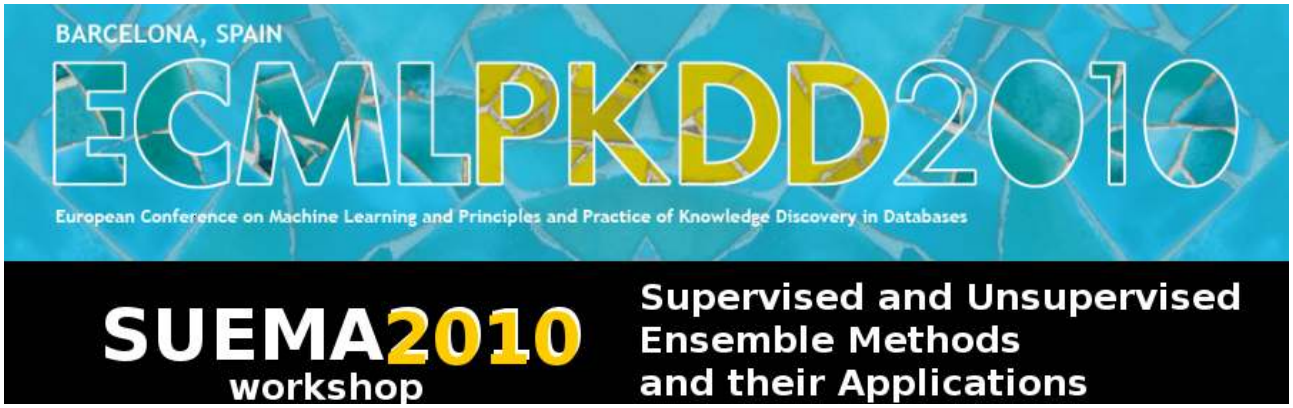
Oleg Okun, Matteo Re and Giorgio Valentini (Eds.)



SUEMA 2010 Proceedings

20th September 2010, Barcelona – Spain

Workshop Chairs

Oleg Okun,
Malmo, Sweden
E-mail: olegokun@yahoo.com

Matteo Re,
DSI - Department of Computer Science, University of Milan
Url: http://homes.dsi.unimi.it/~re/
E-mail: re@dsi.unimi.it

Giorgio Valentini,
DSI - Department of Computer Science, University of Milan
Url: http://homes.dsi.unimi.it/~valenti/
E-mail: valentini@dsi.unimi.it

Programme Committee

Nicolo' Cesa-Bianchi, University of Milano, Italy
Carlotta Domeniconi, George Mason University, USA
Robert Duin, Delft University of Technology, the Netherlands
Mark Embrechts, Rensselaer Polytechnic Institute, USA
Ana Fred, Technical University of Lisboa, Portugal
Joao Gama, University of Porto, Portugal
Giorgio Giacinto, University of Cagliari, Italy
Larry Hall, University of South Florida, USA
Ludmila Kuncheva, University of Wales, UK
Francesco Masulli, University of Genova, Italy
Petia Radeva, Autonomous University of Barcelona, Spain
Juan Jose' Rodriguez, University of Burgos, Spain
Fabio Roli, University of Cagliari, Italy
Paolo Rosso, Polytechnic University Valencia, Spain
Carlo Sansone, Federico II University of Napoli, Italy
Jose' Salvador Sanchez, University Jaume I, Spain
Lambros Skarlas University of Patras, Greece
Grigorios Tsoumakas, Aristotle University of Thessaloniki, Greece
Jordi Vitria', Autonomous University of Barcelona, Spain
Ioannis Vlahavas, Aristotle University of Thessaloniki, Greece
Terry Windeatt, University of Surrey, UK

Table of contents

A brief introduction and acknowledgments

Ensembles of supervised learning machines and, in particular, ensembles of classifiers have been established as one of the main research topics in machine learning. Methods for combining unsupervised clusterings have been recently proposed to improve the reliability of clustering algorithms and to assess the validity of discovered clusters. Statistical, algorithmic, representational, computational and practical reasons can explain the success of ensemble methods. Nevertheless, several problems remain open: for instance in many cases the theoretical reasons of the practical success of several widely used ensemble methods is unclear; the relationships between the diversity and accuracy of base classifiers forming an ensemble and the impact of these characteristics on the effectiveness and the performances of ensemble methods is a controversial question among machine learning researchers.
Though ensemble methods are subject to intensive research, there are also other open questions, related to real-world applications of such methods.
The SUEMA workshop intends to provide a forum for researchers in the field of Machine Learning and Data Mining to discuss topics related to ensemble methods and their applications.
The SUEMA 2010 program comprises 14 accepted papers submitted by scientists from Spain, UK, France, Germany, the Netherlands, Lithuania, Greece, Turkey, USA and Italy, with topics embracing the design and analysis of new ensemble methods and their real-world applications.
This year, SUEMA is proud to host the PASCAL2 invited speaker, *Grigorios Tsoumakas* (University of Thessaloniki, Greece). His talk introduces multi-label learning, an emerging topic in ensemble methods with applications ranging from semantic annotation of images and video, to web page categorization, direct marketing and functional genomics.
PASCAL2 (Pattern Analysis, Statistical Modelling and Computational Learning) is a Network of Excellence funded by the Seventh Framework Programme of the European Union. Its main research focus is on the emerging challenges created by the ever expanding applications of adaptive systems technologies and their central role in the development and application of large scale cognitive systems to real-world problems. From this standpoint the research topics of the workshop largely fall within the scope of PASCAL2.

The workshop chairs would like to thank the *PASCAL 2 Network of Excellence* and the *Dipartimento di Scienze dell'Informazione* (*Università degli Studi di Milano*, Italy) for the partial financial support of this event.

Oleg Okun, *Matteo Re* and *Giorgio Valentini*

PASCAL 2 invited talk:
Grigorios Tsoumakas,
Aristotle University of Thessaloniki, Greece.
e-mail: greg@csd.auth.gr

Ensemble Methods for Multi-Label Data (abstract)

Multi-label data are data consisting of training examples that are associated with a subset of a finite set of labels. Nowadays, such data are becoming ubiquitous. They arise in an increasing number and diversity of applications, such as semantic annotation of images and video, web page categorization, direct marketing, functional genomics and music categorization into genres and emotions. Multi-label data offer an exciting playground for the development of new ensemble methods, because of the additional dimension of the label space. In fact, several recent state-of-the-art algorithms for multi-label learning are ensemble methods. This talk introduces the task of learning from multi-label data and presents recently developed state-of-the-art ensemble methods for multi-label data. It concludes with an inverse interplay between the two areas, where multi-label learning is used for instance-based pruning of an ensemble of classifiers.

Fast-Ensembles of Minimum Redundancy Feature Selection

Benjamin Schowe and Katharina Morik

Technische Universität Dortmund
{schowe,morik}@ls8.cs.tu-dortmund.de

Abstract. Finding relevant subspaces in very high-dimensional data is a challenging task not only for microarray data. The selection of features must be stable, but on the other hand learning performance is to be increased. Ensemble methods have succeeded in the increase of stability and classification accuracy, but their runtime prevents them from scaling up to real-world applications. We propose two methods which enhance correlation-based feature selection such that the stability of feature selection comes with little or even no extra runtime. We show the efficiency of the algorithms analytically and empirically on a wide range of datasets.

1 Introduction

The growing dimensionality of recorded data, especially in bioinformatics, demands dimension reduction methods that identify small sets of features leading to a better learning performance. Along with the high dimensionality comes a high variance, which makes it hard to find adequate feature subsets without being kept in local optima. The large number of features challenges the runtime of the selection algorithms. Hence, the main criteria for its quality are that the algorithm is *a*) **multivariate**, takes into account feature correlations, *b*) **stable**, does not vary much for unseen data of the population; *c*) **amending learning**, the learning performance is enhanced; *d*) **fast**: it scales well for very large numbers of features. Ensemble methods decrease variance and, hence, are frequently used in feature selection. However, they usually slow down the procedure. This paper presents a method that speeds up ensembles in a simple and effective way. A careful evaluation on 9 data sets investigates the quality of our new methods.

2 Related Work

Fast univariate **filter** approaches like the t-test [5] or SAM-statistics [14] compute a scoring function on the features, disregarding feature interplay. **Wrapper** approaches [11] better solve this problem, at the cost of much longer runtime. Each feature set evaluation demands a cross-validated training of the used learning algorithm. Some learning algorithms provide the user with an implicit feature ranking which can easily be exploited for feature selection. Such **embedded** approaches are using the weight vector of a linear SVM [15] or the frequency of

feature use of a *Random Forest* (RF) [3]. They are aware of feature interplay and faster than wrappers but biased towards the learning algorithm used.

A group of new algorithms has come up to bridge the gap between fast but univariate filters on the one hand, and slow but multivariate wrappers on the other hand. Their goal is to find a subset of features which is highly predictive with no or a minimum of redundant information. The *correlation based feature selection* (CFS) [8] performs a *sequential forward search* with a correlation measure in the evaluation step. CFS iteratively adds the feature which has the best ratio between predictive relevance of the feature and its correlation with the already selected features. Both, predictiveness and correlation, are measured by the entropy-based *symmetrical uncertainty* where the information gain IG of feature f_i w.r.t. feature f_j is divided by the sum of the entropies of f_i, f_j. Since CFS uses symmetrical uncertainty, it is only suitable for discrete values.

Ding and Peng [4] reinvented CFS with the capability for handling numerical variables calling it *minimum redundancy maximum relevance FS* (MRMR). For numerical features the *F-test* is used. For a continuous feature X and a nominal class variable Y, both from a data set with n examples and C classes, defined as

$$F(X,Y) = \frac{(n-C)\sum_c n_c(\bar{X}_c - \bar{X})}{(C-1)\sum_c (n_c-1)\sigma_c^2} \tag{1}$$

with class-pooled variance σ_c and n_c the number of examples in class c, $c \in \{1,..,C\}$. The redundancy of a numerical feature set is measured by the absolute value of *Pearson's correlation coefficient*

$$R(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)\cdot Var(Y)}} \tag{2}$$

It detects a linear dependency between X and Y. Another possible measure for the dependency between two nominal variables used by MRMR [4] is the *mutual information*

$$MI(X,Y) = \sum_{x,y} P(x,y)\log_2 \frac{P(x,y)}{P(x)P(y)} \tag{3}$$

where x and y are the possible values of X and Y.

From now on, we will use the term *correlation* and the symbol $Cor(\cdot,\cdot)$ as a synonym for either Pearsons's linear correlation eq. (2), F-test eq. (1) or mutual information eq. (3). The correlation measures are averaged over all combinations of the feature set. Instead of the ratio, one can also use the difference between relevance and redundancy [4]. In any case, the measures are based on variance and, hence, are sensitive to outliers and a high variance of the input data, alike. This instability is not suitable, e.g., for biomedical research, where the relevance of features is in the research focus. The main algorithmic issue is the computation of the correlations. In the first step, selecting the most relevant feature takes p calculations of $Cor(f_i, y)$ with $i = 1..p$. The next steps in MRMR/CFS are

to repeatedly add the feature which has the best ratio between relevance and redundancy to the already selected features.

$$F_{j+1} = F_j \cup \{\arg\max_{f \in \mathbb{F} \setminus F_j} \frac{Cor(f,y)}{\frac{1}{j} \sum_{g \in F_j} Cor(f,g)}\} \tag{4}$$

This takes $p - (j-1)$ correlations in each step. For the whole MRMR/CFS process of selecting k from p features

$$p + \sum_{i=1}^{k-1} (p-i) = p \cdot k - \frac{k^2 - k}{2} \tag{5}$$

correlations must be computed. Variants of the method like, e.g., [7] tried to improve the stability of MRMR by introducing a weighting parameter α for the ratio of relevance and redundancy. Tuning this parameter even increases the overall runtime. The same holds for the approach [12], which evaluates together those pairs of features which are higher correlated than some δ, or for *Fast Correlation-based Filter* (FCBF) [17], which discards all features with relevance $< \delta$. Hence, MRMR/CFS is promising but suffering from a lack of stability.

Ensemble methods A high variance negatively effects prediction algorithms (classification & regression) as well as feature selection schemes. Ensemble methods like *Bagging* [2] or *Boosting* [6] reduce variance. Parallel ensembles, e.g. *Bagging*, do so by repeating the algorithm on different subsamples or bootstrapped samples of the input data. This increases the stability of the set of selected features [13, 9] and - in some cases - even reduces the prediction error [16]. Saeys et al. [13] showed that (bagged) ensembles of *symmetrical uncertainty weighting*, *Relief*, SVM-RFE or RF delivered more stable feature selections than the non-ensembled counterparts, but did not increase classification performance. (For RF accuracy even decreased.) The major problem with *Bagging* are the e-times repetition for ensembles of cardinality e. This increases runtime considerably.

3 Speeding up Ensembles

We now have advantages and shortcomings of the methods which we want to enhance, namely MRMR/CFS (being unstable) and ensemble methods (being too slow). The basis for our algorithm is the "split-sum trick" going back to the displacement law of statistics, the latter of which we apply first. It helps to compute the $Cor(x,y)$ in one pass for any two features. Thanks to the displacement law, cf. (6), the measures (1) to (3) can be rewritten as sums of independent terms. We use Pearson's linear correlation as an example, but the other measures can be split analogously: Since the variance of a variable X can be written as

$$Var(X) = E((X - E(X))^2) = E(X^2) - (E(X))^2 \tag{6}$$

the variance- and covariance-terms now only contain sums of computationally independent terms. Pearson's linear correlation can be computed in only one pass over the data for each feature without prior computation of the mean values of each feature. Eq. (1) and (3) can similarly be split into sums of independent terms. This fact is used by our *Inner Ensemble* method and its further enhancement, the *Fast Ensemble*.

Inner Ensembles Correlation measures like Pearson's linear correlation are very sensitive to outliers [10]. This has a negative effect on the stability of the selected feature set. Our first method, the *Inner Ensembles*, increases the robustness of the correlation measure by performing a parallel *ensemble* of e correlations, instead. The correlation is calculated for e parts on subsets of the examples, each part leaving out $\frac{1}{e}$ of the data - similar to e-fold cross-validation. In contrast to cross-validation, the left-out fraction of the data is not used, at all. The average of the e correlations gives us a more robust correlation estimate.

Let F_j denote the set of selected features in step $j \in \{1..k\}$ of MRMR/CFS. The MRMR/CFS algorithm first picks the feature which has the highest correlation with the label and so is most relevant. This takes p calculations of feature-label-correlation $F_1 = \{\arg\max_{f_i} (Cor(f_i, y))\}$ with $i \in [1, p]$. Usually, for ensembles of size e, this would increase runtime of MRMR/CFS by the factor e to $e \cdot p \cdot k - (k^2 - k)/2)$ correlations to compute. Now, we use that the eqs. (1), (2) and (3) can all be split into sums of sums like

$$\sum_{i=1}^{n} h_i = \sum_{i=1}^{m_1} h_i + \sum_{i=m_1+1}^{m_2} h_i + \cdots + \sum_{i=m_{e+1}}^{n} h_i = \sum_{j=1}^{e} \left[\sum_{i=1+(j-1)\frac{n}{e}}^{j\frac{n}{e}} h_i \right] \tag{7}$$

for equally sized parts and arbitrary terms h_i. This allows us to compute these sums for all e intervals $\left[(j-1)\frac{n}{e} + 1, j\frac{n}{e}\right]$ separately. The final correlation of each part $i \in [1, e]$ of the ensemble is then calculated by using all but the ith partial sums. There is virtually no extra runtime apart from e times adding up the partial sums within Algorithm 1. The memory need increases by factor e which is very small, because Pearson's linear correlation only needs five accumulator variables for the sums; F-test and mutual information only need $1 + 3 \cdot |f_1|$ and $3 \cdot |f_1| \cdot |f_2|$ variables, respectively, where $|f_1|$ and $|f_2|$ are the number of distinct values of the features f_1 and f_2. Algorithm 1 exemplarily shows pseudo-code for Pearsons's linear correlation returning the correlation values for all parts in one pass over the data. For *F-Test* (1) and *MI* (3) the procedure is similar.

Fast Ensembles Doing the calculations on diverse subsets of the data and using the split-sum trick in the *Inner Ensembles* way is extremely fast, but it increases the stability of selected feature sets only marginally, when used on each single correlation step, cf. Section 4. Our 2nd method, *Fast Ensemble*, builds an ensemble of the whole selection process, instead of stabilizing each single correlation. We dramatically reduce runtime of the full ensemble by applying the same split-sum trick.

If a correlation between two features is computed for one part of the ensemble, their correlations can be computed with practically no overhead for all other parts of the ensemble (cf. Algorithm 1). As opposed to the *Inner Ensemble*, here, the partial correlation results are not combined, but cached for later use. Just one pass over the examples is needed: For every ith part of the ensemble, the ith partial sums are left out when aggregating the sums to a correlation measure. Hence, for every two features or a feature-class combination, only one pass over the examples is needed, no matter in how many parts of the ensemble they appear. Where a full ensemble of MRMR/CFS needs $e \cdot p$ passes over the examples in order to select a first feature $f_{i,1}$ for e parts of the ensemble, our method does this just once. For every further feature $f_{i,j}$ in all parts of the ensemble, only those feature correlations need a pass over the examples which were not already considered in any other part of the ensemble.

Algorithm 1 Pearson Correlation Ensemble

Input: Numerical variables X, Y, ensemble size e, number of examples n
Init arrays of size e for split-sums: Cor[],e_n[],e_x[],e_xx[],e_y[],e_yy[],e_xy[]
Init overall sums: E_X=0; E_Y=0; E_XY=0; E_XX=0; E_YY=0
j=0; {index for the ensemble parts 1..e}
for $i = 1$ to N **do** {Iterate over all examples, build overall & split-sums}
    e_x[j]+= X[i]; e_y[j]+= Y[i]; e_xy[j]+= X[i]*Y[i]; e_xx[j]+= X[i]^2; e_yy[j]+= Y[i]^2
    E_X+=X[i]; E_Y+=Y[i]; E_XY+=X[i]*Y[i]; E_XX+=X[i]^2; E_YY+=Y[i]^2
    e_n[j]++; j=(j+1) mod e;
end for
for j=1 to e **do** {Iterate over all ensemble parts: calculate correlation}
    denom=sqrt(abs((E_XX-e_xx[j]-$\frac{(E\_X-e\_x[j])^2}{n-e\_n[j]}$) $\cdot$ (E_YY-e_yy[j]-$\frac{(E\_Y-e\_y[j])^2}{n-e\_n[j]}$)))
    Cor[j]= (E_XY-e_xy[j]-$\frac{E\_X \cdot E\_Y}{n-e\_n[j]}$) / denom
end for
return "average" over Cor[]

Algorithm 2 Fast Ensemble of MRMR/CFS

Input: Set of all features $\mathbb{F}$, desired dimension k, size of the ensemble e, label y
for $i = 1$ to e **do** {For all parts of the ensemble}
    $F_i = \{f_{i,1} \in \mathbb{F} \mid max\ Cor(f_{i,1}, y)[i]\}$
    for j=2 to k **do** {Iteratively add best feature}
        $F_i = F_i \cup \{f_{i,j} \in \mathbb{F} \backslash F_i \mid max\ (\mathrm{Cor}(f_{i,j},\mathrm{y})[\mathrm{i}] / \sum_{g \in F_i} Cor(f_{i,j}, g)[i])\}$
    end for
end for
return "average" over all F_i
Cor(a,b) first looks in cache. If measure for (a,b) or (b,a) was not yet cached, chooses appropriate measure, e.g. Alg. 1, and puts the resulting array into cache.

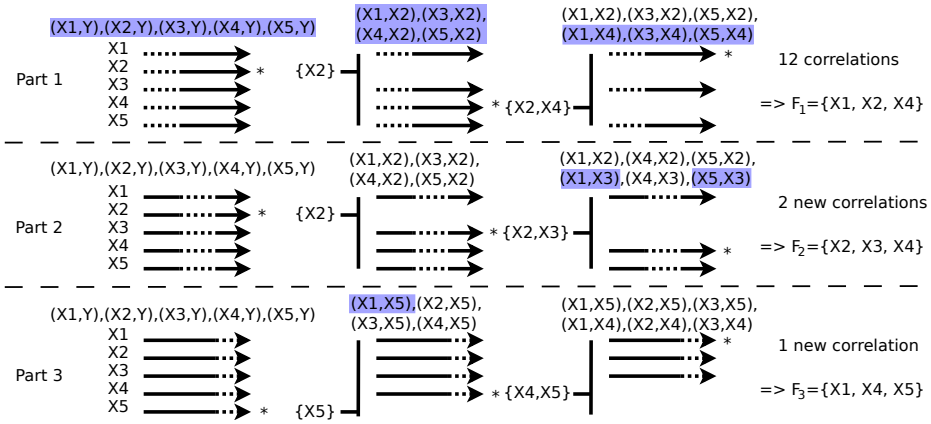


Fig. 1: Fast Ensemble: A simple example with $k = 3$, $n = 5$, $e = 3$. The dashed lines in the arrows represent the subset of the examples which is left out when estimating the correlation. The correlation computations demanding a pass over the data for split-sum calculation are highlighted.

Time-complexity directly depends on the diversity of the ensemble results. If all parts of the ensemble return the same feature set, the needed correlations have all been computed in the first part and the runtime is the same as for a single feature selection. If, in contrast, the resulting feature sets of the feature selections are disjoint, none of the feature pairs has been considered in other parts of the ensemble and thus has not been cached. These extremes are rare.

A simple example of selecting 3 out of 5 features (X_1 to X_5), label Y, illustrates *Fast Ensembles*, cf. Fig. 1. The ensemble consists of three parts. After a first calculation of the relevance of each feature, i.e. the correlation of X_i and Y, the relevance values are cached for each part. To estimate the feature with the best relevance/redundancy ratio the correlations of X_2 and the remaining four features must be computed. X_4 is chosen. In the last step the remaining features must be compared to the newly added X_4. This is repeated for all other parts. Now relevance can be computed from the split sums already computed in the first part. If X_i, X_j have been compared in an earlier part, their split-sums can be reused. The resulting sets are then combined, e.g. via majority vote. Only 15 passes are needed instead of 36 without the split-sum trick.

Our algorithms conduct search in feature subset space like wrappers do. Yet, unlike wrappers, the feature sets are not evaluated in long cross-validation runs but with a fast filter approach. Unlike filters, feature interdependencies are still considered. Hence, *Inner* and *Fast Ensembles* combine the advantages of wrapper and filter approaches. Note, that the inherent parallelism of the algorithms speeds up computation additionally. Every correlation computation between two features is independent of the other features. Thus, not only the parts of the ensemble can be computed in parallel, but also all correlation computations. Only

completed calculations must be reported to some central instance which conducts the search in feature subset space.

4 Evaluation

We evaluated the performance of our algorithm with respect to **stability**, **accuracy** and **runtime** on nine publicly available datasets of a wide range of problem settings and dimensionality (Table 1). For high repeatability plugin, experiments, material and additional plots and figures are available at [1].

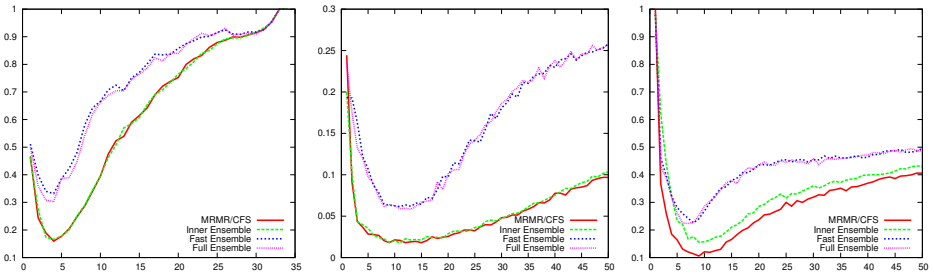
Stability We analyze how the produced feature sets differ under variation of the input data. We compare our two methods *Inner Ensemble* and *Fast Ensemble* to MRMR/CFS and a full ensemble of MRMR/CFS. All ensembles consist of $e = 20$ parts. The stability of the full ensemble is only reported for completeness as it technically does the same as our *Fast Ensemble*.

We use the *Jaccard index* of two feature-sets $J(F_a, F_b) = \frac{|F_a \cap F_b|}{|F_a \cup F_b|}$ to measure the stability of the feature selection. Similar to [13] we draw ten subsets from the example set like ten-fold cross-validation. On each of these subsets a feature selection is computed. The overall stability is further defined as the average of the Jaccard indices for all combinations of those feature selections: $\bar{J} = \frac{2}{l^2+l} \sum_{i=1}^{l} \sum_{j=i+1}^{l} J(F_i, F_j)$, where l is the number of different feature selections in the ensemble. The average over 10 runs are reported. Fig. 2 shows exemplary results for the stability of the four feature selection methods dependent on k, the number of features to select; see [1] for complete results.

The *Fast Ensemble* version clearly outperforms the standard MRMR/CFS, whereas the *Inner Ensemble* shows nearly no visible improvement except for the *Leukemia* dataset in Fig. 3.3. For the *Leukemia* dataset the *mutual information* was used to measure relevance and redundancy. One can see that the inner ensemble only effects nominal datasets. As it increases runtime only in $O(1)$ we

Table 1: Data characteristics and significance of the difference between the stabilities achieved by *Fast Ensemble* and plain MRMR for 10 and 20 features. Values < 0.05 are highly significant.

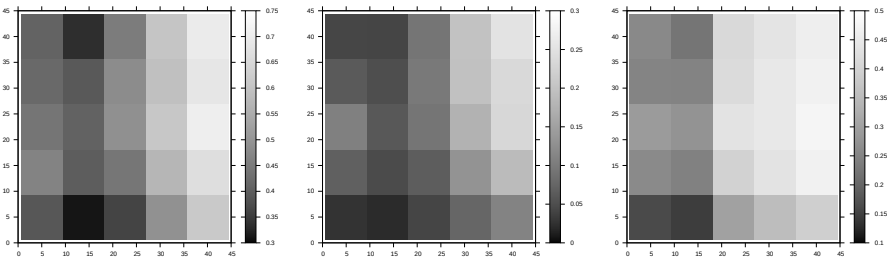
Dataset	p	n	Classes	Data	$k = 10$	$k = 20$
sonar	60	208	2	continuous	0.0085	0.0025
ionosphere	34	351	2	continuous	0.022	0.19951
musk	166	476	2	continuous	$3.1 \cdot 10^{-6}$	$1.4 \cdot 10^{-8}$
lung	325	73	7	nominal	$4.1 \cdot 10^{-7}$	$4.0 \cdot 10^{-14}$
h.w. digits	64	3823	10	continuous	0.082	0.0058
colon	2000	62	2	nominal	$1.4 \cdot 10^{-9}$	$1.1 \cdot 10^{-6}$
lymphoma	4026	96	9	nominal	$1.2 \cdot 10^{-10}$	$4.4 \cdot 10^{-14}$
leukemia	7070	72	2	nominal	$2.6 \cdot 10^{-11}$	$1.9 \cdot 10^{-15}$
nci60	9712	60	9	nominal	$2.4 \cdot 10^{-14}$	0.0



(2.1) Ionosphere (2.2) Lymphoma (2.3) Leukemia

Fig. 2: Stability of the four approaches, MRMR/CFS, *Inner Ensemble*, *Fast Ensemble*, and the full ensemble measured by the average Jaccard index (y-axis), where k, the number of selected features, is the x-axis.

suggest using it for nominal datasets. As is clearly seen, our new *Fast Ensemble* achieves the same performance as a full ensemble of the same size. The benefit is the much smaller number of computations. The visible differences from 2 between methods are significant. We exemplarily report the p-Values for 10 and 20 features in Table 1. The only exception in p-values is selecting 20 features from the *ionosphere* dataset, but this corresponds to the curve in Fig. 2.1. For all datasets stability increases with larger k because the (possible) overlap between subsets selected by different parts of the ensemble increases. The effect of the size e of an ensemble on the selection stability was also investigated, cf. Fig. 3 and [1]. Too small an ensemble does not increase performance and too large an ensemble degrades performance for small n. Stability increases with the number of selected features, but in general a mid-sized ensemble with $e \approx 20$ performs best for all k. We also compared our split-sum based ensembles to classical bagged ensembles of equal size. In 8 of 9 data sets *Bagging* gave less stable results.



(3.1) HW Digits (3.2) Lymphoma (3.3) Leukemia

Fig. 3: Stability of the selected feature sets as a function of e (vertical axis) and k (horizontal axis). Brightness indicates stability - the brighter the better.

Accuracy We analyze if a more stable feature selection benefits classification accuracy on five different learning schemes: *Naïve Bayes*, *5-Nearest-Neighbors*, RF, a linear SVM, and *Logistic Regression* (LR) which all have different strengths and weaknesses. We compare the standard MRMR/CFS approach (Plain) to our *Inner Ensemble* and our *Fast Ensemble* algorithm. Accuracy was averaged over 10 runs of ten-fold cross-validation. SVM and LR were only applied to two-class problems with continuous variables. Pairwise comparisons of the feature selection methods summed up over all experiments gave the following *wins/ties/losses*:

Fast Ensemble vs *MRMR/CFS*: 884/52/634
Inner Ensemble vs *MRMR/CFS*: 785/55/730
Fast Ensemble vs *Inner Ensemble*: 849/50/671

Table 2 shows in more detail the effect of feature selection on classification accuracy for all datasets. Feature selection was only performed on the training set and not on the test set, as this would lead to far too optimistic results. In 7 of the 9 datasets accuracy increased, when a subset of the features was used for prediction instead of all features. More detailed plots of and figures for all datasets can be found in [1]. In general, results show that *Fast Ensembles* deliver better results earlier (with less features) than MRMR/CFS, Except for the *Musk* dataset, where all feature selections degraded performance.

Runtime Let us now compare the runtime of our new *Fast Ensemble* with a standard ensemble of the same size. The overlap (number of features which two sets have in common) can be calculated based on their size k and their Jaccard index as $ol(\bar{J}, k) = 2 \cdot \bar{J} \cdot k / (\bar{J}+1)$. The average Jaccard index of the sets produced by the parts of the ensemble is similar to the average Jaccard index measured for Plain MRMR/CFS for inspecting stability. Considering a *Fast Ensemble* of e parts we now sum up the average amount of correlations in the parts of the ensemble. In the first part of the ensemble, all $p \cdot k - \frac{k^2-k}{2}$ correlations must be computed. In the second part, there are no correlations needed for relevance estimation, and for the redundancy check, only those features, which have not

Table 2: Accuracy by the number of times a method was superior to the others in the order Fast/Inner/Plain. k varied between 1 and 50. Winner is set bold.

Dataset	5NN	NB	RF	SVM	LR
sonar	18/13/**19**	14/**22**/13	**23**/15/11	18/**19**/12	15/**22**/12
ionosphere	**18**/4/11	**17**/7/10	**19**/4/10	**20**/11/2	**18**/9/6
musk	4/17/**29**	2/**27**/21	**14**/6/7	**21**/13/16	18/9/**20**
lung	**23**/11/15	**21**/16/12	16/15/**19**		
H.W. Digits	**37**/7/6	**27**/8/14	17/**19**/14		
colon	**23**/14/13	16/**20**/10	8/**22**/18		
leukemia	17/**21**/11	**20**/11/19	**21**/19/7		
lymphoma	**32**/6/11	**43**/5/2	**44**/2/2		
nci60	**20**/17/10	**31**/8/9	17/**22**/9		

been added in the first part, must be correlated with the remaining $k - ol(\bar{J}, k)$ features. At this point, it is unclear whether these features are added at the end or at the beginning of the selection process, i.e., whether the parts of the ensemble differ at the end or the beginning of the selection process. This determines how many features it is compared to and explains the imprecision of the estimate.

For a rough estimate assume the average probability that a feature f^i in the ith part has already been correlated to all other features in the part before is

$$P_{k,\bar{J}}(f^i) = \sum_{m=1}^{i-1} \frac{ol(\bar{J}, k)}{k}(1 - P_{k,\bar{J}}(f^m)) = \frac{2\bar{J}}{\bar{J}+1} \sum_{m=1}^{i-1} (1 - P_{\bar{J}}(f^m)) =: P_{\bar{J}}(f^i)$$

with $P_{\bar{J}}(f^1) = 0$. As seen from eq. (5), in one part, there are on average $p - (k-1)/2$ correlations to compute. When multiplied with the probability of **not** needing to compute the correlation and adding the initial relevance computation this gives a total average runtime of

$$T(p, k, e, \bar{J}) = p + (k-1)\left(p - \frac{k-1}{2}\right)\left(e - \sum_{i=1}^{e} P_{\bar{J}}(f^i)\right) \tag{8}$$

under the assumption $f^i_j = f^i_l$, $\forall j, l \in [1, k]$.

To give an empirical validation of this average case runtime estimation, Tables 3 and 4 show the number of correlations that must be computed depending on k and e. We compare our approach with a standard ensemble of MRMR/CFS of the same size. High variance and large p can decrease the overlap between the ensemble parts, such increasing runtime as it is this overlap which speeds up runtime of our approach. It is not possible to predict in which order features are selected in different parts of the ensemble. This puts more variance to the number of needed correlations and makes it harder to predict those numbers. Nonetheless, eq. (8) seems to give a good estimate on the average case of correlation computations.

5 Conclusion

We presented two new algorithms towards selecting a maximum relevant and minimum redundant feature set. Our algorithms are more stable than the existing MRMR/CFS approach and much faster than a standard ensemble of MRMR/CFS. The speed-up is due to a faster computation of $Cor(f, f')$ based on the displacement rule and due to caching redundancy calculations from partitions. We showed that our method is well suited for feature selection on high-dimensional data as it is more robust against high variance and outliers than the single version. For the choice of $e = 20$ our algorithm is 1.4 to 19.7 times faster than a usual ensemble of MRMR/CFS.

Our methods do not rely on Mutual Information, Pearson's correlation, or the F-Test, alone. They can make use of any measure of similarity which can be split into sums. The split-sum-trick could also speed-up, e.g., Saeys' *bagged* SU [13], which builds upon MI, when replacing *Bagging* by our subset splits.

Table 3: Runtime measured by the number of passes over the data for a standard ensemble of MRMR/CFS, our *Fast Ensemble* method, the speed gain and the estimated runtime. Results for the *Ionosphere* and *Lung* datasets. *Ionosphere* only contains 34 features. See [1] and Table 4 for more results.

		Ionosphere				Lung			
k	e	Fast	Estimated	Standard	Gain	Fast	Estimated	Standard	Gain
10	5	405	504	1,475	3.64	12,505	11,578	16,025	1.28
20	5	567	587	2,450	4.32	20,844	17,548	31,550	1.51
30	5	594	641	2,925	4.92	26,410	19,937	46,575	1.76
40	5					29,322	22,228	61,100	2.08
50	5					32,472	23,920	75,125	2.31
10	10	424	511	2,950	6.96	16,929	17,342	32,050	1.89
20	10	550	587	4,900	8.91	24,055	20,831	63,100	2.62
30	10	595	641	5,850	9.83	29,539	21,087	93,150	3.15
40	10					31,239	22,716	122,200	3.91
50	10					33,274	24,107	150,250	4.52
10	20	424	511	5,900	13.92	22,099	21,807	64,100	2.90
20	20	540	587	9,800	18.15	27,550	21,576	126,200	4.58
30	20	595	641	11,700	19.66	30,820	21,158	186,300	6.04
40	20					32,269	22,727	244,400	7.57
50	20					33,669	24,109	300,500	8.93

Table 4: Runtime measured by the number of passes over the data for a standard ensemble of MRMR/CFS, our *Fast Ensemble* method, the speed gain and the estimated runtime. Results for the *Colon* and *Lymphoma* datasets. See [1] and Table 3 for more results.

		Colon				Lymphoma			
k	e	Fast	Estimated	Standard	Gain	Fast	Estimated	Standard	Gain
10	5	61,535	47,697	99,775	1.62	176,198	170,514	201,075	1.14
20	5	104,622	64,376	199,050	1.90	354,398	349,559	401,650	1.13
30	5	145,299	94,289	297,825	2.05	526,680	488,337	601,725	1.14
40	5	183,629	119,446	396,100	2.16	677,766	590,451	801,300	1.18
50	5	221,559	142,219	493,875	2.23	800,598	694,083	1,000,375	1.25
10	10	98,775	53,160	199,550	2.02	314,973	305,016	402,150	1.28
20	10	129,855	65,012	398,100	3.07	623,705	617,372	803,300	1.29
30	10	156,840	95,029	595,650	3.80	865,436	787,610	1,203,450	1.39
40	10	193,149	120,023	792,200	4.10	1,065,723	859,902	1,602,600	1.50
50	10	232,860	142,630	987,750	4.24	1,218,735	955,178	2,000,750	1.64
10	20	114,347	53,891	399,100	3.49	565,565	501,460	804,300	1.42
20	20	137,585	65,019	796,200	5.79	1,024,385	985,833	1,606,600	1.57
30	20	166,430	95,035	1,191,300	7.16	1,296,456	1,086,820	2,406,900	1.86
40	20	198,849	120,026	1,584,400	7.97	1,487,010	1,040,597	3,205,200	2.16
50	20	234,740	142,631	1,975,500	8.42	1,631,750	1,091,347	4,001,500	2.45

References

1. http://www-ai.cs.uni-dortmund.de/PUBDOWNLOAD/fastensembles.tar.gz
2. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
3. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
4. Ding, C.H.Q., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. JBCB 3(2), 185–206 (2005)
5. Fox, R.J., Dimmic, M.W.: A two-sample bayesian t-test for microarray data. BMC Bioinformatics 7(126) (2006)
6. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. JCSS 55(1), 119–139 (1997)
7. Gulgezen, G., Cataltepe, Z., Yu, L.: Stable and accurate feature selection. In: ECML/PKDD (2009)
8. Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: ICML (2000)
9. Jurman, G., Merler, S., Barla, A., Paoli, S., Galea, A., Furlanello, C.: Algebraic stability indicators for ranked lists in molec. profiling. Bioinf. 24(2), 258–264 (2008)
10. Koh, J.L.Y., Lee, M.L., Hsu, W., Lam, K.T.: Correlation-based detection of attribute outliers. In: DASFAA (2007)
11. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
12. Michalak, K., Kwaśnicka, H.: Correlation-based feature selection strategy in classification problems. Int. J. Appl. Math. and Comp. Sc. 16(4), 503–511 (2006)
13. Saeys, Y., Abeel, T., de Peer, Y.V.: Robust feature selection using ensemble feature selection techniques. In: ECML/PKDD (2008)
14. Tusher, V.G., Tibshirani, R., Chu, G.: Significance analysis of microarrays applied to the ionizing radiation response. PNAS 98(9), 5116–5121 (April 2001)
15. Vapnik, V.: Statistical Learning Theory. Wiley, Chichester, GB (1998)
16. Xu, X., Zhang, A.: Boost feature subset selection: A new gene selection algorithm for microarray dataset. In: ICCS (2006)
17. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. JMLR 5, 1205–1224 (2004)

Hybrid Correlation and Causal Feature Selection for Ensemble Classifiers

Rakkrit Duangsoithong and Terry Windeatt

Centre for Vision, Speech and Signal Processing
University of Surrey
Guildford, United Kingdom GU2 7XH
{r.duangsoithong,t.windeatt}@surrey.ac.uk

Abstract. PC and TPDA algorithms are robust and well known prototype algorithms, incorporating constraint-based approaches for causal discovery. However, both algorithms cannot scale up to deal with high dimensional data, that is more than few hundred features. This paper presents hybrid correlation and causal feature selection for ensemble classifiers to deal with this problem. The number of eliminated features, accuracy, the area under the receiver operating characteristic curve (AUC) and false negative rate (FNR) of proposed algorithms are compared with correlation-based feature selection (FCBF and CFS) and causal based feature selection algorithms (PC, TPDA, GS, IAMB).

Key words: Feature selection, causal discovery, ensemble classification.

1 Introduction

Feature selection is an important pre-processing step to reduce feature dimensions for classification and generally, can be divided into four categories [1],[2],[3]. Filter method is independent from learning method and uses measurement techniques such as correlation and distance measurement to find a good subset from entire set of features. Wrapper method uses pre-determined learning algorithm to evaluate selected feature subsets that are optimum for the learning process. Hybrid method combines advantage of both Filter and Wrapper method together. It evaluates features by using an independent measure to find the best subset and then uses a learning algorithm to find the final best subset. Finally, Embedded method interacts with learning algorithm but it is more efficient than Wrapper method because the filter algorithm has been built with the classifier.

Feature selection does not usually take causal discovery into account. However, in some cases such as when training and testing dataset do not conform to i.i.d. assumption, testing distribution is shifted from manipulation by external agent, causal discovery can provide some benefits for feature selection under these uncertainty conditions. Causality also can learn underlying data structure, provide better understanding of the data generation process and better accuracy and robustness under uncertainty [4].

Causal relationships are usually uncovered by Bayesian Networks (BNs) which consist of a direct acyclic graph (DAG) that represents dependencies and independencies between variable and joint probability distribution among a set of variables [5].

Generally, the category of BNs can be divided into: Search-and-Score and Constraint-Based approaches. In Search-and-Score approach, BNs search all possible structures to find the one that provides the maximum score. The second approach, Constraint-Based, uses test of conditional dependencies and independencies (CI) from the data by estimation using G^2 statistic test or mutual information, etc. Constraint-Based algorithms are computationally effective and suitable for high dimensional feature spaces. PC algorithm [6], is a pioneer, prototype and well-known global algorithm of Constraint-Based approach for causal discovery. Three Phase Dependency Analysis (TPDA or PowerConstructor) [7] is another global Constraint-Based algorithm that uses mutual information to search and test for CI test instead of using G^2 Statistics test as in PC algorithm. However, both PC and TPDA algorithm use global search to learn from the complete network and can not scale up to more than few hundred features (they can deal with 100 and 255 features for PC and TPDA, respectively) [8]. Recently, many Markov Blanket-based algorithms for causal discovery have been studied extensively and they have ability to deal with high dimensional feature spaces such as GS [9], MMMB, IAMB [8] and HITON [5] algorithms.

An ensemble classifier or multiple classifier system (MCS) is another well-known technique to improve system accuracy [10]. Ensemble combines multiple base classifiers to learn a target function and gathers their prediction together. It has ability to increase accuracy by combining output of multiple experts to reduce bias and variance, improve efficiency by decomposing complex problem into multiple sub problems and improve reliability by reducing uncertainty. To increase accuracy, each classifier in the ensemble should be diverse or unique such as starting with different input, initial weight, random features or random classes [11].

The main objective of this paper is to find algorithm that can scale up PC and TPDA algorithms to deal with high dimensional data. We propose analysis of hybrid correlation and causal feature selection for ensemble classifiers in terms of number of eliminated features, average percent accuracy, the area under the receiver operating characteristic curve (AUC) and false negative rate (FNR).

