$Y = \frac{(X-2)^2}{4*0.25+1}$
Disjunctive	4 disjoint concave polygons
Polynomial	$Y = \frac{1}{2}(x-2)^3 + \frac{1}{2}(x-2.2)^2 + 2$
Checkerboard	9 squares alternating classes



(a) (b)

Fig. 12.1 Examples of Test Patterns:(a) *Annulus* Test Pattern;(b) *Checkerboard* Test Pattern

Validation Procedure. For each test pattern, a stratified sample of 1000 points is used for training and the complete distribution as testing. The validation procedure has been performed five times and results are averaged.

Results. Comparative results of RpBoost are reported in Fig. 12.2 for P projections and in Fig. 12.3 for $U_{-1,1}$ projections. RpBoost performs slightly better than AdaBoost and LsBoost using both types of projections on some patterns. Additionally, RpBoost.sub with P projections always performs considerably worst than RpBoost.same and RpBoost.super and RpBoost.super with $U_{-1,1}$ always performs worst the RpBoost.same and RpBoost.sub. In the *parity* and in the *checkerboard* test patterns, RpBoost always performs better than the compared methods, disregarding the type of projection.

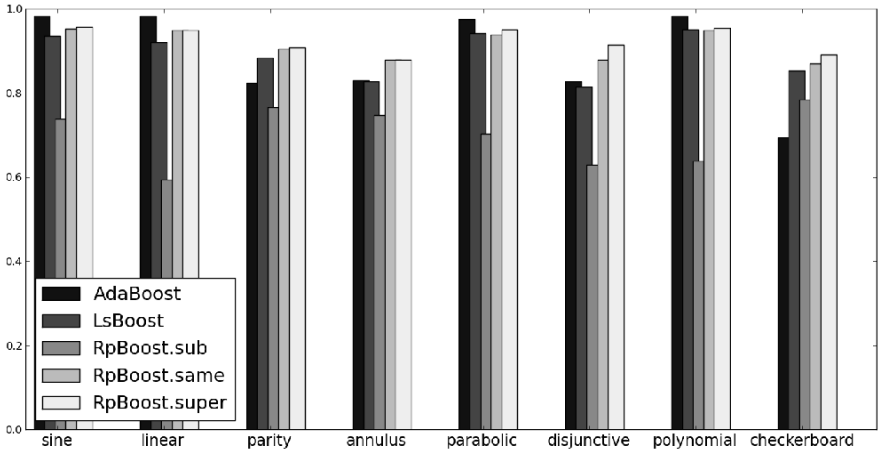


Fig. 12.2 Comparative Results of RpBoost with P projections on Test Patterns

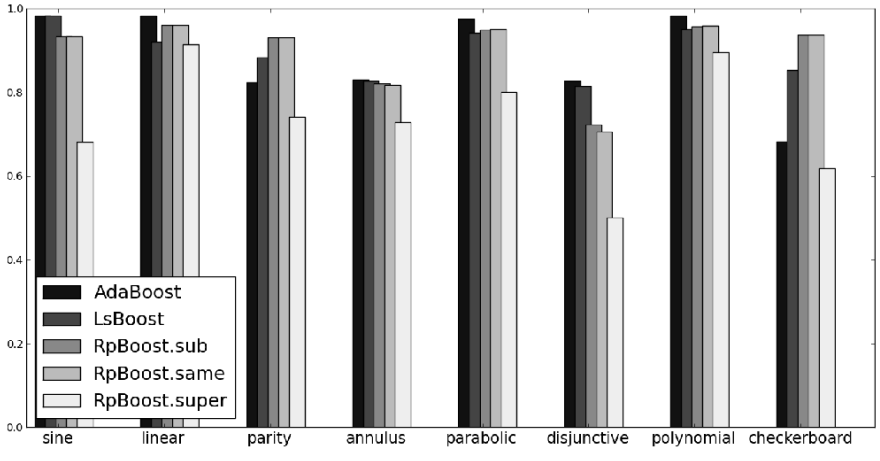


Fig. 12.3 Comparative Results of RpBoost with $U_{-1,1}$ projections on Test Patterns

In Fig. 12.4, comparative results using $N_{0,1}$ projections are shown. Numerical results are reported in Table 12.4. RpBoost.sub and RpBoost.same always provide the best performance. In particular, in the *annulus* and in the *checkerboard* pattern the classification accuracy of RpBoost is considerably improved. Even for the $N_{0,1}$ projections, projecting data to superspaces does not provide benefits for the classification.

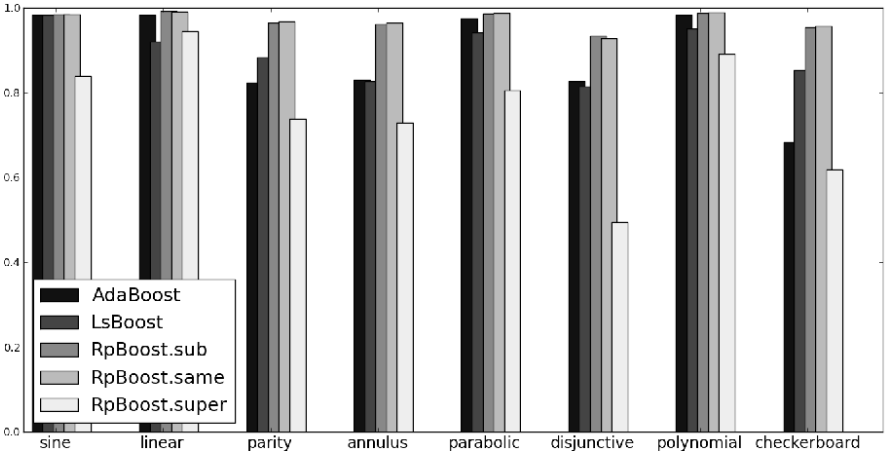


Fig. 12.4 Comparative Results of RpBoost with $N_{0,1}$ projections on Test Patterns

Table 12.4 Numerical Values of Accuracy on Test Patterns obtained with AdaBoost (Ada), LsBoost (Ls), RpBoost.sub (Sub) , RpBoost.same (Same) and RpBoost.super (Super) with $N_{0,1}$ projections

	Ada	Ls	Sub	Same	Super
sine	0.983	0.937	**0.986**	0.985	0.84
linear	0.984	0.921	**0.993**	0.992	0.946
parity	0.824	0.884	0.966	**0.968**	0.738
annulus	0.83	0.828	0.963	**0.965**	0.73
parabolic	0.976	0.943	0.987	**0.989**	0.806
disjunctive	0.827	0.816	**0.935**	0.928	0.495
polynomial	0.984	0.951	0.988	**0.99**	0.892
checkerboard	0.694	0.854	0.955	**0.957**	0.62

12.4.2 UCI Datasets

RpBoost, AdaBoost and LsBoost have been compared on eight datasets from UCI Repository [6]. The results are reported in Table 12.5, with the number of elements per class. All the datasets selected are binary classification problems.

Validation Procedure. Results have been obtained using 10-folds cross-validation. The procedure has been run two times and results have been averaged. The value of the regularization parameter has been selected using 5-fold cross validation on the training set.

Table 12.5 List of UCI Datasets

Dataset	Elements
Monks-1	272,284
Monks-2	300,301
Monks-3	275, 279
Breast	239,485
Liver	100,245
Tic-Tac-Toe	626,332
Ionosphere	126,225
Sonar	97,111

Results. Comparative results of RpBoost are reported in Fig. 12.5 for P projections and in Fig. 12.6 for $U_{-1,1}$ projections. As for synthetic data, there exist problems where RpBoost outperforms AdaBoost and LsBoost. In particular, only for P projections, RpBoost.super provides better classification accuracies. In Fig. 12.7, the mean accuracy obtained with AdaBoost, LsBoost and RpBoost using $N_{0,1}$ is shown. Numerical values are reported in Table 12.6. In Monks-1 and Monks-2 RpBoost outperforms AdaBoost and LsBoost. In Monks-3, performance is slightly improved. A slight improvement can also be noted in Breast, Sonar and Ionoshpere – the datasets having the highest dimensions; it seems that there are no benefits from the dimensionality reduction that RPs provide.