2 Theoretical Approach

In our research, hybrid algorithm of correlation and causal feature selection is compared with Fast Correlation-Based Filter (FCBF), Correlation-based Feature Selection with Sequential Forward Floating Search direction (CFS+SFFS), and with causal feature selection algorithms (PC, TPDA, GS and IAMB) using Bagging (described in Section 2.2).

2.1 Feature Selection

2.1.1 Correlation-based Redundancy and Relevance Analysis The concept of selecting optimal subset from whole features is presented in Figure 1 [12]. where I is irrelevant feature, II is weakly relevant and redundant feature, III is weakly relevant but non redundant feature. IV is strongly relevant feature and III+IV are optimal subset.

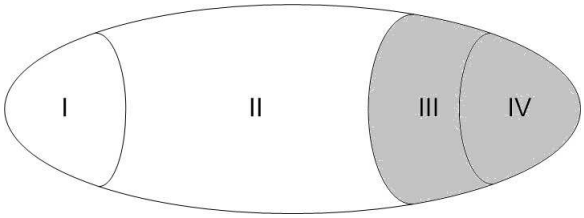


Fig. 1. Optimal Subset

Optimal subset should include all strongly relevant features, subset of weakly relevant features that have no redundancy and none of the irrelevant features.

Table 1 shows the summary analysis of redundancy and relevancy analysis for correlation-based [12], causal-based [4] and proposed hybrid correlation and causal feature selection. Markov Blanket (MB(T)) of target or class (T) is the minimal set of conditional features that all other features are probabilistically independent of T. It consists of the set of parents, children and spouses of T. Approximate Markov Blanket is explained section 2.1.1 a.

Table 1. Summary analysis of correlation, causal and proposed hybrid correlation and causal feature selection for redundancy and relevance analysis.

Relation	Correlation-Based	Causal-Based	Hybrid algorithm
Strongly relevant	$SU_{i,c} = 1$	Features in Markov Blanket	Features in Markov Blanket
Weakly relevant without redundant features	does not has approximate Markov Blanket	connected to classes	connected to classes
Weakly relevant with redundant features	has approximate Markov Blanket	connected to classes	has approximate Markov Blanket
Irrelevant	$SU_{i,c} = 0$	disconnected to classes	disconnected to classes

a) Fast Correlation-Based Filter (FCBF). FCBF [12] algorithm has two stages: relevance analysis and redundancy analysis.

Relevance Analysis. Correlation can be measured by using Symmetrical Uncertainty (SU).

$$SU(X,Y) = 2\Big[\frac{IG(X|Y)}{H(X)+H(Y)}\Big] \tag{1}$$

$$IG(X|Y) = H(X) - H(X|Y) \tag{2}$$

$$H(X) = -\sum_{i} P(x_i) log_2 P(x_i) \tag{3}$$

where $IG(X|Y)$ is the Information Gain of X after observing variable Y. $H(X)$ and $H(Y)$ are the entropy of variable X and Y, respectively. $P(x_i)$ is the probability of variable x.

SU is the modified version of Information Gain that has range between 0 and 1. FCBF removes irrelevant features by ranking correlation (SU) between feature and class. If SU between feature and class equal to 1, it means that this feature is completely related to that class. On the other hand, if SU is equal to 0, the features are irrelevant to this class.

Redundancy analysis. Redundant features can be defined from meaning of predominant feature and approximate Markov Blanket. In Yu and Liu (2004) [12], a feature is predominant (both relevant and non redundant feature) if it does not have any approximate Markov Blanket in the current set.

Approximate Markov Blanket: For two relevant features F_i and F_j $(i \neq j)$, F_j forms an approximate Markov Blanket for F_i if

$$SU_{j,c} \geq SU_{i,c} \; and \; SU_{i,j} \geq SU_{i,c} \tag{4}$$

where $SU_{i,c}$ is a correlation between any feature and the class. $SU_{i,j}$ is a correlation between any pair of feature F_i and F_j $(i \neq j)$.

b) Correlation-based Feature Selection (CFS). CFS [13] is one of well-known techniques to rank the relevance of features by measuring correlation between features and classes and between features and other features.

Given number of features k and classes c, CFS defined relevance of features subset by using Pearson's correlation equation

$$Merit_s = \frac{kr_{kc}}{\sqrt{k+(k-1)r_{kk}}} \tag{5}$$

where $Merit_s$ is relevance of feature subset, r_{kc} is the average linear correlation coefficient between these features and classes and r_{kk} is the average linear correlation coefficient between different features.

Normally, CFS adds (forward selection) or deletes (backward selection) one feature at a time, however, in this research, we used Sequential Forward Floating Search (SFFS) [14] as the search direction.

2.1.2 Causal Discovery Algorithm. In this paper, two standard constraint-based approaches (PC and TPDA) and two Markov Blanket based algorithms (GS, IAMB) are used as causal feature selection methods. In the final output of the causal graph from each algorithm, the unconnected features to classes will be considered as eliminated features.

a) PC Algorithm PC algorithm [6],[4] is the prototype of constraint-based algorithm. It consists of two phases: Edge Detection and Edge Orientation.

Edge Detection: the algorithm determines directed edge by using conditionally independent condition. The algorithm starts with:

i) Undirected edge with fully connected graph.

ii) Remove a share direct edge between A and B ($A - B$) iff there is a subset F of features that can present conditional independence $(A, B|F)$.

Edge Orientation: The algorithm discovers V-Structure $A - B - C$ in which $A - C$ is missing.

i) If there are direct edges between $A - B$ and $B - C$ but not $A - C$, then orient edge $A \rightarrow B \leftarrow C$ until no more possible orientation.

ii) If there is a path $A \rightarrow B - C$ and $A - C$ is missing, then $A \rightarrow B \rightarrow C$.

iii) If there is orientation $A \rightarrow B \rightarrow ... \rightarrow C$ and $A - C$ then orient $A \rightarrow C$.

b) Three Phase Dependency Analysis Algorithm (TPDA) TPDA or PowerConstructor algorithm [7] has three phases: *drafting, thickening and thinning*. In *drafting phase*, mutual information of each pair of nodes is calculated and used to create a graph without loop. After that, in *thickening phase*, edge will be added when that pair of nodes can not be *d-separated*. (node A and B are *d-separated* by node C iff node C blocks every path from node A to node B [15].) The output of this phase is called an independence map (*I-map*). The edge of *I-map* will be removed in *thinning phase* if two nodes of the edge can be *d-separated* and the final output is defined as a *perfect map* [7].

c) Grow-Shrink algorithm (GS) GS [9] algorithm consists of two phases, forward and backward.

Forward phase: GS statistically ranks features by using the strength of association with target or class (T) given empty set. After that the next ordering feature which is not conditionally independent from class T given current Markov Blanket (CMB) will added into CMB.

Backward phase: Identify false positive nodes and remove them from CMB. At this stage, $CMB = MB(T)$. Finally, a feature X will be removed from CMB one-by-one if that feature X is independent of class T given the remaining CMB.

d) Incremental Association Markov Blanket Algorithm. (IAMB) IAMB [8] is one of Markov Blanket detection algorithms using forward selection followed by removing false positive node. IAMB has two phases, forward and backward.

Forward phase: In forward selection phase, the algorithm starts with empty set in CMB, then adding features which maximizes a heuristic function $f(X; T|CMB)$. A feature member in MB(T) will not return zero value of this function.

Backward phase: False positive nodes will be removed from CMB by using condition independent testing of class T given the rest CMB.

2.2 Ensemble Classifier

Bagging [16] or **B**ootstrap **agg**regat**ing** is one of the earliest, simplest and most popular methods for ensemble based classifiers. Bagging uses Bootstrap that randomly samples with replacement and combines with majority vote. The selected data is divided to m bootstrap replicates and randomly sampled with replacement. Each bootstrap replicate contains, on average, 63.2 % of the original dataset. Final output will be selected from majority vote from all classifiers of each bootstrap replicate. Bootstrap is the most well-known strategy for injecting randomness to improve generalization performance in multiple classifier systems and provides out-of-bootstrap estimate for selecting classifier parameters [10]. Randomness is desirable since it increases diversity among the base classifiers, which is known to be a necessary condition for improved performance. However, there is an inevitable trade-off between accuracy and diversity known as the accuracy/diversity dilemma [10].

3 Experimental Setup

3.1 Dataset

The datasets used in this experiment were taken from Causality Challenge [17] and details of each dataset are shown in Table 2.

Table 2. Datasets.

Dataset	Sample	Features	Classes	Missing Values	Data type
LUCAS	2000	11	2	No	Numeric (binary)
LUCAP	2000	143	2	No	Numeric (binary)
REGED	500	999	2	No	Numeric (discrete)
CINA	16033	132	2	No	Numeric (discrete)
SIDO	12678	4932	2	No	Numeric (binary)

3.2 Evaluation

To evaluate feature selection process we use four widely used classifiers: Naive-Bayes(NB), Multilayer Perceptron (MLP), Support Vector Machines (SVM) and Decision Trees (DT). The parameters of each classifier were chosen as follows. MLP has one hidden layer with 16 hidden nodes, learning rate 0.2, momentum 0.3, 500 iterations and uses backpropagation algorithm with sigmoid transfer function. SVM uses polynomial kernel with exponent 2 and the regularization value set to 0.7. DT uses pruned C4.5 algorithm. The number of classifiers in Bagging is varied from 1, 5, 10, 25 to 50 classifiers. The threshold value of FCBF

algorithm in our research is set at zero for LUCAS, REGED, CINA, SIDO and 0.14 for LUCAP dataset, respectively.

The classifier results were validated by 10 fold cross validation with 10 repetitions for each experiment and evaluated by average percent of test set accuracy, FNR and AUC.

Due to large number of samples and limitation of computer memory during validation in CINA and SIDO datasets, the number of samples of both dataset are reduced to 10 percent (1603 and 1264 samples, respectively) from the original dataset.

For causal feature selection, PC algorithm uses mutual information (MI) as statistic test with threshold 0.01 and maximum cardinality equal to 2. In TPDA algorithm, mutual information was used as statistic test with threshold 0.01 and data assumed to be monotone faithful. GS and IAMB algorithm use MI statistic test with significant 0.01 and provides output as Markov Blanket of the classes.

4 Experimental Result

Table 3 presents the number of selected features for correlation-based, causal based feature selection and proposed hybrid algorithm. It can be seen that PC and TPDA algorithms are impractical for high dimensional features due to their complexity. However, if redundant features are removed, the number of selected features will enable both algorithms to be practical as shown in proposed hybrid algorithm. Nevertheless, for some datasets such as REGED, TPDA algorithm might not be feasible because of many complex connections between nodes (features).

Table 3. Number of selected features from each algorithm.

Dataset	Original	Correlation-Based		Causal-Based				Hybrid algorithm			
	Feature	FCBF	CFS	PC	TPDA	GS	IAMB	H-PC	H-TPDA	H-GS	H-IAMB
LUCAS	11	3	3	9	10	9	11	2	3	2	2
LUCAP	143	7	36	121	121	16	14	21	22	17	13
REGED	999	18	18	N/A	N/A	2	2	18	N/A	2	2
CINA	132	10	15	132	N/A	4	4	5	7	10	9
SIDO	4932	23	25	N/A	N/A	17	17	2	3	1	2

Figure 2 to 4 show the average percent accuracy, AUC and FNR of five datasets for all classifiers. From average accuracy in figure 2, correlation-based feature selection (FCBF, CFS) provides the best average accuracy. Hybrid correlation and causal feature selection has better accuracy than original causal feature selection. Hybrid method using PC algorithm (H-PC) has slightly lower average accuracy than correlation-based feature selection but has the ability to

deal with high dimensional features. From figure 3, PC, CFS, TPDA and FCBF algorithm provide the best and comparable AUC. Proposed hybrid algorithm has lower AUC than both correlation and original causal-based algorithms. In figure 4, H-PC has the lowest FNR. In all experiments, hybrid algorithm provides lower FNR than original causal algorithm but still higher than correlation-based algorithm.

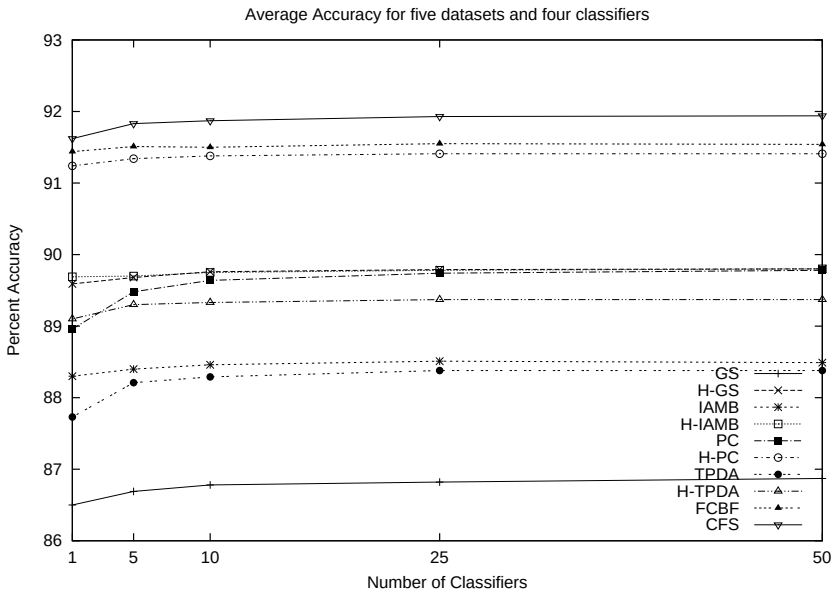


Fig. 2. Average Percent Accuracy of five datasets and four classifiers

Ensemble classifiers using Bagging slightly improves accuracy and AUC for most algorithms. Bagging also reduces FNR for CFS, PC and TPDA algorithm but provides stable FNR for the rest. After increasing number of classifiers to 5-10, the graphs of average accuracy, AUC and FNR all reach saturation point.

5 Conclusion

In this paper, hybrid correlation and causal feature selection for ensemble classifiers is presented to deal with high dimensional features. According to the results, the proposed hybrid algorithm provides slightly lower accuracy, AUC and higher FNR than correlation-based. However, compared to causal-based feature selection, the proposed hybrid algorithm has lower FNR, higher average accuracy and AUC than original causal-based feature selection. Moreover, the proposed hybrid algorithm can enable PC and TPDA algorithms to deal with high dimensional

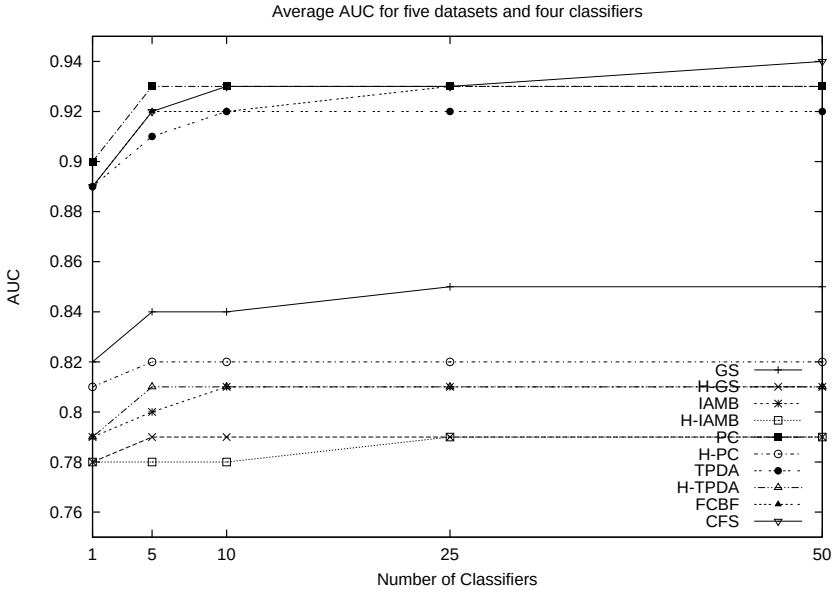


Fig. 3. Average AUC of five datasets and four classifiers

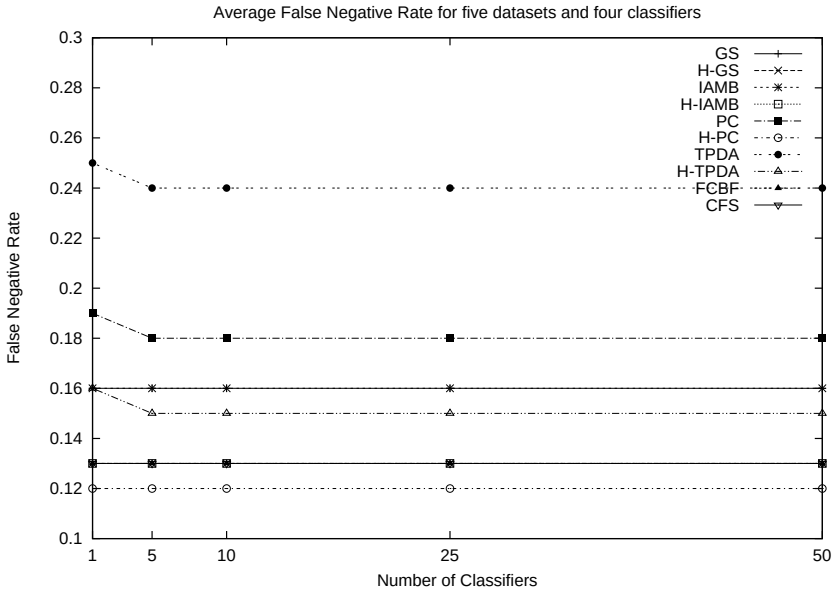


Fig. 4. Average FNR of five datasets and four classifiers

features while maintaining high accuracy, AUC and low FNR. Also the underlying causal structure is more understandable and has less complexity. Ensemble classifiers using Bagging provide slightly better results than single classifier for most algorithms. The future work will improve accuracy of search direction in structure learning for causal feature selection algorithm.

References

1. Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on Knowledge and Data Engineering 17(4), 491–502 (2005)
2. Saeys,Y., Inza,I., Larranaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23(19), 2507–2517 (2007)
3. Duangsoithong, R., Windeatt, T.: Relevance and Redundancy Analysis for Ensemble Classifiers. In: Perner,P. (ed.) Machine Learning and Data Mining in Pattern Recognition, vol. 5362, pp. 206–220, Springer, Heidelberg (2009).
4. Guyon, I., Aliferis, C., Elisseeff, A.: Causal Feature Selection. In Computational Methods of Feature Selection, Liu, H. and Motoda, H. editors. Chapman and Hall (2007)
5. Aliferis, C.F., Tsamardinos, I., Statnikov, A.: HITON, A Novel Markov Blanket Algorithm for Optimal Variable Selection. AMIA 2003 Annual Symposium Proceedings, pp 21–25.(2003)
6. Spirtes, P. Glymour, C., Schinese, R.: Causation, Prediction, and search. Springer, New York (1993)
7. Cheng, J., Bell, D.A., Liu W.: Learning Belief Networks from Data : An Information theory Based Approach. In Proceedings of the Sixth ACM International Conference on Information and Knowledge Management. pp 325–331 (1997)
8. Tsamardinos, I., Aliferis, C.F., Statnikov,A.: Time and Sample Efficient Discovery of Markov Blankets and Direct Causal Relations. KDD2003 Washington DC, USA. (2004)
9. Margaritis, D., Thrun, S.: Bayesian network induction via local neighborhoods. In: Solla, S.A., Leen, T.K., Mller, K.-R. (eds.), Proceedings of the 1999 Conference 2000, vol. 12. MIT Press. pp 505-511 (2000)
10. Windeatt, T.: Ensemble MLP Classifier Design, vol. 137, pp.133–147. Springer, Heidelberg (2008)
11. Windeatt, T.: Accuracy/diversity and ensemble MLP classifier design. IEEE Transactions on Neural Networks 17(5), 1194–1211 (2006)
12. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. J. Mach. Learn. Res. 5, 1205–1224 (2004)
13. Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: Proceeding of the 17th International Conference on Machine Learning, pp. 359–366. Morgan Kaufmann, San Francisco (2000)
14. Pudil, P., Novovicova, J., Kittler, J.: Floating Search Methods in Feature Selection. Pattern Recognition Letters, 15, 1119–1125 (1994)
15. Tsamardinos, I., Brown, L.E, Aliferis, C.F.: The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, vol.65, pp 31–78 (2006)
16. Breiman, L.: Bagging predictors. Machine Learning, 24(2), 123–140 (1996)
17. Guyon, I.: Causality Workbench (2008)
    `http://www.causality.inf.ethz.ch/home.php`

Feature Selection for Unsupervised Learning using Random Cluster Ensembles

Haytham Elghazel and Alex Aussem

Université de Lyon, F-69000, Lyon ; Université de Lyon 1 ;
Laboratoire LIESP, 69622 Villeurbanne, France
haytham.elghazel@univ-lyon1.fr

Abstract. In this paper, we propose another extension of the Random Forests paradigm to unlabeled data, leading to localized unsupervised feature selection (FS). We show that the way internal estimates are used to measure variable importance in Random Forests are also applicable to FS in unsupervised learning. We first illustrate the clustering performance of the proposed method on various data sets based on widely used external criteria of clustering quality. We then assess the accuracy and the scalability of the FS procedure on UCI and real labeled data sets and compare its effectiveness against other FS methods.

Keywords: Unsupervised learning, feature selection, Random Forest.

1 Introduction

The identification of relevant subsets of random variables among thousands of potentially irrelevant and redundant variables is a challenging topic of pattern recognition research that has attracted much attention over the last few years [13, 16, 24, 22]. In supervised learning, feature selection (FS) algorithms maximize some function of predictive accuracy. But in unsupervised learning, we are not given class labels. It becomes unclear which features we should keep as there are no obvious criteria to guide the search. Intuitively, all features are not equally important. Some of the features may be redundant, some may be irrelevant, and some can even misguide clustering results. Broadly speaking, the FS in unsupervised learning aims at finding relevant subsets of variables that produce "natural" groupings by grouping "similar" objects together based on some similarity measure. Reducing the number of features increases comprehensibility and ameliorates the problem that some unsupervised learning algorithms break down with high dimensional data.

Databases have increased many fold in recent years. Important recent problems (i.e., DNA data in biology) often have the property that there are hundreds or thousands of variables, with each one containing only a small amount of information. A single clustering model is known to produce very bad groupings as the learning algorithms break down with high dimensional data. Clustering ensembles is an effective solution to overcome the dimensionality problem and to improve the robustness of the clustering [9, 27, 28, 8, 11]. The idea is to combine

the results of miltiple clusterings into a single data partition without accessing to the original features. The strategy follows a split-and -merge approach: 1) construct a diverse and accurate ensemble committee of clusterings, and 2) combine the clustering results of the committee using a consensus function. Although considerable attention has been given on the problem of constructing accurate and diverse ensemble committee of clusterings, little attention has been given to exploiting the multiple clusterings of the ensemble with a view to identify and remove the irrelevant features.

The framework pursued in this article attempts to bridge the gap between supervised and unsupervised FS approaches in ensemble learning. The way internal estimates are used to measure variable importance in the Random Forests (RF) paradigm [2] have been influential in our thinking. In this study, we show that these ideas are also applicable to unsupervised FS. We emphasize that RF was already extended for unsupervized clustering by Breiman however we think his approach suffers from many problems as explained in the sequel. In our proposal, we extend the RF paradigm to unlabeled data by introducing a clustering ensemble termed as RCE (for Random Cluster Ensembles). RCE combines both data resampling (*bagging*) and random selection of features (*random subspaces*) strategies for generating an ensemble of component clusterings. A combination of these two main strategies for producing clustering ensembles leads to exploration of distinct views of inter-pattern relationships. Many approaches can be used to combine the multiple obtained partitions [11]. For sake of simplicity, we will use the *evidence accumulation technique* proposed in [9] in our experiments. The method consists of taking the co-occurences of pairs of patterns in the same cluster as votes for their association. The co-association matrix of patterns represents a new similarity measure between patterns. The final (consensus) clustering is obtained by running a traditional average-link hierarchical agglomerative algorithm on this matrix. Once the consensus clustering is obtained, we select the relevant features locally in each final cluster based on the out-of-bag importance measure discussed by Breiman [2]. Strictly speaking, our method computes the feature relevance; the FS is performed afterwards by a statistical test called the Scree Test [4]. Empirical results on UCI and real labeled data sets will be presented to answer the following questions: (1) Is our FS for unsupervised learning algorithm better than clustering on all features? (2) Is it competitive with other unsupervised FS methods? (3) How does the performance degrade as more irrelevant variables are included?

2 Unsupervised FS background

The problem of unsupervised FS has attracted a great deal of interest recently. Like in supervised FS, the methods can be divided into three categories, depending on how they interact with the clustering algorithm: *wrapper*, *embedded* and *filter* approaches. *Wrapper methods* perform a search in the space of feature subsets, guided by the outcome of the clustering model. For that, they wrap unsupervised FS process around a clustering algorithm. Typically, a criterion

is firstly defined for evaluating the quality of a candidate feature subset and wrapper approaches aim to identify a feature subset such that the clustering algorithm trained on this feature subset can achieve the optimal value of the predefined criterion, such as normalized scatter separability (for k-means) [7] or normalized likelihood (for EM clustering) [7] or DB-index [20]. Another example is given by the algorithm described in [14]. It tries to search for a subset of all features such that the clustering algorithm trained on this feature subset can achieve the most similar clustering solution to the one obtained by an ensemble learning algorithm. Furthermore, RF has been also extended to unlabeled data leading to unsupervised learning. The idea is to construct an RF predictor that distinguishes the observed data from suitably generated synthetic data [3, 26]. This method will be included in our experiments. In the aforementioned algorithms, the candidate feature subsets are evaluated globally. Regardless of the evaluation criteria, global FS approaches compute them over the entire dataset. Thus, they can only find one relevant feature subset for all clusters. However, it is the local intrinsic properties of data that counts during clustering [29]. As a solution, authors in [29] proposed a localized FS algorithm for clustering. The proposed algorithm computes adjusted and normalized scatter separability for individual clusters and a sequential backward search is then applied to search for the optimal feature subsets for each cluster. However this approach doest not scale to high-dimensional data. Several important works have proposed *embedded formalisms* using variable weighting to solve the problem of unsupervised FS [10, 17]. For such approaches, the search for an optimal subset of features is built into the clustering construction making these techniques specific of a given learning algorithm. In contrast to wrapper and embedded approaches, *filter methods* discover the relevant and redundant features through analyzing the correlation and dependence among features without involving any clustering algorithms [19, 5]. The most common filter strategies are based on feature ranking. Recently, [15] proposed a consensus unsupervised feature ranking approach that combines multiple rankings of the full set of features into a single consensus one. The ranking of features is obtained using their relevance measured by the linear correlation coefficient and symmetrical uncertainty. Unfortunately, the authors only report experimental results on very low-dimensional data sets.

3 RCE: Random Cluster Ensembles for Unsupervised FS

3.1 Building and combining multiple clustering solutions

Ensemble methods have been applied successfully in the area of unsupervised learning to improve the accuracy and the robustness of clustering algorithms. *Resampling methods* such as Bagging, was one of the first approaches that exploited this idea of ensemble learning. A group of clustering models was built over a bootstrapped replicate of the former dataset. Finally, the last partition was made by a consensus function over the set of partitions of each single clustering model [27, 6]. Another trend appeared with the use of *random subspaces*. Likewise bagging, random subspaces are another excellent source of clustering

diversity that provides different views of the data and allow to improve the quality of unsupervised classification solutions [27, 28]. Since ensemble methods are designed to improve performance of supervised and unsupervised classification methods, it is reasonable to think that they can also be used to tackle the unsupervised FS problem.

Contrary to all the previous described approaches, we propose to combine both bagging and random subspaces for producing an ensemble of component clusterings. RCE achieves a population of clustering solutions with the following steps employed: a new training set is drawn, with replacement, from the original data set. Then m features are randomly selected from the entire feature set (leading to a partial view of the bootstrap data set) and a clustering solution is obtained through executing the "base" clustering algorithm on the selected features. The same steps is repeated r times. There are many reasons for using bagging in tandem with random feature subspaces. The first is that bagging can be used to give estimates of both the *variable importance* and the *pattern proximities* that will serve to build final consensus clustering from the ensemble of clusterings. The second is that the ensemble method combines many weak learners in an attempt to produce a strong learner [9]. Moreover, clustering in the projected subspace allows us to mitigate the curse of dimensionality.

Assume a method h for constructing a clustering from any training set. Given a specific data set $\mathcal{D} = \{x_1, \ldots, x_n\}$ with M input variables, form r boostrap training sets $\mathcal{D}_k$ ($k \in \{1, \ldots, r\}$) in a random feature subspace (a mD view of the bootstrap training set where $m = \sqrt{M}$), and construct clustering $\mathbf{C}^k = h(x, \mathcal{D}_k)$. The way these $\mathbf{C}^k$ ($k \in \{1, \ldots, r\}$) should be combined together is called the cluster ensemble problem. Several feasible approaches are introduced in the literature to solve the cluster ensemble problem. In this study, we adopt the average-link consensus function based on co-association values between data, proposed in [9]. Proximity between pair of cases simply counts the fraction of clusters shared by these objects in the initial partitions. Numerous similarity-based clustering algorithms can be applied to the proximity (or similarity) matrix to obtain the final partition. We chosed the Agglomerative Hierarchical Classification. The number of clusters is the one that maximizes the average mutual information criterion [27]. The overall framework is also applicable to any feasible consensus clustering procedure.

3.2 Out-of-bag estimates to measure variable importance

In the RCE method, bagging is used in tandem with feature subspace to give ongoing estimates of the feature importance of the combined ensemble of clusterings. A start on this problem is made by using internal out-of-bag (oob) estimates, and verification by reruns using only selected variables. These estimates are done out-of-bag, exactly as done in RF. After each clustering is constructed, the values of the vth variable in the oob patterns are randomly permuted and the oob data are re-assigned into clusters. This is repeated for $v = 1, 2, \ldots, M$. At the end of the run, the oob cluster assignments for x, with the vth variable

permuted, is compared with the original cluster assignment of x[1]. The average number of times the oob pattern x, with the variable v permuted, is misclassified divided by the number of clusterings in the ensemble r is the local importance score for variable v for this pattern. Now, the importance of the vth variable for a given cluster (local) in the final consensus clustering is calculated as the sum of all the importance values over all the patterns that fall into this particular cluster. Various methods can be employed to select the most important local variables in view of their importance estimates, including: 1) statistical tests (e.g., Scree Test [4]), 2) selecting a predefined percentage of variables [23] 3) the same recursive feature elimination scheme used with SVM [13]. In this study, the Scree Test [4] is used. It consists in selecting the features preceding a threshold value called the "Scree". The "Scree" corresponds to the point where the maximum deceleration of the curve occurs. See [4] for more details.

Our FS algorithm has several advantages when compared to other existing unsupervised FS algorithms: First, as mentioned in [14], most existing unsupervised FS algorithms are dimensionality-biased. For example, if the scatter separability based FS algorithm is adopted, high-dimensional feature subsets are selected more easily [7, 14]. The problem should be circumvented as RCE operates on low-dimensional feature spaces. Second, as RCE leverages different clustering solutions to measure feature importance, it is expected to improve the robustness and stability of the feature subset compared to FS algorithms based on a single clustering method. Third, the estimates of variable importance is obtained from the oob patterns only. As noted by Breiman, it should therefore be as accurate as using a test set of the same size as the training set. Therefore, using the oob error estimate removes the need for a set aside test set. In each bootstrap training set, about one-third of the instances are left out. Therefore, the oob estimates are based on combining only about one-third as many clustering models as in the ongoing main combination.

Given a data set $\mathcal{D} = \{x_1, \ldots, x_n\}$ with M input features $\mathcal{F} = \{f_1, \ldots, f_M\}$, the overall proposed RCE framework for localized unsupervised FS is summarized below:

1. Initialize an $n \times n$ matrix A and $n \times M$ matrix I to zero.
2. Form the kth bootstrap sample $\mathcal{D}_k = \{x_k^1, \ldots, x_k^n\}$ and the kth oob sample $\mathcal{D}_{k,oob} = \{x_{oob}^1, \ldots, x_{oob}^{n_{oob}}\}$ such that $\mathcal{D}_{k,oob} = \mathcal{D} \setminus \mathcal{D}_k$.
3. Create a random feature subset $\mathcal{F}_k = \{f_k^1, \ldots, f_k^m\}$ where $m = \sqrt{M}$.
4. Project $\mathcal{D}_k$ and $\mathcal{D}_{oob}^k$ onto feature subset $\mathcal{F}_k$: $\mathcal{D}_k = \mathcal{D}_{k_{|\mathcal{F}_k}}$ and $\mathcal{D}_{k,oob} = \mathcal{D}_{k,oob_{|\mathcal{F}_k}}$.
5. Apply the clustering procedure h to the bootstrap learning set $\mathcal{D}_k$ and obtain cluster labels $\mathbf{C}^k$.
6. For each pair of observations (x_k^i, x_k^j), update the matrix A as follow:

$$A(x_k^i, x_k^j) := A(x_k^i, x_k^j) + \tfrac{1}{r} \ \textit{iff} \ h(x_k^i, \mathcal{D}_k) = h(x_k^j, \mathcal{D}_k).$$

[1] An oob instance x is assigned to the closest cluster, where the distance from an instance to one cluster C is given by the distance between the instance and the centroid of C.

7. Classify each oob data x^i_{oob} into $\mathbf{C}^k$ and obtain its label $\mathbf{C}^k(i)$.
8. For each feature v in $\mathcal{F}_k$ randomly permute the values of v in oob data, re-assign each x^i_{oob} to a new label $\mathbf{C}^k_{new}(i)$ and update the matrix I as follow:

$$I(x^i_{oob}, v) := I(x^i_{oob}, v) + \frac{1}{r} \; \textit{iff} \; \mathbf{C}^k_{new}(i) \neq \mathbf{C}^k(i).$$

9. Repeat Steps 2-8 r times and define a new dissimilarity matrix D, by $D(i,j) := 1 - A(i,j)$.
10. Cluster the n original observations on the basis of this new dissimilarity matrix using average-link hierarchical agglomerative algorithm and obtain the combined partition $\mathbf{C}$.
11. For each obtained cluster c_k in $\mathbf{C}$ and each feature v in $\mathcal{F}$ measure the importance of v in c_k as $imp(v, c_k) = \sum_{i=1;x_i \in c_k}^{n} I(x_i, v)$.
12. For each obtained cluster c_k in $\mathbf{C}$ use the scree test to select the most important local variables.

4 Experiments

In this section, several benchmark (UCI) data sets [1] were selected to test the performance of RCE (*c.f.* Table 1). It should be noted that most of these data sets have frequently been used as benchmark data sets for testing the performance of some state-of-the-art unsupervised FS algorithms which will be compared to RCE. In addition, RCE is applied on two real gene expression data *Ovarian* [25] and *Leukemia* [12] in order to assess its performance against data with a large number of features and a small number of samples. The evaluation of the performance of RCE will be conducted as follows: 1) quality of selected features using *k-means* as the "base" clustering algorithm, 2) performances of RCE on large feature/small sample size domains and 3) impact of noisy features on RCE performances.

4.1 Evaluation using k-means as a base clustering algorithm

First, the k-means clustering algorithm was adopted as the "base" clustering algorithm. The clusters number of the "base" k-means clustering algorithm was set to the one optimizing Davies Bouldin index. The size of clustering ensembles r was 200 in our experiments. In these experiments, the number and quality of

Table 1. Characteristics of used data sets.

Data set	n	M	#labels
Ecoli	325	7	8
Iris	150	4	3
Lung	32	56	3
Vehicle	846	18	4
Wave	5000	40	3
Ovarian	54	1536	2
Leukemia	72	7129	2

the features selected by RCE was studied and compared with those obtained by state-of-the-art unsupervised FS algorithms: the scatter separability wrapper unsupervised FS algorithm [7], the DB-index wrapper unsupervised FS algorithm [20], the clustering ensembles guided FS algorithm (CEFS) [14] and the unsupervised RF FS algorithm [3][2]. To make fair comparisons, the same experimental approach (protocol and evaluation measure) in [14] was adopted here. The quality of the features selected by each approach on the first four data sets (which were used in [14]) were evaluated through executing k-means on them and the clustering accuracy was returned as the quality of the selected features. As in [14], accuracies of all clustering solutions were calculated by the Rand Index method [21] and all experiments were executed for 20 independent runs since k-means is very unstable[3]. On the other hand, since RCE provides a local FS and for fair comparison, the feature subset considered as the union of all local important features was considered as the final selected feature subset. Experimental results are shown in Table 2. The accuracies of k-means on the features selected by RCE are consistently higher than those obtained by k-means on all the features. k-means on all the features achieved accuracies around 80.54% for Ecoli data set, 86.32% for Iris data set, 63.25% for Lung data set and 64.17% for Vehicle data set. However, k-means clustering algorithm on the features selected by RCE achieves accuracies around 84.02% for Ecoli data set, 93.20% for Iris data set, 68.27% for Lung data set and 66.98% for Vehicle data set. This confirms the ability of our approach to improve the quality of clustering and generate meaningful clusters. In addition, RCE always identifies a better feature subset when compared with those obtained by the scatter separability wrapper method, the DB-index wrapper algorithm, the clustering ensembles guided FS algorithm (CEFS) and the unsupervised RF FS algorithm. Considering the unsupervised RF approach, we note that the missclassfication rate for Ecoli, Iris, Lung and Vehicule data sets are less than 50% indicating that unsupervised RF is unable to distinguish between the two classes (original and synthetic).

4.2 Results on large feature/small sample size domains

In this section, we present the results of our RCE unsupervised FS technique on large feature/small sample size domains. The*Self-Organizing Map clustering algorithm* (SOM) is used as a base clustering algorithm in RCE. The unsupervised FS techniques is assessed using 10-fold cross validation on two real data sets *Ovarian* and *Leukemia*. The data set was split into ten disjoint subsets of equal size (subsample contains 90% of the data for training and 10% for test). This percentage was chosen because we use small sample datasets and thus cannot discard too much data when building models. To demonstrate the effectiveness of RCE, its performance was compared to those of our gold standard ensemble supervised FS algorithm (RF). For each approach, the strategy explained before was used. Accuracy was used as the classification performance measure. For

[2] We note that the number of tree in the RF classifier was set to 500
[3] Results of three state-of-the-art approaches are given from [14].

Table 2. Experimental Results using k-means.

	Ecoli data set		**Iris data set**	
Methods	#features	Accuracy(%)	#features	Accuracy(%)
All features	7.00±0.00	80.54±0.82	4.00±0.00	86.32±2.83
RCE	5.30±0.65	84.02±0.89	1.78±0.57	93.20±1.87
Unsupervised RF	1.00±0.00	74.57±0.57	1.00±0.00	88.38±6.30
CEFS	5.96±0.10	82.51±0.24	2.00±0.00	92.56±2.96
Scatter separability	6.40±0.63	80.74±0.13	3.00±0.00	87.35±5.34
DB-index	2.00±0.00	70.95±0.43	1.00±0.00	91.22±4.28
	Lung data set		**Vehicle data set**	
Methods	#features	Accuracy(%)	#features	Accuracy(%)
All features	56.0±0.00	63.25±2.87	18.0±0.00	64.17±1.14
RCE	51.0±2.03	68.27±1.98	13,89±1.47	66.98±0.36
Unsupervised RF	53.0±0.00	62.12±3.45	10.0±0.00	64.56±1.34
CEFS	32.0±3.21	64.72±0.77	13.2±0.28	66.81±0.66
Scatter separability	40.4±1.79	64.28±0.27	13.0±0.00	64.19±1.99
DB-index	8.87±2.32	63.46±0.56	4.00±0.00	64.96±0.57

Table 3. Accuracy comparison for RCE and RF. Each entry in the table represents the average accuracy using 10-fold cross-validation.

Data sets	Consensus (all)	Consensus (sel. RCE)	RF (all)	RF (sel. RF)
Ovarian	83.33	94.66	86.00	96.00
Leukemia	76.60	83.75	85.00	94.00

each fold, FS was performed using only the training part of the data, and each approach (RCE and RF) was run again using the 1% best features returned by its first run, as it was used in [23] and observed in these domains that only such a small amount of features was relevant. Each model was then evaluated on the test part of the data for each fold, and results were averaged over all 10 folds. The results of this experiment are displayed in Table 3.

The returned results confirm again the usefulness of the FS strategy. The results output by the consensus clustering of RCE on the selected features are shown to be more relevant in view of the accuracy than that obtained by the consensus clustering of RCE on all the features. On the other hand, regarding accuracy values, the performances of RCE are acceptable compared to those returned by the supervised FS approach (RF), that has access to labels during the learning step.

4.3 Effect of number of noisy features on RCE performances

The *wave* data set [1] has 40 variables where the last 19 ones are totally irrelevant with mean 0 and variance 1. We experiment on this data set in order to study the impact of adding noisy features on the performance of RCE, where SOM algorithm is used as a base clustering algorithm. We first performed a FS on the original data set; then 190 noisy variables were added sequentially (the 19

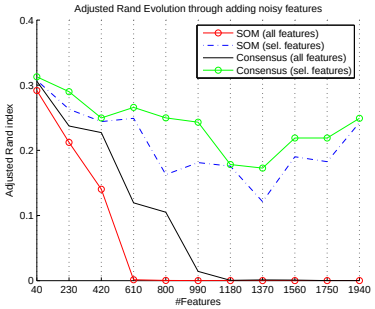


Fig. 1. Adjusted Rand index as a function of the number of irrelevant variables in the input.

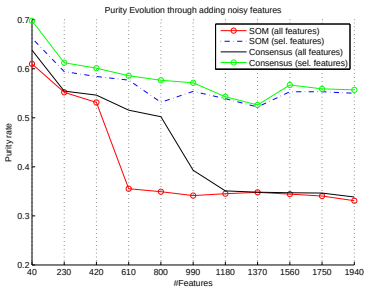


Fig. 2. Purity as a function of the number of irrelevant variables in the input.

irrelevant features duplicated 10 times) and the procedure was repeated several times. At each step, the clustering quality of SOM on all features is compared to the one returned from (1) clustering ensemble before FS (*consensus with all features*), (2) SOM clustering on features selected by RCE (*SOM with sel. features*) and (3) clustering ensemble on features selected by RCE (*consensus with sel. features*). Comparisons were made using two performance metrics: (*i*) *Purity rate* used to evaluate the *clustering accuracy* and (*ii*) *Adjusted Rand index* proposed by Hubert and Arabie [18] to assess the *degree of agreement* between two partitions (the one obtained with the clustering algorithm (*clusters*) and the correct predefined one (*labels*)) by considering relations upon patterns. Figures 1 and 2 show the evolution of Adjusted Rand and Purity values according to the number of features. The following conclusions can be drawn from these experiments. (1) The performance of a single application of SOM on all features deteriorated markedly with an increasing number of noisy features especially during the first steps. (2) The effectiveness of the clustering ensembles method to achieve better clustering results than single method is confirmed. (3) The pertinence of our RCE FS approach to offer an important preprocessing step for unsupervised learning and its robustness to noisy features are confirmed. It appears clearly that *consensus with sel. features* and *SOM with sel. features* achieve better results than *consensus with all features* and a significant improvement over SOM clustering on all features. Despite the degradation of *consensus with all features*, the output selected features are all correct as there are no false positives among the selected variables.