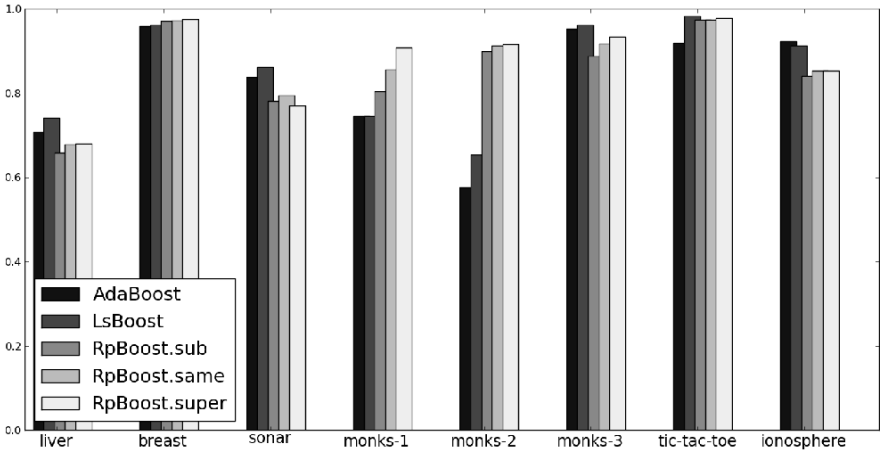


Fig. 12.5 Comparative Results of RpBoost with P projections on UCI datasets

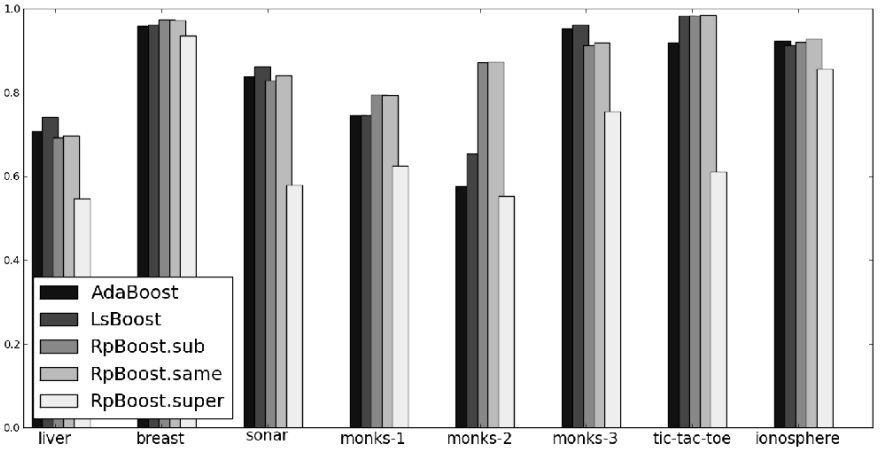


Fig. 12.6 Comparative Results of RpBoost with $U_{-1,1}$ projections on UCI datasets

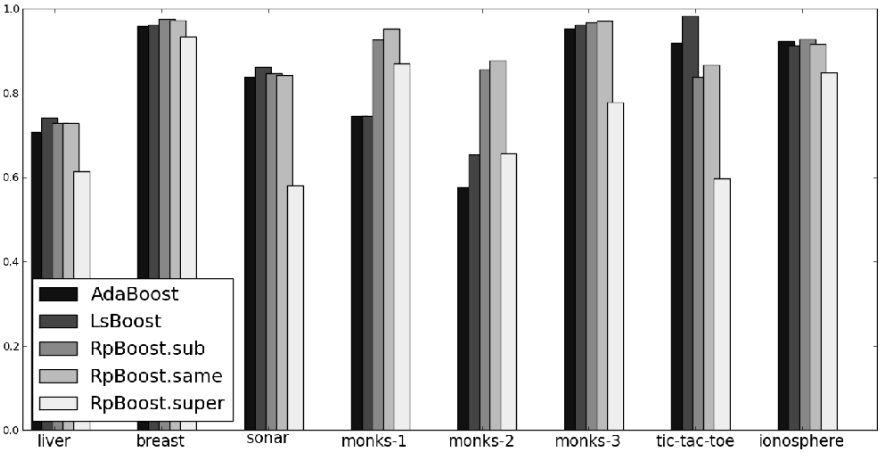


Fig. 12.7 Classification Accuracy on UCI Datasets obtained with AdaBoost, LsBoost and RpBoost with $N_{0,1}$ projections

12.4.3 The Effect of Regularization in RpBoost

In order to study the effect of the regularization parameter λ, a 10-fold cross validation over two runs of cross validation for

$$\lambda \in \{1, 5, 10, 50, 100, 500, 1000, 5000, 10000\}$$

for each dataset and for each type of projection has been performed. From the experiments, it is evident how the effect of the regularization parameter can be noted

Table 12.6 Numerical Values of Accuracy on Uci Datasets obtained with AdaBoost (Ada), LsBoost (Ls), RpBoost.sub (Sub), RpBoost.same (Same) and RpBoost.super (Super) with $N_{0,1}$ projections

	Ada	Ls	Sub	Same	Super
liver	0.708	**0.742**	0.692	0.698	0.547
breast	0.959	0.962	**0.974**	0.973	0.937
sonar	0.839	**0.862**	0.829	0.841	0.579
monks-1	0.746	0.746	**0.796**	0.794	0.625
monks-2	0.577	0.654	0.872	**0.873**	0.554
monks-3	0.953	**0.963**	0.913	0.919	0.756
tic-tac-toe	0.92	0.983	0.983	**0.986**	0.611
ionosphere	0.924	0.914	0.921	**0.928**	0.857

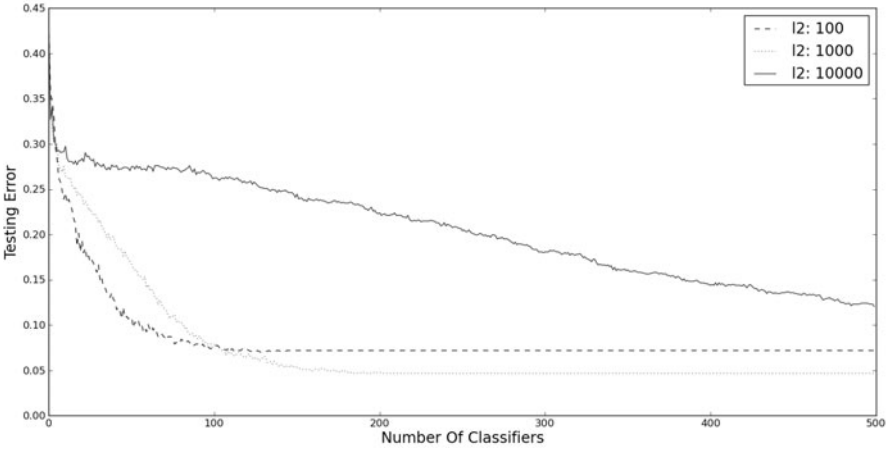


Fig. 12.8 Testing Error of RpBoost.same using $N_{0,1}$ in Monks-1 dataset for different regularization parameters.

at the initial phase of the construction of the ensemble, before the convergence of the optimization process. The principal benefits of using a proper value for λ are tied to the speed of the optimization process. In Fig. 12.8, the testing error RpBoost.same using $N_{0,1}$ projections on the Monks-1 dataset is shown. Values of λ are in $\{100, 1000, 10000\}$. Here, a typical trend is shown where it is possible to see how the optimization process converges slower when the value of λ increases. It is also evident how, in the former steps of the optimization process, λ influences the construction of the ensemble. In Fig. 12.9, the testing error RpBoost.same using $U_{-1,1}$ projections on the Liver dataset is shown. Here, it is evident how, with a proper value of λ, the testing error rapidly slows down and the overfitting is prevented. It is evident that for $\lambda = 100$, overfitting is present. In Fig. 12.10, the testing error

RpBoost.same using $U_{-1,1}$ projections on the Breast dataset is shown. In this figure, overfitting is more evident and, in particular, only for $\lambda = 10000$ the classifier does not tend to overfit. We should also note about the order of magnitude of the regularization parameter. In the last case, a very big value is needed. In Eq. (12.12) the regularization parameter is present in the denominator. This fact means that very little weights are needed in the step-wise approximation process.