5 Conclusion

In this paper, the permutation accuracy importance measure in Random Forest was extended to unlabeled data. We showed that the way internal estimates are used to measure variable importance in Random Forests are also applicable to FS in unsupervised learning. We first illustrated the clustering performance of

the proposed method on various data sets based on widely used external criteria of clustering quality. We then assessed the accuracy and the scalability of the FS procedure on UCI and real labeled data sets and compared its effectiveness against powerful FS methods. Future substantiation through more experiments on databases containing several thousands of variables are currently being undertaken and comparisons with other unsupervised FS methods will be reported in due course.

References

1. C.L Blake and C.J Merz. Uci repository of machine learning databases, 1998.
2. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
3. L. Breiman and A. Cutler. *Random Forests Manual v4.0, Technical report, UC Berkeley*, 2003.
4. R. B. Cattell. The scree test for the number of factors. *Multivariate Behavioral Research*, 2:245–276, 1966.
5. M. Dash, K. Choi, P. Scheuermann, and H. Liu. Feature selection for clustering-a filter solution. In *International Conference on Data Mining*, pages 115–122, 2002.
6. S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.
7. J.G. Dy and C.E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5:845–889, 2004.
8. A.L.N. Fred and A.K. Jain. Data clustering using evidence accumulation. In *16^{th} International Conference on Pattern Recognition*, pages 276–280, 2002.
9. A.L.N. Fred and A.K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(6):835–850, 2005.
10. H. Frigui and O. Nasraoui. Unsupervised learning of prototypes and attribute weights. *Pattern recognition*, 37(3):567–581, 2004.
11. R. Ghaemi, N. Sulaiman, H. Ibrahim, and N. Mustapha. A survey: Clustering ensembles techniques. *World Academy Sc., Engineering and Technology*, 38, 2009.
12. T.R. Golub, Slonim, D.K., P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, and H. Coller. Molecular classication of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
13. I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
14. Y. Hong, S. Kwong, Y. Chang, and R. Qingsheng. Unsupervised feature selection using clustering ensembles and population based incremental learning algorithm. *Pattern Recognition*, 41(9):2742–2756, 2008.
15. Y. Hong, S. Kwong, Y. Chang, and Q. Ren. Consensus unsupervised feature ranking from multiple views. *Pattern Recognition Letters*, 29(5):595–602, 2008.
16. J. Hua, W. Tembe, and E. Dougherty. Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognition*, 42:409–424, 2009.
17. J. Huang, M. Ng, H. Rong, and Z. Li. Automated variable weighting.in k-means type clustering. *IEEE Trans Pattern Anal. Mach. Intell.*, 27(5):657–668, 2005.
18. L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

19. P. Mitra, A. Murthy, and S.K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):301–312, 2002.
20. M. Morita, R. Sabourin, F. Bortolozzi, and C.Y. Suen. Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. In *International Conference on Document Analysis and Recognition*, 2003.
21. W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.
22. S. Rodrigues de Morais and A. Aussem. A novel Markov boundary based feature subset selection algorithm. *Neurocomputing*, 73:578–584, 2010.
23. Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In *ECML PKDD*, pages 313–325, 2008.
24. Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23:95–116, 2007.
25. M. Schummer, W. V. Ng, and R. E. Bumgarnerd. Comparative hybridization of an array of 21,500 ovarian cdnas for the discovery of genes overexpressed in ovarian carcinomas. *Gene*, 238(2):375–385, 1999.
26. T. Shi and S. Horvath. Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics*, 15(1):118–138, 2006.
27. A. Strehl and J. Ghosh. Cluster ensembles-a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.
28. A. Topchy, A.K. Jain, and W. Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(12):1866–1881, 2005.
29. L. Yuanhong, D. Ming, and H. Jing. Localized feature selection for clustering. *Pattern Recognition Letters*, 29(1):10–18, 2008.

Embedding Random Projections in Regularized Gradient Boosting Machines

Pierluigi Casale, Oriol Pujol, and Petia Radeva

Computer Vision Center, Bellaterra, Barcelona, Spain
Dept. Applied Mathematics and Analysis, University of Barcelona, Barcelona, Spain
{pierluigi,oriol,petia}@cvc.uab.es

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.

Keywords: Random Projection, Gradient Boosting Machines, L2 Regularization.

1 Introduction

Recently, 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 [7] or Rotboost [8]. Using RPs, we can get multiple rotated views of the dataset. Diversity can be generated by projecting data (i) into random subspaces, (ii) into random spaces of the same dimension of the features spaces or (iii) superspaces of higher dimensionality than the original features spaces. In our work, we study different techniques to apply random projections and apply them in the construction of a gradient boosting ensemble.

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 [1]. Gradient Boosting Machines (GBMs) generalize this idea for any arbitrary loss function. We embed RPs in LsBoost, projecting data into random spaces at each step of the optimization process, thus searching for the classifier that

best fits the data in this new space. Nevertheless, the optimization of ill-posed problems generally yields to poor results. For this reason, we also take into account the effect of the L2 penalization term for regularization in the creation of the ensemble.

We evaluate our approach on using RPs on synthetic datasets. In addition, we evaluate the effect of both the regularization parameter and RPs on eight datasets from UCI repository [10]. We compare our approach with AdaBoost and LsBoost both with decision stumps as weak classifiers. Results show that, for some kind of problems and in particular Xor-type problems, using RPs significantly improves the classification accuracy. Furthermore, the use of the L2 regularization parameter is specially justified as a way to model noise and for ensuring smoothness in the solutions.

This paper is organized as follows. In the next section, we briefly describe previous works about how RPs have been used in the machine learning community. In Section 3 we first formulate the Gradient Boosting, then we present different types of RPs and finally, we describe how we embed RPs into the GBMs. In Section 4, we show the expetimental results and finally, in Section 5, we will discuss the results and conclude.

2 Related Works on RPs

Arriaga and Vempala [2] 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 in a random subspace and running the algorithm in that space, taking advantage of working with a lower dimensionality.

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 [4] 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.

3 Methods

In this section, first we present the formulation of both GBMs and RPs, and then we explain our proposal to embed RPs into the GBMs.

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 $F(\mathbf{x}; \mathbf{P})$:

$$F(\mathbf{x}; \mathbf{P}) = \sum_{m=0}^{M} \beta_m h(\mathbf{x}; \mathbf{a}_m), \tag{1}$$

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 :

$$\Phi(F(\mathbf{x})) = E_y[\Psi(y, F(\mathbf{x})|\mathbf{x})], \tag{2}$$

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

$$F^*(\mathbf{x}) = \sum_{m=0}^{M} f_m(\mathbf{x}), \tag{3}$$

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 :

$$f_m(\mathbf{x}) = -\varrho_m g_m(\mathbf{x}), \tag{4}$$

where, assuming that differentiation and integration can be interchanged,

$$g_m(\mathbf{x}) = E_y \left[\frac{\partial \Psi(y, F(\mathbf{x}))}{\partial F(\mathbf{x})} | \mathbf{x} \right]_{F(\mathbf{x}) = F_{m-1}(\mathbf{x})} \tag{5}$$

and

$$F_{m-1}(\mathbf{x}) = \sum_{i=0}^{m-1} f_i(\mathbf{x}). \tag{6}$$

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 perfom 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 :

$$\varrho_m = argmin_{\varrho} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\mathbf{x}_i) + \varrho h(\mathbf{x}_i; \mathbf{a}_m)) \tag{7}$$

and the approximation updated in the following way :

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \varrho_m h(\mathbf{x}; \mathbf{a}_m). \tag{8}$$

In the special case where $y \in \{-1, 1\}$ 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 1, is called LsBoost.

Table 1. Algorithm: LsBoost

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

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 Johson-Lindenstrauss Lemma [5] states that, given m points in $\mathbb{R}^n$ and $0 < \gamma < 1$, 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. If we consider points as row-vectors of length n, the projection can be performed by multiplying all the n points by an $n \times d$ matrix. The matrix should be one of the following:

- 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. Furthermore, if we would project the data into a superspace, we can not rely on any theoretical results.

3.3 Random Projections in Boosting Machine.

In order to embed RPs in LsBoost, we perfom a slight modification of the original *LsBoost* algorithm. In Step 4 of the algorithm, we can first search analitycally for the optimal set of weighting values for each candidate classifier and after select the classifier that best approximates the negative gradient with the correspondent precomputed weight. Following the procedure described by Pujol [6], we obtain that the values of all possible weights are given by :

$$\varrho_m = \frac{\tilde{\mathbf{y}}^m A}{N + \lambda}, \tag{9}$$

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 is the L2 penalization term. It has to be noted that, when $\lambda = 0$, regularization is not taken into account.

We define the *RpBoost* as Regularized Gradient Boosting where before considering the data by the weak classifiers they are projected by a transformation representing by a specific RPs technique. Table 2 defines the algorithm to apply the RpBoost method. In addition, we define :

Table 2. Algorithm: RpBoost

Initialize:	
	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
Step 1	$F_0(\mathbf{x}) = argmin_{\varrho} \sum_{i=1}^{N} \Psi(y_i, \varrho)$
Step 2	For $m = 1$ to M do:
Step 3	$\tilde{y}_i^m = y_i - F_{m-1}, i = 1, N$
Step 4	Set a new R_p
Step 5	$A_r = A \cdot R_p$
Step 6	$\varrho_m = \frac{\tilde{y}_m A_r}{N+\lambda}$
Step 7	$\mathbf{a}_m = argmin_{\mathbf{a}} \sum_{i=1}^{N} [\tilde{y}_i^m - \varrho_m h(\mathbf{x}_i; \mathbf{a})]^2$
Step 8	$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \varrho_m h(\mathbf{x}_i; \mathbf{a}_m)$
Step 9	end For

Def. 1: *Rpboost.sub* as RpBoost working of data projected into a random subspace;

Def. 2: *Rpboost.same* as RpBoost working of data projected into a random space of the same dimension than the original feature space;

Def. 3: *Rpboost.super* as RpBoost working of data projected into a random superspace.

4 Experiments and Results.

In order to test the performance of RpBoost, we perform several tests on synthetic and real datasets from the UCI database. Results obtained with RpBoost are compared with the results obtained with AdaBoost and LsBoost. In our setting, we use decision stumps as weak classifiers. Dimension of the ensemble has been set to 500 classifiers. In order to have a straightforward comparison, in all the experiments, we set the dimension of the subspace and of the superspace equal to the half and the double of the dimension of the features space, respectively. RPs are performed using a matrix $M \in \{P, U_{-1,1}, N_{0,1}\}$ as defined in the Johnson-Lindenstrauss Lemma. The only difference is that we do not orthogonalize column vectors of the P projections because an orthonormalization process could slow down the training phase. In this section we show results obtained on test patterns, UCI datasets and results about the effect of regularization in the creation of the ensemble.

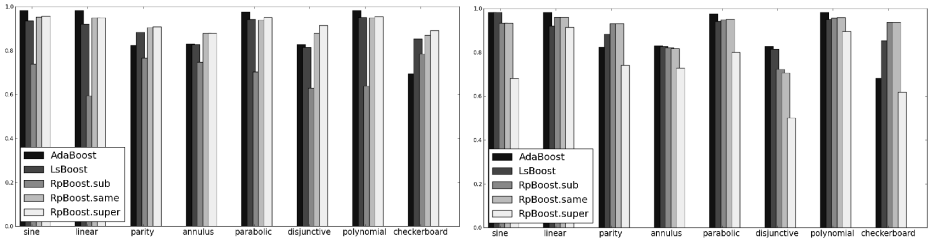
4.1 Test Patterns

Test patterns are synthetic bidimensional dataset proposed by Fawcett in [9] for comparing performances of different classifiers. These patterns are randomly generated on a 2-D grid of points, in [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 3 we give a description and some examples of the patterns. For each test pattern, we use a stratified sample of 1000 points as training set and all the distribution as testing set.

Table 3. Test Patterns.

Test Pattern	Description	Examples of Test Patterns
Sine	$Y = 0.84sin(1.78X)$	
Linear	$Y = 1.87 * X \pm 1.74$	
Parity	9 parity circles	
Annulus	Annulus at (2.00, 2.00)	Annulus
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	Checkerboard

In Figure 1(a) we show the classification accuracy obtained with AdaBoost, LsBoost and RpBoost using P projections. In Figure 1(b) we show the classification accuracy obtained with AdaBoost, LsBoost and RpBoost using $U_{-1,1}$ projections. In Figure 2(a) we show the classification accuracy obtained with AdaBoost, LsBoost and RpBoost using $N_{0,1}$ projections and in Figure 2(b) their respective numerical values. RpBoost.sub and RpBoost.same with $N_{0,1}$ projections give always better classification accuracies than AdaBoost or LsBoost and, in most of the cases, the performance increases considerably respect to AdaBoost and LsBoost. One can observe that P projections are the only ones where superpaces perform better than random subspace and random space of the same dimension of the feature space. Finally, it has to be noted that all types of projection always outperform AdaBoost and LsBoost on the parity pattern and, significantly, on the checkerboard pattern. These patterns represent XOR-type problems.

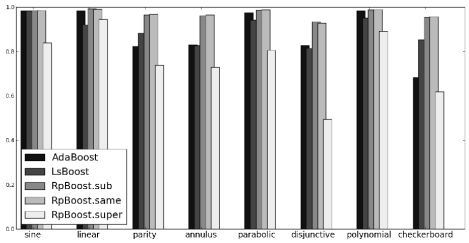


(a) (b)

Fig. 1. Classification Accuracy on Test Patterns obtained with AdaBoost, LsBoost and RpBoost with (a) P projections and (b) $U_{-1,1}$ projections.

4.2 UCI Datasets

We compare the behaviour of RpBoost, AdaBoost and LsBoost on eight datasets from UCI Repository in comparation with AdaBoost and LsBoost. All the results are 10-fold cross validated over two runs of cross validation. We use decision stumps as week classifiers. The value of the regularization parameter is found by 5-fold cross validation on the training set. We use the following datasets:



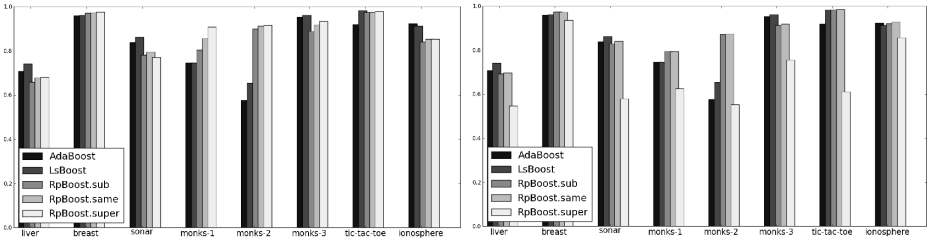
(a)

	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

(b)

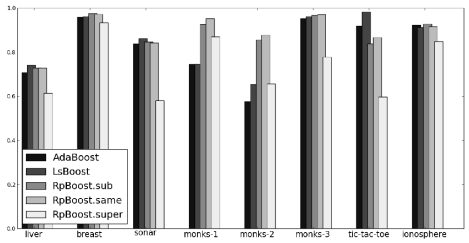
Fig. 2. (a) Classification Accuracy on Test Patterns obtained with AdaBoost, LsBoost and RpBoost with $N_{0,1}$ projections.
(b) Numerical Values of Accuracy on Test Patterns obtained with AdaBoost (Ada), LsBoost (Ls), RpBoost.sub (Sub) , RpBoost.same (Same) and RpBoost.super (Super) with $N_{0,1}$ projections.

Monks-1, Monks-2, Monks-3, Bupa Liver Disorders, Tic-tac-toe, Breast cancer Winsconwin (original), Ionosphere and Sonar. We choose to take into account the three separated Monks problems. In Figure 3(a) we show the mean accuracy obtained with AdaBoost, LsBoost and RpBoost using P. In Figure 3(b) we show the mean accuracy obtained with AdaBoost, LsBoost and RpBoost using $U_{-1,1}$. In Figure 4(a) we show the mean accuracy obtained with AdaBoost, LsBoost and RpBoost using $N_{0,1}$ and in Figure 4(b) their respective numerical values. $N_{0,1}$ and $U_{-1,1}$ projections give better classification accuracy with both RpBoost.sub and RpBoost.same. On the other side, superspaces provide good results only with P projections. For all projections and all space dimensions, RpBoost outperforms significantly both AdaBoost and LsBoost in Monks-1 and Monks-2 while , in Monks-3, only using $N_{0,1}$ projection slightly improves the performance. A slight improvement can be also noted in Breast for all projections. Sonar and Ionoshpere, the datasets having the highest dimensions, do not benefit of the dimensionality reduction that RPs provide.



(a) (b)

Fig. 3. Classification Accuracy on Uci Datasets obtained with AdaBoost, LsBoost and RpBoost with (a) P projections and (b) $U_{-1,1}$ projections.



(a)

	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

(b)

Fig. 4. (a) Classification Accuracy on Uci Datasets obtained with AdaBoost, LsBoost and RpBoost with $N_{0,1}$ projections.
(b) 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.

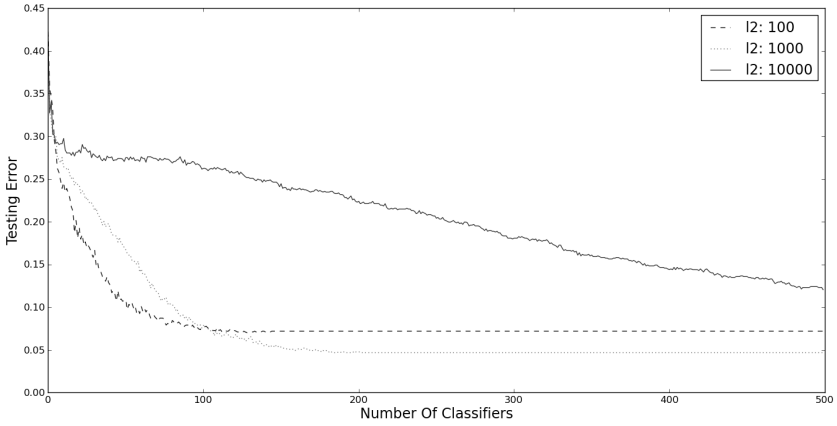
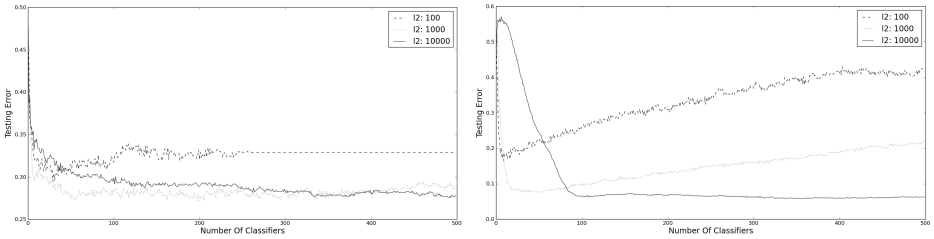


Fig. 5. Testing Error for Different Values of the Regularization Parameter for RpBoost.same using $N_{0,1}$ in Monks-1 dataset



(a) (b)

Fig. 6. Testing Error for Different Values of the Regularization Parameter for (a) RpBoost.same using $U_{-1,1}$ in Liver dataset and (b) RpBoost.super using $U_{-1,1}$ in Breast dataset.

4.3 The effect of Regularization in RPs.

In order to study the effect of the regularization parameter λ in the construction of the ensemble, we perform 10-fold cross validation over two run of cross validation for $\lambda \in \{1, 5, 10, 50, 100, 500, 1000, 5000, 10000\}$ for each dataset and for each type of projection. We can note the effect of the regularization parameter at the initial phase in the costruction of the ensemble, before the convergence of the optimization process. In Figure 5 we show the testing error of the Monks-1 dataset obtained in the construction of RpBoost.same using $N_{0,1}$ projection for $\lambda \in \{100, 1000, 10000\}$. In Figure 6(a) we show the testing error of the Liver dataset obtained in the construction of RpBoost.super using $N_{0,1}$ projection for same values of λ. In Figure 6(b) we show the testing error of the Breast Dataset obtained in the construction of RpBoost.super using $U_{-1,1}$ projection for the same values λ. The typical trend is shown in Figure 5 where we can see how the optimization process converges slower when the value of λ increases. It is evident how, in the former steps of the optimization process, λ influences the construction of the ensemble. With a proper value of λ, the testing error slows down fastly and overfitting is prevented, as shown in Figure 6(a). Furthermore, we note that for $\lambda = 100$, the ensemble overfits. This fact is evident in Figure 6(a) and, in particular, in Figure 6(b) where we can see that only for $\lambda = 10000$ the classifier does not tend to overfitting.

Table 4. Best classification accuracies obtained.

Test Patterns				Uci Datasets				
Test Pattern	Accuracy	Classifier	R_p	Dataset	Accuracy	Classifier	R_p	λ
Sine	98.6%	RpBoost.sub	N	Liver	74.2%	LsBoost	-	1000
Linear	99.3%	RpBoost.sub	N	Breast	97.6%	RpBoost.super	P	10000
						RpBoost.sub	N	10000
Parity	96.8%	RpBoost.same	N	Sonar	86.2%	LsBoost	-	1000
Annulus	96.5%	RpBoost.same	N	Monks-1	95.3%	RpBoost.same	N	1000
Parabolic	98.9%	RpBoost.same	N	Monks-2	91.6%	RpBoost.super	P	1000
Disjunctive	93.5%	RpBoost.sub	N	Monks-3	97.2%	RpBoost.same	N	1000
Polynomial	99.0%	RpBoost.same	N	Tic-tac-toe	98.6%	RpBoost.same	U	10
Checkerboard	95.7%	RpBoost.same	N	Ionosphere	92.8%	RpBoost.same	U	5000
						RpBoost.sub	N	1000

5 Discussion and Conclusions.

In this work, we propose to use random projections(RPs) to generate diversity in the construction of regularized Gradient Boosting Machines. In particular, we propose to embbed RPs in a modified version of LsBoost, that we call RpBoost. At each step of the optimization process, we project the data in a new random space and search for the classifier that best fits the data in this new space. Spaces can be random subspaces, random spaces of the same dimension than the original features space and random superspaces.

In Table 4, we report the best classification accuracies obtained for both test patterns, on the right, and Uci Datasets, on the left. For each test pattern, we report the accuracy, the classifier and the projections, denoted with R_p, providing that value of accuracy. For the datasets we report the accuracy, the classifier, the projection, denoted with R_p, and the value of regularization parameter λ providing that value of accuracy. Experimental results show that such type of

embedding, especially when RPs are drawn from a normal distribution, always provides better accuracies in the classification of synthetic data. In particular, RPs help significantly in the classification of Xor-type problems. Boosting with decision stumps is unable to solve such type of problems. This fact is evident in the checkerboard test pattern and in the Monks-1 and Monks-2 datasets. In these cases, the performance of RpBoost is considerably improved compared to the performance obtained with AdaBoost or LsBoost.

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. With 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 other types of projections.

The regularization parameter influeces 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 minor number of classifiers in the ensemble would have to provide better classification accuracy.

Finally, results clearly show that RpBoost is a promising technique and encourage in following in its study and, in particular, for more realistic problems.

Acknowledgments. This work is partially supported by a research grant from projects TIN2009-14404-C02 and CONSOLIDER (CSD2007-00018).

References

1. Friedman, J.H. : Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics 29, 1189–1232 (2000)
2. Arriaga, R., Vempala, S. : Algorithmic Theories of Learning. Foundations of Computer Science (1999).
3. Dasgupta, S.: Experiments with Random Projections In: 16th Conf. on Uncertainity in Artificial Intelligence, 2000
4. Fradkin, D. , Madigan, D.: Experiments with Random Projections for Machine Learning In: KDD 03: 9th ACM SIGKDD International conference on Knowledge discovery and data mining, pp. 517–522. ACM Press, (2003)
5. Johnson, W. , Lindenstrauss. J. : Extensions of lipschitz maps into a hilbert space: Contemporary Mathematics, 26:119-139, (1984)
6. Pujol, O. : Boosted Geometry-Based Ensembles In: El Gayar, N. , Kittler, J., and Roli, F. (eds.) MCS 2010. LCNS, vol. 5997, pp. 195–204. Springer, Heidelberg (2006)
7. Rodriguez, J.J. Kuncheva, L.I. Alonso, C.J. : Rotation Forest: A New Classifier Ensemble Method. In: TPAMI 28:10:1619–1630 , (2006)
8. Zhang, C.X. Zhang, J.S. : A technique for combining Rotation Forest and AdaBoost In : PRL, 29:10:1524–1536, (2008)
9. Comparing Patterns Classifiers. `http://home.comcast.net/~tom.fawcett/public_html/ML-gallery/pages/index.html`
10. UCI machine learning repository. `http://archive.ics.uci.edu/ml/datasets.html`

Trading-off diversity and accuracy for optimal ensemble tree selection in random forests

Haytham Elghazel, Alex Aussem, and Florence Perraud

Université de Lyon, F-69000, Lyon ; Université de Lyon 1 ;
Laboratoire LIESP, 69622 Villeurbanne, France
haytham.elghazel@univ-lyon1.fr

Abstract. We discuss an effective method for optimal ensemble tree selection in random forests by trading-off diversity and accuracy of the ensemble during the selection process. As the chances of overfitting increase dramatically with the size of the ensemble, we wrap cross-validation around the ensemble selection to maximize the amount of validation data considering, in turn, each fold as a validation fold to select the trees from. The aim is to increase performance by reducing the variance of the tree ensemble selection process. We demonstrate the effectiveness of our approach on several UCI and real-world data sets.

Keywords: Ensemble selection, random forest, decision trees

1 Introduction

Many advances in Machine Learning suggest using a set of individual classifiers, or ensemble of classifiers, instead of a single predictor to address supervised classification problems [15]. A large number of studies show that ensemble of classifiers generally achieve better results compared to a single classifier in terms of misclassification error [21, 11]. This improvement of performances relies on the concept of diversity which states that a good classifier ensemble is an ensemble in which the examples that are misclassified are different from one individual classifier to another. However, the practical trade-off between diversity and accuracy of the ensemble learning is still an open question in machine learning [6]. Dietterich [10] states that "A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse". Many methods have been proposed to generate accurate, yet diverse, sets of models. Bagging [4], boosting [12], random forests [5] and their variants are the most popular examples of this methodology. Boosting and random forests are comparable and sometimes better than state-of-the-art methods in classification and regression [8]

Random forests (RF) is a popular and very efficient algorithm, based on model aggregation ideas, for both classification and regression problems [5]. The principle of RF is to combine many binary decision trees built using several bootstrap samples coming from the learning sample and choosing randomly at each node a subset of explanatory variables. We assume the reader already is

familiar with details of the RF procedure. RF algorithm becomes more and more popular and appears to be computationally effective and offers good prediction performance in a lot of different applications [23]. Breiman sketches an explanation of the good performance of RF related to the good quality of each tree together with the small correlation (denoting high diversity) among the trees of the forest. However, the mechanisms that explain this good performance of RF are not clearly elucidated from a mathematical point of view [2]. Indeed, it appears that using random selection of variables during the design stage of RF makes individual trees rather weak predictors and does not always give the expected performances. In addition, Breiman observed that above a certain number of trees, adding more trees does not allow to improve the performance [5]. Precisely he stated that for an increasing number of trees in the forest, the generalization error converges to a maximum. This result indicates that the number of trees in a forest does not have to be as large as possible to produce an accurate RF. On the other hand, pruning the irrelvant trees from huge forest is not as easy as one may think. The tree selection process is subject to overfitting problems especially when the number of validation samples provided is not sufficient [1, 9]. Also, we believe there is still room for improvement.

In this work, we discuss a simple framework called *Fitselect* to improve RF under Ensemble pruning (also called Overproduce and Choose Paradigm). The main idea of our approach is to perform classifier selection from an initial pool of decision trees obtained with the RF algorithm while focusing on the trade-off between accuracy and diversity of selected trees. The proposed method works by evaluating the qualities of all obtained trees in terms of *accuracy-diversity trade-off* on a hillclimb (validation) set, and selectively choosing part of promising trees to build the final RF. To alleviate the overfitting problem, we wrap cross-validation around ensemble selection to maximize the amount of validation data considering, in turn, each fold as a validation fold. A distinct tree selection is performed in each RF model. Improvements are demonstrated on several classification data sets. Empirical results show that the selected subset of trees performs similar to or better than the original ensemble (RF).

The rest of the paper is organized as follow: Section 2 reviews recent studies on Ensemble pruning. Section 3 introduces the $fitselect$ framework for improving Random Forest. Experiments using relevant benchmarks and real data sets are presented in Section 4.

2 Background of Ensemble Selection

The Ensemble Selection (also called Ensemble Pruning or Overproduce and Choose paradigm) consists in selecting the ensemble members from a set of individual classifiers that are subject to less resource consumption and response time with accuracy that is similar to or better than the original ensemble. In supervised classification, it has been known that selective classifier ensembles can always achieve better solutions when compared with traditional ensemble methods [24, 19]. Given an ensemble of size M, the problem of finding the sub-set

of ensemble members with the optimal generalization ability involves searching the space of 2^{M-1} non-empty sub ensembles, which was proved to be an NP-complete problem [17]. Like ensemble learning approaches, the performance gain of the ensemble pruning methods stems from the accuracy-diversity trade-off, where choosing only the most accurate individual classifiers to form the sub ensemble is theoretically unsound [24] and a strategy that only considers diversity for pruning does not always give the expected performances in terms of misclassification error [19]

Many pruning algorithms exist for selecting good sub ensembles to reduce the size of ensemble without compromising its performance. [24] formulated the ensemble selection problem as a combinatorial optimization problem to look for a subset of classifiers that has the optimal accuracy-diversity trade-off. Their quadratic programming method was not significantly better than other metric-based pruning heuristics though. Most of the pruning approaches that appeared in the literature reorder the original ensemble members based on some predefined criteria and select a subset of ensemble members from the sorted list [18, 16, 9]. A straightforward classifiers selection method is to rank the classifiers according to their individual performance on a hillclimb (validation) set and pick the best ones. The Choose Best heuristic described in consists in selecting the L classifiers (among M initial classifiers) which possess the highest individual accuracy. This heuristic does not take into account diversity. To this purpose, Margineantu and Dietterich [18] use the Adaboost algorithm to train an ensemble of decision trees which is pruned with different selection heuristics. Of them, the Kappa Pruning heuristic aims at maximizing the pair-wise difference between the selected ensemble members. In [16], the authors proposed to use the clustering algorithm to prune redundant neural networks for maintaining the diversity of the ensemble committee of neural networks. The major problem with the above algorithms is that they do not consider the trade-off between accuracy and diversity. To solve this limitation, Margineantu and Dietterich [18] suggest pruning the initial set of M classifiers using Kappa-error convex hull criteria that is a diagram-based heuristic targeting at a good *accuracy-diversity trade-off* among the selected subsets. [13] proposed a genetic algorithm to study the trade-off between accuracy and diversity for ensemble pruning. Using an iterative process, the proposed approach evaluates an *accuracy-diversity trade-off* measure for different sub ensemble solutions and the sub ensemble with the highest value is returned by the algorithm. This approach is often subject to overfitting problems especially when the number of validation samples provided is not sufficient. On the other hand, Ensemble selection also employs greedy forward selection to select models to add to the ensemble [9]. Compared to previous works, ensemble selection uses many more classifiers, allows optimizing to arbitrary performance metrics, and includes refinements to prevent overfitting to the ensembles hillclimb data. Ensemble selection of classifiers from a large and diverse library of base classifiers have been shown to be most competitive learning principles in a recent world-wide KDD'09 Cup Orange Challenge [20].

Most of the methods proposed in the literature are based on a single parameter to select the ensemble members. Selection-based methods in [18, 13] consider both diversity and accuracy to train the ensemble but we currently lack effective pruning principles to consider the individual *accuracy-diversity trade-off* "contribution" that each ensemble member can bring to the ensemble. So it seems that there are still many unanswered questions: (1) What is a good balance between accuracy and diversity for each individual classifier in the ensemble? (2) Should different data sets have different control parameters? and (3) which refinements could be included to prevent overfitting ? We address some of these above concerns in the next section.

3 Contribution

In this section, we describe our algorithm, called $Fitselect$, to select the classifier ensemble under the Overproduce and Choose paradigm. We restrict our approach to the RF but it is straightforward to generalize the method to any library of general classifiers. $Fitselect$ belongs to the pruning approaches that reorder the original ensemble members based on some predefined criteria and select a subset of ensemble members from the sorted list. The training data set is subdivided into "training" and "validation" subsets. The training subset serves to construct the RF and the validation set is used for ensemble selection. $Fitselect$ works by evaluating the accuracy and diversity of the decision trees in the RF and selecting the promising trees. The final solution is achieved by combining all the selected trees from the original forest. To study the trade-off between accuracy and diversity, we use the fitness function employed in [21, 13] to evaluate each decision tree h_i. Given a RF $\mathcal{H}$ with M decision trees $\{h_1, h_2, \ldots, h_M\}$, the fitness function of each decision tree h_i is given by:

$$fitness(h_i) = \alpha \times \frac{a(h_i)}{m_a} + (1 - \alpha) \times \frac{d(h_i)}{m_d} \quad (1)$$

where $a(h_i)$ and $d(h_i)$ stands respectively for the accuracy and the diversity of the tree h_i computed on the validation data set. m_a and m_d denotes respectively the maximal accuracy and diversity overall trees in the forest $\mathcal{H}$. $0 \leq \alpha \leq 1$ is a control parameter that balances the accuracy and the diversity. The accuracy $a(h_i)$ is defined as the average correct predictions of h_i on the validation set.

Two classifiers C_i and C_j are said diverse if they assign different class labels to the same examples. Various measures have been proposed to quantify the diversity between two classifiers from their respective outputs [15, 18, 13]. In this study, we define the diversity $d(h_i)$ to be the average Hamming distance, computed on the validation set, between the prediction of h_i and the other trees in the forest. Accuracy and diversity values are normalized separately (using respectively the m_a and m_d factors) so that the values range from 0 to 1. Normalizing both terms allows α to have the same meaning across multiple domains. Once the fitness scores have been calculated for a given α, we rank all the trees according to their fitness values and we select the first $L < M$ trees that maximize the accuracy only of the ensemble on the validation set. In other terms,

there is no $L' \neq L$ such that the L' first trees are more accurate than the L first trees in the ordered list. The search is in $O(M)$. The selected trees form the new forest. To summarize, the fitness is used for ranking only, and accuracy is used for selecting. Note that any arbitrary performance metric could be useed instead of accuracy, like the area under the ROC curve [20].

As there are a large number of trees to select from, the chances of overfitting increase dramatically [9, 7]. In addition, ensemble selection is still prone to overfitting when the validation set is small. We therefore propose to wrap cross-validation around ensemble selection to maximize the amount of validation data considering, in turn, each fold as a validation fold. This has a bagging like effect that combat overfitting. A cross-validated RF is created by training a tree in each fold. If there are K folds, there will be K individual RF models (each trained on a fraction of $K - 1/K$ training points). These RF differ due to their different training samples. Tree ensemble selection is performed in each RF model. The aim is to increase performance by reducing the variance of the forward stepwise selection process. While adding cross-validation to ensemble selection is computationally expensive, the ensemble selection method is far simpler than the sequential forward selection (SFS) and sequential backward selection (SBS) discussed in [1], as well as the more complex genetic algorithm proposed in [13], since the ranking is performed once in $O(M \log(M))$. The methods is of course sub-optimal however it is very fast and simple. The final tree ensemble is obtained through combining all the selected trees obtained over the different K cross-validation steps. As the combined selected trees are obtained from K different training and model selection process, the chances of finding combinations of trees that overfit the validation sets is minimized.

The balance between diversity and accuracy is a controversial issue, so it is unclear what value α should take. We believe it should be adjusted to the data set at hand. Therefore, we perform multiple runs with increasing values of α (10% increase at each step) as done in [21]. The value of α which produces the highest *average overall accuracy* on the K different sub ensembles of selected trees $\{S_{(\alpha,1)}, \ldots, S_{(\alpha,K)}\}$ is used to rank and select the trees. Note that simple majority voting is used to combine the predictions of the ensembles. The *average overall accuracy* Acc_α over the K cross-validation steps $\{S_{(\alpha,1)}, \ldots, S_{(\alpha,K)}\}$ for a given value α is measured as:

$$Acc_\alpha = \frac{1}{K} \sum_{j=1}^{K} Acc_{(\alpha,j)} \tag{2}$$

where $Acc_{(\alpha,j)}$ corresponds to the ensemble's accuracy of tree sub ensemble $S_{(\alpha,j)}$ selected for the cross-validation step j. Algorithm 1 summarizes the overall approach.

Algorithm 1 pseudo code for $FitSelect$

for each cross-validation step $j \in [1, K]$ **do**
  $\mathcal{H}_j$: Construct a RF with M decision trees $\{h_{1,j}, h_{2,j}, \ldots, h_{M,j}\}$ on the training set.
  $\{a(h_{1,j}), ..., a(h_{M,j})\}$: Calculate (on the validation set) the tree accuracies in $\mathcal{H}_j$.
  $\{d(h_{1,j}), ..., d(h_{M,j})\}$: Calculate (on the validation set) the tree diversity values using the average Hamming distance between prediction of each tree and all the other trees in $\mathcal{H}_j$
end for
for each value of $\alpha \in [0, 1]$; $\alpha = \alpha + 0, 1$ **do**
  for each cross-validation step $j \in [1, K]$ **do**
    $\{fitness_\alpha(h_{1,j}), \ldots, fitness_\alpha(h_{M,j})\}$: Calculate the tree fitness values as $fitness_\alpha(h_{i,j}) = \alpha \times \frac{a(h_i)}{m_a} + (1 - \alpha) \times \frac{d(h_i)}{m_d}$
    Sort the trees $\{h_{1,j}, \ldots, h_{M,j}\}$ in decreasing order of the fitness
    Select the $(L < M)$ first trees $S_{(\alpha,j)} = \{h_{1,j}, \ldots, h_{L,j}\}$ that maximize, $A_{(\alpha,j)}$, the ensemble's accuracy (using majority voting) of the selected trees in $S_{(\alpha,j)}$ on the validation set
  end for
  Calculate the average overall accuracy $Acc_\alpha = \frac{1}{K} \sum_{j=1}^{K} Acc_{(\alpha,j)}$ of the K selected sub ensembles $\{S_{(\alpha,1)}, \ldots, S_{(\alpha,K)}\}$
end for
$\alpha_{opt} = argmax_{\alpha \in [0,1]}(Acc_\alpha)$
$RF^{(final)}$ = Combine $\{S_{(\alpha_{opt},1)}, \ldots, S_{(\alpha_{opt},K)}\}$

4 Empirical Results

4.1 Experiments on benchmark data sets

The *Fitselect* RF selection algorithm was tested on 11 binary classification problems: Clean, Haberman, Madelon, Pima, Spamb, Transfusion, Wdbc from the UCI repository [3], Engytime [22] and Leukemia [14]. The characteristics of data sets are reported in Table 1. For each problem we splitted the data set into a training and testing sets of equal sizes. In order to guarantee the original class distribution within training and testing sets, a proportionate stratified sampling was applied. The training set was splitted using a 3-fold cross-validation: 3 RF models with 500 decision trees were trained on a fraction of 2/3 training points. Selection was performed on the 1/3 withheld data points using *Fitselect*. The testing set was used to evaluate the performance of the final tree ensemble returned by *Fitselect*. Four performance metrics were used. Three threshold metrics: accuracy (Acc), recall (Rec), F-Score (F) and one ordering/rank metric: the area under the ROC curve (AUC).

For each dataset, the performance of the classifier ensemble obtained with *Fitselect* was compared to (1) the unpruned tree ensemble obtained from learning the RF algorithm on the whole training set and (2) the forward ensemble

Table 1. Data sets used in the experiments.

Data sets	# instance	# features
Clean	476	166
Engytime	4096	2
Haberman	306	3
Leukemia	72	7129
Madelon	2600	500
Pima	768	8
Spamb	4601	57
Transfusion	748	4
Wdbc	569	30

selection method (which we will refer to as *Forward*) [9] used by the winner of the recently KDD Cup Orange Challenge (2009). To test the statistical relevancy of the results, we used Wilcoxon signed-rank test, a non-parametric equivalent of paired t-test. Table 2 shows that the performances of the *Fitselect* outperformed both RF and the ensemble selection algorithm proposed in [9] for the majority of the data sets (except for the recall metric for Madelon and the AUC value for Transfusion). The difference in accuracy, over all data sets, between *Fitselect* and the *Forward* selection algorithm is significant at 0.01 level using the Wilcoxon signed-rank test, at 0.04 for Recall, at 0.01 for F-Score and at 0.02 for AUC. The difference in accuracy, over all data sets, between *Fitselect* and the RF is significant at 0.01 for accuracy, at 0.04 for Recall, at 0.01 for F-Score and at 0.02 for AUC. In all cases, the test statistics were less than the critical value for a two-tailed p-value of 0.05 so the differences in performnce between *Fitselect* and both RF and *Forward* approaches were significant at this level. In a way of conclusion, we suggest that adaptively trading off diversity and accuracy during the tree selection on cross-validated data sets is adequate for improving RF predictions. It seems however difficult to extract any general conclusion about the best trade-off between accuracy and diversity. The value of α_{opt} varied significantly from on data set to another, from 0.3 up to 1 (for Leukemia and Transfusion), indicating that accuracy tend to be favored more than diversity during the ensemble pruning. Although its effectiveness is confirmed for a library of heterogeneous models, our experiments suggest that the ensemble selection method proposed in [9] is not effective in the RF framework.

4.2 Experiments on real data sets

In this section, we report very briefly on some investigations with *Fitselect* on real-world data to illustrate the usefulness of the method in a real breast cancer (BC) epidemiological study conducted by Dr. Corbex at the World Health Organization located in Cairo. The overall purpose of the study was to investigate if the psychological, economic, or socio/cultural profile of women in Egypt can

Table 2. Performance scores on benchmark data sets. Best scores are in boldface. The number of trees returned by the *Forward* method for each data set is given in parentheses; α_{opt} and the number of trees selected by *Fitselect* are given in parentheses.

	Clean data set			**Engytime data set**		
	RF	Fwd(**37**)	Fitsel.(**0.6,80**)	RF	Fwd(**37**)	Fitsel.(**0.9,90**)
Acc	0.7322	0.7029	**0.7657**	0.9639	0.9624	**0.9663**
Rec	**0.6346**	0.5288	0.6635	**0.9600**	0.9502	**0.9600**
F	0.6735	0.6077	**0.7113**	0.9637	0.9619	**0.9661**
AUC	0.7953	0.7574	**0.8278**	0.9883	0.9840	**0.9886**
	Haberman data set			**Leukemia data set**		
	RF	Fwd(**21**)	Fitsel.(**0.7,60**)	RF	Fwd(**21**)	Fitsel.(**1,30**)
Acc	0.6883	0.7078	**0.7208**	0.8919	0.9189	**0.9459**
Rec	**0.5122**	0.3902	**0.5122**	0.6923	0.7692	**0.8462**
F	0.4667	0.4156	**0.4941**	0.8182	0.8696	**0.9167**
AUC	0.7086	0.7127	**0.7182**	0.9792	0.9744	**0.9812**
	Madelon data set			**Pima data set**		
	RF	Fwd(**163**)	Fitsel.(**0.3,230**)	RF	Fwd(**136**)	Fitsel.(**0.6,100**)
Acc	**0.7423**	0.7231	0.7408	0.7786	0.7682	**0.7865**
Rec	0.6631	**0.6923**	0.6708	0.5448	0.5448	**0.5672**
F	0.7201	0.7143	**0.7213**	0.6320	0.6213	**0.6496**
AUC	**0.8080**	0.7865	0.8077	0.8463	0.8241	**0.8569**
	Spamb data set			**Transfusion data set**		
	RF	Fwd(**54**)	Fitsel.(**0.8,70**)	RF	Fwd(**25**)	Fitsel.(**1,30**)
Acc	0.9166	0.8979	**0.9183**	0.7727	0.7701	**0.7781**
Rec	0.9261	0.9107	**0.9261**	0.1685	0.2022	**0.2135**
F	0.8974	0.8755	**0.8994**	0.2609	0.2951	**0.3140**
AUC	0.9675	0.9598	**0.9690**	0.6366	**0.6481**	0.6439
	Wdbc data set					
	RF	Fwd(**31**)	Fitsel.(**0.8,40**)			
Acc	0.9544	0.9544	**0.9719**			
Rec	0.9497	0.9609	**0.9665**			
F	0.9632	0.9636	**0.9774**			
AUC	0.9899	0.9921	**0.9931**			

be predictive of the delays between: 1) the first symptoms and the first visit to a doctor, and 2) the first visit to a doctor and the effective diagnosis. The first delay is mainly women dependent, while the second is mainly dependent on the health system. These delay values were binned into two bins according to the epidemiologist: "short delay" (class 1) and "long delay" (class 2). 204 patients treated in Cairo were interviewed according to a questionnaire with up to 70 questions (medical journey, personal journey and socio-cultural barriers). Explanatory categorical variables included socio-economic status (education level and economic capacity), socio-economic status (age, marital status, residence, household, etc), awareness and beliefs (knowledge about BC, stigma, modesty issues, etc.), behaviors and attitudes (what women do, what they share about

Table 3. The data sets description.

Data sets	# instance	# features	long	short
Delay between symptoms and consultation	201	40	99	102
Delay between consultation and diagnosis	173	36	86	87

Table 4. Performance results for the two tasks.

	Delay between first symptoms and doctor consultation			**Delay between doctor consultation and effective diagnosis**		
	RF	FWD(**44**)	FITSEL.(**0.6,130**)	RF	FWD(**92**)	FITSEL.(**0.7,70**)
ACC	0.6238	0.5446	0.6634	0.6437	0.5977	0.6782
REC	0.5686	0.4510	0.5686	0.5000	0.5227	0.6364
F	0.6042	0.5000	0.6304	0.5867	0.5679	0.6667
AUC	0.6425	0.5539	0.6531	0.6641	0.6414	0.6942

their disease and with who, etc.). For each explained variable, the epidemiologist has selected a subset of explanatory variables and a subset of women, see Table 3.

Here again, the performance of the classifier ensemble obtained, for both classification tasks, with *Fitselect* was compared to (1) the unpruned tree ensemble obtained from learning the RF algorithm on the whole training set and (2) the forward ensemble selection method. Results are reported in Table 4. As may be seen, *Fitselect* outperformed the other algorithms by a noticeable margin on the both classification tasks. Surprisingly, the *Fwd* method performed worse than RF. Future work will aim to extract the most important variables that explains a promising accuracy of about 67% for two tasks, from the selected trees.

5 Conclusion

This paper introduced a tree ensemble selection method to improve efficiency and effectiveness of RF by adaptively trading off diversity and accuracy according to the data. We wrapped cross-validation around ensemble selection to maximize the amount of validation data considering, in turn, each fold as a validation fold. Tree ensemble selection was performed in each RF model. The tree selection method called *Fitselect* was shown to increase performance by reducing the variance of the ensemble selection process, however the gain in performance was relatively modest in our experiments. It would be interesting for work to be performed to ascertain the classification problems for which the *Fitselect* is most suited. Moreover, it seems difficult to extract any general conclusion about the best trade-off between accuracy and diversity in view of our experiments.

References

1. S. Bernard, L. Heutte, and S. Adam. On the selection of decision trees in random forests. In *International Joint Conference on Neural Network (IJCNN'09)*, pages 302–307, 2009.
2. G. Biau, L. Devroye, and G. Lugosi. Consistency of random forests and other averaging classiers. *Journal of Machine Learning Research*, 9:2039–2057, 2008.
3. C.L Blake and C.J Merz. Uci repository of machine learning databases, 1998.
4. L. Breiman. Bagging predictors. *Machine Learning*, 26(2):123–140, 1996.
5. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
6. G. Brown and L.I. Kuncheva. "good" and "bad" diversity in majority vote ensembles. In *Proceedings of Multiple Classifier Systems 2010, LNCS 5997*, pages 124–133, 2010.
7. R. Caruana, A. Munson, and A. Niculescu-Mizil. Getting the most out of ensemble selection. In *Proceedings of the 6th International Conference on Data Mining (ICDM '06)*, December 2006.
8. R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, 2006.
9. R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *21^{st} International Conference on Machine Learning*, 2004.
10. T.G. Dietterich. Ensemble methods in machine learning. In *First International Workshop on Multiple Classier Systems*, pages 1–15, 2000.
11. T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting and randomization. *Machine Learning*, 40:139–157, 2000.
12. Y. Freund and Robert E. Shapire. Experiments with a new boosting algorithm. In *13^{th} International Conference on Machine Learning*, pages 276–280, 1996.
13. D. Gacquer, V. Delcroix, F. Delmotte, and S. Piechowiak. On the effectiveness of diversity when training multiple classifier systems. In *European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty ECSQARU'09*, pages 493–504, 2009.
14. T.R. Golub, Slonim, D.K., P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, and H. Coller. Molecular classication of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
15. L.I. Kuncheva. *Combining Pattern Classiers: Methods and Algorithms*. Wiley Interscience, 2004.
16. G. Li, J. Yang, A.S. Kong, and N. Chen. Clustering algorithm based selective ensemble. *Journal of Fudan University*, 43:689–695, 2004.
17. Z. Lu, X. Wu, and J. Bongard. Ensemble pruning via individual contribution ordering. In *ACM Knowledge Discovery and Data Mining SIGKDD*, 2010.
18. D.D. Margineantu and T.G. Dietterich. Pruning adaptive boosting. In *14^{th} International Conference on Machine Learning*, pages 211–218, 1997.
19. G. Martinez-Munoz, D. Hernandez-Lobato, and A. Suarez. An analysis of ensemble pruning techniques based on ordered aggregation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):245–259, 2009.
20. A. Niculescu-Mizil and et al. Winning the kdd cup orange challenge with ensemble selection. In *Journal of Machine Learning Research, to appear*, 2010.
21. D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Articial Intelligence Research*, 11:169–198, 1999.

22. A. Ultsch. Fundamental clustering problems suite, 2005.
23. R. Diaz Uriarte and S. Alvarez de Andres. Gene selection and classication of microarray data using random forest. *BMC Bioinformatics*, 7(3):1–13, 2006.
24. Y. Zhang, S. Burer, and W. N. Street. Ensemble pruning via semi-denite programming. *Journal of Machine Learning Research*, 7:1315–1338, 2006.

Facial Action Unit Recognition using Filtered Local Binary Pattern Features with Bootstrapped and Weighted ECOC Classifiers

R.S.Smith and T.Windeatt

Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, Surrey GU2 7XH, UK

{Raymond.Smith, T.Windeatt}@surrey.ac.uk.

Abstract. Within the context face expression classification using the facial action coding system (FACS), we address the problem of detecting facial action units (AUs). The method adopted is to train a single error-correcting output code (ECOC) multiclass classifier to estimate the probabilities that each one of several commonly occurring AU groups is present in the probe image. Platt scaling is used to calibrate the ECOC outputs to probabilities and appropriate sums of these probabilities are taken to obtain a separate probability for each AU individually. Feature extraction is performed by generating a large number of local binary pattern (LBP) features and then selecting from these using fast correlation-based filtering (FCBF). The bias and variance properties of the classifier are measured and we show that both these sources of error can be reduced by enhancing ECOC through the application of bootstrapping and class-separability weighting.

1 Introduction

The facial-action coding system (FACS) of Ekman and Friesen [1,2] is commonly employed in applications which perform automatic facial expression recognition. In this method, individual facial movements are characterised as one of 44 types known as action units (AUs). Groups of AUs may then be mapped to emotions using a standard code book. Note however that AUs are not necessarily independent as the presence of one AU may affect the appearance of another. They may also occur at different intensities and may occur on only one side of the face. In this paper we focus on recognising six AUs from the region around the eyes, as illustrated in Fig. 1.

Initial representation methods for AU classification were based on measuring the relative position of a large number of landmark points on the face [2]. It has been found, however, that comparable or better results can be obtained by taking a more holistic approach to feature extraction using methods such as Gabor wavelets or principal components analysis (PCA) [3]. In this paper we compare two such methods, namely PCA [4] and local binary pattern (LBP)

Fig. 1. Some example AUs and AU groups from the region around the eyes. AU1 = inner brow raised, AU2 = outer brow raised, AU4 = brows lowered and drawn together, AU5 = upper eyelids raised, AU6 = cheeks raised, AU7 = lower eyelids raised. The images are shown after manual eye location, cropping, scaling and histogram equalisation.

features [5]. The latter is a computationally efficient texture description method that has the benefit that it is relatively insensitive to lighting variations. LBP has been successfully applied to facial expression analysis [6] and here we take as features the individual histogram bins that result when LBP is applied over multiple sub-regions of an image and at multiple sampling radii.

One problem with the holistic approach is that it can lead to the generation of a very large number of features and so some method must be used to select only those features that are relevant to the problem at hand. For PCA a natural choice is to use only those features that account for most of the variance in the set of training images. For the LBP representation, AdaBoost has been used to select the most relevant features [6]. In this paper, however, we adopt the very efficient fast correlation-based filtering (FCBF) [7] algorithm to perform this function. FCBF operates by repeatedly choosing the feature that is most correlated with class, excluding those features already chosen or rejected, and rejecting any features that are more correlated with it than with the class. As a measure of correlation, the information-theoretic concept of symmetric uncertainty is used.

To detect the presence of particular AUs in a face image, one possibility is to train a separate dedicated classifier for each AU. Bartlett et. al. for example [8], have obtained good results by constructing such a set of binary classifiers, where each classifier consists of an AdaBoost ensemble based on selecting the most useful 200 Gabor filters, chosen from a large population of such features. An alternative approach [6] is to make use of the fact that AUs tend to occur in distinct groups and to attempt, in the first instance, to recognise the different AU groups before using this information to infer the presence of individual AUs. This second approach is the one adopted in this paper; it treats the problem of AU recognition as a multiclass problem, requiring a single classifier for its solution. This classifier generates confidence scores for each of the known AU groups and these scores are then summed in different combinations to estimate the likelihood that each of the AUs is present in the input image.

One potential problem with this approach is that, when the number positive indicators for a given AU (i.e. the number of AU groups to which it belongs)

differs from the number of negative indicators (i.e. the number of AU groups to which it does not belong), the overall score can be unbalanced, making it difficult to make a correct classification decision. To overcome this problem we apply Platt scaling [9] to the total scores for each AU. This technique uses a maximum-likelihood algorithm to fit a sigmoidal calibration curve to a 2-class training set. The re-mapped value obtained from a given input score then represents an estimate of the probability that the given point belongs to the positive class.

The method used in this paper to perform the initial AU group classification step is to construct an error-correcting output code (ECOC) ensemble of multi-layer perceptron (MLP) neural networks. The ECOC technique [13] has proved to be a highly successful way of solving a multiclass learning problem by decomposing it into a series of 2-class problems, or dichotomies, and training a separate base classifier to solve each one. These 2-class problems are constructed by repeatedly partitioning the set of target classes into pairs of super-classes so that, given a large enough number of such partitions, each target class can be uniquely represented as the intersection of the super-classes to which it belongs. The classification of a previously unseen pattern is then performed by applying each of the base classifiers so as to make decisions about the super-class membership of the pattern. Redundancy can be introduced into the scheme by using more than the minimum number of base classifiers and this allows errors made by some of the classifiers to be corrected by the ensemble as a whole.

In addition to constructing vanilla ECOC ensembles, we make use of two enhancements to the ECOC algorithm with the aim of improving classification performance. The first of these is to promote diversity among the base classifiers by training each base classifier, not on the full training set, but rather on a bootstrap replicate of the training set [14]. These are obtained from the original training set by repeated sampling with replacement and this results in further training sets which contain, on average, 63% of the patterns in the original set but with some patterns repeated to form a set of the same size. This technique has the further benefit that the out-of-bootstrap samples can also be used for other purposes such as parameter tuning.

The second enhancement to ECOC is to apply weighting to the decoding of base-classifier outputs so that each base classifier is weighted differently for each target class (i.e. AU group). For this purpose we use a method known as class-separability weighting (CSEP) ([15] and section 2) in which base classifiers are weighted according to their ability to distinguish a given class from all other classes.

When considering the sources of error in statistical pattern classifiers it is useful to group them under three headings, namely Bayes error, bias (strictly this is measured as bias^2) and variance. The first of these is due to unavoidable noise but the latter two can be reduced by careful classifier design. There is often a tradeoff between bias and variance [10] so that a high value of one implies a low value of the other. The concepts of bias and variance originated in regression theory and several alternative definitions have been proposed for

extending them to classification problems [11]. Here we adopt the definitions of Kohavi and Wolpert [12] to investigate the bias/variance characteristics of our chosen algorithms. These have the advantage that bias and variance are non-negative and additive. A disadvantage, however, is that no explicit allowance is made for Bayes error and it is, in effect, rolled into the bias term.

Previous investigation [15,16,17] has suggested that the combination of bootstrapping and CSEP weighting improves ECOC accuracy and that, for general problems at least, this is achieved through a reduction in both bias and variance error. In this paper we extend this work, for the specific problem of FACS recognition, in three main ways: firstly we compare two different image feature extraction strategies (namely PCA and LBP plus FCBF), secondly we show that Platt scaling improves AU recognition accuracy and thirdly we perform a bias-variance analysis on the AU group recognition problem.

2 ECOC Weighted Decoding

The ECOC method consists of repeatedly partitioning the full set of N classes Ω into L super-class pairs. The choice of partitions is represented by an $N \times L$ binary *coding matrix* $\mathbf{Z}$. The rows $\mathbf{Z}_i$ are unique *codewords* that are associated with the individual target classes ω_i and the columns $\mathbf{Z}^j$ represent the different super-class partitions. A separate base classifier is trained to solve the 2-class problem represented by each column.

Given an input pattern vector $\mathbf{x}$ whose true class $y(\mathbf{x}) \in \Omega$ is unknown, let the soft output from the jth base classifier be $s_j(\mathbf{x}) \in [0,1]$. The set of outputs from all the classifiers can be assembled into a vector $\mathbf{s}(\mathbf{x}) = [s_1(\mathbf{x}), \ldots, s_L(\mathbf{x})]^{\mathrm{T}} \in [0,1]^L$ called the *output code* for $\mathbf{x}$. In its general form, a *weighted* decoding procedure makes use of an $N \times L$ weights matrix $\mathbf{W}$ that assigns a different weight to each target class and base classifier combination. For each class ω_i we may use the L^1 metric to compute a class score $F_i(\mathbf{x}) \in [0,1]$ as follows:

$$F_i(\mathbf{x}) = 1 - \sum_{\mathrm{j}=1}^{\mathrm{L}} \mathbf{W}_{\mathrm{ij}} \left| \mathrm{s_j}(\mathbf{x}) - \mathbf{Z}_{\mathrm{ij}} \right|, \tag{1}$$

where it is assumed that the rows of $\mathbf{W}$ are normalized so that $\sum_{j=1}^{L} \mathbf{W}_{ij} = 1$ for $i = 1 \ldots N$. Patterns may then be assigned to the target class $y(\mathbf{x}) = \arg\max_{\omega_i} F_i(\mathbf{x})$. If the base classifier outputs $s_j(\mathbf{x})$ in Eqn. 1 are replaced by hardened values $h_j(\mathbf{x})$ then this describes the weighted Hamming decoding procedure.

In the context of this paper Ω is the set of known AU groups and we are also interested in combining the class scores to obtain values that measure the likelihood that AUs are present; this is done by summing the $F_i(\mathbf{x})$ over all ω_i that contain the given AU and dividing by N . That is, the score $G_k \in [0,1]$ for $\mathrm{AU_k}$ is given by:

$$G_k(\mathbf{x}) = \frac{1}{N} \sum_{AU_k \in \omega_i} F_i(\mathbf{x}) \tag{2}$$

The values of $\mathbf{W}$ may be chosen in different ways. For example, if $\mathbf{W}_{ij} = \frac{1}{L}$ for all i, j then the decoding procedure of Eqn. 1 is equivalent to the standard unweighted L^1 or Hamming decoding scheme. In this paper we make use of the CSEP measure [15,17] to obtain weight values that express the ability of each base classifier to distinguish members of a given class from those of any other class. The algorithm for computing CSEP weights is shown in Fig. 2.

Inputs: matrix of training patterns $\mathbf{T} \in \mathbb{R}^{P \times M}$, binary coding matrix $\mathbf{Z} \in \{0,1\}^{N \times L}$, trained ECOC coding function $E : \mathbb{R}^M \mapsto [0,1]^L$.
Outputs: weight matrix $\mathbf{W} \in [0,1]^{N \times L}$ where $\sum_{j=1}^{L} \mathbf{W}_{ij} = 1$, for $i = 1 \ldots N$.
Apply E to each row of $\mathbf{T}$ and round to give prediction matrix $\mathbf{H} \in \{0,1\}^{P \times L}$.
Initialise $\mathbf{W}$ to $\mathbf{0}$.
for $c = 1$ to N
  for $i =$ indices of training patterns belonging to class c
    for $j =$ indices of training patterns *not* belonging to class c
      let d be the true class of the pattern $\mathbf{T}_j$.
      for k $= 1$ to L
        if $\mathbf{H}_{ik} = \mathbf{Z}_{ck}$ and $\mathbf{H}_{\mathrm{jk}} = \mathbf{Z}_{\mathrm{dk}}$, add 1 to $\mathbf{W}_{ck}$
          as the predictions for both patterns $\mathbf{T}_i$ and $\mathbf{T}_j$ are correct.
        if $\mathbf{H}_{ik} \neq \mathbf{Z}_{ck}$ and $\mathbf{H}_{\mathrm{jk}} \neq \mathbf{Z}_{\mathrm{dk}}$, subtract 1 from$\mathbf{W}_{ck}$
          as the predictions for both patterns $\mathbf{T}_i$ and $\mathbf{T}_j$ are incorrect.
      end
    end
  end
end
Reset all negative entries in $\mathbf{W}$ to 0.
Normalize $\mathbf{W}$ so that each row sums to 1.

Fig. 2. Pseudo-code for computing the class-separability weight matrix for ECOC.

3 Experiments

In this section we present the results of performing classification experiments on the Cohn-Kanade face expression database [18]. This dataset contains frontal video clips of posed expression sequences from 97 university students. Each sequence goes from neutral to target display but only the last image has available a ground truth in the form of a manual AU coding. In carrying out these experiments we focused on detecting AUs from the the upper face region as shown in Fig. 1. Neutral images were not used and AU groups with three or fewer examples were ignored. In total this led to 456 images being available and these were distributed across the 12 classes shown in Table 1.

Each 640 x 480 pixel image we converted to greyscale by averaging the RGB components and the eye centres were manually located. A rectangular window

Table 1. Classes of action unit groups used in the experiments.

Class number	1	2	3	4	5	6	7	8	9	10	11	12
AUs present	None	1,2	1,2,5	4	6	1,4	1,4,7	4,7	4,6,7	6,7	1	1,2,4
Number of examples	152	23	62	26	66	20	11	48	22	13	7	6

around the eyes was obtained and then rotated and scaled to 150 x 75 pixels. Histogram equalization was used to standardise the image intensities. LBP features were extracted by computing a uniform (i.e. 59-bin) histogram for each sub-window in a non-overlapping tiling of this window. This was repeated with a range of tile sizes (from 12 x 12 to 150 x 75 pixels) and sampling radii (from 1 to 10 pixels). The histogram bins were concatenated to give 107,000 initial features; these were then reduced to approximately 120 features by FCBF filtering.

ECOC ensembles of size 200 were constructed with single hidden-layer MLP base classifiers trained using the Levenberg-Marquardt algorithm. A range of MLP node numbers (from 2 to 16) and training epochs (from 2 to 1024) was tried; each such combination was repeated 10 times and the results averaged. Each run was based on a different randomly chosen stratified training set with a 90/10 training/test set split. The experiments were performed with and without CSEP weighting and with and without bootstrapping. The ECOC code matrices were randomly generated but in such a way as to have balanced numbers of 1s and 0s in each column. Another source of random variation was the initial MLP network weights. When bootstrapping was applied, each base classifier was trained on a separate bootstrap replicate drawn from the complete training set for that run. The CSEP weight matrix was, in all cases, computed from the full training set.

3.1 Classifier accuracy

Table 2 shows the mean AU classification error rates and area under ROC figures obtained using these methods (including Platt scaling); the best individual AU classification results are shown in Table 3. From Table 2 it can be seen that the LBP feature extraction method gives greater accuracy than PCA. Furthermore, LBP is able to benefit from the application of bootstrapping and CSEP weighting, whereas PCA does not. The LBP method thus exhibits behaviour similar to that found on other data sets [15], in that bootstrapping and CSEP weighting on their own each lead to some improvement and the combination improves the results still further. By contrast, PCA performance is not improved by either technique, whether singly or in combination. The reasons for this anomaly, in terms of a bias/variance decomposition of error, are discussed in section 3.3.

3.2 The effect of Platt scaling

Platt scaling was used to convert the soft scores G_k from Eqn. 2 into approximate measures of the probability that AU_k is present. An example of the kind of

Table 2. Best mean error rates and area under ROC curves for the AU recognition task.

Bootstrapping Applied	CSEP Weighting Applied	Error (%)		Area Under ROC	
		PCA	LBP + FCBF	PCA	LBP + FCBF
No	No	9.5	9.0	92.8	93.7
Yes	No	9.8	8.8	92.8	94.4
No	Yes	9.5	9.0	93.0	94.2
Yes	Yes	9.6	**8.5**	93.0	**94.8**

Table 3. Best error rates and area under ROC curves for individual AU recognition. LBP feature extraction was used, together with bootstrapping and CSEP weighting. MLPs had 16 nodes and 8 training epochs.

AU no.	1	2	4	5	6	7
Error (%)	8.9	5.4	8.7	4.8	11.2	12.3
Area under ROC	94.4	96.2	96.1	97.0	92.1	92.2

calibration curves that result from this algorithm is shown in Fig. 3 and the effect of applying the mapping to the test set is shown in Fig. 4. Note that, before calibration all scores are below 0.5 and hence would be classed as AU not present. After calibration (Fig. 4(b)) most of the test patterns that contain AU2 fall to the right hand side of the 0.5 threshold and hence are correctly classified.

Table 4 shows the effect on mean error rates and area under ROC curve. It can be seen that AU detection error rates are approximately halved by this procedure but that it has no effect on the area under ROC curve values. The reason for this is that the application of any monotonically increasing function to G_k does not affect the shape of the ROC curve, it only affects the threshold values associated with each point on the ROC curve.

3.3 A bias/variance analysis

It is instructive to view the performance of these algorithms from the point of view of a bias/variance decomposition of error. Fig. 5 shows bias and variance curves for AU group recognition when the number of training epochs is varied and other parameter settings are fixed at their respective optimal values. It is notable that, for both types of feature extraction, bias error (which, as noted in section 1, includes an unknown amount of Bayes error) predominates. Bias is, however, somewhat higher for PCA (at around 40%) than for LBP (at around 35%). This indicates that LBP is more successful at capturing subtle variations in face expressions than PCA. The downside to this is that LBP feature extraction is more heavily influenced by chance details of the training set and hence shows higher variance (at around 8%) than PCA (at around 4.5%). It is thus evident that these two feature extraction methods are operating at different points on the bias/variance tradeoff curve.

Table 4. The effect of applying Platt scaling on error rates and area under ROC curves for AU recognition

Scaling Applied	Error (%)		Area Under ROC	
	PCA	LBP + FCBF	PCA	LBP + FCBF
No	17.5	16.6	93.0	94.8
Yes	9.6	8.5	93.0	94.8

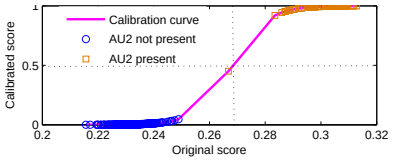


Fig. 3. Calibration curve for AU2 training set (bootstrapping plus CSEP weighting applied).

One notable difference between LBP and PCA is that, when ECOC is augmented with bootstrapping and CSEP weighting, the former method benefits by a reduction in both bias and variance; this is consistent with results found on other datasets [16]. For PCA, by contrast, variance is reduced but this is cancelled by an increase in bias so that PCA does not benefit from these methods. This increase in bias appears to be largely due to the application of bootstrapping.

4 Discussion and Conclusions

In this paper we have shown that good results on the problem of AU classification can be achieved by using a single multi-class classifier to estimate the probabilities of occurrence of each one of a set of AU groups and then combining the values to obtain individual AU probabilities. An ECOC ensemble of MLP neural networks has been shown to perform well on this problem, particularly when enhanced by the application of bootstrapping and CSEP weighting. When combining ECOC outputs it has been found necessary to apply a score-to-probability calibration technique such as Platt scaling to avoid the bias introduced by different AU group membership numbers.

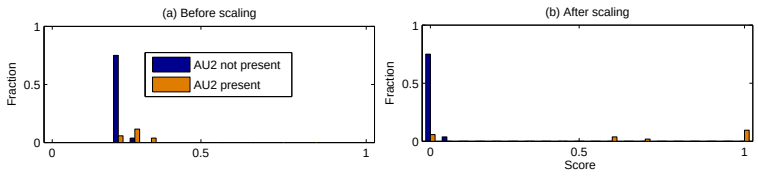


Fig. 4. The effect of Platt scaling on the distribution of test-set scores for AU2.

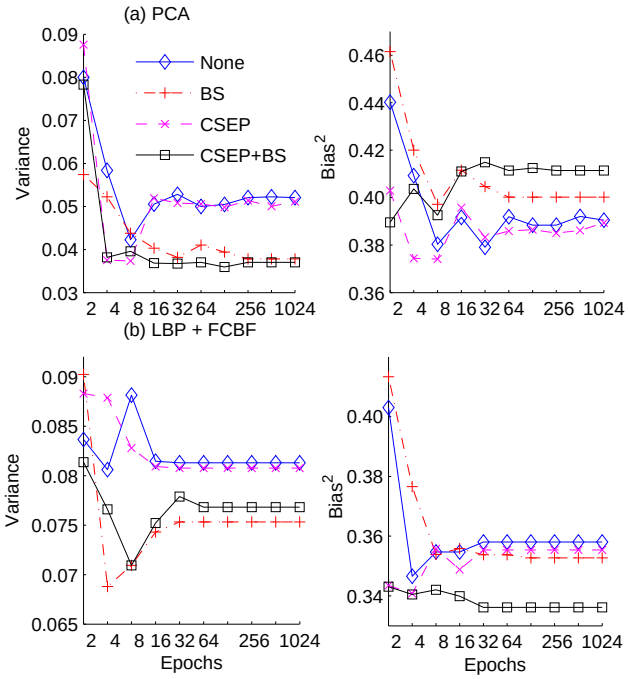


Fig. 5. Bias and variance curves for different feature extraction methods using 16-node base classifiers.

Two methods of feature extraction have been examined, namely PCA as applied directly to the input images, and the use of LBP to extract a wide range of texture features followed by FCBF filtering to reduce their number. The LBP-based method has been found to be more effective. This is particularly true when combined with bootstrapping and CSEP weighting which lead to a reduction in both bias and variance error.

From an efficiency point of view, it is worth noting that both LBP and FCBF (which is only required during training) are fast lightweight techniques. The use of a single classifier, rather than one per AU, also helps to minimise the computational overheads of AU detection.

5 Acknowledgements

This work was supported by EPSRC grant E061664/1.

References

1. P. Ekman and W.V. Friesen, The Facial Action Coding System: A Technique for The Measurement of Facial Movement. San Francisco: Consulting Psychologists Press, 1978.

2. Ti an Y-I, Kanade T, Cohn JF. Recognizing action units for facial expression analysis. Pattern Analysis and Machine Intelligence, IEEE Transactions on Volume: 23(2), pp: 97-115, Feb 2001.
3. Donato G, Bartlett MS, Hager JC, Ekman P, Sejnowski TJ. Classifying facial actions. *IEEE Trans. PAMI* 21(10), pp. 974-989, 1999.
4. Turk M, Pentland A. Eigenfaces for recognition. *J. Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.
5. Ahonen T, Hadid A, Pietikainen M. Face Description with Local Binary Patterns: Application to Face Recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(12), pp. 2037-2041, Dec. 2006.
6. Shan C, Gong S, McOwan PW. Facial expression recognition based on Local Binary Patterns: A comprehensive study. Image and Vision Computing, 27(6), pp. 803-816, 2009.
7. Yu L, Liu H. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proc 12th Int Conf on Machine Learning* (ICML-03), pp. 856-863, 2003.
8. Bartlett MS, Littlewort G, Frank M, Lainscsek C, Fasel I, Movellan J. Fully Automatic Facial Action Recognition in Spontaneous Behavior. Proc 7th Conf. On Automatic Face and Gesture Recognition, pp. 223-238, 2006.
9. Platt J. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pp. 61-74, A.J. Smola, P. Bartlett, B. Scholkopf, D. Schuurmans, eds., MIT Press, 1999.
10. Geman S, Bienenstock E. Neural networks and the bias / variance dilemma. Neural Computation, 4, pp. 1-58, 1992.
11. James G. Variance and Bias for General Loss Functions. *Machine Learning*, 51 (2), 115-135, 2003.
12. Kohavi R, Wolpert D. Bias plus variance decomposition for zero-one loss functions. *Proc. 13th International Conference on Machine Learning*, pp. 275-283, 1996.
13. Dietterich TG, Bakiri G. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research* 2: 263-286, 1995.
14. Efron B, Tibshirani RJ. An Introduction to the Bootstrap. Chapman & Hall, 1993.
15. Smith RS, Windeatt T. Class-Separability Weighting and Bootstrapping in Error Correcting Output Code Ensembles. *Proc. 9th Int. Conf. on Multiple Classifier Systems*, LNCS 5997, pp. 185-194, 2010.
16. Smith RS, Windeatt T. A Bias-Variance Analysis of Bootstrapped Class-Separability Weighting for Error-Correcting Output Code Ensembles. *Proc. 22nd Int. Conf. on Pattern Recognition (ICPR)*, August 2010, accepted.
17. Windeatt T, Smith RS, Dias K. Weighted Decoding ECOC for Facial Action Unit Classification. *18th European Conference on Artificial Intelligence* (ECAI), pp. 26-30, Patras, Greece, July 2008.
18. Kanade T, Cohn JF, Tian Y. Comprehensive Database for facial expression analysis, Proc. 4th Int. Conf. Automatic Face and Gesture Recognition, pp. 46-53, March 2000.

DDAG K-TIPCAC: an ensemble method for protein subcellular localization

A. Rozza, G. Lombardi, M. Re, E. Casiraghi, and G. Valentini

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano,
Via Comelico 39-41, 20135 Milano, Italy
Rozza@dico.unimi.it, WWW home page:
http://security.dico.unimi.it/∼fox721/

Abstract. Protein subcellular location prediction is one of the most difficult multiclass prediction problems in modern computational biology. Many methods have been proposed in the literature to solve this problem, but all the existing approaches are affected by some limitations. In this contribution we propose a novel method for protein subcellular location prediction that performs multiclass classification by combining kernel classifiers through DDAG. Each base classifier, called K-TIPCAC, projects the points on a Fisher subspace estimated on the training data by means of a novel technique. Experimental results clearly indicated that DDAG K-TIPCAC performs equally, if not better, than state-of-the-art ensemble methods for protein subcellular location.

Keywords: Bioinformatics, protein subcellular location prediction, Fisher subspace, ensemble of classifiers.

1 Introduction

Since many different protein molecules are present in one or more subcellular locations, a better understanding of their distribution and function is advisable to understand the complex biological systems that regulate the biological life of each cell. To this aim, the first and fundamental problem to be solved is the subcellular protein localization. Since biochemical experiments aimed at this task are both costly and time-consuming, and new proteins are continuously discovered (increasing the gap between the newly found proteins and the knowledge about their subcellular location), an efficient and effective automatic method for protein subcellular location prediction is required.

This problem can be formulated as a multiclass classification problem as follows. The training dataset, $\boldsymbol{\mathcal{P}}_{Train}$, is composed of N protein vectors, $\boldsymbol{\mathcal{P}}_{Train} = \{\boldsymbol{p}_i\}_{i=1}^{N}$, where each protein sequence can be represented as a vector $\boldsymbol{p} = [R_s^j]$, R_s^j being the amino acid residue whose ordered position in the sequence is $s = 1, \cdots, S$ (S is the protein length, which differs in each protein), while the superscript $j = 1, \ldots, 20$ indicates which native amino acid is present in the s-th position of the sequence. The proteins in $\boldsymbol{\mathcal{P}}_{Train}$ are classified into M subsets $\boldsymbol{\mathcal{S}} = \bigcup_{i=1}^{M} \boldsymbol{\mathcal{S}}_i$, where each subset, $\boldsymbol{\mathcal{S}}_m$ ($m = 1, 2, \ldots, M$), is composed of proteins with the same subcellular component, and the cardinality of $\boldsymbol{\mathcal{S}}$ is

$|\boldsymbol{\mathcal{S}}| = N = N_1 + N_2 + \cdots + N_M$. The classifier's aim is to learn the information provided by $\boldsymbol{\mathcal{P}}_{Train}$ to predict the subcellular location of a query protein $\boldsymbol{p}_q$.

In the past decade many authors have tried to handle this problem, and several classification methods have been proposed [11]. Nevertheless, the problem is still open due to several difficulties that make the task of protein subcellular location prediction very challenging. At first, the protein data are usually encoded with high dimensional vectors, so that the employed classifiers should be designed in order to minimize the computational complexity. Secondly, the number of subcellular locations that should be discriminated is at most 22 (that is $M \leq 22$), and some proteins, called multiplex proteins, might be present in more than one cellular component, or they might move from one location to another. Finally, the protein subcellular distribution is highly unbalanced since some cellular components contain a significantly lower number of protein molecules. To achieve satisfactory results in such (multiclass, high dimensional, and highly unbalanced) classification problem, a dataset of high cardinality is needed. Unfortunately, the training datasets have a limited number of proteins, due to the following reasons: some proteins must be discarded since they contain less than 50 amino acids, or they are annotated as 'fragments'; to avoid homology bias proteins with $\geq 25\%$ sequence identity to any other in the same subcellular organelle must be eliminated; proteins belonging to components with less than 20 proteins are generally excluded, because of lacking statistical significance; several proteins cannot be used as robust data for training a solid predictor since they have not been experimentally annotated yet. Finally, further deletions might be performed by some authors focusing on proteins with a unique subcellular location, or belonging to a specific organism. These difficulties motivate the large number of research works devoted to the task of protein location prediction; these methods can be grouped according to either the data representation, or the employed algorithm.

Representing a protein $\boldsymbol{p}$ with a vector that codes its entire amino acid sequence is unfeasible since this representation produces too long vectors of different dimensionality. A more compact representation is provided by the amino acid composition (`AAC`) descriptor [6], whose elements are the normalized occurrence frequencies of the 20 native amino acids. Since the `AAC` lacks the ability of representing the sequence order effects, several alternative non sequential descriptors have been proposed in the literature. More precisely, these descriptors represent both single and evolutionarily related groups of proteins: `PseAAC`) encodes proteins by taking into account correlations between pairs of aminoacids at different sequence distance w.r.t a given chemico-physical property [7]; the k-peptide encoding vector, which is the normalized occurrence of the k-letter pattern that appears in a window being shifted along the sequence, is another popular representation for single proteins [21]; evolutionarily related groups of proteins can be encoded through the `SeqEvo` representation, based on the normalized occurrence of the changes in the protein sequence for each native amino acid (that is insertions, deletions, substitutions of amino acid residues) that are due to proteins evolution [13]. While the aforementioned protein representation schemes are all strictly based on the protein amino acid sequence, alternative encodings

are possible by considering the availability of large amount of information contained in public databases like the Functional Domain (FunD) [8] and the Gene Ontology (GO) [1]. According to the content of FunD it is possible to code each proteins in the form of a boolean vector indicating the presence/absence of any of the 7785 functional protein domains annotated in the database and a similar encoding scheme can be adopted by considering the annotations stored in the Cellular Component division of the Gene Ontology.

Regarding the employed predictors, they are: the Covariant Discriminant (CD) algorithm [7]; modified versions of the K-Nearest-Neighbor (KNN) technique [15, 21, 29], or its extension, called Optimized Evidence-Theoretic KNN (OET-KNN) [33, 10], Support Vector Machines (SVMs) [14, 22, 18], and the naive Bayes classifier [3]. All the aforementioned methods are depending on critical parameters, defining both the protein representation mode, the dataset dimensionality, and different settings of the learning algorithm. Recently, simple ensemble methods have been proposed: given an engine learning algorithm (e.g. OET-KNN or SVM), these techniques create different predictors by changing the values of their parameters, and produce the final classification result by a simple majority vote algorithm [10, 31, 13].
Although promising results have been obtained, the computational efficiency and the classification performance of all the above mentioned techniques are highly affected both by the high unbalancing of the training set, and by the low cardinality of some classes compared to the high data dimensionality. To overcome such weaknesses, in this paper we propose our ensemble method whose engine algorithm, hereafter referred as Kernel Truncated Isotropic Principal Component Analysis Classifier (K-TIPCAC ,see Section 2), is an evolution of the K-IPCAC and the O-IPCAC algorithms [26, 28], which project the points on the Fisher subspace estimated by a novel technique on the training data (see Section 2). The ensemble method combines the results computed by different K-TIPCAC predictors through a Decision Directed Acyclic Graph (DDAG) technique [24]. Experimental results and the comparison to existing techniques, reported in Section 4, demonstrate the effectiveness of the proposed method.