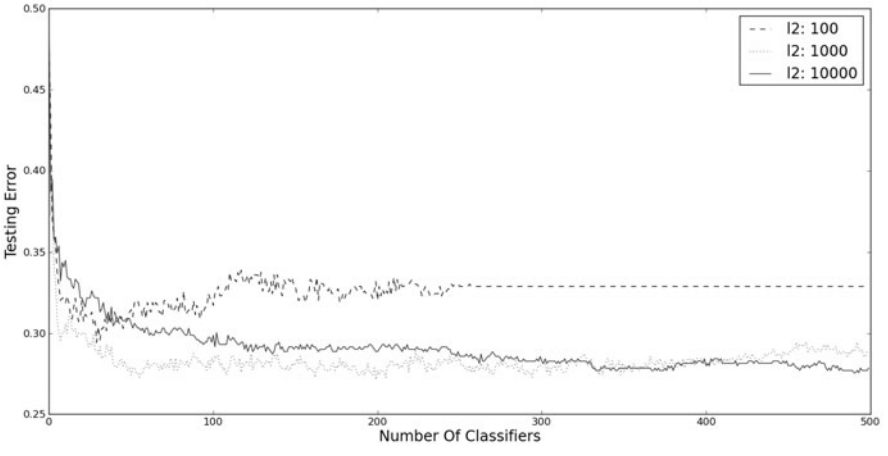


Fig. 12.9 Testing Error of RpBoost.same using $U_{-1,1}$ in Liver dataset for different regularization parameters.

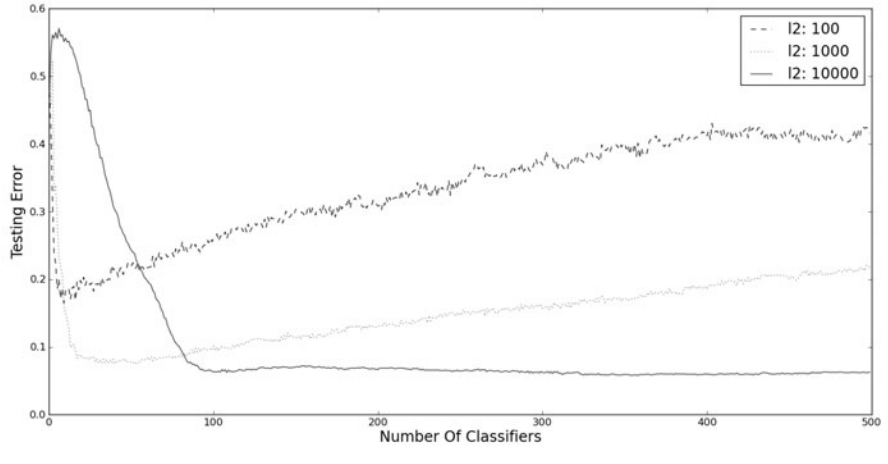


Fig. 12.10 Testing Error of RpBoost.super using $U_{-1,1}$ in Breast dataset for different regularization parameters.

12.4.4 Discussion

In Table 12.7, best classification accuracies obtained on both test patterns and Uci Datasets are reported. For each test pattern, accuracy, classifier and type of projections R_p are shown. For UCI datasets, the value of regularization parameter λ is shown too. RpBoost always performs better than AdaBoost. In addition, RpBoost always provides the best accuracy in the classification of synthetic data using projections drawn from a normal distribution. Very significant improvements are reported in the *annulus* pattern and in the *checkerboard*, *parity* and *disjunctive* patterns where performance increased by more than 10%. In Fig. 12.11, the classification of the *annulus* is shown. Although classification is not perfect, the effect of using RPs is evident. RPs allow to follow the non linear boundary even when using linear weak classifiers as decision stumps. Figure 12.12 shows the classification of the *checkerboard* pattern. Here, in contrast to the others, RpBoost is capable to