2 IPCAC, O-IPCAC, and K-TIPCAC

The first version of O-IPCAC, called IPCAC, has been initially proposed in [26]. It is a binary classifier exploiting theoretical results presented in [4] to efficiently estimate the Fisher subspace (Fs). More precisely, in [4] it is demonstrated that, given a set of N clustered points sampled from an isotropic Mixture of Gaussians, Fs corresponds to the span of the class means; as a consequence, when a binary classification problem is considered, Fs is spanned by unit vector $\boldsymbol{f} = \frac{\boldsymbol{\mu}_A - \boldsymbol{\mu}_B}{||\boldsymbol{\mu}_A - \boldsymbol{\mu}_B||}$, being A and B the two classes, and $\boldsymbol{\mu}_{A/B}$ the class means.
IPCAC exploits this result by whitening the training set $\boldsymbol{\mathcal{P}}_{Train}$, computing $\boldsymbol{f}$, and classifying a new point $\boldsymbol{p}$ as follows:

$$\theta((\boldsymbol{W}_D^T \boldsymbol{f}) \cdot \boldsymbol{p} - \gamma) = \theta(\boldsymbol{w} \cdot \boldsymbol{p} - \gamma) \; ; \qquad \gamma = \left\langle \underset{\bar{\gamma} \in \{\boldsymbol{w} \cdot \boldsymbol{p}_i\}_{i=1}^N}{\text{argmax}} \; Score(\bar{\gamma}) \right\rangle \tag{1}$$

where $\theta(x) = A$ if $x \geq 0$, $\theta(x) = B$ if $x < 0$, the matrix $\boldsymbol{W}_D$ represents the whitening transformation estimated on the N training points, $Score(\bar{\gamma})$ computes the number of correctly classified training points when $\bar{\gamma}$ is used as threshold, and $\langle\cdot\rangle$ represents the average operator (we may have multiple $\bar{\gamma}$ corresponding to the maximum of the $Score$ function).
Unfortunately, the high computational complexity of classifiers based on the estimation of `Fs` prevents their application to high dimensional datasets. Moreover, these techniques often fail when the training-set cardinality is equal or lower than the space dimensionality. To address these problems, `O-IPCAC` (Online `IPCAC`) [28] improves `IPCAC`, and reduces the computational complexity, by replacing the first step of data whitening by a 'partial whitening' process; if the points to be classified belong to a D dimensional space, this method whitens the data in the linear subspace $\pi_d = \mathbf{Span}\langle \boldsymbol{v}_1, \cdots, \boldsymbol{v}_d \rangle$, spanned by the first $d \ll D$ principal components, while maintaining unaltered the information related to the orthogonal subspace $(\pi_d)^{\perp} = \mathbf{Span}\langle \boldsymbol{v}_{d+1}, \cdots, \boldsymbol{v}_D \rangle$.
More precisely, the linear transformation $\boldsymbol{W}_D$ representing the partial whitening operator is estimated as follows. The Truncated Singular Value Decomposition [19] is applied to estimate the first $d = min(log_2^2 N, D)$ principal components, obtaining the low-rank factorization $\boldsymbol{P} \simeq \boldsymbol{U}_d \boldsymbol{Q}_d \boldsymbol{V}_d^T$ (where $\boldsymbol{P}$ is the matrix representing the training set $\boldsymbol{\mathcal{P}}_{Train}$ since it contains the training vectors). The d largest singular values on the diagonal of $\boldsymbol{Q}_d$, and the associated left singular vectors, are employed to project on the subspace $\boldsymbol{\mathcal{SP}}_d$, spanned by the columns of $\boldsymbol{U}_d$, and to perform the whitening on the points contained in $\boldsymbol{P}$:

$$\bar{\boldsymbol{P}}_{\boldsymbol{W}_d} = q_d \boldsymbol{Q}_d^{-1} \boldsymbol{P}_{\perp SP_d} = q_d \boldsymbol{Q}_d^{-1} \boldsymbol{U}_d^T \boldsymbol{P} = \boldsymbol{W}_d \boldsymbol{P}$$

where q_d is the smallest singular value of the points projected in $\boldsymbol{\mathcal{SP}}_d$. Note that, to obtain points whose covariance matrix best resembles a multiple of the identity, we have chosen to set the value of the d largest singular values to q_d instead of 1, thus avoiding the gap between the d-th and the $(d+1)$-th singular value. The obtained matrix $\boldsymbol{W}_d$ projects and whitens the points in the linear subspace $\boldsymbol{\mathcal{SP}}_d$; however, dimensionality reduction during the whitening estimation might delete discriminative information, decreasing the classification performance. To avoid this information loss, we add to the partially whitened data the residuals $\boldsymbol{R}$ of the points in $\boldsymbol{P}$ with respect to their projections on $\boldsymbol{\mathcal{SP}}_d$:

$$\begin{aligned} \boldsymbol{R} &= \boldsymbol{P} - \boldsymbol{U}_d \boldsymbol{P}_{\perp \boldsymbol{\mathcal{SP}}_d} = \boldsymbol{P} - \boldsymbol{U}_d \boldsymbol{U}_d^T \boldsymbol{P} \\ \bar{\boldsymbol{P}}_{\boldsymbol{W}_D} &= \boldsymbol{U}_d \bar{\boldsymbol{P}}_{\boldsymbol{W}_d} + \boldsymbol{R} = \left(q_d \boldsymbol{U}_d \boldsymbol{Q}_d^{-1} \boldsymbol{U}_d^T + \boldsymbol{I} - \boldsymbol{U}_d \boldsymbol{U}_d^T \right) \boldsymbol{P} = \boldsymbol{W}_D \boldsymbol{P} \end{aligned} \tag{2}$$

where $\boldsymbol{W}_D \in \Re^{D \times D}$ represents the linear transformation that whitens the data along the first d principal components, while keeping unaltered the information along the remaining ones.
In case of binary classification problems, once the partial whitening step has been performed the two whitened class means, and the vector $\boldsymbol{f}$ representing the estimated `Fs` in the partially whitened space, are computed; this allows the binary predictor to compute the class labels by employing the procedure described in [27].
The described approach increases the performance and guarantees a greater stability during the classification task. We note that `O-IPCAC` has been implemented

to perform both batch and online training. For convenience, in this contribution, we refer to the batch method as TIPCAC (Truncated-whitening IPCAC).

A Kernel version of TIPCAC.

To relax the linear separability constraint imposed by the IPCAC algorithm, it is possible to exploit the kernel trick as in the Kernel Principal Component Analysis [32], thus obtaining the Kernel Isotropic PCA Classifier (KIPCAC, [26]). More precisely, Rozza et al. demonstrate that a given point $\boldsymbol{p}$ can be projected on Fs in the kernel space as follows:

$$proj_{\boldsymbol{F}}(\boldsymbol{p}) = \boldsymbol{Ker}(\boldsymbol{p})^T \left((N_A N_B)^{\frac{1}{2}} \tilde{\boldsymbol{A}} \tilde{\boldsymbol{\Lambda}}^{-1} \tilde{\boldsymbol{A}}^T \boldsymbol{N}_{A|B}^{-1} \right) = \boldsymbol{Ker}(\boldsymbol{p})^T \boldsymbol{w} \tag{3}$$

where N is the cardinality of the training set, $\boldsymbol{Ker}(\boldsymbol{p}) = \{KerFunction(\boldsymbol{p}_i, \boldsymbol{p})\}_{i=1}^{N}$ is the vector of the kernel values computed between the point $\boldsymbol{p}$ and the set of the training points $\boldsymbol{p}_i$, $\tilde{\boldsymbol{\Lambda}}$ are the eigenvalues obtained by the decomposition of the kernel matrix, $\tilde{\boldsymbol{A}}$ are the associated eigenvectors, N_A, N_B are the cardinalities of the two classes, and $\boldsymbol{N}_{A|B}^{-1} = \left(\underbrace{N_A^{-1} \cdots}_{N_A \text{ times}} \underbrace{-N_B^{-1} \cdots}_{N_B \text{ times}} \right)^T$.

In this work we extend this method by exploiting the same concept at the basis of the TIPCAC partial whitening step. More precisely, we select the largest eigenvalues that represent a fixed amount of variance defined a-priori, and we set the remaining part of the spectrum to 1; this process reduces the overfitting problems produced by the smallest part of the spectrum without performing any kind of dimensionality reduction.

3 Experimental setting

3.1 Dataset

We evaluated the proposed method on a publicly available dataset[1] involved in the training of the EukP-loc method described in [12].

This dataset contains 5.618 different proteins, classified into 22 eukaryotic subcellular locations. Among the 5.618 considered proteins, 5.091 belong to one subcellular location, 495 to two locations, 28 to three locations, and 4 to four locations. None of the proteins has $\geq 25\%$ sequence identity to any other in the same subset. The collection of sequences was then evaluated to compute the Pseudo Amino Acid compositions (PseAAC) of each protein using the PseAAC web server [30]. For each protein we produced a 495-elements vector composed by 20 numbers describing the standard amino acid composition, 400 values representing the PseAAC based on the dipeptide representation of the protein and further 75 values representing three groups of 25 PseAACs values obtained by setting the λ parameter to 25 and computing the PseAACs based on three pairs of chemico-physical properties: Hydrophobicity-Hydrophilicity, pK1 (alpha-COOH)-pK2 (NH3) and Mass-pI. In this preliminary investigation we focused on the location prediction of the 5091 proteins with a single experimentally annotated subcellular location. Some characteristics of this dataset are depicted in Table 1.

[1] The protein sequences were downloaded in fasta format from the web site http://www.csbio.sjtu.edu.cn/bioinf/euk-multi/Supp-A.pdf.

It is worth noting that the problem is highly unbalanced, ranging the number of proteins associated to a subcellular location from 13 (hydrogenosome, melanosome and synapse) to 1077 (nucleus).

Table 1. Protein subcellular localization prediction dataset (5091 proteins and 22 locations). This table reports the number of annotated proteins per location; labels are mutually exclusive, thus the problem is multiclass but not multilabel.

Dataset			
acrosome proteins	17	cell wall proteins	47
Golgi proteins	157	spindle pole body proteins	17
hydrogenosome proteins	13	synapse proteins	13
lysosome proteins	59	vacuole proteins	91
melanosome proteins	13	centriole proteins	45
microsome proteins	23	chloroplast proteins	497
mitochondrion proteins	488	cyanelle proteins	85
nucleus proteins	1077	cytoplasm proteins	741
peroxisome proteins	92	cytoskeleton proteins	46
plasma membrane proteins	647	endoplasmic reticulum proteins	275
extracell proteins	609	endosome proteins	39

3.2 Methods

Decision DAG K-TIPCAC: In Section 2 an efficient binary classifier (TIPCAC) and its kernel version (K-TIPCAC) are described, that are based on the projection of the data on the one dimensional Fs estimated in a partially whitened subspace. The ensemble classifier proposed in this paper is a C-class classifier that projects the data on a $C-1$ dimensional Fs estimated in a partially whitened subspace, and then combines many binary K-TIPCACs to obtain the final prediction.
More precisely, the first step of this method evaluates the Fs of the overall C classes by generalizing the approach used by TIPCAC; accordingly to what observed in the previous work [28], this step reduces the training time complexity. To this aim, after the partial whitening of the data, the whitened class means $\{\boldsymbol{\mu}_c\}_{c=1}^{C}$ are computed: $\boldsymbol{\mu}_c = \boldsymbol{W}_D\hat{\boldsymbol{\mu}}_c = q_d\boldsymbol{U}_d\boldsymbol{Q}_d^{-1}\boldsymbol{U}_d^T\hat{\boldsymbol{\mu}}_c + \hat{\boldsymbol{\mu}}_c - \boldsymbol{U}_d\boldsymbol{U}_d^T\hat{\boldsymbol{\mu}}_c$, and the orthonormal basis, Π_{C-1}, composed of $C-1$ vectors spanning the Fs, is computed by orthonormalizing the $C-1$ linearly independent $\boldsymbol{\mu}_c$ vectors through the Gram-Schmidt procedure. The partially whitened training points $\boldsymbol{\mathcal{P}}_{\boldsymbol{W}_D}$ are then projected on the subspace Π_{C-1}, obtaining the set of points $\boldsymbol{\mathcal{P}}_{\Pi_{C-1}} = \{\boldsymbol{F}\boldsymbol{S}^T\boldsymbol{p}_i | \boldsymbol{p}_i \in \boldsymbol{\mathcal{P}}_{\boldsymbol{W}_D}\}$, where $\boldsymbol{F}\boldsymbol{S}$ is the matrix whose columns span Fs. Exploiting the points in $\boldsymbol{\mathcal{P}}_{\Pi_{C-1}}$, $C(C-1)/2$ K-TIPCAC binary classifiers are trained, each discriminating two classes in a *one-against-one* fashion (1-vs-1), and their results are combined by means of the Decision Directed Acyclic Graph (DDAG) approach [24].

Support Vector Machine (SVM): Since SVM is a binary classifier, a problem transformation is required before the application of this method to the considered multiclass prediction problem. The existing approaches to cast a multiclass classification problem to a series of binary classification problems can be roughly divided into two main classes: *one-against-all* and 1-vs-1. We applied the latter,

and thus we trained a committee of 231 probabilistic SVMs [23]. The probabilities produced by each classifier were then reconciled to a multiclass prediction via pairwise coupling [20] and a simple max rule over all the class probability estimates was applied to make a final decision.

Ensemble of nested dichotomies (END): Nested dichotomies [17] is a standard statistical technique applied in polytomous classification problems where logistic regression is applied by fitting binary logistic regression models to the internal nodes composing a tree. In absence of domain knowledge it is difficult to decide, among all the possible trees of nested dichotomies, the one to be adopted. A possible solution [16] is to consider all the hierarchies of nested dichotomies equally likely, and to use an ensemble of these hierarchies for prediction. In our experiments we used the END implementation provided in WEKA and we tuned across nd (number of dichotomies) $\in \{5, 10, 20, 40\}$.

Random Forest (RF): Random Forest [2] has been applied as an effective tool for biomolecular and bioinformatics research. This method grows many classification trees. Instances whose class needs to be predicted are classified using the trees composing the forest. Each tree computes its own prediction, and the forest employs a plurality voting (over all the trees in the forest) to choose the final classification. We tuned the method using a grid search over nt (number of trees of the forest) $\in \{10, 20, 30, 40, 50\}$ and nf (number of features) $\in \{10, 100\}$.

Performance evaluation: All the compared methods were evaluated according to a canonical 10 fold stratified cross-validation scheme. Given that the considered problem is a multiclass prediction problem affected by severe unbalance, accuracy is not suitable for performance evaluation. Performances were thus collected in form of F-score (harmonic mean of Precision and Recall). All the experiments, apart those involving the DDAG K-TIPCAC, which is implemented in MATLAB, were performed using the WEKA machine learning library [34].

4 Results

The performances achieved by the evaluated approaches averaged across all the classes are reported in Table 2 (top). The table shows, for each method, the best combination of parameters, Precision, Recall and F-measure. The F-scores obtained by the evaluated methods for each subcellular location averaged across the 10 stratified cross validation folds are reported in Table 2 (bottom). In order to investigate if the differences between the collected per class performances are statistically significant we performed a Wilcoxon signed ranks sum (U) test. Results are reported in Table 3 (direction of the comparison is row-vs-column).

Considering the performances averaged across all the classes achieved by the compared ensemble methods (see Table 2 (top)) the best performing approach is DDAG K-TIPCAC (weighted F-score 0.390) immediately followed by the 1-vs-1 ensemble of SVMs (weighted F-score 0.368). A closer look to this table highlights that, while all the evaluated approaches produced comparable Recall scores, on average this comes at the cost of a reduced precision, the only exception being represented by the DDAG K-TIPCAC ensemble. We note that input space

Table 2. Estimated performances (top table) and per class performances (bottom table) obtained by 10 fold stratified cross validation.

Method	Parameters	Precision	Recall	F-score
DDAG_K-TIPCAC	kernel=RBF, $\sigma = 8$, $var = 0.955$	0.383	0.408	**0.390**
Multiclass SVM	$C = 10.0$ $G = 0.01$	0.369	0.409	0.368
END	$nd = 40$	0.351	0.393	0.355
RF	$nt = 50$ $nf = 100$	0.349	0.391	0.340

END	MCSVM	RF	DDAG_K-TIPCAC	proteins	location
0.211	0.000	0.300	0.560	17	acrosome proteins
0.046	0.000	0.024	0.030	157	Golgi proteins
0.375	0.375	0.375	0.316	13	hydrogenosome proteins
0.000	0.000	0.033	0.213	59	lysosome proteins
0.632	0.000	0.556	0.522	13	melanosome proteins
0.000	0.000	0.000	0.114	23	**microsome proteins**
0.295	0.312	0.241	0.355	488	mitochondrion proteins
0.529	0.535	0.523	0.533	1077	nucleus proteins
0.000	0.000	0.000	0.047	92	**peroxisome proteins**
0.484	0.522	0.489	0.470	647	plasma membrane proteins
0.493	0.482	0.494	0.479	609	extracell proteins
0.175	0.218	0.157	0.267	47	cell wall proteins
0.000	0.000	0.000	0.306	17	**spindle pole body proteins**
0.700	0.700	0.700	0.383	13	synapse proteins
0.000	0.043	0.000	0.071	91	vacuole proteins
0.000	0.000	0.000	0.125	45	**centriole proteins**
0.424	0.504	0.459	0.518	497	chloroplast proteins
0.056	0.189	0.022	0.255	85	cyanelle proteins
0.247	0.235	0.211	0.290	741	cytoplasm proteins
0.000	0.000	0.000	0.059	46	cytoskeleton proteins
0.143	0.159	0.027	0.236	275	endoplasmic reticulum proteins
0.000	0.000	0.000	0.067	39	**endosome proteins**

Table 3. Statistical comparison of per class performances through Wilcoxon test (alternative hypothesis: "greater", direction of comparison: rows versus columns).

	END	MCSVM	RF	DDAG_K-TIPCAC
END	–	0.6876	0.1317	0.9970
MCSVM	0.3375	–	0.1813	0.9950
RF	0.8826	0.8348	–	0.9874
DDAG_K-TIPCAC	$2.689E^{-05}$	$3.073E^{-05}$	$4.449E^{-05}$	–

reduction is present in our approach and also in other types of ensemble evaluated in this experiment, as in the case of Random Forests. Nevertheless, the space reduction computed by RF might be affected by a more relevant information loss, since the input space dimensionality is reduced by means of a random selection of subsets of features of a priori defined size. We can hypothesize that the data transformation applied by our approach is able to produce a more informative representation of the data than feature selection, thus leading to better performances also in highly unbalanced multiclass classification problems as the one involved in our experiments.

This interpretation is supported by the collected per class performances (see Table 2 (bottom)). As we can see, despite the multiclass SVM ensemble (MCSVM)

ranks second in terms of overall F-score (after a weighted averaging of the per class F-scores), its performances are often worse that those obtained by DDAG K-TIPCAC. The hypothesis that the performances, on a per class basis, of DDAG K-TIPCAC are better than those produced by all the other evaluated methods is also supported by the Wilcoxon signed ranks sum test (see Table 3).

5 Conclusions

In this contribution we evaluated the performances of an ensemble of K-TIPCAC classifiers in proteins subcellular location prediction. We demonstrated that the multiclass version of K-TIPCAC is competitive with state-of-the-art methods in one of the most difficult unbalanced multiclass classification problems in bioinformatics. It is worth noting that the ability of the proposed approach to effectively control the precision-recall trade-off also in the prediction of small classes is of paramount importance in real applications, when we need to reduce the costs associated with the biological validation of new protein locations discovered through in silico methods.
Considering that F-score accounts both for precision and recall and that most of the compared methods failed completely to predict the membership of proteins to particularly difficult subcellular locations (reported in bold-face in Table 2), we conclude that DDAG K-TIPCAC is a promising line of research in this application domain and we plan both to extend the proposed approach, and to provide a deeper characterization of its performances in further investigations.

Acknowledgement

The authors would like to thank Professor Paola Campadelli for her invaluable support.

References

1. The Gene Ontology Consortium: Gene Ontology: tool for the unification of biology. Nature Genet., 25:25-29, (2000)
2. Breiman, L.: Random Forests. Machine Learning 45(1), (2001)
3. Briesemeister, S., Rahnenfuhrer, J., Kohlbacher, O.: Going from where to why - interpretable prediction of protein subcellular localization. Bioinformatics (2010)
4. Brubaker, S. C. and Vempala, S.: Isotropic PCA and Affine-Invariant Clustering. Proceedings of IEEE Symposium on Foundations of Computer Science (2008)
5. Cai, Y.D., Chou, K.C.: Nearest Neighbour algorithm for predicting protein subcellular location by combining functional domain composition and pseudo-amino acid composition. Biochemical and Biophysical Research Communications (2003)
6. Chou, K.C.: A novel approach to predicting protein structural classes in a (20-1)-D amino acid composition space. Proteins: Structure, Function, and Genetics (1995)
7. Chou, K.C.: Prediction of protein cellular attributes using pseudo amino acid composition. Proteins: Structure, Function, and Genetics (2001)
8. Chou, K.C., Cai, Y.D.: Using functional domani composition and support vector machines for prediction of protein subcellular location. J. Biol. Chem.(2002)
9. Chou, K.C., Cai, Y.D.: Prediction of protein subcellular locations by GO-FunD-PseAA predictor. Biochemical and Biophysical Research Communications (2004)

10. Chou, K.C., Shen, H.B.: Predicting eukaryotic protein subcellular locations by fusing optimized evidence-theoretic K-nearest neighbor classifiers. J. Prot. Res.(2006)
11. Chou, K.C., Shen, H.B.: Recent progress in protein subcellular location prediction. Analitical Biochemistry (2007)
12. Chou, K., Shen, H.: Cell-Ploc: a package of web servers for predicting subcellular localization of proteins in various organisms. Nature protocol (2008)
13. Chou, K., Shen, H.: A new method for predicting the subcellular localization of eukariotic proteins with both single and multiple sites: Euk-mPLoc. Plos One (2010)
14. Cortes, C., Vapnik, V.: Support Vector Networks. Machine learning (1995)
15. Cover, T.M, Hart, P.E.: Nearest neighbour pattern classification. IEEE Transactions on Information Theory (1967)
16. Frank, E., Kramer, S.: Ensembles of nested dichotomies for multi-class problems. Proceedings of the 21st ICML, Banff, Canada, 2004
17. Fox, J.: Applied Regression Analysis, linear models, and related methods. Sage
18. Garg, A., Bhasin, M., Raghava, G.P.: Support vector machine-based method for subcellular localization of human proteins using amino acid compositions, their order, and similarity search. Journal of Biological Chemistry (2005)
19. Hansen, P. C.: The truncated SVD as a method for regularization. Technical Report, Standford University, CA, USA, (1986)
20. Hastie, T., Tibshirani, R.: Classification by pairwise coupling. Proceedings of Adv. in Neural Information Processing Systems (1998)
21. Huang, Y., Li, Y.: Prediction of protein subcellular locations using fuzzy K-NN method. Bioinformatics (2004)
22. Lei, Z., Dai, Y.: An SVM-based system for predicting protein subnuclear localizations. BMC Bioinformatics (2005)
23. Platt, J.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. in Large Margin Classifiers. MIT press (1999)
24. Platt, C., Cristianini, N., Shawe-taylor, J.: Large Margin DAGs for Multiclass Classification. Proceedings of NIPS (2000)
25. Qi, Y., Bar, Z., Klein, J.: Evaluation of different biological data and computational classification methods for use in protein interaction prediction. Fun. Bioinf. (2006)
26. Rozza, A., Lombardi, G., Casiraghi, E.: Novel IPCA-Based Classifiers and Their Application to Spam Filtering. Proceedings of ISDA. IEEE C. S. (2009)
27. Rozza, A., Lombardi, G., Casiraghi, E.: PIPCAC: a Novel Binary Classifier Assuming Mixtures of Gaussian Functions. Proceedings of AIA (2010)
28. Rozza, A., Lombardi, G., Rosa, M., Casiraghi, E.: O-IPCAC and its Application to EEG Classification. WAPA, JMLR (to appear)
29. Shen, H.B., Chou, K.C.: Virus-PLoc: a fusion classifier for predicting the subcellular localization of viral proteins within host and virus-infected cells. Biopol. (2006)
30. Shen, H., Chou, K.: PseAAC: a flexible web server for generating various kinds of protein pseudo amino acid composition. Analytical Biochemistry (2008)
31. Shen, H.B., Chou, K.C.: Hum-mPLoc: an ensemble classifier for large-scale human protein subcellular location prediction by incorporating samples with multiple sites. Biochemical and biophysical research communications (2007)
32. Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computing (1998)
33. Zouhal, L.M., Denoeux, T.: An evidence theoretic K-NN rule with parameter optimization. IEEE Transactions on System, Man, and Cybernetics (1999)
34. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations (2009)

Random Oracles for Regression Ensembles*

Carlos Pardo, Juan J. Rodríguez, José F. Díez-Pastor, and César García-Osorio

University of Burgos, Spain
{cpardo,jjrodriguez,cgosorio}@ubu.es,jdp0014@alu.ubu.es

Abstract. This paper considers the use of Random Oracles in Ensembles for regression tasks. A Random Oracle model (Kuncheva and Rodríguez, 2007) consists of a pair of models and a fixed, randomly created oracle that selects between them. They can be used as the base model for any ensemble method. Previously, they have been used for classification. The use of Random Oracles for regression is studied using 61 data sets, regression trees as base models and several ensemble methods: Bagging, Random Subspaces, AdaBoost.R2 and Iterated Bagging. For all the considered methods and variants, ensembles with Random Oracles are better than the corresponding version without the Oracles.

1 Introduction

Ensembles [10] are combinations of models. In many situations, an ensemble gives better results than any of its members. Although they have been studied mainly for classification, there are also ensemble methods for regression.

The models to be combined have to be different, otherwise the ensemble is unnecessary. One way to have different models is to construct them with different methods. Nevertheless, there are ensemble methods that combine models obtained from the same method. Most of these ensemble methods change the dataset in some way.

In Bagging [1] each member is trained with a sample of the training data. Normally, the size of the sample is the same than the size of the original training data, but the sample is *with replacement*. Hence, some training examples will appear several times in the sample while others will not appear. The prediction of the ensemble is the average of its members predictions.

In Random Subspaces [9] each member is trained with all the training examples, but with a subset of the attributes. The dimension of the subspaces is a parameter of the method. The prediction of the ensemble is also the average of the predictions.

Bagging and Random Subspaces can be used for classification and for regression. AdaBoost [8] initially was a method for classification, but there are some variants for regression, such as AdaBoost.R2 [5]. In these methods, each training example has a weight. Initially, all the examples have the same weight. The construction of the ensemble members must take into account the examples weights. After an ensemble member is constructed, the examples weights are adjusted. The idea is to give more weight to the examples with greater errors in the previous iterations. Hence, in the construction of

* This work was supported by the Project TIN2008-03151 of the Spanish Ministry of Education and Science.

the next member, these examples will be more important. The ensemble members also have weights, they depend on their error. In AdaBoost.R2, the predicted value of the ensemble is a weighted median. In [18], this method was the one with the best results among several ensemble methods for regression.

Iterated Bagging [2] is a method for regression based on Bagging. It combines several Bagging ensembles. The first Bagging ensemble is constructed as usual. Based on the predictions of the previous Bagging ensemble, the values of the predicted variable are altered. The next Bagging ensemble is trained with these altered values. These values are the *residuals*: the difference between the real and the predicted values. Nevertheless, these predictions are not obtained using all the members in the Bagging ensemble. The error of the predictions for a training example would be too optimistic, the majority of the ensemble methods have been trained with that example. These predictions are obtained using the *out-of-bag* estimation: the prediction for an example is obtained using only those ensemble members that were not trained with that example. The prediction of an Iterated Bagging ensemble is the sum of the predictions of its Bagging ensembles. According to [15], Iterated Bagging is in general the most effective method.

Random Oracles [12] are mini-ensembles formed by two models, they can be used as base models for other ensemble methods. The objective of using random oracles is to have more diversity among the base classifiers that form an ensemble. This additional diversity can improve the accuracy of the ensembles.

The rest of the paper is organised as follows. Next section explains Random Oracles. Section 3 describes the experimental setting. The results are presented and discussed in Section 4. In Section 5, diversity-error diagrams are used to analyze the behavior of the base classifiers. Finally, Section 6 presents some concluding remarks.

2 Random Oracles

A Random Oracle classifier [12] is a mini-ensemble formed by a pair of classifiers and a Random Oracle that chooses between them. It can be thought of as a random discriminant function which splits the data into two subsets with no regard of any class labels or cluster structure. A Random Oracle model can be used as the base model of any ensemble method. Given a base method, the training of a Random Oracle model consists of:

- Select randomly the Random Oracle.
- Split the training data in two subsets using the Random Oracle.
- For each subset of the training data, build a model. The Random Oracle model is formed by the pair of models and the oracle itself.

The prediction of a test instance is done in the following way:

- Use the Random Oracle to select one of the two models.
- Return the prediction given by the selected model.

If the computational complexity of the oracle is low, both in training and prediction, the computational complexity of a Random Oracle model is very similar to the complexity of the base method. In the prediction phase, only one of the two models is used.

In the training phase, two models are built. Nevertheless, they are trained with a disjoint partition of the training examples and the training time of any method depends, at least linearly, on the number of training examples.

Different types of Oracles can be considered. In this work, the Linear Random Oracle is used. This oracle divides the space into two subspaces using a hyperplane. To build the oracle, two different training objects are selected at random (these can be from the same class). The points that are at the same distance from the two training objects define the hyperplane.

The distances are calculated according to the Euclidean distance, numerical attributes are scaled within [0,1], for nominal attributes we consider that the distance is 0 or 1 depending if the two valued are different or equal.

3 Experiments

The experiments were conducted using 5×2 fold cross validation [4]. The performance of the different methods over different datasets was measured using *root mean squared error* (RMSE). The base models were Regression Trees. They were used pruned (*P*) or unpruned (*U*). The method used for pruning was *Reduced Error Pruning* (REP) [6]. Ensemble size was 100.

Several ensemble methods were considered:

- Randomization. When the base method has a random element, different models can be obtained from the same training data. Randomization is an ensemble of such randomizable models, which prediction is the average of the members predictions. In the case of of regression trees, when pruning is used, the training data is partitioned randomly: one subset for building the tree and another for pruning. For unpruned trees, there is not a random element.
- Bagging [1].
- Random Subspaces [9]. For the dimension of the subspaces, two values were considered: 50% and 75% of the number of attributes.
- AdaBoost.R2 [5]. This method can be used with different loss functions. Three are proposed in [5] and used in this work: linear, square and exponential. The suffixes "-Li", "-Sq" and "-Ex" are used to denote the used function. Moreover, methods based on AdaBoost can be used in two ways [7]. In the reweighting version, the base model is trained with all the training data, it must take into account the weight distribution. In the resampling version, the base model is trained with a sample from the training data. This sample is constructed taken into account the weights. These versions are denoted with "-W" and "-S".
- Iterated Bagging [2]. Two configurations were considered 10×10 (Bagging is iterated 10 times, the ensemble size of each Bagging is 10) and 5×20 (Bagging is iterated 5 times, the ensemble size of each Bagging is 20). In both cases, the maximum ensemble size is 100.

For all the configurations of these methods, two versions were considered: combined or not with Random Oracles.

Moreover, other methods were included in the study, as a baseline for the comparisons:

- A single Regression Tree, with or without pruning.
- Linear regression. Two versions were considered: using all the features and using only the selected features with the method described in [16].
- Nearest neighbors. There are two versions, in the first one the number of neighbors is 1. In the other, the number of neighbors is selected using "leave one out".

Weka [17] was used for the experiments. It includes the base method (Multilayer Perceptron), Bagging and Random Subspaces. The rest of the methods (i.e., Iterated Bagging and AdaBoost.R2), were implemented in this library.

Table 1 shows the characteristics of the 61 considered datasets. They are available in the format used by Weka[1]. 30 of them were collected by Luis Torgo[2].

4 Results

In order to compare all the configurations considered, average ranks [3] were used. For each dataset, the methods are sorted according to their performance. The best method has rank 1, the second rank 2 and so on. If there are ties, these methods have the same rank, the average value. For each method, its average rank is obtained as the average value over all the considered datasets. According to [3], "*average ranks by themselves provide a fair comparison of the algorithms*".

Table 2 shows the methods sorted according to their average ranks. The prefix "$\mathcal{O}$-" denotes methods that use Random Oracles. The 12 top positions are for methods that use Random Oracles.

For a method that uses Random Oracles, the benefit is defined as the difference between the average ranks of the corresponding method without Random Oracles and the method with Random Oracles. In Table 2, all the benefits are positive.

When comparing two methods, the number of datasets where one method has better, equal, or worse results than the other is calculated. According to [3], using a sign test, one method is significantly better than other, with a confidence level of 0.05, if the number of wins plus half the ties is at least $N/2+1.96\sqrt{N}/2$. For $N = 61$ datasets, this number is 39. Table 2 shows, in the columns denoted as W/T/L, the paired comparisons of methods with Random Oracles and the corresponding methods without Random Oracles. The symbol "•"denotes significant differences.

The number of wins, ties and losses and the average ranks are calculated using a direct comparison of the results for the different methods: less, equal or greater. Nevertheless, they do not take into account the size of the differences. For this purpose, we use the *quantitative scoring* [14, 18]. Given the results for two methods i and j in one dataset, this score is defined as

$$S_{i,j} = \frac{RMSE_j - RMSE_i}{\max(RMSE_i, RMSE_j)}$$

Where $RMSE_i$ is the root mean squared error for the method i. Unless both methods have zero error, this measure will be between -1 and 1, although it can be expressed as a percentage. The sign indicates which method is better.

[1] `http://www.cs.waikato.ac.nz/ml/weka/index_datasets.html`
[2] `http://www.liaad.up.pt/~ltorgo/Regression/DataSets.html`

Table 1. Datasets used in the experiments.

Dataset	Examples	Numeric	Nominal
2d-planes	40768	10	0
abalone	4177	7	1
ailerons	13750	40	0
auto93	93	16	6
auto-horse	205	17	8
auto-mpg	398	4	3
auto-price	159	15	0
bank-32nh	8192	32	0
bank-8FM	8192	8	0
baskball	96	4	0
bodyfat	252	14	0
bolts	40	7	0
breast-tumor	286	1	8
cal-housing	20640	8	0
cholesterol	303	6	7
cleveland	303	6	7
cloud	108	4	2
cpu-act	8192	21	0
cpu	209	6	1
cpu-small	8192	12	0
delta-ailerons	7129	5	0
delta-elevators	9517	6	0
detroit	13	13	0
diabetes-numeric	43	2	0
echo-months	130	6	3
elevators	16599	18	0
elusage	55	1	1
fishcatch	158	5	2
friedman	40768	10	0
fruitfly	125	2	2
gascons	27	4	0

Dataset	Examples	Numeric	Nominal
house-16H	22784	16	0
house-8L	22784	8	0
housing	506	12	1
hungarian	294	6	7
kin8nm	8192	8	0
longley	16	6	0
lowbwt	189	2	7
machine-cpu	209	6	0
mbagrade	61	1	1
meta	528	19	2
mv	40768	7	3
pbc	418	10	8
pharynx	195	1	10
pole	15000	48	0
pollution	60	15	0
puma32H	8192	32	0
puma8NH	8192	8	0
pw-linear	200	10	0
pyrimidines	74	27	0
quake	2178	3	0
schlvote	38	4	1
sensory	576	0	11
servo	167	0	4
sleep	62	7	0
stock	950	9	0
strike	625	5	1
triazines	186	60	0
veteran	137	3	4
vineyard	52	3	0
wisconsin	194	32	0

Table 2. Average ranks.

Average	Method	Benefit	W/T/ L
13.76	$\mathcal{O}$-AdaBoostR2-S-Ex (P)	8.74	•51/0/10
15.02	$\mathcal{O}$-Bagging (U)	9.01	•54/0/ 7
15.91	$\mathcal{O}$-Iterated Bagging 5x20 (P)	5.66	•46/1/14
16.86	$\mathcal{O}$-AdaBoostR2-S-Li (P)	8.73	•49/1/11
18.30	$\mathcal{O}$-Iterated Bagging 5x20 (U)	8.31	•52/1/ 8
18.53	$\mathcal{O}$-AdaBoostR2-S-Sq (P)	6.70	•47/2/12
19.35	$\mathcal{O}$-AdaBoostR2-S-Li (U)	10.25	•56/0/ 5
19.48	$\mathcal{O}$-AdaBoostR2-S-Ex (U)	11.06	•54/0/ 7
19.60	$\mathcal{O}$-AdaBoostR2-W-Sq (U)	10.21	•55/0/ 6
21.03	$\mathcal{O}$-AdaBoostR2-W-Ex (U)	14.98	•60/0/ 1
21.30	$\mathcal{O}$-Random Subspaces 75% (P)	13.70	•60/0/ 1
21.57	$\mathcal{O}$-AdaBoostR2-S-Sq (U)	6.70	•48/1/12
21.57	Iterated Bagging 5x20 (P)		
21.95	$\mathcal{O}$-AdaBoostR2-W-Li (U)	13.07	•56/0/ 5
21.98	$\mathcal{O}$-Bagging (P)	4.15	•43/1/17
22.50	AdaBoostR2-S-Ex (P)		
22.54	$\mathcal{O}$-Randomization (U)	24.76	•61/0/ 0
24.03	Bagging (U)		
24.07	$\mathcal{O}$-Iterated Bagging 10x10 (P)	2.96	•38/2/21
24.32	$\mathcal{O}$-Randomization (P)	6.57	•51/1/ 9
25.24	AdaBoostR2-Sq-S (P)		
25.43	$\mathcal{O}$-AdaBoostR2-W-Ex (P)	5.29	•44/0/17
25.59	AdaBoostR2-S-Li (P)		
26.12	Bagging (P)		
26.61	Iterated Bagging 5x20 (U)		
26.76	$\mathcal{O}$-Iterated Bagging 10x10 (U)	5.79	•47/1/13
26.98	$\mathcal{O}$-Random Subspaces 50% (U)	4.56	•47/2/12
27.02	Iterated Bagging 10x10 (P)		
28.26	AdaBoostR2-Sq-S (U)		
28.69	$\mathcal{O}$-AdaBoostR2-W-Li (P)	3.04	•39/1/21
28.89	$\mathcal{O}$-Random Subspaces 75% (P)	3.32	35/2/24
29.49	$\mathcal{O}$-AdaBoostR2-W-Sq (P)	3.18	38/0/23
29.61	AdaBoostR2-S-Li (U)		
29.81	AdaBoostR2-W-Sq (U)		
30.53	AdaBoostR2-S-Ex (U)		
30.72	AdaBoostR2-W-Ex (P)		
30.89	Randomization (P)		
31.54	Random Subspaces 50% (U)		
31.73	AdaBoostR2-W-Li (P)		
31.79	Linear Regression (all)		
31.79	K-NN (square)		
31.90	Linear Regression (selection)		
32.21	Random Subspaces 75% (P)		
32.55	Iterated Bagging 10x10 (U)		
32.67	AdaBoostR2-W-Sq (P)		
33.53	K-NN (absolute)		
34.99	Random Subspaces 75% (U)		
35.02	AdaBoostR2-W-Li (U)		
35.50	$\mathcal{O}$-Random Subspaces 50% (P)	0.80	30/0/31
36.02	AdaBoostR2-W-Ex (U)		
36.30	Random Subspaces 50% (P)		
43.43	Tree (P)		
46.39	1-NN		
47.30	Tree (U)		

Fig. 1 shows these scores (as percentages) for the considered methods, comparing the versions with and without Random Oracles. The score is calculated for each dataset and the datasets are sorted according to its score. The number of values above and below zero corresponds with the number of wins and losses in Table 2.

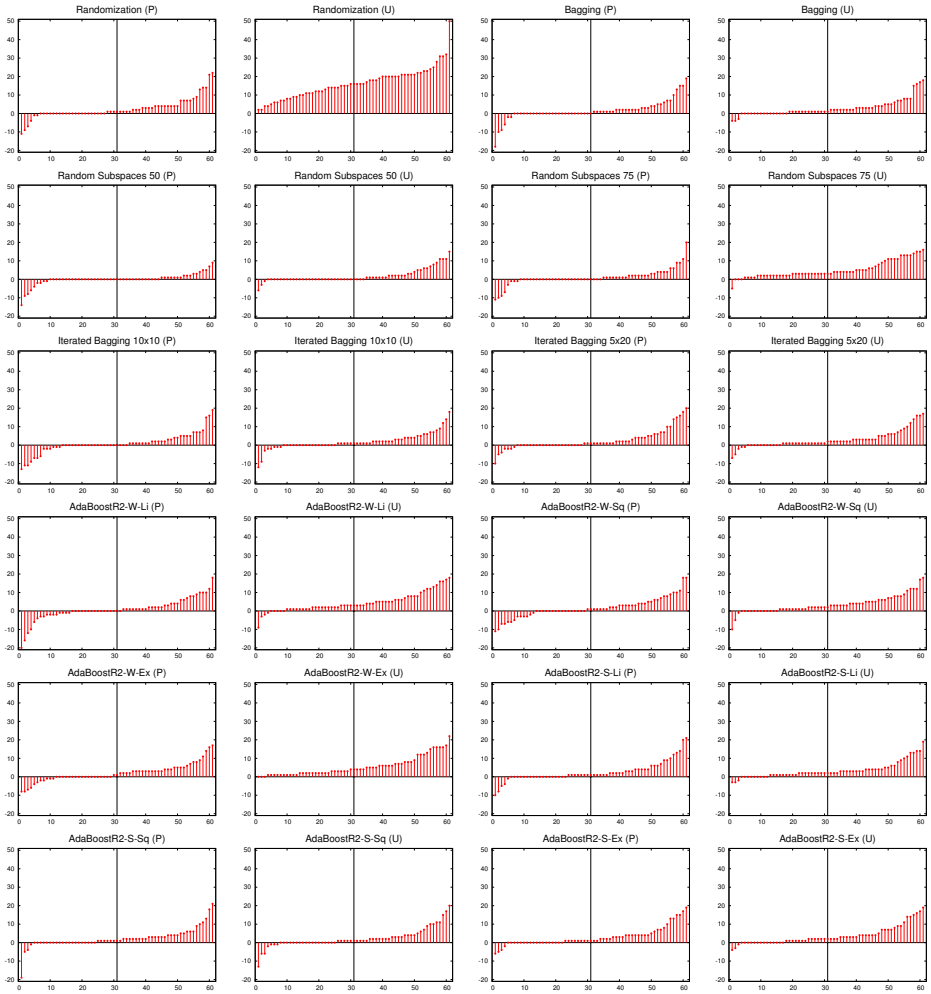


Fig. 1. Comparison scores.

When comparing two methods with these graphs, it is desired to have more positive values than negative, but it is also desirable that the absolute values were greater for positive scores than for negative scores. In this case there are more positive values, and the greater absolute scores are also for positive values.

Results of ensemble methods depend on the ensemble size, the number of combined models. Figures. 2 and 3 show, for the different data sets, the error as a function of the number iterations (from 1 to 100). The graphs show the error for AdaBoostR2-S-Ex (*P*) with and without Oracles. In general, the results are better with the Oracles. For some data sets (e.g., strike), the error increases with the number of iterations. This indicated that AdaBoostR2 is not always robust with respect to the ensemble size.

Fig. 4 shows a comparative of the versions with and without Oracles, as a function of the ensemble size. For Bagging (*U*) and AdaBoostR2-S-Ex (*P*), it shows the proportion of data sets where the version with Oracles is better than the version without Oracles. In the case of ties, they are considered as half a victory. If the ensemble size is very small, five or less, the version without Oracles can be better, but otherwise the advantage is for the version with Oracles.

5 Diversity-error diagrams

Successful ensembles are formed by models with low errors, but that are diverse. These two objectives are contradictory, because if the errors of two models are small, they cannot be very different. Several diversity measures had been proposed in order to analyze the behaviour of ensemble methods [11].

One of the techniques used is diversity-error diagrams [13]. They are scatter plots, there is a point for each pair of models. The horizontal axis represents the diversity between the two models, for classification, usually κ (kappa) is used. The vertical axis represents the average error of the two models.

In regression, several error measures can be considered, in this work RMSE was used:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(a_i - p_i)^2}{n}}$$

Where a_i are the actual values and p_i are the predicted values.

For measuring the diversity, the RMSE of one of the models with respect to the other was used:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(q_i - p_i)^2}{n}}$$

Where p_i and q_i are the predictions of the two models. Note that with this measure, bigger values indicate more diversity, while for kappa, bigger values indicated less diversity.

Fig. 5 shows these diagrams for AdaBoostR2-S-Ex (*P*) and Bagging (*U*) with and without oracles. In general, when using oracles the base classifiers are more diverse.

6 Conclusions

The performance of Random Oracles for regression ensembles have been studied, using 61 data sets and regression trees as base models. They have been combined with

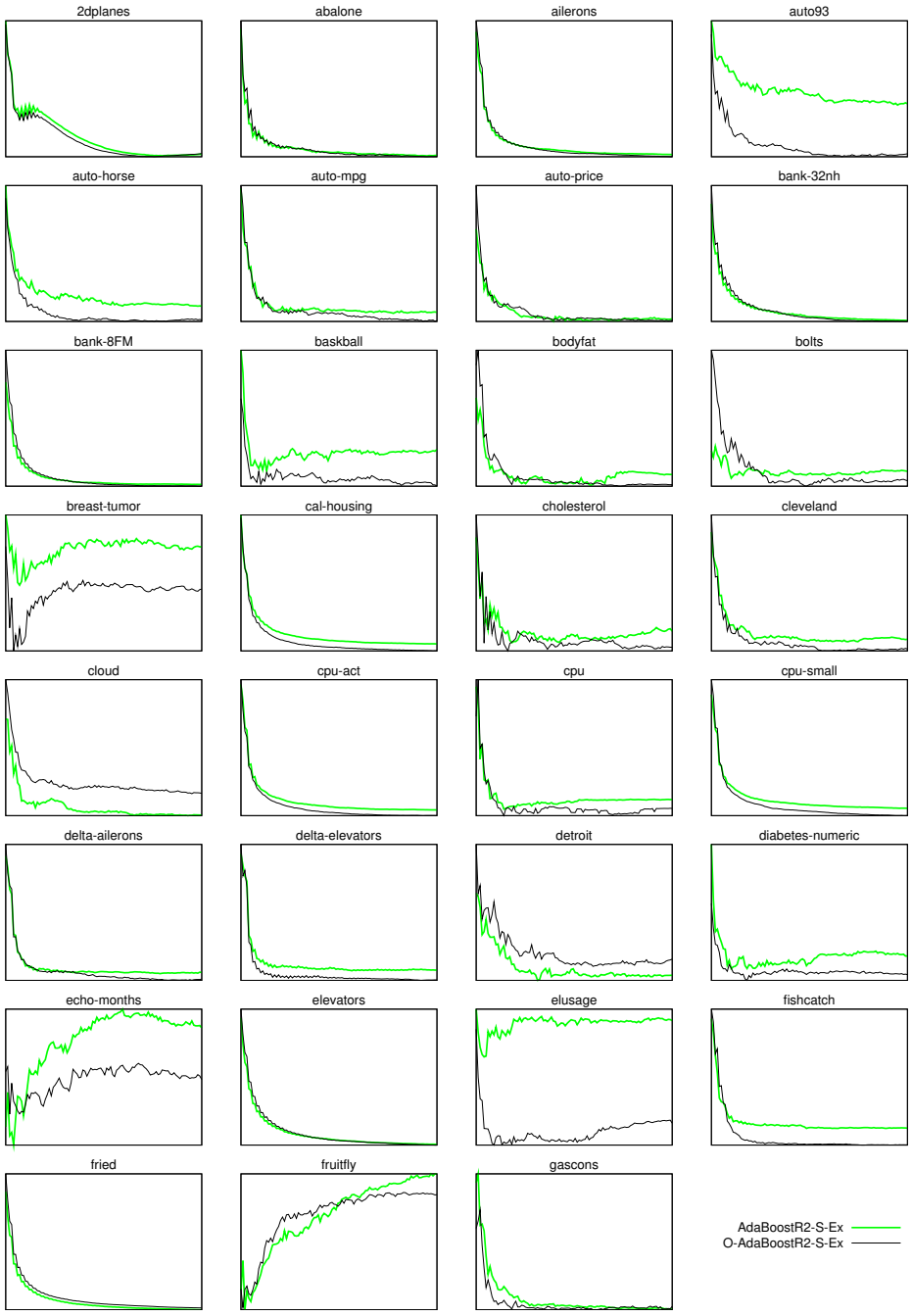


Fig. 2. Error as a function of the number of iterations for the different data sets, first part.

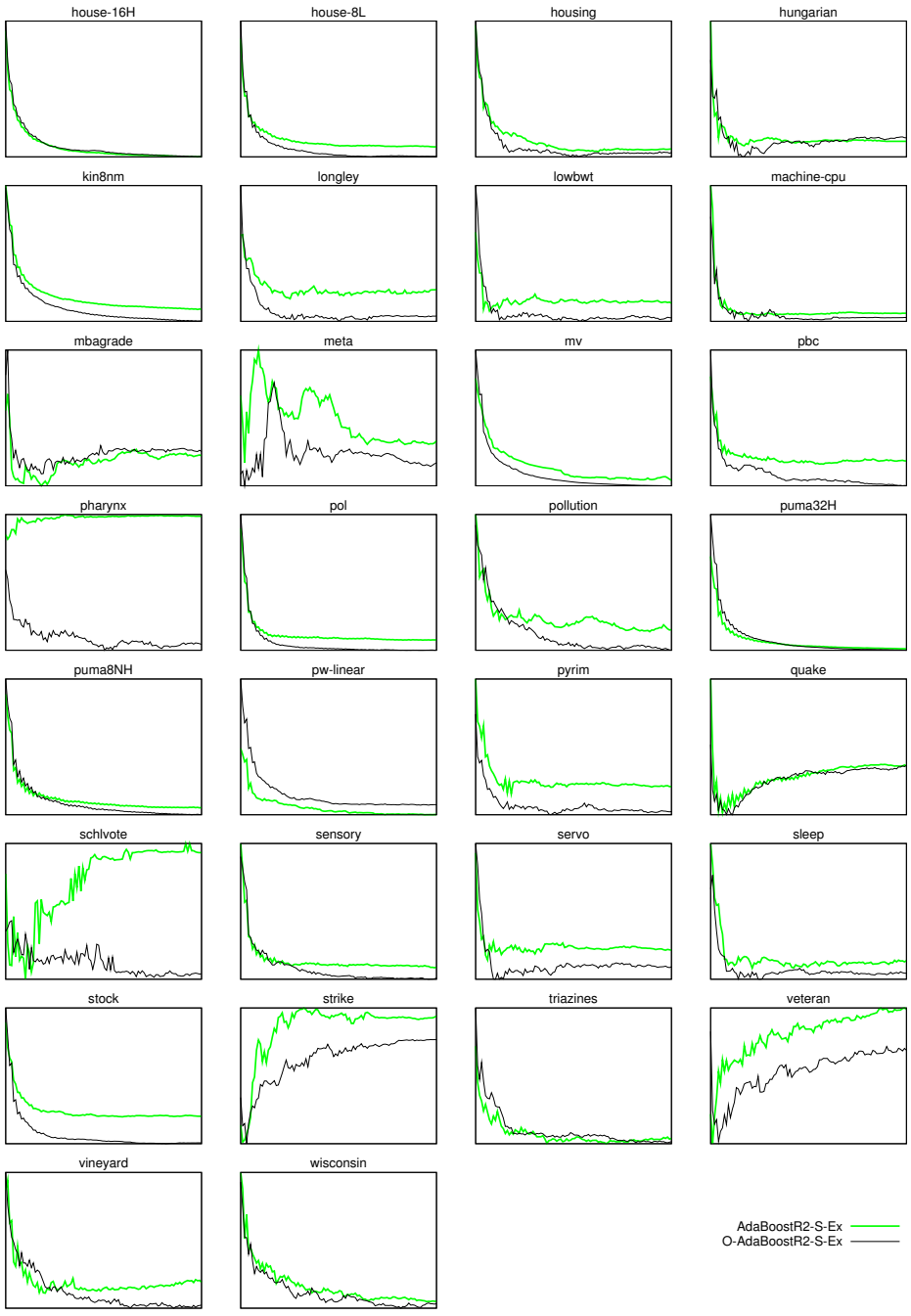


Fig. 3. Error as a function of the number of iterations for the different data sets, second part.

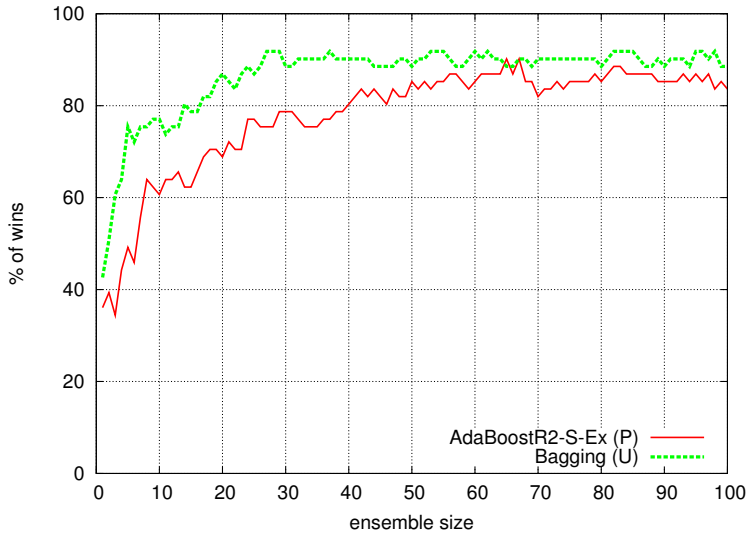


Fig. 4. Evolution of the percentage of wins when comparing the version with Oracle and the version without.

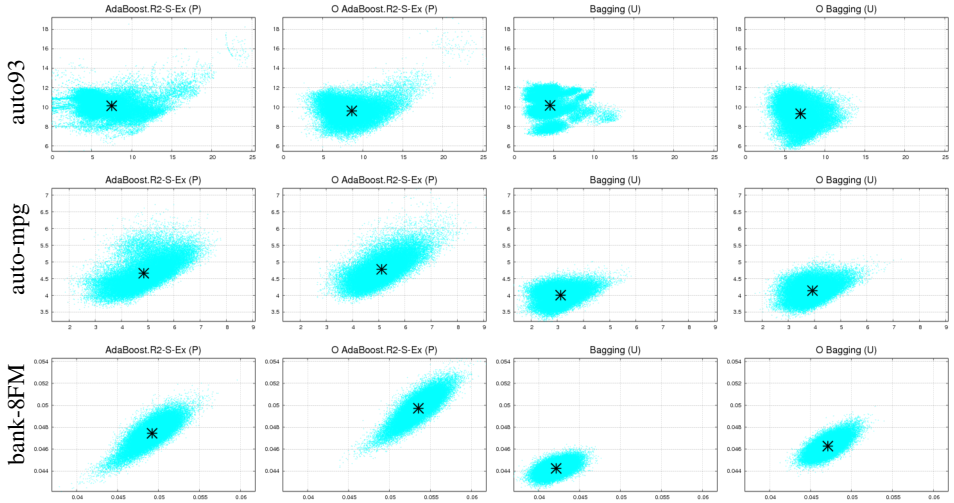


Fig. 5. Diversity error diagrams. The centers are marked with the symbol ✳.

Bagging, Random Subspaces, AdaBoost.R2 and Iterated Bagging. For all the configurations considered using Random Oracles gives better results. The cause for these improvements can be the increased diversity of the base classifiers, as shown by the diversity-error diagrams.

Acknowledgements. We wish to thank the developers of Weka. We also express our gratitude to the donors of the different datasets.

References

1. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
2. Breiman, L.: Using iterated bagging to debias regressions. Machine Learning 45(3), 261–277 (December 2001), http://dx.doi.org/10.1023/A:1017934522171
3. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
4. Dietterich, T.G.: Approximate statistical test for comparing supervised classification learning algorithms. Neural Computation 10(7), 1895–1923 (1998)
5. Drucker, H.: Improving regressors using boosting techniques. In: ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning. pp. 107–115. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1997), http://portal.acm.org/citation.cfm?id=645526.657132
6. Elomaa, T., Kääriäinen, M.: An analysis of reduced error pruning. Journal of Artificial Intelligence Research 15, 163–187 (2001)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)
8. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: 13th International Conference on Machine Learning. pp. 148–156. Morgan Kaufmann, San Francisco (1996)
9. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
10. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience (2004)
11. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles. Machine Learning 51, 181–207 (2003)
12. Kuncheva, L.I., Rodríguez, J.J.: Classifier ensembles with a random linear oracle. IEEE Transactions on Knowledge and Data Engineering 19, 500–508 (2007)
13. Margineantu, D.D., Dietterich, T.G.: Pruning adaptive boosting. In: Proc. 14th International Conference on Machine Learning. pp. 211–218. Morgan Kaufmann (1997)
14. Shrestha, D.L., Solomatine, D.P.: Experiments with AdaBoost.RT, an improved boosting scheme for regression. Neural Computation 18(7), 1678–1710 (2006), http://dx.doi.org/10.1162/neco.2006.18.7.1678
15. Suen, Y., Melville, P., Mooney, R.: Combining bias and variance reduction techniques for regression trees. In: Machine Learning: ECML 2005. pp. 741–749. Springer (2005), http://dx.doi.org/10.1007/11564096_76
16. Wang, Y., Witten, I.H.: Induction of model trees for predicting continuous classes. In: Poster papers of the 9th European Conference on Machine Learning. Springer (1997)
17. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, 2nd edn. (2005), http://www.cs.waikato.ac.nz/ml/weka
18. Zhang, C., Zhang, J., Wang, G.: An empirical study of using rotation forest to improve regressors. Applied Mathematics and Computation 195(2), 618–629 (February 2008), http://dx.doi.org/10.1016/j.amc.2007.05.010

Learning Markov Blankets for Continuous or Discrete Networks via Feature Selection

Houtao Deng[1], Saylisse Davila[1], George Runger[1], Eugene Tuv[2]

[1] Arizona State University, Tempe, AZ
[2] Intel, Chandler, AZ

Abstract. Learning Markov Blankets is important for classification/regression, causal discovery, and Bayesian network learning. We present an argument that ensemble masking measures can provide an approximate Markov blanket. Consequently, an ensemble feature selection method can be used to learn Markov blankets for either discrete or continuous networks (without linear, Gaussian assumptions). We use masking measures for redundancy and statistical inference for feature selection criteria. We compare our performance in the causal structure learning problem to a collection of common feature selection methods. We also compare to Bayesian local structure learning. These results can also be easily extended to other casual structure models such as undirected graphical models.

1 Introduction

Structure learning in Bayesian networks is an important step for causal inference, and Markov Blanket causal discovery algorithms can be helpful for learning the structure of Bayesian networks. Here we argue that ensemble masking measures (applied in decision tree ensembles) can provide an approximate Markov blanket. This result implies that an ensemble feature selection method can effectively learn Markov blankets for either discrete or continuous networks. Thus, Markov blanket learning can initialize a causal structure learning algorithm. In particular, the ensemble methods can be used in continuous network structure learning without the strong linear, Gaussian assumptions. There are well-known contexts where the linear and Gaussian assumptions common in continuous Bayesian networks (BN) do not hold (e.g, fMRI [1]). Mixed BNs provide other examples. The sensitivity of BN learning to linearity assumptions was described by [2]. Also, [3](LIMGAM) considered for the learning of causal structure, along with the inference in such models. More recent work by [4] indicated the importance of relaxed assumptions.

There are few studies that applied feature selection methods to learning causal structure in continuous networks. A linear regression model was extended by [5] to learn structure in a Gaussian undirected graph model. [6] applied SVM-RFE method to discover causal structure and proposed their methods for learning continuous Gaussian Bayesian networks with linear causal relations. A more related work was by [7](C5C). C5C uses the features selected from C5.0 rules

to identify Markov blankets of a discrete Bayesian network. However, the C5C algorithm still needs prior specification for the importance threshold. Furthermore, it is based on only one decision tree, and the greedy nature of a single tree could lead to local optimum.

The masking measure relationship to approximate Markov blanket learning suggests an ensemble-based feature selection method such as ACE [8] should be effective to initialize structure learning. Consequently, we compare a set of commonly used minimum relevancy, maximum redundancy feature selection methods to learn Markov blankets in Bayesian networks along with ACE. Therefore, the structure learning problem is studied under a collection of feature selection methods. We focus here on continuous learning experiments and illustrate performance without the common strong assumptions. A discrete network example illustrates that an ensemble-based method can also generalize to discrete networks. Finally, our work focuses on learning causal structure in Bayesian networks. However, it can be easily extended to other causal graphical models such as undirected graphical models.

In Section 2 we describe the feature selection approach and provide an argument that a masking measure defines an approximate Markov blanket. In Section 3 we provide experiments for local structure learning in continuous networks for both linear and nonlinear models and with and without Gaussian assumptions. We also provide an example for one discrete Bayesian network to illustrate that our method can generalize. Section 4 provides conclusions.

1.1 Learning Bayesian networks via feature selection

Let F be a full set of variables. Given a target variable Y, let $MB(Y) \subset F$ and $Y \notin MB(Y)$, $MB(Y)$ is said to be a Markov Blanket (MB) for Y if $Y \perp (F - MB)|MB$. That is, Y is conditionally independent of other features given MB. A MB can be considered the objective of a feature selection algorithm. However, redundant features can replace others in a feature subset. In reality, it is not so straightforward to determine feature redundancy if a feature is partially correlated to a set of features.

An MB is important for learning a Bayesian Network structure because a MB is useful for discovering the causal relationship of Y. Under certain conditions (faithfulness to a Bayesian Network), MB(Y) is identical to Y's parents, its children, and its children's other parents (co-parents) [9]. Therefore, Markov Blanket causal discovery algorithms can be helpful for learning the structure of Bayesian networks. By [10], MB(Y) for all Y are identified first, and then MB(Y) are used to construct the Bayesian network of the domain. Algorithms (e.g., [9]) have been proposed for identifying Markov Blankets in a Bayesian Network.

MB learning is also closely related to feature selection and [11] stated that MB is an optimal solution for a feature selection method. The MB definition is similar to the maximal relevancy and minimal redundancy principle used in [12], and [13]. There are common characteristics between current MB learning and feature selection algorithms. For example, the feature selection methods [14], [13], select relevant features in a forward phase and remove redundant features

in a backward phase (similar to the two phases described in a Markov Blanket learning algorithm [9]).

Therefore, those feature selection methods maximizing relevancy and minimizing redundancy could be used for learning Markov blankets and thus for learning causal structure in networks. [7] and [6] showed the advantages of feature selection methods over a Bayesian network learning algorithm for learning causal structure. Furthermore, most Bayesian network learning algorithms are designed to learn either networks with discrete variables or networks with continuous variables under the gaussian-distribution, linear-relation assumptions. The ACE feature selection method used here can deal with mixed categorical and continuous variables free of distributions and relations [8]. Therefore, the method can be used to learn local causal structure in both discrete and continuous Bayesian networks without any distribution and relation assumption.

Current feature selection approaches have been successfully used in ranking the importance of variables [15], [16] or selecting a maximal relevancy and minimal redundancy set [12], [14]. In classification and regression problems, it is well known that selecting a combination of most important individual features can not necessarily produce the best result. Therefore, a feature selection for purpose of supervised learning should be designed to maximize relevancy and minimize redundancy.

2 Feature selection framework

The framework of our method is outlined in Algorithm 1 (shown for a regression problem) and with notation summarized in Table 1. A similar algorithm applies to classification problems. Several iterations of feature selection are considered to include important features, but possibly weaker than a primary set. In each iteration, only the important features are used to predict the target and generate residuals (targets minus model predictions for regression). In subsequent iterations, the feature selection is applied to the residuals. However, all variables are input to the feature selection module that builds the ensembles—not only the currently important ones. This is to recover partially masked variables that still contribute predictive power to the model. This can occur after the effect of a masking variable is completely removed, and the partial masking is eliminated. Based on important features, the redundancy elimination module selects a non-redundant feature subset. Brief comments for the functions *SelectFeatures* and *RemoveRedundant* are provided below and further details were provided by [8].

2.1 Feature importance measure

Relevant feature selection is based on an ensemble of decision tees. Trees handle mixed categorical and numerical data, capture nonlinear interactions, are simple, fast learners. Trees also provide intrinsic feature selection scores through split values. We briefly summarize here and details were provided for the ACE feature

Algorithm 1: Ensemble-Based Feature Selection

1. Set $\Phi \leftarrow \{\}$; set $F \leftarrow \{X_1, \ldots, X_M\}$; set $I = 0$ ($|I| = M$)
2. Set $[\hat{\Phi}, \Delta I] = SelectFeatures(F, Y)$
3. Set $\hat{\Phi} = RemoveRedundant(\hat{\Phi})$
4. If $\hat{\Phi}$ is empty, then quit
5. $\Phi \leftarrow \Phi \cup \hat{\Phi}$
6. $I(\hat{\Phi}) = I(\hat{\Phi}) + \Delta I(\hat{\Phi})$
7. $Y = Y - g_Y(\hat{\Phi}, Y)$
8. Go to 2.

F	set of original variables
Y	target variable
M	Number of variables
I	cumulative variable importance vector
Φ	set of important variables
ΔI	current vector of variable importance scores from an ensemble
$\Delta I(\hat{\Phi})$	current variable importance scores for the subset of variables $\hat{\Phi}$
$g_Y(F, Y)$	function that trains an ensemble based on variables F and target Y, and returns a prediction of Y

Table 1. Notation in Algorithm 1

selection algorithm by [8]. For a single decision tree, the measure of variable importance is $VI(X_i, T) = \sum_{t \in T} \triangle I(X_i, t)$ where $\triangle I(X_i, t)$ is the impurity decrease due to an actual split on variable X_i at a node t of tree T. Impurity measure $I(t)$ for regression is defined as $\sum_{i \in t}(y_i - \bar{y})^2/N(t)$, where y_i is the response of observation i in node t, and $\bar{y}$ is the average response for all $N(t)$ observations in node t. For classification, $I(t)$ equals the Gini index at node t

$$\text{Gini}(t) = \sum_{i \neq j} p_i^t p_j^t \tag{1}$$

where p_i^t is the proportion of observations with $y = i$ and i and j run through all target class values. The split weight measure $\triangle I(X_i, t)$ can be improved if out-of-bag (OOB) samples are used. The split value for the selected variable is calculated using the training data. However, only the OOB samples are used to select the feature as the primary splitter. The experiments show that this provides a more accurate estimate of variable importance, and mitigates the cardinality problem of feature selection with trees [15] (where features with greater numbers of attributes values are scored higher by the usual metrics). Then, the

importance score in a ensemble can be obtained by averaging over the trees

$$E(X_i) = \frac{1}{M}\sum_{m=1}^{M} VI(X_i, T_M) \tag{2}$$

Furthermore, a statistical criterion is determined through the use of artificial features (permutations of the actual features). Variable importance scores for actual features are compared to the distribution of scores obtained for the artificial features. Replicates of the ensembles are also used so that a statistical t-test can generate a p-value for the importance score of an actual feature. Further comments are provided below.

2.2 Feature masking measure and its relationship to Markov blanket

Next we define a feature masking measure and argue the measure can be used to define an approximate Markov blanket. An important issue for variable importance in tree-based models is how to evaluate or rank variables that were masked by others with slightly higher splitting scores but could provide as accurate a model if used instead. One early approach in the CART methodology used surrogate splits [15]. The predictive association of a surrogate variable X^s for the best splitter X^* at a tree node T is defined through the probability that X^s predicts the action of X^* correctly and this is estimated as

$$p(X^s, X^*) = \frac{p_L(X^s, X^*) + p_R(X^s, X^*)}{N} \tag{3}$$

where $p_L(X^s, X^*)$ and $p_R(X^s, X^*)$ define the estimated probabilities that both X^s and X^* send a case in T left (right).

The predictive measure of association $\lambda(X^*|X^s)$ between X^s and X^* is defined as [8]:

$$\lambda(X^s|X^*) = \frac{min(\pi_L, \pi_R) - (1 - p(X^s, X^*))}{min(\pi_L, \pi_R)} \tag{4}$$

where $\pi_L(\pi_R)$ are the proportions of cases sent to left(right) by X^*. Here $1 - p(X^s, X^*)$ measures the error using the surrogate X^s for X^*, $min(\pi_L, \pi_R)$ measures the error of the *naive* surrogate that assigns all cases according to $max(\pi_L, \pi_R)$. If $\lambda(X^s, X^*) < 0$, then X^s is disregarded as a surrogate for X^*. Sometimes a small, nonnegative threshold is used instead of 0.

Equation (4) only measures the association at a node, we now extend it to define a masking score as follows. Variable i is said to mask variable j in a tree, if there is a split in variable i in a tree with a surrogate on variable j. We define the masking measure for a pair of variables i, j in tree T as

$$M_{ij}(T) = \sum_{\{t \in T, split\ on\ X_i\}} w(X_i, t)\lambda(X_i|X_j) \tag{5}$$

where $w(X_i, t) = \Delta I(X_i, t)$ is the decrease in impurity from the primary split on variable X_i, and summation is done over the nodes where primary split was made on variable X_i. Here we take into account both the similarity between variables X_i, X_j at the node, and the contribution of the actual split of variable X_i to the model. For an ensemble the masking measure is simply averaged over the trees. Note that in general the measure is not symmetric in the variables,e.g., X_i may mask X_j, but the reverse may not be true (one variable may mask several others, but for a single selected masked variable the reverse may not be true).

In order to show that the masking measure corresponds to the Markov Blanket criterion, we now to proceed show that if masking is strong, that is, if the predictive measure of association λ approaches one, then excluding the masked variable has no effect on the conditional distribution of the target as measured by cross-entropy. Of course, this only determines conditional independence, which is weaker than the Markov Blanket condition, but can well be used to define an approximate Markov Blanket.

Now, it is intuitively sensible that masking variable X_i, with globally high predictive association with masked variable X_j, might be a good candidate for a Markov Blanket for X_j. We use expected KL-divergence $\delta(X_i|X_j)$ to estimate how close X_i is to being a Markov Blanket for X_j. Consider

$$\delta(X_i|X_j) = \sum_{x_i, x_j} Pr(X_i = x_i, X_j = x_j) \cdot D(Pr(C|X_i = x_i, X_j = x_j), Pr(C|X_i = x_i)) \tag{6}$$

where the KL-divergence $D(p, q)$ between two distributions p and q is defined as $\sum_{c \in C} p_c \log \frac{p_c}{q_c}$. In fact, it is easy to see that our masking measure between two variables computed in a tree node behaves very similar to cross-entropy $\delta(X_i|X_j)$ locally. Specifically $\lambda(X_i|X_j) \to 1$ leads to $\delta(X_i|X_j) \to 0$.

Consider a case when node T, has a stronger primary splitter X_i masking a surrogate X_j with a high predictive association $\lambda(X_i|X_j) \sim 1$. Then a four-node tree T^* with a split on X_i followed by splits on X_j locally approximates $P(C|X_i, X_j)$, and a four-node tree T^s with three splits using only X_i approximates $P(C|X_i)$. We will show that $\delta(X_i|X_j) \sim 0$. Because trees T^* and T^s have a common root split, it suffices to demonstrate $\delta(X_i|X_j) \sim 0$ between the left (or right) two-node subtrees of T^* and T^s, constructed using X_i and X_j splitters, correspondingly. For simplicity we keep the same notations T^* and T^s for the corresponding two-node subtrees, and assume that the root nodes of both T^* and T^s have n samples. Note that n samples could be partitioned in four disjoint sets: $n = n_{LL} + n_{LR} + n_{RL} + n_{RR}$ where n_{LL} is a number of samples sent by both X_i and X_j to the left nodes of T^* and T^s correspondingly; n_{LR} is a number of samples sent by X_i to the left node of T^* but sent to the right node of T^s by X_j; n_{RL} and n_{RR} are defined in the same way. Let also n^*_{Lc} be a number of samples in T^* of class c sent to the left by X_i; quantities n^*_{Rc} , n^s_{Rc} ,n^s_{Lc} are defined similarly. Then

$$\delta(X_i|X_j) = \sum_{c=1}^{C} \left(\frac{n_{LL}}{n} \cdot \frac{n^*_{Lc}}{n} \log \frac{n^*_{Lc}}{n^s_{Lc}} + \frac{n_{RR}}{n} \cdot \frac{n^*_{Rc}}{n} \log \frac{n^*_{Rc}}{n^s_{Rc}} + \frac{n_{LR}}{n} \cdot \frac{n^*_{Lc}}{n} \log \frac{n^*_{Lc}}{n^s_{Rc}} + \frac{n_{RL}}{n} \cdot \frac{n^*_{Rc}}{n} \log \frac{n^*_{Rc}}{n^s_{Lc}}\right) \tag{7}$$

For the last two terms in (7) we see that

$$\frac{n_{LR}}{n} \cdot \frac{n^*_{Lc}}{n} \log \frac{n^*_{Lc}}{n^s_{Rc}} + \frac{n_{RL}}{n} \cdot \frac{n^*_{Rc}}{n} \log \frac{n^*_{Rc}}{n^s_{Lc}} \leq \frac{n_{LR}}{n} \cdot \log n + \frac{n_{RL}}{n} \cdot \log n$$

$$= \log n \cdot (1 - \frac{n_{LL} + n_{RR}}{n}) < log(n) \cdot (1 - \lambda(X_i|X_j)) \rightarrow 0 \; as \; \lambda \rightarrow 1$$

Denote n_{LLc} a subset of n_{LL} samples that belongs to class c, then for the first term in (7) we have $\frac{n_{LL}}{n} \cdot \frac{n^*_{Lc}}{n} \log \frac{n^*_{Lc}}{n^s_{Lc}} \leq \log \frac{n^*_{Lc}}{n^s_{Lc}} = \log \frac{n_{LLc}+n_{LRc}}{n_{LLc}+n_{RLc}}$, but $\max(n_{LRc}, n_{RLc}) \leq \max(n_{LR}, n_{RL}) \leq n_{LR}+n_{RL} = n-n_{LR}-n_{RL} \rightarrow 0 \; as \; \lambda \rightarrow 1$ hence, the upper bound for the first term $\log(n_{LLc} + n_{LRc})/(n_{LLc} + n_{RLc}) \rightarrow 0 \; as \; \lambda \rightarrow 1$ The same exact argument applies for the second term in (7), and therefore $\delta(X_i|X_j) \rightarrow 0 \; as \; \lambda(X_i|X_j) \rightarrow 1$.

We have just shown that the defined masking measure indeed corresponds to KL-divergence and thus provides an approximately optimal means to remove redundant variables based on the Markov Blanket criterion. We next describe an efficient algorithm to make use of the measure.

2.3 Statistical criteria for identifying relevant and redundant features

For deleting irrelevant or redundant features, a threshold is needed. Artificial contrasts can be used to construct and specify the threshold in an efficient way. Let the number of variables be M. Denote the variables set as $S_X = \{X_j, j = 1, 2, ...K\}$. In each replicate $r, r = 1, 2, ...R$, artificial variables are generated as follows. For every variable X_j in S_X, a corresponding artificial variables Z^r_j is generated from randomly permutating values of X_j, let $S^r_Z = \{Z^r_j, j = 1, 2, ...K\}$. Then the new variables set can be denoted as $S^r_{X,Z} = \{S_X, S^r_Z\}$.

Consider relevant variables selection. Denote the importance score of $S^r_{X,Z}$ as $I^r_{X,Z} = \{I^r_X, I^r_Z\}$, where $I^r_X = \{I^r_{X_j}, j = 1, 2, ...M)\}$ and $I^r_Z = \{I_{Z^r_j}, j = 1, 2, ...K)\}$, $I^r_{X_j}$ and $I_{Z^r_j}$ are the importance scores of X_j and Z^r_j at the r^{th} replicate respectively. Denote $I_{X_j} = \{I^r_{X_j}, r = 1, 2, ...R\}$. Then $I^r_{X,Z}$ can be obtained by using relevant feature selection methods to $S^r_{X,Z}$. Denote I^r_α as the $1 - \alpha$ percentile value of I^r_Z and $I_\alpha = \{I^r_\alpha, r = 1, 2, ..., R\}$. For each variable X_j, a paired t-test compares I_{X_j} to I_α. A test that results in statistical significance, i.e., a suitably small p-value, identifies an important variable. Therefore, an important variable here need consistently score higher than the artificial variables over multiple replicates.

Consider redundancy elimination. Let $M^r_{X_i,X_j}$ for $j = 1, 2, ...i-1, i+1, ..., K$ and $M^r_{X_i,Z_j}$ for $j = 1, 2, ..., K$ denote the masking score of X_i over X_j, and over Z^r_j for replicate $S^r_{X,Z}$ respectively. Denote $M^r_{X_i,\alpha}$ as the $1 - \alpha$ percentile value of $M^r_{X_i,Z_j}$ and $M_{X_i,\alpha} = \{M^r_{X_i,\alpha}, r = 1, 2, ..., R\}$. A paired t-test compares between $M^r_{X_i,X_j}$ and $M_{X_i,\alpha}$. Variable X_j is masked by variable X_i if the test is significant.

2.4 Residuals for multiple iterations

A single iteration in Algorithm 1 can select a relevant and non-redundant feature set, but it may fail to detect some variables that are important but possibly weaker than a primary set. Thus, more iterations are considered here. At the end of each iteration, a subset of features $\hat{\Phi}$ can be obtained. An ensemble model $g_Y(\hat{\Phi})$ is built on $\hat{\Phi}$. Denote $\hat{Y}$ as the OOB prediction of $g_Y(\hat{\Phi})$. Then residuals are calculated and form a new target. For a regression problem, the new target is simply formed by: $Y = Y - \hat{Y}$. For a classification problem, residuals are calculated from a multi-class logistic regression procedure. Log-odds of class probabilities for each class are predicted (typically a gradient boosted tree [17]is used), and then pseudo-residuals are taken as residuals.

In a Bayesian network, sometimes non-causal, but relevant variables, can also contribute to the target. Though the contribution from those non-causal but relevant variables could be small compare to causal related variables, ACE adds them into the feature set. Therefore, false alarm rates might be increased. The Bonferroni correction is a multiple-comparison correction used when several statistical tests are performed simultaneously. The Bonferroni correction is used here to reduce the false positive rate. For example, if the p-value of t-test in the previous sections is α, the p-value is reduced to α/N when there are N features.

3 Experiments

The work here focuses on continuous Bayesian networks, but we add an example from a discrete network that also illustrates that the method easily generalizes– the discrete networks results are equally good. We applied our method and the feature selection methods CFS [14], SVM-RFE [13], and FCBF [18] to learn the MB of the target nodes. The performance is also compared to a well-known Bayesian local structure learning algorithm (MMPC) [19]. In the experiments, ACE [8] is programmed in C, and RWeka [20, 21] and bnlearn [22] in R [23] are used to run the other algorithms. The default parameter setting for the methods in the software are used. To evaluate the performance of an algorithm, we measure the sensitivity and specificity for a given task. The sensitivity is the ratio of the number of correctly identified variables in the MB over the size of the true MB. The specificity is the ratio of the number of correctly identified variables as not belonging in the MB over the true number of variables not in MB [19]. To compare different algorithms, we follow the same terminology that was used by [19] and define a combined measure d:

$$d = \sqrt{(1 - \text{sensitivity})^2 + (1 - \text{specificity})^2} \tag{8}$$

A better algorithm implies a smaller d value.

3.1 Continuous, Gaussian local structure learning

There are few available continuous benchmark causal-structure network (the focus is on discrete networks). Therefore, we simulated a causal-structure network

with continuous nodes as shown Figure 1. Bayesian structure learning often assumes Gaussian models whereas the ensemble-based ACE method is not limited to the such models. The first experiment uses the common Gaussian distributions for these experiments, and a second experiment relaxes this assumption. Because FCBF and SVM-RFE (in RWeka [20, 21]) do not work with continuous target variables, only ACE, MMPC and CFS with best first search (CFSBestFirst) and gene search (CFSGene) are applied to this data.

Consider the network in Figure 1. For the first experiment nodes A, B, C are root nodes and follow normal distributions N(1,1), N(2,1) N(3,1), respectively, where $N(\mu, \sigma^2)$ denotes a normal distribution with mean μ and variance σ^2. Denote a node (not a root node) as N_i, and denote the parent nodes of N_i as $N_i^p(j), j = 1, ..., |N_i^p|$, where $|N_i^p|$ is the number of parent nodes of N_i. The causal relation between N_i and $N_i^p(j)$ is expressed by $N_i = f(N_i^p(j))$. We considered $N_i = \sum_{j=1}^{|N_i^p|}(N_i^p(j)) + \varepsilon$ or $N_i = \prod_{j=1}^{|N_i^p|}(N_i^p(j)) + \varepsilon$ where $\varepsilon \sim N(0, 1)$. Therefore, we can investigate both linear and nonlinear causal relationships in the network. For example, in Figure 1, the linear causal relationship between node D and its parent nodes A, C is $D = A + C + \varepsilon$. The nonlinear causal relationship is $D = A * C + \varepsilon$. For each of the continuous Bayesian networks, 5000 rows of data are simulated. The objective is to learn the MB of the output nodes.

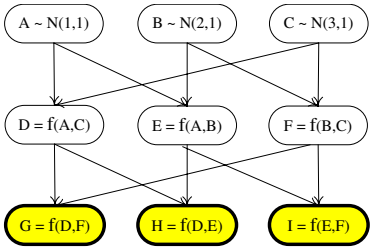


Fig. 1. Nodes with thick edges (yellow) are taken as targets. The function f is taken as either an additive or multiplicative function of the inputs.

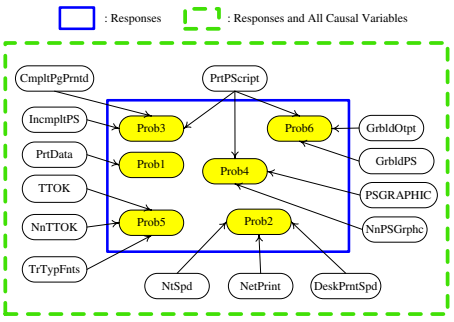


Fig. 2. Local structure of the Windows printer network with regard to targets

The results for the linear and non-linear cases are shown in Table 2. For the linear Bayesian network, it is well known that linear relationships are not optimal for a tree representation, but well-suited for correlation-based methods. Still, ACE has the lowest d value. The other three methods have the same d value. For the non-linear network, the challenge of learning increases, and the d of all methods increase. ACE still produces the smallest d value.

3.2 Continuous, nonGaussian local structure learning

For the nonGaussian experiment the distributions for nodes A, B, C were changed to $Normal(0, 1)$, $Exponential(1)$, $Uniform(-1, 1)$ respectively. Other charac-

Table 2. Measure d for each output node from different algorithms learning continuous, Gaussian, linear and nonlinear Bayesian networks

	Linear				NonLinear			
	G	H	I	**Average**	G	H	I	**Average**
ACE	0.00	0.00	0.17	**0.06**	0.00	0.33	0.00	**0.11**
CFSBestFirst	0.33	0.33	0.33	**0.33**	0.50	0.67	0.83	**0.67**
CFSGene	0.33	0.33	0.33	**0.33**	0.50	0.67	0.67	**0.61**
MMPC	0.33	0.33	0.33	**0.33**	1.00	0.33	0.97	**0.77**

teristics of the experiment (including the linear and nonlinear target functions) were the same as in the Gaussian case.

Table 3. Measure d for each output node from different algorithms learning continuous, nonGaussian, linear and nonlinear Bayesian networks

	Linear				NonLinear			
	G	H	I	**Average**	G	H	I	**Average**
ACE	0.00	0.00	0.33	**0.11**	0.50	0.50	0.67	**0.56**
CFSBestFirst	0.33	0.17	0.17	**0.22**	1.12	1.20	0.71	**1.01**
CFSGene	0.33	0.17	0.17	**0.22**	1.12	1.20	0.71	**1.01**
MMPC	0.50	0.50	0.33	**0.44**	1.12	1.20	0.71	**1.01**

For both nonGaussian linear and nonlinear networks, ACE is still better than the other three methods. CFSBestFirst outperforms MMPC in the nonGaussian linear case while they have similar performance in other cases. Consequently, feature selection methods can provide reasonable alternatives to the MMPC algorithm in continuous networks. Furthermore, it is more difficult for all methods to learn a nonlinear relationship than a linear relationship in the nonGaussian cases.

3.3 Discrete local structure learning

Although our focus is continuous network structure, a discrete Bayesian network is also considered. The network is Windows printer trouble shooting with 76 features and 10000 observations were generated with the GeNIe structural modeling tool (http://genie.sis.pitt.edu/). Due to limitations of space, only the local structure with regard to the targets of the network are illustrated in Figure 2. Here 6 nodes (printer problem nodes) are considered as the targets (each with binary classes). ACE, MMPC, FCBF, CFSBestFirst, CFSGene, SVM-RFE are compared based on learning the local structure of the Bayesian networks. Because SVM-RFE requires the number of features to be selected as an input, we assign the number of features in two ways: the size of the correct Markov Blanket and the number of features selected by ACE. We refer the SVM-RFE with these

two parameters as SVM(MB) and SVM(ACE), respectively. The results from the Windows printer trouble shooting network are shown in Table 4.

Table 4. Measure: d of outputs from different algorithms learning the Windows printer network. SVM(MB) is given the correct number of features, and SVM(ACE) is given the number of features selected by ACE

	Pro1	Prob2	Prob3	Prob4	Prob5	Prob6	**Average**
ACE	0.00	0.00	0.33	0.00	0.00	0.67	**0.167**
CFSBestFirst	0.11	0.03	0.34	0.04	0.34	0.34	**0.199**
CFSGene	0.34	0.19	0.41	0.37	0.25	0.19	**0.292**
FCBF	0.00	0.03	0.67	0.04	0.34	0.67	**0.291**
MMPC	0.00	0.04	0.67	0.33	0.03	0.67	**0.289**
SVM(ACE)	0.00	0.00	0.67	0.00	0.00	0.67	**0.222**
SVM(MB)	0.00	0.00	0.33	0.00	0.00	0.33	**0.111**

For the Windows printer network, ACE and SVM(MB) have the lowest d values. SVM(MB) only outperforms ACE for the target Prob6. However, SVM(MB) is given the priori knowledge of the size of true MBs. With the number of variables selected by ACE as input, SVM(ACE) does not perform as well as ACE. Another feature selection method CFSBestFirst also provides better results than MMPC.

4 Conclusions

Structure learning is important for both discrete and continuous networks, and relaxed Gaussian assumptions are important for continuous networks. A relationship between ensemble masking and Markov Blankets is argued here and exploited for a generalized feature selection method to handle discrete and continuous cases for local structure learning. Common feature selection methods, along with a Bayesian structure algorithm, are compared for the structure learning problem, and experiments illustrates the strength of an ensemble-based feature selection approach in these cases.