Table 12.7 Resume of Results

Test Pattern	Accuracy	Classifier	R_p	Dataset	Accuracy	Classifier	R_p	λ
Sine	98.6%	Sub	N	Liver	74.2%	Ls	-	1000
Linear	99.3%	Sub	N	Breast	97.6%	Super/Sub	P N	10000/ 10000
Parity	96.8%	Same	N	Sonar	86.2%	Ls	-	1000
Annulus	96.5%	Same	N	Monks-1	95.3%	Same	N	1000
Parabolic	98.9%	Same	N	Monks-2	91.6%	Super	P	1000
Disjunctive	93.5%	Sub	N	Monks-3	97.2%	Same	N	1000
Polynomial	99.0%	Same	N	Tic-tac-toe	98.6%	Same	U	10
Checkerboard	95.7%	Same	N	Ionosphere	92.8%	Same/Sub	U N	5000/1000

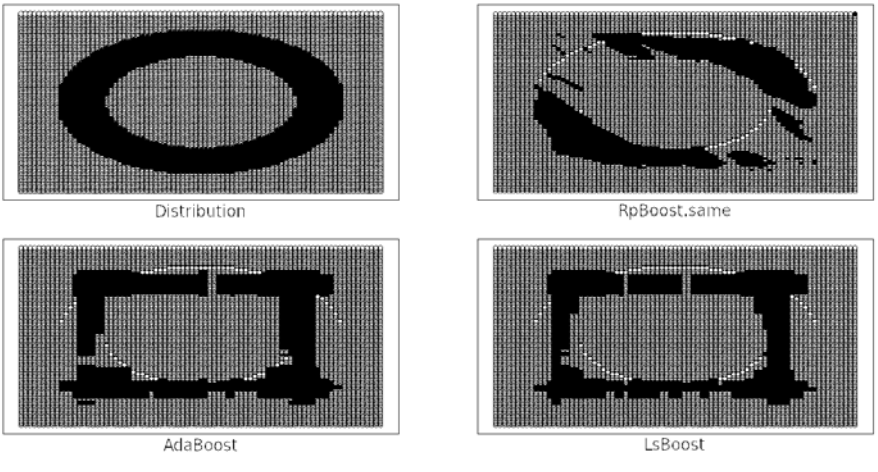


Fig. 12.11 Classification of the *annulus* test pattern using RpBoost, AdaBoost and LsBoost

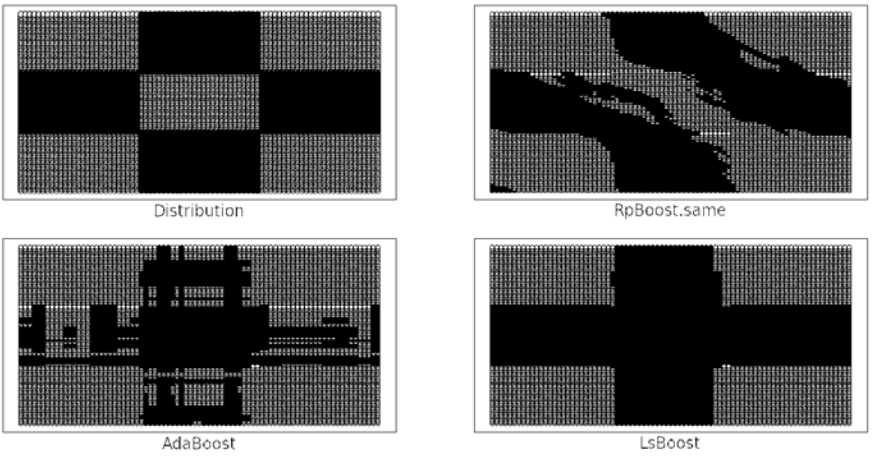


Fig. 12.12 Classification of the *checkerboard* test pattern using RpBoost, AdaBoost and LsBoost

grasp the different class in the central part of the pattern. *Checkerboard* represents a XOR-type problem. Similarly, *parity* and *disjunctive* represent XOR-type problems. On these problems, RpBoost behaves as in the case previously analyzed. This fact is confirmed in the Monks-1 and Monks-2 datasets, both representing XOR-type problems [12]. In these cases, too, the performance of RpBoost is considerably improved, compared to the performance obtained with AdaBoost or LsBoost.