Acknowledgements. This research was partially supported by ONR grant N00014-09-1-0656

References

1. Tillman, R., Gretton, A., Spirtes, P.: Nonlinear directed acyclic structure learning with weakly additive noise models. Advances in Neural Information Processing Systems (NIPS) **22** (2009) 18471855
2. Voortman, M., Druzdzel, M.: Insensitivity of constraint-based causal discovery algorithms to violations of the assumption of multivariate normality. In: Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference (FLAIRS) 2008. (2008)

3. Shimizu, S., Hoyer, P., Hyvarinen, A., Kerminen, A.: A linear non-gaussian acyclic model for causal discovery. Journal of Machine Learning Research **7** (2006) 2003–2030
4. Hoyer, P., Janzing, D., Mooij, J., Peters, J., Scholkopf, B.: Nonlinear causal discovery with additive noise models. Advances in Neural Information Processing Systems (NIPS) **21** (2009) 689–696
5. Li, F., Yang, Y.: Use modified lasso regressions to learn large undirected graphs in a probabilistic framework. In: 20th National Conference on Artificial Intelligence (AAAI 2005). (2005)
6. Pellet, J.P., Elisseeff, A.: Using Markov blankets for causal structure learning. Journal of Machine Learning Research **9** (2008) 1295–1342
7. Frey, L., Fisher, D., Tsamardinos, I., Aliferis, C., Statnikov, A.: Identifying Markov blankets with decision tree induction. Third IEEE International Conference on Data Mining (Nov 2003) 59 – 66
8. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature selection with ensembles, artificial variables, and redundancy elimination. Journal of Machine Learning Research **10** (2009) 1341–1366
9. Tsamardinos, I., Aliferis, C., Statnikov, A.: Algorithms for large scale Markov blanket discovery. In: Proceedings of the 16th International FLAIRS Conference. (2003)
10. Margaritis, D., Thrun, S.: Bayesian network induction via local neighborhoods. In: In Advances in Neural Information Processing Systems 12(NIPS). (1999)
11. Koller, D., Sahami, M.: Toward optimal feature selection. In: Proceedings of ICML-96: 13th International Conference on Machine Learning. (1996) 284–292
12. P. Pudil, J.N., Kittler, J.: Floating search methods in feature selection. Pattern Recognition Letters **15**(11) (1994) 1119–1125
13. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning **46**(1-3) (2002) 389–422
14. Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: Proceedings of the 17th International Conference on Machine Learning. (2000) 359–366
15. Breiman, L.: Random forests. Machine Learning **45**(1) (2001) 5–32
16. Robnik-Sikonja, M., Kononenko., I.: Theoretical and empirical analysis of relief and relieff. Machine Learning **53** (2003) 23–69
17. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: A statistical view of boosting. Annals of Statistics **28** (2000) 832–844
18. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. Journal of Machine Learning Research **5** (2004) 1205–1224
19. Tsamardinos, I., Aliferis, C., Statnikov, A.: Time and sample efficient discovery of Markov blankets and direct causal relations. In: Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining (KDD) 2003. (2003) 673–678
20. Hornik, K., Buchta, C., Zeileis, A.: Open-source machine learning: R meets Weka. Computational Statistics **24**(2) (2009) 225–232
21. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. 2nd edn. Morgan Kaufmann, San Francisco (2005)
22. Scutari, M.: Learning bayesian networks with the bnlearn R package. Journal of Statistical Software **35**(3) (2010) 1–22
23. Ihaka, R., Gentleman, R.: R: A language for data analysis and graphics. Journal of computational and graphical statistics **5**(3) (1996) 299–314

Bias-Variance Analysis of ECOC and Bagging Using Neural Nets

Cemre Zor[1], Terry Windeatt[1] and Berrin Yanikoglu[2]

[1]Center for Vision, Speech and Signal Processing, University of Surrey, UK, GU2 7XH
(c.zor, t.windeatt)@surrey.ac.uk
[2]Sabanci University, Tuzla, Istanbul, Turkey, 34956
berrin@sabanciuniv.edu

Abstract. One of the methods used to evaluate the performance of ensemble classifiers is bias and variance analysis. In this paper, we analyse bagging and ECOC ensembles using bias-variance domain of James [1] and make a comparison with single classifiers, when using Neural Networks (NNs) as base classifiers. As the performance of the ensembles depends on the individual base classifiers, it is important to understand the overall trends when the parameters of the base classifiers, nodes and epochs for NNs, are changed. We show experimentally on 5 artificial and 4 UCI MLR datasets that there are some clear trends in the analysis that should be taken into consideration while designing NN classifier systems.

1 Introduction

Within machine learning research, many techniques have been proposed in order to understand and analyse the success of ensemble classification methods over single-classifier classifications. One of the main approaches considers tightening the generalization error bounds by using the margin concept [6]. Though theoretically interesting, bounds are not usually tight enough to be used in practical design issues. Bias and variance analysis is another method used to show why ensembles work well. In this paper, we try to analyse the success of bagging [22] and Error Correcting Output Coding (ECOC) [4] as ensemble classification techniques, by using Neural Networks (NNs) as the base classifiers and zero-one loss as the loss function within the bias and variance framework of James [1]. As the characteristics of the ensemble depend on the specifications of the base classifiers, having a detailed look at the parameters of the base classifiers within the bias-variance analysis is of importance. Similar work for bagged Support Vector Machines (SVMs) within Domingos' bias-variance framework [7] can be found in [19].

ECOC is an ensemble technique [4], in which multiple base classifiers are trained according to a preset binary *code matrix*. Consider an ECOC matrix C, where a particular element $C_{ij} \epsilon (+1, -1)$ indicates the desired label for class i, to be used in training the base classifier j. The base classifiers are the dichotomizers which carry out the two-class classification tasks for each column of the matrix, according to the input labelling. Each row, called a *codeword*, represents the

desired output for the whole set of base classifiers for the class it indicates. During decoding, a given test sample is classified by computing the similarity between the output (hard or soft decisions) of each base classifier and the codeword for each class by using a distance metric, such as the Hamming or the Euclidean distance. The class with the minimum distance is then chosen as the estimated class label. The method can handle incorrect base classification results up to a certain degree. Specifically, if the minimum Hamming distance (HD) between any pair of codewords is d, then up to $\lfloor (d-1)/2 \rfloor$ single bit errors can be corrected.

As for bias and variance analysis, after the initial work of Geman [8] on the regression setting using squared-error loss, others like Breiman [20], Kohavi and Wolpert [10], Dietterich and Kong [9], Friedman [11], Wolpert [23], Heskes [12], Tibshirani [13], Domingos [7] and James [1] have tried to extend the analysis for the classification setting. One of the problems with the above definitions of bias and variance is that most of them are given for specific loss functions such as the zero-one loss, and it is hard to generalize them for all the other loss functions. Usually, new definitions are driven for each loss function. Even if the definitions are proposed to be general, they may fail to satisfy the additive decomposition of the prediction error defined in [8]. The definition of James has advantages over the others as it proposes to construct a scheme which is generalizable to any symmetric loss function. Furthermore, it proposes two more concepts called "systematic effect" and "variance effect" which help assure the additive prediction error decomposition for general loss functions and realize the effects of bias and variance on the prediction error.

Some characteristics of the other definitions which make James' more preferable for us are as follows: 1) Dietterich allows a negative variance and it is possible for the Bayes classifier to have positive bias. 2) Experimentally, the trends of Breiman's bias and variance closely follow James' systematic effect and variance effect ones respectively. However, for each test input pattern, Breiman separates base classifiers into two sets, as biased and unbiased; and considers each test pattern only to have either bias or variance accordingly. 3) Kohavi and Wolpert also assign a nonzero bias to the Bayes classifier but the Bayes error is absorbed within the bias term. Although it helps avoid the need to calculate the Bayes error in real datasets through making unwarranted assumptions, it is not preferable since the bias term becomes too high. 4) The definitions of Tibshirani, Heskes and Breiman are difficult to generalize and extend for the loss functions other than the ones for which they were defined. 5) Friedman proposes that bias and variance do not always need to be additive.

In addition to all these differences, it should also be noted that the characteristics of bias and variance of Domingos' definition are actually close to James', although the decomposition can be considered as being multiplicative [1].

In the literature, attempts have also been made to explore the bias-variance characteristics of ECOC and bagging ensembles. Examples can be found in [1] [9] [20][14][15]. In this paper, a detailed bias-variance analysis of ECOC and bagging ensembles using NNs as base classifiers is given while systematically changing parameters, namely nodes and epochs, based on James' definition.

2 Bias and Variance Analysis of James

James [1] extends the prediction error decomposition, which is initially proposed by Geman et al [8] for squared error under regression setting, for all symmetric loss functions. Therefore, his definition also covers zero-one loss under classification setting, which we use in the experiments.

In his decomposition, the terms "systematic effect" and "variance effect" satisfy the additive decomposition for all symmetric loss functions, and for both real valued and categorical predictors. They actually indicate the effect of bias and variance on the prediction error. For example, a negative variance effect would mean that variance actually helps reduce the prediction error. On the other hand, the "bias" term is defined to show the average distance between the response and the predictor; and the "variance" term refers to the variability of the predictor. As a result, both the meanings and the additive characteristics of the bias and variance concepts of the original setup have been preserved. Following is a summary of the bias-variance derivations of James:

For any symmetric loss function L, where $L(a,b) = L(b,a)$:

$$\begin{aligned} E_{Y,\tilde{Y}}[L(Y,\tilde{Y})] &= E_Y[L(Y,SY)] + E_Y[L(Y,S\tilde{Y}) - L(Y,SY)] \\ &\quad + E_{Y,\tilde{Y}}[L(Y,\tilde{Y}) - L(Y,S\tilde{Y})] \\ prediction\,error &= Var(Y) + SE(\tilde{Y},Y) + VE(\tilde{Y},Y) \end{aligned}$$

where $L(a,b)$ is the loss when b is used in predicting a , Y is the response, $\tilde{Y}$ is the predictor, SE is the systemmatic effect and VE is the variance effect. $SY = argmin_\mu E_Y[L(Y,\mu)]$ and $S\tilde{Y} = argmin_\mu E_Y[L(\tilde{Y},\mu)]$. We see here that prediction error is composed of the variance of the response (irreducible noise), systematic effect and variance effect.

Using the same terminology, the bias and variance for the predictor are defined as follows:

$$\begin{aligned} Bias(\tilde{Y}) &= L(SY, S\tilde{Y}) \\ Var(\tilde{Y}) &= E_{\tilde{Y}}[L(\tilde{Y}, S\tilde{Y})] \end{aligned}$$

When the specific case of classification problems with zero-one loss function is considered, we end up with the following formulations:

$L(a,b) = I(a \neq b)$, $Y \epsilon \{1,2,3..N\}$ for an N class problem, $P_i^Y = P_Y(Y=i)$, $P_i^{\tilde{Y}} = P_{\tilde{Y}}(\tilde{Y}=i)$, $ST = argmin_i E_Y[I(Y \neq i)] = argmax_i P_i^Y$

Therefore,

$$\begin{aligned} Var(Y) &= P_Y(Y \neq SY) = 1 - max_i P_i^Y \\ Var(\tilde{Y}) &= P_{\tilde{Y}}(\tilde{Y} \neq S\tilde{Y}) = 1 - max_i P_i^{\tilde{Y}} \\ Bias(\tilde{Y}) &= I(S\tilde{Y} \neq SY) \end{aligned}$$

$$VE(\tilde{Y},Y) = P(Y \neq \tilde{Y}) - P_Y(Y \neq S\tilde{Y}) = P^Y_{S\tilde{Y}} - \sum_i P^Y_i P^{\tilde{Y}}_i$$

$$SE(\tilde{Y},Y) = P_Y(Y \neq S\tilde{Y}) - P_Y(Y \neq SY) = P^Y_{SY} - P^Y_{S\tilde{Y}}$$

where $I(q)$ is 1 if q is a true argument and 0 otherwise.

3 Experiments

3.1 Experimental Setup

Experiments have been carried out on 5 artificial and 4 UCI MLR [21] datasets. 3 of the artificial datasets are created according to Breiman's description in [20]. Detailed information about the sets can be found in Table 1. The optimization method used in NNs is the Levenberg-Marquart (LM) technique; the level of training (epochs) varies between 2 and 15; and the number of nodes between 2 and 16.

The ECOC matrices are created by randomly assigning binary values to each matrix cell and Hamming Distance is used as the metric in the decoding stage. In the experiments, 3 classification methods are analysed: Single classifier, bagging, and ECOC. In each case, 50 base classifiers are created for bias-variance analysis. Each base classifier is either a single classifier, or an ensemble consisting of 50 bagged classifiers or ECOC matrices of 50 columns.

Experiments have been repeated 10 times for the artificial datasets by using different training & test data, as well as different ECOC matrices in each run; and the results are averaged[1]. The number of training patterns per base classifier is equal to 300; and the number of test patterns is 18000. For the UCI datasets having separate test sets, the analysis has been done just once for the single classifier and bagging settings, and 10 times with different matrices for the ECOC setting. Here, bootstrapping is applied on the base classifiers, as it is expected to be a close enough approximation to random & independent data generation from a known underlying distribution [20]. As for the UCI datasets without separate test sets, the *ssCV* cross-validation method of Webb and Conilione [16], which allows the usage of the whole dataset both in training and test stages, has been implemented. In *ssCV*, the shortcomings of the hold-out approach like the usage of small training and test sets; and the lack of inter-training variability control between the successive training sets has been overcome. In our experiments, we set the inter-training variability constant δ to 1/2.

The Bayes error, namely $Var(Y)$ for the zero-one loss function, is analytically calculated for the artificial datasets, as the underlying likelihood probability distributions are known. As for the real datasets; either the need for the underlying probability distributions has been overcome by assuming zero noise level [7], or some heuristic methods like using nearest neighbours [1] have been proposed to

[1] On the two class problems, ECOC has not been used, as it would be nothing different than applying bagging. The effect of bootstrapping of bagging would be satisfied by the random initial weights of LM.

Table 1. Summary of the datasets used

	Type	# Training Samples	# Test Samples	# Attributes	# Classes	Bayes Error (%)
TwoNorm [20]	Artificial	300*	18000*	20	2	2.28
ThreeNorm [20]	Artificial	300*	18000*	20	2	10.83
RingNorm [20]	Artificial	300*	18000*	20	2	1.51
ArtificalMulti1	Artificial	300*	18000*	2	5	21.76
ArtificalMulti2	Artificial	300*	18000*	3	9	14.33
Glass Identification	UCI	214	-	10	6	38.66
Dermatology	UCI	358	-	33	6	9.68
Segmentation	UCI	210	2100	19	7	4.21
Yeast	UCI	1484	-	8	10	43.39

*: The training and test samples for the artificial datasets change per each base classifier and per each run respectively.

estimate the underlying probability distributions and therefore the Bayes error in the literature. The first approach has the shortcoming of a wrong estimate on bias. Therefore, we also use a heuristic method in our experiments to do the estimation. Our motivation is to find the optimal classifier parameters giving the lowest error rate possible, through cross-fold validation (CV); and then to use these parameters to construct a classifier which is expected to be close enough to the Bayes classifier. This classifier is then used to calculate the output probabilities per pattern in the dataset. For this, we first find an optimal set of parameters for RBF SVMs by applying 10 fold CV; and then, obtain the underlying probabilities by utilizing the leave-one-out approach. Using the leave-one-out approach instead of training and testing the whole dataset with the found CV parameters helps us avoid overfitting. It is assumed that the underlying distribution stays almost constant for each fold of the leave-one-out procedure.

3.2 Results

In this section, some clear trends found in the analysis are discussed. Although the observations are made using 9 datasets, for brevity reasons we only present a number of representative graphs.

Prediction errors obtained by bagging and ECOC ensembles are always lower than those of the single classifier; and the reduction in the error is almost always a result of reductions both in variance effect (VE) and in systematic effect (SE). This observation means that the contributions of bias and (predictor) variance to the prediction error are smaller when ensembles are used (Fig 1, Fig 2). Note that, reductions in VE have greater magnitude, and in two-class problems, the reduction in SE is almost zero (Fig 3). As for bias and variance themselves, it has been observed that ECOC and bagging induce reduction in both, but especially in variance, in almost all the cases. The fact that NNs are high variance - low bias classifiers also plays a role in these observations, where the high variance is more easily reduced compared to the already lower bias; and VE is reduced in greater

amounts compared to SE. In [20] and [9], bagging and ECOC are also stated to have low variance in the additive error decomposition, and Kong-Dietterich framework [9] also acknowledges that ECOC reduces variance.

The convergence of single classifiers to the optimal prediction error are usually achieved at higher number of epochs than those of bagging; and ECOC ensemble convergence is mostly at even lower epochs than bagging. The prediction errors are also in the same descending order: single classifier, bagging and ECOC. The only exceptions to these happen when high number of nodes and epochs are used. Under these circumstances, the VE, SE, and therefore the prediction errors of both ECOC and bagging are similar. However, it should also be noted that ECOC outperforms bagging in sense of speed due to the fact that it divides multi-class classification problems into binary classification ones.

It is also almost always the case that the prediction error of ECOC converges to its optimum in 2 nodes, whereas a single classifier requires a higher number of nodes. Moreover, for ECOC, the number of epochs at the optimum is also lower than or equal to that of the single classifier. In other words, compared to a single classifier trained with high number of epochs and nodes, an ECOC can yield better results with fewer nodes and epochs. The trend is similar when bagging is considered. It usually stands between the single classifier and ECOC, in sense of accuracy and convergence points.

When the single classifier case is taken into account; we see that VE does not necessarily follow the trend of variance. It happens especially when the number of nodes and epochs is small, that is when the network is relatively weak (Fig 2). In this scenario, the variance decreases while the VE increases. This is actually an expected observation as one would expect having high variance to help hitting the right target class, when the network is relatively less decisive. Ensemble methods do not show this property as much as the single classifier. A possible explanation might be that each base ensemble classifier already makes use of variance coming from the base classifiers it is composed of; and this compensates for the decrease in VE of single classifiers with high variance, in weak networks.

Therefore, more variance among base ensemble classifiers does not necessarily help having less VE. However, an example of bagging creating negative VE, which clearly states that having variance reduces prediction error; and then going back to positive when variance increases, can be observed on ArtificialMulti2 data when it is processed with 4 node NNs. A similar observation is that although the variance has high values in networks with small number of nodes and epochs, the magnitude of its effect is relatively smaller (Fig 1, Fig 2).

In the above mentioned scenario of VE showing an opposite trend of variance, the bias-variance trade-off can be observed. At the points where the VE increases, SE decreases to reveal an overall decrease in the prediction error. However, these points are not necessarily the optimal points in terms of the prediction error; the optima are mostly where there is both VE and SE reduction (Fig 2). Apart from this case, bias and variance are mostly correlated with SE and VE respectively. This is also pointed out in [1] (Fig 2, Fig 3).

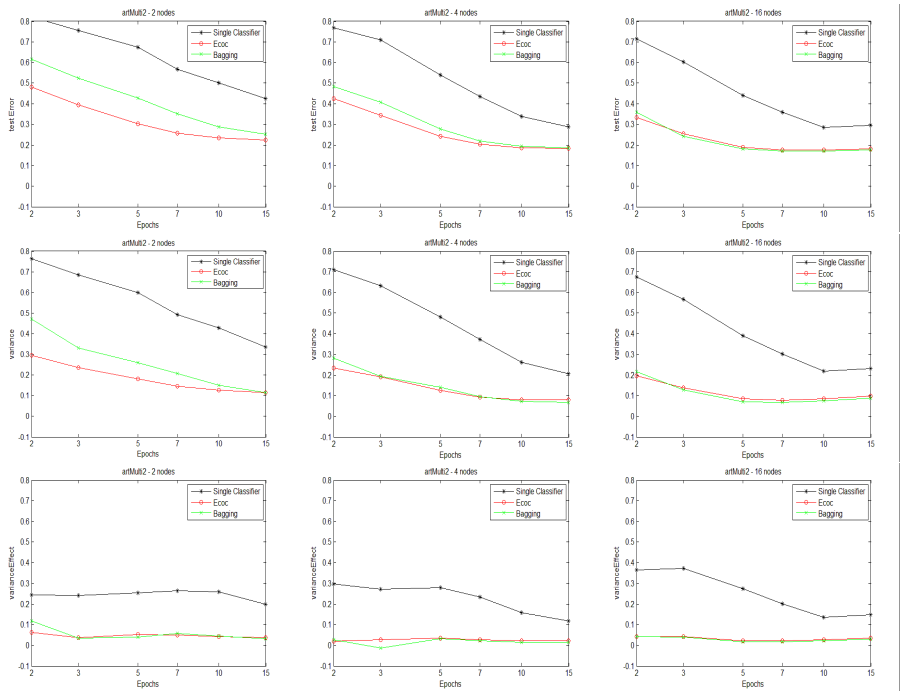


Fig. 1. Bias Variance Analysis for ArtificalMulti2 data. First Row: Overall prediction error. Second Row: Variance. Third Row: Variance effect. First Column: For 2 Nodes. Second Column: For 4 Nodes. Third Column: For 16 Nodes. Black lines indicate the results for single classifier, red for ECOC and green for bagging

4 Discussion

By analysing bagging, ECOC and single classifiers consisting of NNs through the bias-variance definition of James, we have found some clear trends and relationships that offer hints to be used in classifier design. For multi-class classification problems, the increase in the overall prediction performance obtained with ECOC makes it preferable over the single classifiers. The fact that it converges to the optimum by using smaller number of nodes and epochs is yet another advantage. It also outperforms bagging mostly, while in other cases gives similar results. As for the two-class problems, bagging always outperforms the single classifier; and the optimum number of nodes and epochs is relatively smaller.

The increase in the performance of bagging and ECOC is a result of the decrease in both variance effect and systematic effect, although the reductions in the magnitude of the variance effect are bigger. Also, when the NNs are weak, that is when they have been trained with few nodes and epochs, we see that the trends of variance and variance effect might be in opposite directions in the single classifier case. This implies that having high variance might help

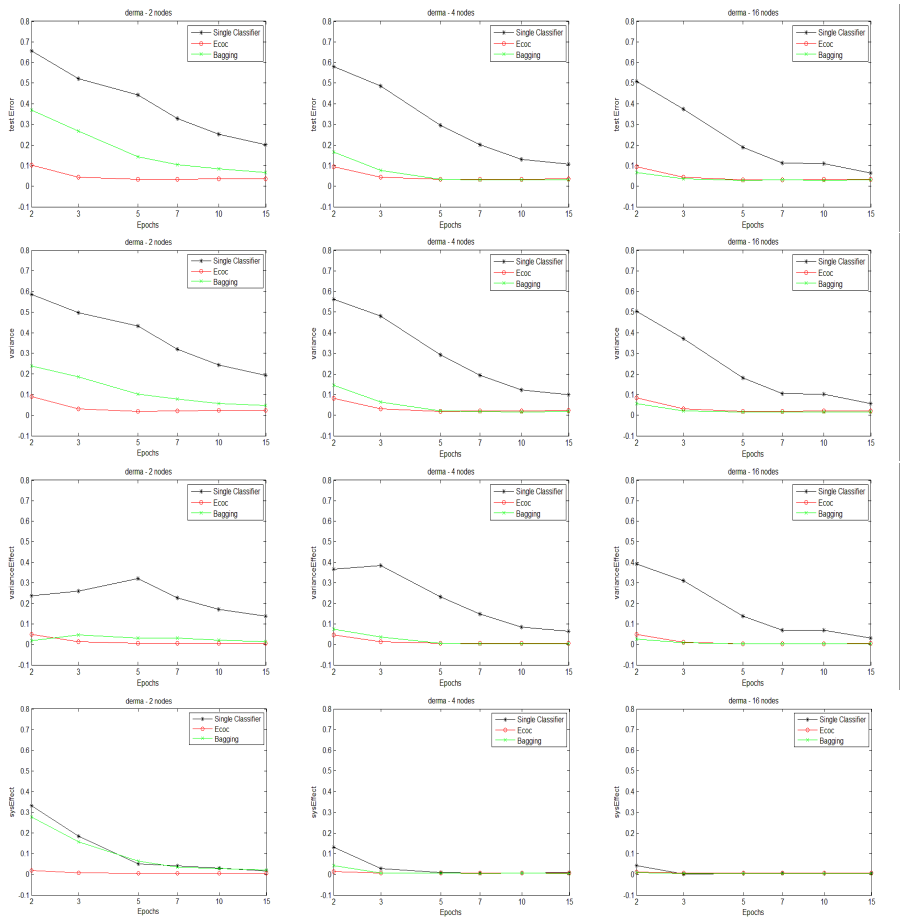


Fig. 2. Bias Variance Analysis for Dermatology data. First Row: Overall prediction error. Second Row: Variance. Third Row: Variance effect. Fourth Row: Systematic effect. First Column: For 2 Nodes. Second Column: For 4 Nodes. Third Column: For 16 Nodes. Black lines indicate the results for single classifier, red for ECOC and green for bagging

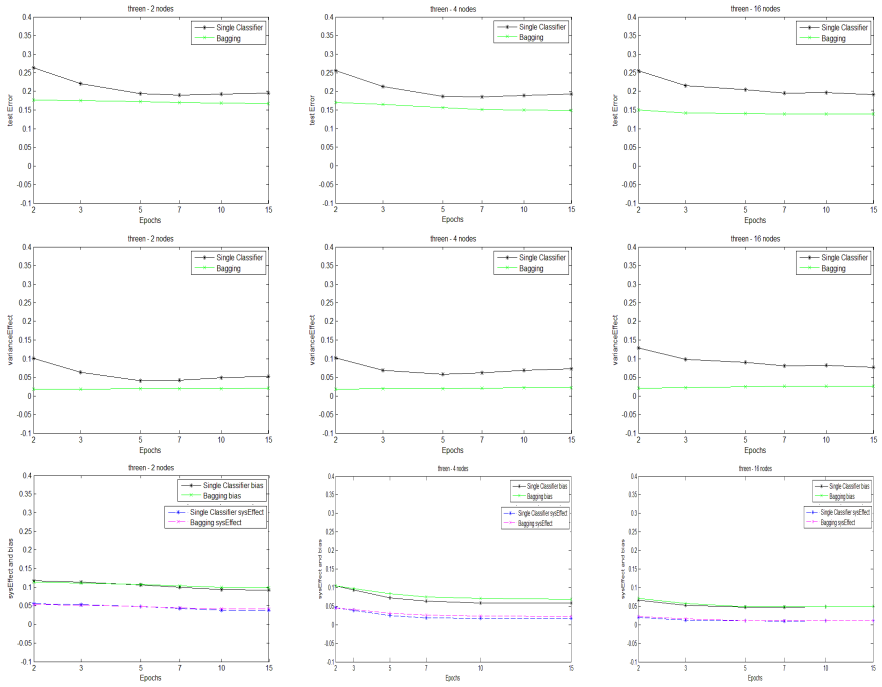


Fig. 3. Bias Variance Analysis for ThreeNorm data. First Row: Overall prediction error. Second Row: Variance effect. Third Row: Systematic effect and Bias. First Column: For 2 Nodes. Second Column: For 4 Nodes. Third Column: For 16 Nodes. Black & blue lines indicate the results for single classifier (bias and systematic effect) and green & magenta for bagging

improve the classification performance in weak networks when single classifiers are used. However, they are still outperformed by ensembles, which have even lower variance effects.

As for further possible advantages of ensembles, the fact that they are expected to avoid overfitting might be shown by using more powerful NNs with higher number of nodes, or other classifiers such as SVMs that are more prone to overfitting. Future work is also aimed at understanding and analysing the bias-variance domain within some mathematical frameworks such as [17] [18] and using the information in the design of ECOC matrices.

References

1. James, G.: Variance and Bias for General Loss Functions, Machine Learning, 51(2), 115–135 (2003)
2. Dietterich, T.G., Bakiri, G.: Solving Multi-class Learning Problems via Error-Correcting Output Codes. J. Artificial Intelligence Research 2. 263–286 (1995)
3. Allwein, E., Schapire, R., Singer, Y.: Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. JMLR 1. 113–141 (2002)

4. Dietterich, T.G., Bakiri, G.: Solving Multi-class Learning Problems via Error-Correcting Output Codes. J. Artificial Intelligence Research 2. 263–286 (1995)
5. Allwein, E., Schapire, R., Singer, Y.: Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. JMLR 1. 113–141 (2002)
6. Schapire, R. E., Freund, Y., Bartlett, P., Lee, W. S.: Boosting the margin: a new explanation for the effectiveness of voting methods. The Annals of Statistics, 26(5):1651–1686 (1998)
7. Domingos. P.: A Unified Bias-Variance Decomposition for Zero-One and Squared Loss. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence, pp. 564–569 (2000)
8. Geman, S., Bienenstock, E., Doursat R.: Neural networks and the bias/variance dilemma, Neural Comput., vol. 4, no. 1, pp. 1-58 (1992)
9. Kong, E. B., Dietterich, T. G.: Error-correcting Output Coding Corrects Bias and Variance. In: Proceedings of the Twelfth International Conference on Machine Learning, pp. 313-321 (1995)
10. Kohavi, R., & Wolpert, D. H.: Bias plus variance decomposition for zero-one loss functions. In: Proceedings Thirteenth International Conference on Machine Learning, pp.275- 283 (1996)
11. Friedman, J. H.: On bias, variance, 0/1 loss and the curse of dimensionality. Data Mining and Knowledge Discovery, 1, pp. 55–77 (1997)
12. Heskes, T.: Bias/Variance Decompostion for Likelihood-Based Estimators. Neural Computation, 10, pp. 1425–1433 (1998)
13. Tibshirani, R.: Bias, variance and prediction error for classification rules. Technical Report, University of Toronto, Toronto, Canada (1996)
14. Smith, R. S., Windeatt, T.: The Bias Variance Trade-Off in Bootstrapped Error Correcting Output Code Ensembles. MCS, pp.1–10 (2009)
15. Domingos, P.: Why does bagging work? A Bayesian account and its implications .Proceedings of the 3rd International Conf. on Knowledge Discovery and Data Mining, pp. 155–158 (1997)
16. Webb, G.I., Conilione, P.: Estimating bias and variance from data. Technical Report, (2005)
17. Tumer, K., Ghosh, J.: Error correlation and error reduction in ensemble classifiers. Connection Science 8 (3-4), pp. 385–403 (1996)
18. Tumer, K., Ghosh, J.: Analysis of decision boundaries in linearly combined neural classifiers. Pattern Recognition, 29(2), pp. 341–348 (1996)
19. Valentini, G., Dietterich, T.: Bias–variance analysis of Support Vector Machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research, vol. 5, pp. 725-775 (2004)
20. Breiman L.: Arcing classifiers. The Annals of Statistics, 26(3), 801-849 (1998)
21. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html. School of Information and Computer Science, University of California, Irvine, CA (2007)
22. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the 13th ICML, pp. 148–156 (1996)
23. Wolpert, D. H.: On bias plus variance. Neural Computation. 9, pp. 1211-1244, (1996)

Compact Evolutive Design of Error-Correcting Output Codes

Miguel Àngel Bautista[1], Xavier Baró[1,2,3], Oriol Pujol[1,2], Petia Radeva[1,2], Jordi Vitrià[1,2], and Sergio Escalera[1,2]

[1]Dept. Matemàtica Aplicada i Anàlisi, Gran Via les Corts Catalanes 585, Barcelona
[2]Computer Vision Center, Campus UAB, Edifici O, 08193, Bellaterra, Barcelona
[3]Department of Computer Science, Multimedia, and Telecomunications, Universitat Oberta de Catalunya, Rambla del Poblenou 156, 08018, Barcelona
miguelangelbautistamartin@gmail.com,
{xevi, oriol, petia, jordi, sergio}@maia.ub.es

Abstract. The classification of large number of object categories is a challenging trend in the Machine Learning field. In literature, this is often addressed using an ensemble of classifiers. In this scope, the Error-Correcting Output Codes framework has demonstrated to be a powerful tool for the combination of classifiers. However, most of the state-of-the-art ECOC approaches use a linear or exponential number of classifiers, making the discrimination of a large number of classes unfeasible. In this paper, we explore and propose a minimal design of ECOC in terms of the number of classifiers. Evolutionary computation is used for tuning the parameters of the classifiers and looking for the best Minimal ECOC code configuration. The results over several public UCI data sets and a challenging multi-class Computer Vision problem show that the proposed methodology obtains comparable and even better results than state-of-the-art ECOC methodologies with far less number of dichotomizers.

Keywords: Ensemble of Dichotomizers, Error-Correcting Output Codes, Evolutionary optimization

1 Introduction

Nowadays challenging applications of Machine Learning deal with changing environments, online adaptations, contextual information, etc. In order to deal with all these problems, efficient ways for processing huge amount of data are often required. Usual machine learning strategies are effective for dealing with small number of classes. The choices are limited when the number of classes becomes large. In that case, the natural algorithms to consider are those that model classes in an implicit way, such as instance based learning (i.e. nearest neighbors). However, this choice is not necessarily the most adequate for a given problem. Moreover, we are forgetting many algorithms of the literature such as ensemble learning (i.e. Adaboost) or kernel based discriminant classifiers (i.e. support vector machines) that have been proven to be very powerful tools.

Most of state-of-the-art multi-class architectures need to deal with the discrimination of each class either by modeling its probability density function, or by storing a classification boundary and using some kind of aggregation/selection

function to obtain a final decision. Another way to deal with this kind of problems is to use a divide-and-conquer approach, such as flat strategies (voting), hierarchical classifiers, or Error-Correcting Output Codes (ECOC). ECOC encodes different partitions of the problem in a matrix of codewords (one codeword per class) and the final decision is obtained by looking at the most similar codeword at the test step. ECOC allows the inclusion of flat strategies as well as hierarchical classifiers [1]. Moreover, the analysis of the ECOC error evolution has demonstrated that ECOC corrects errors caused by the bias and the variance of the learning algorithm [2]. However, note that by construction or in order to obtain the desired performance, most of the strategies need between N and N^2 classifiers, given N different classes. Although this is adequate and acceptable when the number of classes is small, it becomes prohibitive when the number of classes becomes large. This number of classifiers has been recently reduced in some ECOC designs, such as the DECOC approach of [1], that requires $N-1$ classifiers. The Dense Random and Sparse Random designs also reduce this number of classifiers to $15 \cdot log_2(N)$ and $10 \cdot log_2(N)$, respectively. However this kind of approaches design the problems without taking into account the underlying distribution of the class characteristics.

The goal of this paper is to propose and evaluate different general ways of making the multi-class machine learning problem tractable when the number of categories makes most of the models computationally unfeasible. In particular, we are interested in methods that scale sub-linearly with the number of classes, allowing their applicability in general Machine Learning problems. The proposal relies on the Error Correcting Output Codes framework, reducing the number of binary classifiers that have to be trained in the ensemble. Following the Occam razor principle, we propose a minimal ECOC design of size $log_2(N)$ in terms of the number of classifiers. An evolutionary approximation, similar to the one proposed in [3] is proposed for tuning the parameters of the classifiers and looking for a Minimal design with high generalization capabilty. Moreover, this design is problem dependent in the sense that the evolved ECOC fits the distribution of the object characteristics. The novel Minimal ECOC is compared with the state-of-the-art ECOC approaches, obtaining comparable and even better results when classifying several object categories in different Machine Learning applications with far less cost.

The paper is organized as follows: Section 2 presents the Minimal ECOC design. Section 3 evaluates the novel methodology comparing with the state-of-the-art approaches on public and challenging Pattern Recognition Applications. Finally, Section 4 concludes the paper.

2 Minimal Error-Correcting Output Codes

In this section, we review the ECOC framework and propose a Minimal ECOC design in terms of the number of classifiers.

2.1 Error-Correcting Output Codes

Given a set of N classes to be learnt in an ECOC framework, n different bi-partitions (groups of classes) are formed, and n binary problems (dichotomizers)

over the partitions are trained. As a result, a codeword of length n is obtained for each class, where each position (bit) of the code corresponds to a response of a given dichotomizer (coded by +1 or -1 according to their class set membership). Arranging the codewords as rows of a matrix, we define a *coding matrix* M, where $M \in \{-1, +1\}^{N \times n}$ in the binary case. In Figure 1 we show an example of a binary coding matrix M. The matrix is coded using five dichotomizers $\{h_1, ..., h_5\}$ for a 4-class problem $\{c_1, ..., c_4\}$ of respective codewords $\{y_1, ..., y_4\}$. The hypotheses are trained by considering the labeled training data samples $\{(\rho_1, l(\rho_1)), ..., (\rho_m, l(\rho_m))\}$ for a set of m data samples. The white and black regions of the coding matrix M are coded by +1 and -1, respectively. For example, the first classifier is trained to discriminate c_3 against c_1, c_2, and c_4; the second one classifies c_2 and c_3 against c_1 and c_4, etc., as follows:

$$h_1(x) = \begin{cases} 1 & \text{if } x \in \{c_3\} \\ -1 & \text{if } x \in \{c_1, c_2, c_4\} \end{cases}, \ldots, \quad h_5(x) = \begin{cases} 1 & \text{if } x \in \{c_2, c_4\} \\ -1 & \text{if } x \in \{c_1, c_3\} \end{cases} \tag{1}$$

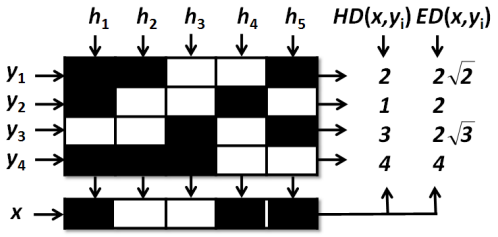


Fig. 1. Binary ECOC design for a 4-class problem. An input test codeword x is classified by class c_2 using the Hamming or the Euclidean Decoding.

The standard binary coding designs are the one-versus-all [4] strategy with N dichotomizers and the dense random strategy [5], with $10 \log_2 N$ classifiers. In the case of the ternary symbol-based ECOC, the coding matrix becomes $M \in \{-1, 0, +1\}^{N \times n}$. In this case, the symbol zero means that a particular class is not considered for a given classifier. In this ternary framework, the standard designs are the one-versus-one strategy [6] and the sparse random strategy [5], with $\frac{N(N-1)}{2}$ and $15 \log_2 N$ binary problems, respectively.

During the decoding process, applying n binary classifiers, a code x is obtained for each data sample ρ in the test set. This code is compared to the base codewords $(y_i, i \in [1, .., N])$ of each class defined in the matrix M, and the data sample is assigned to the class with the *closest* codeword. In Figure 1, the new code x is compared to the class codewords $\{y_1, ..., y_4\}$ using Hamming [4] and Euclidean Decoding [5]. The test sample is classified by class c_2 in both cases, correcting one bit error.

In literature there roughly exists three different lines for decoding [7]: those based on similarity measurements, including the Hamming and Euclidean decoding, probabilistic approaches, and loss-functions strategies.

2.2 Minimal ECOC Coding

Although the use of large codewords was initially suggested in order to correct as many errors as possible at the decoding step, high effort has been put into improving the robustness of each individual dichotomizer so that compact codewords can be defined in order to save time. In this way, the one-versus-all ECOC coding has been widely applied for several years in the binary ECOC framework. Although the use of a reduced number of binary problems often implies dealing with more data per classifier (i.e. compared to the one-versus-one coding), this approach has been defended by some authors in the literature demonstrating that the one-versus-all technique can reach comparable results to the rest of combining strategies if the base classifier is properly tuned [8]. Recently, this codeword length has been reduced to $N-1$ in the DECOC approach of [1], where the authors codify $N-1$ nodes of a binary tree structure as dichotomizers of a ternary problem-dependent ECOC design. In the same line, several problem-dependent designs have been recently proposed [9, 1, 10]. The new techniques are based on exploiting the problem domain by selecting the representative binary problems that increase the generalization performance while keeping the code length "relatively" small.

Although one-versus-all, DECOC, dense, and sparse random approaches have a relatively small codeword length, we can take advantage of the information theory principles to obtain a more compact definition of the codewords. Having a N-class problem, the minimum number of bits necessary to codify and univocally distinguish N codes is $B = \lceil \log_2 N$, where $\lceil . \rceil$ rounds to the upper integer.

For instance, we can think in a codification where the class codewords correspond to the N first Gray or binary code sequences of B bits, defining the Gray or binary Minimal ECOC designs. Note that this design represents the minimal ECOC codification in terms of the codeword length. An example of a binary Minimal ECOC, Gray Minimal ECOC, and one-versus-all ECOC designs for a 8-class problem are shown in Figure 2. The white and black positions correspond to the symbols +1 and -1, respectively.

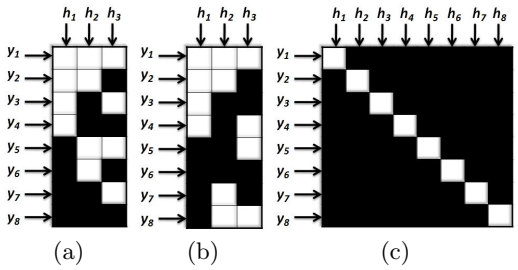


Fig. 2. (a) Binary Minimal, (b) Gray Minimal, and (c) one-versus-all ECOC coding designs of a 8-class problem.

Besides exploring predefined binary or Gray minimal coding matrices, we also propose the design of different minimal codification based on the distribution of the data and the characteristics of the applied base classifier, which can

increase the discrimination capability of the system. However, finding a suitable minimal ECOC matrix for an $N-$class problem requires to explore all the possible $N \times B$ binary matrices, where B is the minimum codeword length in order to define a valid ECOC matrix. For this reason, we also propose an evolutionary parametrization of the Minimal ECOC design.

Evolutionary Minimal Parametrization When defining a minimal design of an ECOC, the possible lost of generalization performance has to be taken into account. In order to deal with this problem an evolutionary optimization process is used to find a minimal ECOC with high generalization capability.

In order to show the parametrization complexity of the Minimal ECOC design, we first provide an estimation of the number of different possible ECOC matrices that we can build, and therefore, the search space cardinality. We approximate this number using some simple combinatorial principles. First of all, if we have an $N-$class problem and B possible bits to represent all the classes, we have a set CW with 2^B different words. In order to build an ECOC matrix, we select N codewords from CW without reposition. That is, taking N from a variation of 2^B elements and considering the symmetry of binary partitions, we can construct $V_N^{2^B} = \frac{2^B!}{2N(2^B-N)!}$ different ECOC matrices.

In this type of scenarios evolutionary approaches are often introduced with good results. Evolutionary algorithms are a wide family of methods that are inspired on the Darwin's evolution theory, and used to be formulated as optimization processes where the solution space is not differentiable or is not well defined. In these cases, the simulation of natural evolution process using computers results in stochastic optimization techniques which often outperform classical methods of optimization when applied to difficult real-world problems. Although the most used and studied evolutionary algorithms are the Genetic Algorithms (GA), from the publication of the *Population Based Incremental Learning* (PBIL) in 1995 by Baluja and Caruana [11], a new family of evolutionary methods is striving to find a place in this field. In contrast to GA, those new algorithms consider each value in the chromosome as a random variable, and their goal is to learn a probability model to describe the characteristics of good individuals. In the case of PBIL, if a binary chromosome is used, a uniform distribution is learnt in order to estimate the probability of each variable to be one or zero.

In this paper we experiment with both evolutionary strategies, GA and PBIL.

Problem encoding: The first step in order to use an evolutionary algorithm is to define the problem encoding, which consists of the representation of a certain solution or point in the search space by means of a *genotype* or alternatively a *chromosome* [12]. Binary encoding is the most common, mainly because first works about GA used this type of encoding. In binary encoding, every chromosome is a string of bits 0 or 1. Each ECOC is encoded as a binary chromosome $\zeta =< h_1^{c_1}, \ldots, h_B^{c_1}, h_1^{c_N}, \ldots, h_B^{c_N} >$, where $h_i^{c_j} \in \{0, 1\}$ is the expected value of the $i-th$ classifier for the class c_j, which corresponds to the $i-th$ bit of the class c_j codeword.