12.5 Conclusion

In this work, Random Projections are used to generate diversity in the construction of regularized Gradient Boosting Machines. In particular, RPs are embedded in a modified version of LsBoost, named RpBoost. At each step of the optimization process, data are projected into a random space and, in the new space, the classifier that best fits the data is selected to be added to the ensemble. Projecting spaces can be subspaces, random spaces of the same dimension than the original feature space and random superspaces.

RpBoost always performs better than AdaBoost on synthetic data and, in the majority of the cases, performs better than LsBoost on real data especially when projections into subspaces or space of the same dimension than the original spaces are used. In these spaces, RpBoost performs well with all types of projections on most of the problems. The use of superspaces yields to better classification accuracy only when the projection is drawn completely at random. In this case, the performance appears to be slightly better than with other types of projections.

The regularization parameter influences the creation of the ensemble, in particular, when high values of regularization are provided. Finding the “optimal” value for

the regularization parameter is crucial especially when there exists a trend to overfitting. Obviously, in the cases where overfitting is present, using a small number of classifiers in the ensemble would have to provide better classification accuracy.

Finally, results clearly show that RpBoost is a promising technique and encourages future research with studies on real-world problems.

Acknowledgements. This work is partially supported by a research grant from projects TIN2009-14404-C02, La Marato de TV3 082131 and CONSOLIDER (CSD2007-00018).

References

1. Arriaga, R.I., Vempala, S.: An algorithmic theory of learning: Robust concepts and random projection. Machine Learning 63, 161–182 (2006)
2. Blum, A.: Random projection, margins, kernels, and feature-selection. In: Saunders, C., Grobelnik, M., Gunn, S., Shawe-Taylor, J. (eds.) SLSFS 2005. LNCS, vol. 3940, pp. 52–68. Springer, Heidelberg (2006)
3. Dasgupta, S.: Experiments with random projection. In: Proc. the 16th Conf. Uncertainty in Artif. Intell., Stanford, CA, pp. 143–151. Morgan Kaufmann, San Francisco (2000)
4. Fawcett, T.: Comparing patterns classifiers, `http://home.comcast.net/ tom.fawcett/public_html/ML--gallery/pages/index.html`
5. Fradkin, D., Madigan, D.: Experiments with random projections for machine learning. In: Proc. the 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, Washington, DC, pp. 517–522. ACM Press, New York (2003)
6. Frank, A., Asuncion, A.: UCI machine learning repository. University of California, School of Information and Computer Sciences, Irvine (2010)
7. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Annals of Stat. 29, 1189–1232 (2000)
8. Johnson, W.B., Lindenstrauss, J.: Extensions of Lipschitz maps into a Hilbert space. Contemporary Mathematics 26, 189–206 (1984)
9. Pujol, O.: Boosted geometry-based ensembles. In: El Gayar, N., Kittler, J., Roli, F. (eds.) MCS 2010. LNCS, vol. 5997, pp. 195–204. Springer, Heidelberg (2010)
10. Rahimi, A., Recht, B.: Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In: Advances in Neural Inf. Proc. Syst., vol. 21, pp. 1313–1320. MIT Press, Cambridge (2008)
11. Rodriguez, J.J., Kuncheva, L.I., Alonso, C.J.: Rotation Forest: A new classifier ensemble method. IEEE Trans. Pattern Analysis and Machine Intell. 28, 1619–1630 (2006)
12. Thrun, S., Bala, J., Bloedorn, E., Bratko, I., Cestnik, B., Cheng, J., De Jong, K., Dzeroski, S., Hamann, R., Kaufman, K., Keller, S., Kononenko, I., Kreuziger, J., Michalski, R.S., Mitchell, T., Pachowicz, P., Roger, B., Vafaie, H., Van de Velde, W., Wenzel, W., Wnek, J., Zhang, J.: The MONK's problems: A performance comparison of different learning algorithms. Technical Report CMU-CS-91-197, Carnegie Mellon University (1991)
13. Zhang, C.-X., Zhang, J.-S.: RotBoost: A technique for combining Rotation Forest and AdaBoost. Pattern Recogn. Letters 29, 1524–1536 (2008)