Adaptation function: Once the encoding is defined, we need to define the adaptation function, which associates to each individual its adaptation value to the environment, and thus, their survivor probability. In the case of the ECOC framework, the adaptation value must be related to the classification error.

Given a chromosome $\zeta =< \zeta_0, \zeta_1, \ldots, \zeta_L >$ with $\zeta_i \in \{0, 1\}$, the first step is to recover the ECOC matrix M codified in this chromosome. The values of M allows to create binary classification problems from the original multi-class problem, following the partitions defined by the ECOC columns. Each binary problem is addressed by means of a binary classifier, which is trained in order to separate both partitions of classes. Assuming that there exists a function $y = f(\mathbf{x})$ that maps each sample $\mathbf{x}$ to its real label y, to train a classifier means to find the best parameters w^* of a certain function $y = f'(\mathbf{x}, \mathbf{w})$, in the manner that for any other $\mathbf{w} \neq \mathbf{w}^*$, $f'(\mathbf{x}, \mathbf{w}^*)$ is a better approximation to f than $f'(\mathbf{x}, \mathbf{w})$. Once $\mathbf{w}^*$ are estimated for each binary problem, the adaptation value corresponds to the classification error. In order to take into account the generalization power of the trained classifiers, the estimation of $\mathbf{w}^*$ is performed on a subset of samples, while the rest of the samples are reserved for validation. The adaptation value for an individual represented by a certain chromosome ζ_i can be formulated as:

$$\varepsilon_i(X, Y, M_i) = \sum_j \delta_j(x_j, M_i \neq y_j) \tag{2}$$

where M_i is the ECOC matrix encoded in ζ, $X = \langle x_1, \ldots, x_N \rangle$ a set of samples, $Y = \langle y_1, \ldots, y_N \rangle$ the expected labels for samples in X, and δ_i is the function that returns the classification label applying the decoding strategy.

Evolutionary process: Once the encoding and adaptation function have been defined, we use standard implementation for GA and PBIL, in order to evolve the Minimal ECOC matrices. In the case of GA, scattered crossover operator is used, in which, we generate a random binary vector, with a binary value assigned to each gene. The first child is created using all the genes from the first parent in those positions with a value of one, and the genes of the second parent with positions with the value zero. The second child is created as the complementary of the first one. That is, taking genes from second parent for values one and from first parent for values zero. In order to introduce variations to the individuals, we use mutation operator that adds a unit Gaussian distributed random value to the chosen gene. The new gene value is clipped if it falls outside of the user-specified lower or upper bounds for that gene. For PBIL, the best two individuals of each population are used to update the probability distribution. At each generation, this probability distribution is used to sample a new population of individuals. A uniform random noise is applied to the probability model to avoid convergence to local minima.

Finally, in both evolutionary strategies we adopt an *Island Model* evolution scheme in order to exploit a more coarse grain parallel model. The main idea is to split a population of K individuals into S sub-populations of K/S individuals. If each sub-population is evolved independently from the others, genetic drift will

tend to drive these populations in different directions. By introducing migration, the *Island Model* is able to exploit differences in the various sub-populations (this variation in fact represents a source of genetic diversity). Each sub-population is an island and there is a chance movement of genetic material from one island to another.

Training the binary classifiers: In [8], Rifkin concludes that the number of classifiers in the ECOC problem can be reduced using more accurate classifiers. Therefore, in this paper we adopt the Support Vector Machines with Gaussian Radial Basis Functions kernel (SVM-RBF). Training a SVM often implies the selection of a subset of data points (the support vectors), which are used in order to build the classification boundaries. In the specific case of Gaussian RBF kernels, we need to optimize the kernel parameter γ and the regularizer C, which have a close relation to the data distribution. While the support vectors selection is part of the SVM learning algorithm, and therefore, is clearly defined, finding the best C and γ is addressed in literature with various heuristic or brute-force approaches. The most common approach is the use of cross-validation processes which select the best pair of parameters for a discretization of the parameters space. Nevertheless, this can be viewed as another optimization problem. An therefore, it can be faced using evolutionary algorithms. For each binary problem, defined by one column of the ECOC matrix, we use Genetic Algorithms in order to find good values for C and γ parameters, using the same settings than in [3], where individuals correspond to a pairs of genes, and each gene corresponds to the binary codification of a floating point value.

3 Results

In order to present the results, first, we discuss the data, methods, and evaluation measurements of the experiments.

• *Data*: The first data used for the experiments consists of twelve multi-class data sets from the UCI Machine Learning Repository database [13]. Then, we apply the classification methodology in the public *Labeled Faces in the Wild* [14] data set to perform the multi-class face classification of a large problem consisting of 610 face categories.

• *Methods*: We compare the one-versus-one [6] and one-versus-all [4] ECOC approaches with the binary and evolutionary Minimal approaches. For simplicity we omitted the Gray Minimal design. The Hamming decoding is applied at the decoding step [15]. The ECOC base classifier is the OSU implementation of SVM with Radial Basis Function kernel [16]. The SVM C and γ parameters are tuned via Genetic Algorithms and PBIL for all the methods, minimizing the classification error of a two-fold evaluation over the training sub-set.

• *Evaluation measurements*: The classification performance is obtained by means of a stratified ten-fold cross-validation, and testing for the confidence interval with a two-tailed t-test. We also apply the Friedman test [17] in order to look for statistical significance among the obtained performances.

3.1 UCI categorization

The classification results obtained for all the UCI data sets considering the different ECOC configurations are shown in Table 1. In order to compare the performances provided for each strategy, the table also shows the mean rank of each ECOC design considering the twelve different experiments. The rankings are obtained estimating each particular ranking r_i^j for each problem i and each ECOC configuration j, and computing the mean ranking R for each design as $R_j = \frac{1}{N}\sum_i r_i^j$, where N is the total number of data sets. We also show the mean number of classifiers (#) required for each strategy.

Table 1. UCI classification results.

	Binary Minimal ECOC		Evol. Minimal ECOC		one-vs-all ECOC		one-vs-one ECOC	
Data set	Perf.	Classif.	Perf.	Classif.	Perf.	Classif.	Perf.	Classif.
Derma	96.0±2.9	3	**96.3±2.1**	3	95.1±3.3	6	94.7±4.3	15
Iris	96.4±6.3	2	**98.2±1.9**	2	96.9±6.0	3	96.3±3.1	3
Ecoli	80.5±10.9	3	**81.4±10.8**	3	79.5±12.2	8	79.2±13.8	28
Vehicle	72.5±14.3	2	76.99±12.4	2	74.2±13.4	4	**83.6±10.5**	6
Wine	95.5±4.3	2	**97.2±2.3**	2	95.5±4.3	3	**97.2±2.4**	3
Segment	96.6±2.3	3	96.6±1.5	3	96.1±1.8	7	**97.18±1.3**	21
Glass	56.7±23.5	3	50.0±29.7	3	53.85±25.8	6	**60.5±26.9**	15
Thyroid	**96.4±5.3**	2	93.8±5.1	2	95.6±7.4	3	96.1±5.4	3
Vowel	57.7±29.4	3	**81.78±11.1**	3	80.7±11.9	8	78.9±14.2	28
Balance	80.9±11.2	2	87.1±9.2	2	89.9±8.4	3	**92.8±6.4**	3
Shuttle	80.9±29.1	3	83.4±15.9	3	**90.6±11.3**	7	86.3±18.1	21
Yeast	50.2±18.2	4	**54.7±11.8**	4	51.1±18.0	10	52.4±20.8	45
Rank & #	2.9	**2.7**	**2.0**	**2.7**	2.7	5.7	2.2	15.9

In order to analyze if the difference between method ranks is statistically significant, we apply a statistical test. In order to look if the measured ranks differ from the mean rank we use the Friedman test. The Friedman statistic value is computed as $X_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right]$. In our case, with $k = 4$ ECOC designs to compare, $X_F^2 = -4.94$. Since this value is undesirable conservative, Iman and Davenport proposed a corrected statistic: $F_F = \frac{(N-1)X_F^2}{N(k-1)-X_F^2}$. Applying this correction we obtain $F_F = -1.32$. With four methods and twelve experiments, F_F is distributed according to the F distribution with 3 and 33 degrees of freedom. The critical value of $F(3, 33)$ for 0.05 is 2.89. As the value of F_F is no higher than 2.98 we can state that there are no statistical different among the ECOC performances. This means that all four strategies are suitable in order to deal with multi-class categorization problems. This result is very satisfactory and encourages the use of the Minimal approach since similar (or even better) results can be obtained with far less number of classifiers. Moreover, the GA evolutionary version of the Minimal design improves in the mean rank to the rest of classical coding strategies, and in most cases outperforms the binary Minimal approach in the present experiment. This result is expected since the evolutionary version looks for a minimal ECOC matrix configuration that minimizes the error over the training data. In particular, the advantage of the evolutionary version over the binary one is more significant when the number of classes increases, since more minimal matrices are explored, and hence, on an average, it is capable of finding a better solution.

On the other hand, possible reasons why the evolutionary Minimal ECOC design obtains similar or even better performance results than the one-versus-one and one-versus-all approaches with far less number of dichotomizers can be

the few classifiers considered for tuning, and the use of all the classes in balanced binary problems, which can help the system to increase generalization if a good decision boundary can be found by the classifier. Note that the one-versus-one classifier looks for binary problems that split just two classes. In those cases, though good and fast solutions could be found in training time, the use of less data does not assure a high generalization capability of the individual classifiers.

In terms of testing time, since all the trained classifiers spend the same time for testing, classification time is proportional to the number of trained classifiers. The mean number of dichotomizers used for each strategy is shown in the last row of Table 1. Observe the great difference in terms of the number of classifiers between the minimal approaches and the classical ones. The Minimal approaches obtain an average speed up improvement of 111% respect the one-versus-all approach in testing time. Meanwhile in the case of the one-versus-one technique this improvement is of 489%.

In the next section we test if the same behavior occurs classifying a challenging Computer Vision problem with several object categories.

3.2 Labeled Faces in the Wild categorization

This data set contains 13000 faces images taken directly from the web from over 1400 people. This images are not constrained in terms of pose, light, occlusions or any other relevant factor. For the purpose of this experiment we used a specific subset, taking only the categories which at least have four or more examples, having a total of 610 face categories. Finally, in order to extract relevant features from the images, we apply an Incremental Principal Component Analysis procedure [18], keeping 99.8% of the information. An example of face images is shown in 3.

Fig. 3. Labeled Faces in the Wild data set.

The results in Table 2 show that the best performance is obtained by the Evolutionary GA and PBIL Minimal strategies. One important observation is that Evolutionary strategies outperform the classical one-versus-all approach, with far less number of classifiers (10 instead of 610). Note that in this case we omitted the one-vs-one strategy since it requires 185745 classifiers for discriminating 610 face categories.

Table 2. Labeled Faces in the Wild data set classification results.

	Binary M. ECOC		GA M. ECOC		PBIL M. ECOC		one-vs-all		one-vs-one	
Data set	Perf.	#	Perf.	#	Perf.	#	Perf.	#	Perf.	#
FacesWild	26.4±2.1	10	**30.7±2.3**	10	30.02.4±	10	25.0±3.1	610	-	185745

4 Conclusion

We presented a general methodology for the classification of several object categories which only requires $\lceil \log_2 N \rceil$ classifiers for a N-class problem. The methodology is defined in the Error-Correcting Output Codes framework, designing a minimal coding matrix in terms of dichotomizers which univocally distinguish N codes. Moreover, in order to speed up the design of the coding matrix and the tuning of the classifiers, evolutionary computation is also applied.

The results over several public UCI data sets and a challenging multi-class Computer Vision problem with several object categories show that the proposed methodology obtains statistically equivalent results as the state-of-the-art ECOC methodologies with far less number of dichotomizers. For example, the Minimal approach trained 10 classifiers to split 610 face categories, meanwhile the one-versus-all and one-versus-one approaches required 610 and 185745 dichotomizers, respectively.

References

1. O. Pujol, P. Radeva, J. Vitrià, Discriminant ECOC: A heuristic method for application dependent design of error correcting output codes, in: Trans. on PAMI, Vol. 28, 2006, pp. 1001–1007.
2. T. Dietterich, E. Kong, Error-correcting output codes corrects bias and variance, in: ICML (Ed.), S. Prieditis and S. Russell, 1995, pp. 313–321.
3. A. C. Lorena, A. C. de Carvalho, Evolutionary tuning of svm parameter values in multiclass problems, Neurocomputing 71 (16-18) (2008) 3326 – 3334.
4. M. Pelikan, D. E. Goldberg, Learning machines, in: McGraw-Hill, 1965.
5. E. Allwein, R. Schapire, Y. Singer, Reducing multiclass to binary: A unifying approach for margin classifiers, in: JMLR, Vol. 1, 2002, pp. 113–141.
6. T.Hastie, Classification by pairwise grouping, NIPS 26 (1998) 451–471.
7. S. Escalera, O. Pujol, P. Radeva, On the decoding process in ternary error-correcting output codes, PAMI 99 (1).
8. R. Rifkin, A. Klautau, In defense of one-vs-all, JMLR 5 (2004) 101–141.
9. K. Crammer, Y. Singer, On the learnability and design of output codes for multiclass problems, in: Machine Learning, Vol. 47, 2002, pp. 201–233.
10. S. Escalera, O. Pujol, P. Radeva, Error-correcting output codes library, Journal of Machine Learning Research 11 (2010) 661–664.
11. S. Baluja, R. Caruana, Removing the genetics from the standard genetic algorithm, in: ICML, 1995, pp. 38–46.
12. J. Holland, Adaptation in natural and artificial systems: An analysis with applications to biology, control, and AI, University of Michigan Press, 1975.
13. A. Asuncion, D. Newman, UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 2007.
14. G. B. Huang, M. Ramesh, T. Berg, E. L. Miller, Labeled faces in the wild, Tech. Rep. University of Massachusets Amherst, 07-49 (October 2007).
15. T. Dietterich, G. Bakiri, Solving multiclass learning problems via error-correcting output codes, in: JAIR, Vol. 2, 1995, pp. 263–286.
16. OSU-SVM-TOOLBOX, http://svm.sourceforge.net/.
17. J. Demsar, Statistical comparisons of classifiers over multiple data sets, JMLR 7 (2006) 1–30.
18. W. Hwang, J. Weng, Y. Zhang, Candid covariance-free incremental principal component analysis, PAMI 25 (8) (2003) 1034–1040.

Combining Expertise-Driven and Semi-Supervised Bayesian Networks for Classification of Early Glaucoma

Stefano Ceccon[1], David Garway-Heath[2], David Crabb[3], and Allan Tucker[1]

[1] Department of Information Systems and Computing, Brunel University, Uxbridge UB8 3PH, London, UK,
stefano.ceccon@brunel.ac.uk,
[2] NIHR Biomedical Research Centre for Ophthalmology, Moorfields Eye Hospital NHS, EC1V 2PD, London, UK
[3] Department of Optometry and Visual Science, City University, EC1V 0HB, London, UK

Abstract. Bayesian Networks (BNs) are probabilistic graphical models that are popular in numerous fields. Here we propose these models to improve the classification of glaucoma, a major cause of blindness worldwide. We use visual field and retinal data to predict the early onset of glaucoma. In particular, the ability of BNs to deal with missing data and encode prior expert knowledge allows us to compare a visual field semi-supervised network and an expertise-driven network. Two BN-based combinations of these networks are also proposed and compared to the others. The best performances are obtained with the semi-supervised data-driven network, however combining it with the expertise-driven network improves performance in many cases and leads to interesting insights about the datasets, networks and metrics.

Keywords: Bayesian networks, glaucoma, AGIS, visual field, classification, ensembles of classifiers, combining classifiers

1 Introduction

Glaucoma is the second cause of blindness worldwide [17], but its underlying mechanisms are still not clear. However, early treatment has been shown to slow the progression of the disease, thus early diagnosis is desirable [22]. To this purpose, several medical instruments available nowadays provide a large amount of anatomical and functional data, which is exploited using statistical and A.I. techniques. Our study aims to set up and combine Bayesian Network (BNs) classifiers in order to obtain more precise early diagnosis of glaucoma and to learn insights from the models. BNs are models which seem to be appropriate for this issue, being able to integrate different datasets and model expert knowledge in the field [16]. Moreover, their ability in handling missing data is very useful in the context of glaucoma because no gold standard disease detection method is available. BNs are white-box models, so that it is possible to look at the

underlying relations of the model to improve knowledge in the field. BNs have already been successfully tested with glaucoma data in [19]. In the first stage of this study, two different BN models are obtained using the Advanced Glaucoma Intervention Study (AGIS) scoring system [7] as the class variable, and imposing an expertise-driven network based on the anatomy of the optic nerve. The AGIS defect scoring system depends on the number and depth of clusters of adjacent depressed test sites in the total deviation printout of the threshold program visual field (VF) test STATPAC-2 analysis. The VF test aims to measure the functional ability of an eye by exposing different stimulus to different locations of the patient field and it's a routine test when screening for glaucoma. The anatomy-based network is obtained by modelling the relations between sectors of the optic nerve head (ONH) and the VF sectors as proposed by [10]. This is a widely used structure-function map which has been proved to be correlated with previous studies. The second stage of this study aims to combine the results of the two BN classifiers in order not only to maximize the results, but also to learn new insights about how the two different approaches interact. Two techniques are explored: a BN combiner with 3 nodes and a more classical weighted vote technique with accuracy-based weights. A BN combiner weighted on accuracy is also proposed in this section. In the last part of the study the results of the different models are presented and discussed in terms of performance and qualitative outcome to better understand glaucoma and the associated clinical metrics.

2 Methods

2.1 Datasets

In this study two independent datasets were used (Table 1). Dataset A is a cross-sectional dataset of 78 early glaucomatous patients and 102 healthy control subjects. Inclusion criteria for early glaucomatous patients were Intraocular Pressure (IOP) greater than 21 mmHg and early VF defects on at least 3 occasions. Control subjects had IOP $<$ 21 mmHg and known to be healthy. Dataset B is a longitudinal dataset of 19 controls and 43 patients from Ocular Hypertensive Treatment group, who developed glaucoma in the time span observed. Initial eligibility criteria were in this case IOP $>$ 21 mmHg and negative VF test, and conversion was defined as positive AGIS score on 3 consecutive tests. Control subjects had negative VF test in 2 tests and IOP $<$ 21 mmHg. Data consists of VF point sensibility obtained with Humphrey Field Analyzer II and retinal sector-based parameters data from Heidelberg Retina Tomograph. Retinal data was pre-processed for both datasets by applying the 95% prediction interval MRA regression equation as indicated in [9], which is a linear combination of Age, Optic Disc Area (ODA) and Retinal Rim Area (RRA) into one single parameter. Sensibility values were grouped in six sectors as suggested in [10] for computational and simplicity reasons in correspondence with retinal parameters.

Table 1. Characteristics of the datasets used in the study

	Dataset A	Dataset B
Control Subjects (values)	102 (102)	19 (155)
Converters (values)	78 (78)	43 (474)
Total Subjects (values)	180 (180)	62 (629)
Mean Age (controls)	67.6 (57.5)	65.7 (66.7)

2.2 Bayesian Networks

BNs are probabilistic directed graphical models in which each node represents a variable, and lack of arcs between nodes represents conditional independence assumption. Each variable is associated with a conditional probability distribution (CPD), so that given the set of all CPDs in the network it is possible to inference about any value of any node. For classification, BNs are used by selecting the most probable value of the class node (i.e. glaucomatous vs healthy) given the values observed for all the other nodes. To build a BN, the CPDs of all the variables need to be estimated from data. This is typically obtained using the Maximum Likelihood Estimation (MLE), which consists in maximizing the likelihood of the data given the parameters. In case of missing data, however, this technique cant be used. Therefore, the Expectation-Maximization algorithm (EM) was used for estimating the parameters in the unsupervised network. This technique iteratively estimates the hidden values for the unobserved data in the first step and maximizes the estimated likelihood function at the next step. When the algorithm converges to a local maximum, the parameters are estimated [3]. The structure of the network (i.e. the links between nodes) must be chosen as well: it can be either learned from the data or imposed using expert knowledge. The learning algorithm used in this study was a Simulated Annealing (SA) technique, which is a quasi-greedy algorithm that searches through the space of possible structures to obtain the optimal structure [14]. To explore different structures 2100 operations were carried by the algorithm, so that in each iteration a link was added, removed or two nodes were swapped. The algorithm was repeated 100 times using bootstrapping on the dataset [5]. This resampling technique has been widely used in order to reduce noise and obtain more precise results. The score used to select the best structure after each operation was the Bayesian Information Criterion (BIC) score, which is a metric based on the likelihood of the observed data given the structure and the parameters, with a penalizing factor related to the number of parameters that aims to prevent overfitting. Bayesian Model Averaging was used on the learned structures in order to calculate the posterior probability for each arc as proposed in [6], i.e. by weighting each arc on the network score and a factor related to the complexity of the network. The arcs with highest posterior probability were then selected to obtain the final structure. In the first stage of this study, two structures were learned using dataset A. The first structure was learned using only control subjects (Fig. 1). The rationale is to model the healthy controls

in order to avoid the potential bias introduced by the scoring metric AGIS. In fact, the AGIS score is based only on visual field test and its performance seems insufficient especially for early glaucoma, therefore a bias is introduced by supervising the learning process just on it [22], [11], [2]. Learning the network only on healthy subjects provides a more reliable framework, which can then be combined with AGIS score by training the parameters of the network on AGIS labeled data. This approach can be seen as a form of semi-supervised learning based on control subjects using the AGIS score (which can be seen to accurately identify controls). This network appears to perform better in classification than the complete AGIS supervised learned networks or the controls supervised one. A second stage involved the exploration of expert knowledge modeling and its performance in classification. To this purpose, we modeled the relations showed in the structure-function map proposed in [10] imposing arcs between correspondent sector-based VF values and retinal values. The network was then trained on unsupervised data, in order to be independent on all clinical metrics. The aim of using this BN is to assess the different results obtained with a network based only on prior anatomical knowledge, and to explore whether its possible to understand more about glaucoma and to improve classification performance together with the data-driven network.

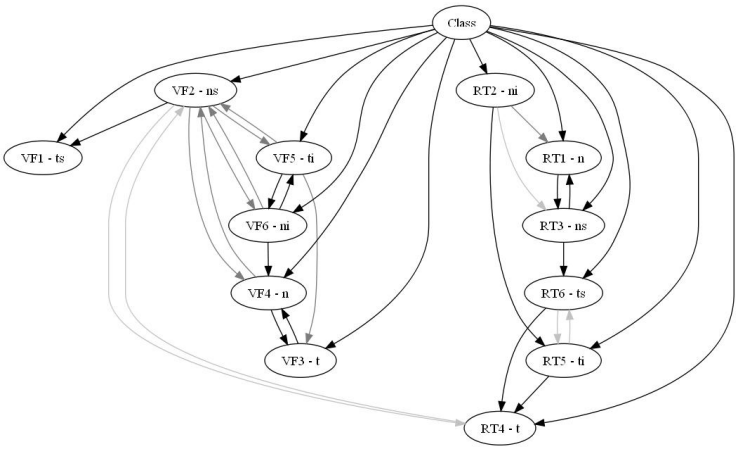


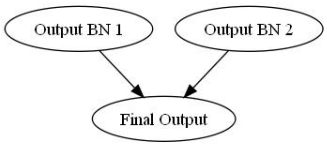
Fig. 1. Structure learned on dataset A using Simulated Annealing on controls subjects. Lighter arcs represent lower probabilities.

2.3 Combining Networks

The natural next step was thus to combine the two networks described above, to exploit base classifiers outputs and obtain the best performing and illustrative results. In fact the data-driven network seems to be better at excluding

negative patients than the anatomy based, whilst the latter seems to produce often different results, having a higher sensitivity at low specificities. There is a broad literature on ensembles of classifiers [15], [18] and given the input available (i.e. probabilistic outputs from base BNs), a non-generative stacked structure ensemble was chosen [4]. Non-generative ensemble methods try to combine existing base classifiers, without acting on the base classifiers structure. Stacked ensembles use Machine Learning techniques on the top of the base learners, using their output as a set of input data for the final combiner. In particular, since the aim is not just in terms of performance but also in understanding how data and anatomy driven networks interacts to improve results, a BN model was chosen. This type of BN has been proved to be an effective combiner in [8], although improved performances can be obtained only if the individual classifiers disagree with each other [12]. A third BN was then imposed to combine the outputs from the other two networks (Fig. 2A): each output was discretized in 8 states and then linked to a 2 states class node. As a result, an 8x8 matrix showing the interactions between the two networks was obtained by observing the CPD of the class node (Fig. 2B). In order to optimize the CPD, the matrix was then smoothed using a mean-filter window of size 3. For validation purpose, a combination of the outputs was also performed using a weighted voting approach [21]. The weights were adjusted in relation to the accuracy of the base networks. An optimized combining BN was also set up by weighting its output with the

A



B

Low values → High values

Controls and AGIS BN discretized outputs (High values)	Anatomy-based BN discretized outputs (Low values → High values)							
	0.27	0.15	0.25	0.33	0.33	0.00	0.00	0.00
	0.47	0.33	0.09	0.64	0.50	0.43	0.40	0.00
	0.58	0.75	0.52	0.77	0.67	0.92	1.00	0.00
	1.00	0.60	0.80	0.89	0.85	0.95	0.73	1.00
	1.00	1.00	1.00	0.89	0.94	1.00	0.94	0.75
	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.79
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.88
	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00

Fig. 2. A. Combining Bayesian Network. B. Instance of a raw CPD of the class node. Each value represents the probability of positive output for each combination of base classifiers inputs (i.e. the final output of the combining network).

accuracy of the base networks. In particular, the output of the BN combined network was weighted on the accuracies of each base BN classifier as follows:

```
> for i = 1:size(TestDataset)

>    Weighted_output(i) = (output_BN1(i) * accuracy_BN1(i) +
                          output_BN2(i) * accuracy_BN2(i)) /
                          sum(accuracy_BN1, accuracy_BN2)

>    Final_output(i) = mean(output(i), Weighted_output(i))

> end
```

The probabilistic outputs is in this way biased towards the output provided by the most accurate base classifier.

3 Results

Results were evaluated using 3-fold cross validation on dataset A and 6-fold cross validation on dataset B. Different folds reflect the different sizes of the datasets. The performance of the single and the combined networks are shown in Table 2 and Fig. 3. Same tests were performed considering only the pre-diagnosis data subset of dataset B, in order to take into account only patients converting to glaucoma. This leaded to results qualitative very similar to those in Table 2. Considering the overall performances on the full datasets A and B,

Table 2. Performances of different BNs tested in terms of mean AUROC and total errors at maximum accuracy and at 90% specificity.

	Dataset A			Dataset B		
	AUROC	Errors	Errors at 90% spec	AUROC	Errors	Errors at 90% spec
Semi-Supervised BN	0.98	6	13	0.87	84	206
Anatomy-based BN	0.98	7	13	0.75	110	326
Weighted Vote Combined	0.98	5	13	0.84	90	252
BN-based Combined	0.98	8	12	0.87	93	186
Accuracy Weighted BN	0.98	5	12	0.85	93	118

the semi-supervised data-driven BN performs better than the expertise-driven based one and at least comparable to the combined ones. On dataset A, however, the performances of all the classifiers are comparable. Among the combined classifiers, the Weighted Vote is outperformed only at high specificities, but in dataset B its outperformed by both the other combined networks. The BN combined networks perform well in both datasets, the accuracy weighted one

being better on dataset A but worse on dataset B. Their performances were the best for higher specificities. Looking at the ROC curves in a particular test on dataset B it can be seen that the performances of the base semi-supervised data-driven BN are the highest at mid-specificities. However, at higher specificity, the BN combined networks are comparable or outperform it, as showed also in Table 2. On dataset A the performance of the two base BNs are more similar and often the expertise-driven BN outperforms the other (Fig. 4). The Weighted Vote classifier performs better at low specificities as pointed out considering the total errors.

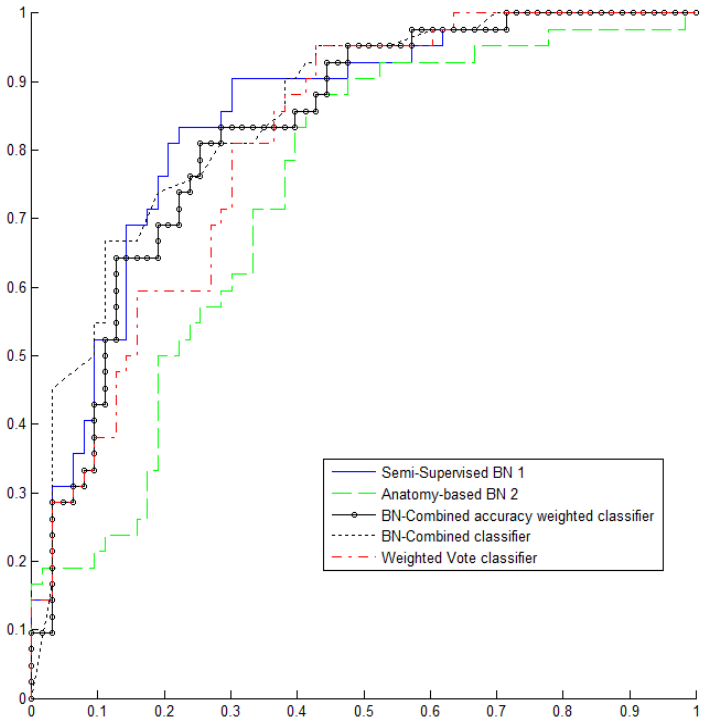


Fig. 3. ROC curves of classifiers in a test with 6-fold cross validation on dataset B.

4 Discussion

The semi-supervised network is clearly performing very well at all specificities. The conservative nature of AGIS metric score and the idea of modeling controls show the effectiveness of data and BNs in classification. However, the bias introduced by using AGIS score in the inclusion criteria and its unreliability must be

kept in mind, especially for dataset B where conversion diagnosis is a more difficult task and it's defined using the AGIS score. The expertise-driven network on the other hand seems not to perform as well as the other. However, variability was found in the results obtained, so that improvements in performances can be theoretically obtained with a combination of them. In fact, as pointed out in [20], diversity and accuracy of base classifiers are key factors for better performing ensembles of classifiers. Diversity (and correlated independency) was obtained by using an unsupervised approach on the expertise-network. This consequently decreased the performances of the base classifier but increased those of the combined ones in many cases, especially at higher specificities. Performances at different specificities are of great interest. For glaucoma, which is a relative low frequency disease, high specificities are preferred. This reflects the higher weight put on the FP to minimize its occurrence. In this context, the performances of the accuracy Weighted Vote classifier decrease when the specificity increases. This occurs also in the BN combining network weighted on the accuracy, as expected. On the other hand, the simple BN combining classifier is not biased toward the accuracy and this allows the classifier to outperform all the others at high specificities. In fact, looking at the ROC curves it can be seen that with an increase of accuracy of the anatomy-based network there is a decrease in performance of the accuracy weighted ones. In this particular case this is due to the expertise-driven network that doesnt outperform the other at any specificity. However, on dataset A, the opposite situation was observed. In Fig. 4 a ROC curve is showed for dataset A, showing the higher performances of the BN Accuracy-Weighted classifier with respect to the non-weighted one. The

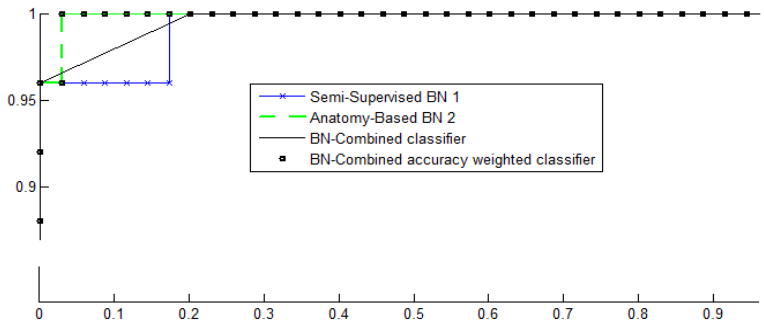


Fig. 4. Top-left zoom of a ROC graph for a 3-fold cross validation test on dataset A.

higher number of total errors in Dataset A could therefore be explained by the differences in the datasets and the different performances on them: on dataset A the diversity between the results of the base classifiers is lower than in dataset B, leading to worse combined performance. Therefore, a single BN achieved strong results on dataset A (being more easy to classify), so that adding a weaker BN

didn't lead to any improvement. This highlights the importance of choosing the most efficient base classifiers, and could lead to further study in generative ensembles of classifiers (i.e. active learning of base classifiers to improve diversity and accuracy of base classifiers) [13]. It must be pointed also out that the structures of the base classifiers were learned and trained on dataset A, introducing a bias with respect to the performances. Again, AGIS score was in the inclusion criteria for both datasets, adding difficulties in comparing performances of classifiers supervised on AGIS itself or unsupervised. The difference in performances obtained with the two base classifiers and the two datasets points out another key point about datasets and a strength of combining them. Data is very noisy due to the absence of a gold standard and to the high variability of the measurements [22], therefore a generalized algorithm that accords itself to the best performing network independently on the data is desirable. This seems not correctly obtained using accuracy weight, as the training and the testing dataset can be very different: considering an example, if dataset A is used to train an accuracy-based network and dataset B is used to test it, results will be insufficient as the accuracies are not similar for the same networks in both datasets. The idea here would be to use an independent dataset of the same type (cross-sectional or longitudinal) for training and testing: using cross-sectional data to build models used with longitudinal data is often not advisable [1]. Further, a broader search on both datasets for a network that shows more accurate and diverse performances than the other could lead in this case to better results for both datasets. An interesting advantage offered by the BN combining classifiers is the possibility to observe at the CPD for the class node. This gives us interesting insights about how the network works and how the two outputs combine to obtain better results. In Fig. 3B a CPD is showed: for each discretized value of the base classifiers a probability is learned from the data and can be observed. This reflects the output of the final classifier, in particular the probability of diagnosis of glaucoma for each combination of inputs. Observing the CPDs obtained in an instance of a CPD in Fig. 2, the higher values in the matrix are skewed towards the bottom-left, i.e. the AGIS semi-supervised network is more reliable at high specificities (for all values of the expertise-driven network, if the data-driven network's output is low then the combined output must be low). Some 0 values are obtained in the corners of the matrix, due to the lack of data for these combinations of outputs: these cells are smoothed with the mean-filter application. Looking at lower values of the data-driven base network (e.g. value 3), the output of the expertise-driven network increases the probability of glaucoma sensibly, adding its knowledge to the result. Several different instances of this matrix have been found in this study, showing again variability between different datasets used. Thus, the exploration of the CPD of the combined network confirms the higher performances at high specificity of semi-supervised data-driven network, but also gives a quantitative measure of the improvement that each single base classifier brings to the final output. Further study using this approach will be carried in the future, for example by acting on the CPDs or

averaging on them, as well as using different discretization and more complete datasets.

5 Conclusion

This preliminary study has showed the possibilities of using BNs for classification and exploration of different dataset in a real problem. Data-driven BN classifier outperformed the expertise-driven one, even if with different magnitude on different datasets. Combining networks have been showed to be both effective in terms of performances and illustrative. In particular, the performances of the combined classifier seem to be better for datasets with more different results for the base classifiers. Given the issues of the datasets, the idea of using simulated data for future work could give a start to more deep analysis of BNs potential in modeling data and combining classifiers, keeping in mind the importance of an effective choice of the base classifiers.

References

1. Artes, P.H., Chauhan, B.C.: Longitudinal changes in the visual field and optic disc in glaucoma. Progress in retinal and eye research 24(3), 333–354 (2005)
2. Chauhan, B.C., Drance, S.M., Douglas, G.R.: The use of visual field indices in detecting changes in the visual field in glaucoma. Investigative ophthalmology & visual science 31(3), 512 (1990)
3. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society.Series B (Methodological) 39(1), 1–38 (1977)
4. Duin, R., Tax, D.: Experiments with classifier combining rules. Multiple Classifier Systems pp. 16–29 (2000)
5. Efron, B., Tibshirani, R., Tibshirani, R.J.: An introduction to the bootstrap. Chapman & Hall/CRC (1993)
6. Friedman, N., Koller, D.: Being bayesian about network structure. a bayesian approach to structure discovery in bayesian networks. Machine Learning 50(1), 95–125 (2003)
7. Gaasterland, D.E., Ederer, F., Sullivan, E.K., Caprioli, J., Cyrlin, M.N.: Advanced glaucoma intervention study. 2. visual field test scoring and reliability. Ophthalmology 101(8), 1445–1455 (1994)
8. Garg, A., Pavlovic, V., Huang, T.S.: Bayesian networks as ensemble of classifiers. Urbana 51, 61801 (2002)
9. Garway-Heath, D.F.: Moorfields regression analysis. The Essential HRT Primer.San Ramon, California: Jocoto Advertising Inc p. 3139 (2005)
10. Garway-Heath, D.F., Poinoosawmy, D., Fitzke, F.W., Hitchings, R.A.: Mapping the visual field to the optic disc in normal tension glaucoma eyes. Ophthalmology 107(10), 1809–1815 (2000)
11. Goldbaum, M.H., Sample, P.A., White, H., Colt, B., Raphaelian, P., Fechtner, R.D., Weinreb, R.N.: Interpretation of automated perimetry for glaucoma by neural network. Investigative ophthalmology & visual science 35(9), 3362 (1994)

12. Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(10), 993–1001 (1990)
13. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural computation 3(1), 79–87 (1991)
14. Kirkpatrick, S., Jr, C.D.G., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671 (1983)
15. Kittler, J.: Combining classifiers: A theoretical framework. Pattern Analysis & Applications 1(1), 18–27 (1998)
16. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (1988)
17. Resnikoff, S., Pascolini, D., Etya'ale, D., Kocur, I., Pararajasegaram, R., Pokharel, G.P., Mariotti, S.P.: Global data on visual impairment in the year 2002. Bulletin of the World Health Organization 82, 844–851 (2004)
18. Sharkey, A.J.C.: On combining artificial neural nets. Connection Science 8(3), 299–314 (1996)
19. Tucker, A., Vinciotti, V., Liu, X., Garway-Heath, D.: A spatio-temporal bayesian network classifier for understanding visual field deterioration. Artificial Intelligence in Medicine 34(2), 163–177 (2005)
20. Valentini, G., Masulli, F.: Ensembles of learning machines. Neural Nets pp. 3–20 (2002)
21. Woods, K., Bowyer, K., Jr, W.P.K.: Combination of multiple classifiers using local accuracy estimates. In: 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96. pp. 391–396 (1996)
22. Yanoff, M., Duker, J.S.: Ophthalmology 2nd Edition. Mosby, St. Louis (August 25 2003)

Hierarchical Mixture of Random Prototype-based Experts

Giuliano Armano and Nima Hatami

DIEE-Department of Electrical and Electronic Engineering,
University of Cagliari, Piazza D'Armi, I-09123 Cagliari, Italy
{armano, nima.hatami}@diee.unica.it

Abstract. The Hierarchical Mixture of Experts (HME) is a well-known tree-structured architecture which can be considered a natural extension of the Mixture of Experts (ME) model. The standard HME model hierarchically splits the input space into a *nested* set of subspaces, the gating networks being used to specialize the experts on each subspace. Recently, data splitting based on random prototypes has been proposed to increase the diversity of classifier ensembles. In this paper, instead of stochastically partitioning the input space, we propose to split the space into centralized regions derived from randomly selecting some prototypes from the data points available for training. The training and testing strategies of the standard HME are modified accordingly. In particular, the proposed approach does not require to train the gating networks as they are implemented with simple distance-based rules. This way, the overall time required for training an HME is considerable lower. It is also worth pointing out that centralizing input subspaces and adopting a random strategy for selecting prototypes permits to increase at the same time individual accuracy and diversity of HME modules, which in turn increases the accuracy of the overall ensemble. Experiments are performed on an artificial toy problem and on selected datasets from the UCI machine learning repository and obtained results point to the robustness of the proposed method compared to the standard HME model.

Keywords: Classifier ensembles, Hierarchical Mixture of Experts (HME), Neural Networks, and Random prototype-based data splitting.

1 Introduction

Most real-world pattern recognition problems are too complicated for a single classifier to solve. Divide-and-conquer has proved to be efficient in many of these complex situations, using a combination of classifiers which have complementary properties. The issues are (i) how to divide the problem into simple subproblems, (ii) how to assign base classifiers to solve these subproblems, and (iii) how to obtain the final decision using their outputs.

If it is possible to naturally decompose the problem then this can be done manually. However, in most real-world problems, we either know too little about the problem, or it is too complex to have a clear understanding of how to manually decompose it into subproblems. Thus, a method for automatically decomposing a complex problem into a set of overlapping or disjoint subproblems is desirable,

assigning one or more classifiers (experts hereinafter) to each subproblem. The remaining question is how to combine the outputs of these experts if the decomposition scheme is unknown a priori.

Jacobs et al. [6, 7] have proposed an ensemble method based on the divide-and-conquer principle called Mixture of Experts (ME), in which a set of networks referred to as *expert networks* is trained together with a *gate network*. This tight coupling mechanism (i) encourages diversity between the single experts by automatically localizing them in different regions of the input space and (ii) achieves good combination weights of the ensemble members by training the gate which computes the dynamic weights together with the experts.

The Hierarchical Mixture of Experts (HME) [8] is a well-known tree-structured architecture, which can be thought of as a natural extension of the Mixture of Experts (ME) model. The expert networks form the leaves of the tree, whereas gating networks are located at the branch-points. Tasks are approached using a "recursive" divide-and-conquer strategy: complex tasks are decomposed into subtasks which in turn are themselves decomposed into sub-subtasks. Like many known classical neural network ensemble methods, diversity in the standard HME is promoted by randomly initializing their weight parameters. This choice drives experts to start learning their task from different points in the search space, with the goal of getting them specialized on different subspaces.

Many of the earlier works on the ME and HME models use preprocessing to partition or transform the input space into simpler and more separable spaces. An expert is then specialized on each subspace without altering the learning rules established by the standard ME model. As a consequence, the major effort in earlier works has been spent in the task of increasing the individual accuracy of experts –so to facilitate their task on the corresponding areas of expertise. Waterhouse and Cook [13] and Avnimelech and Intrator [3] proposed to combine ME with the boosting algorithm. Since boosting encourages classifiers to become experts on patterns that previous experts disagree on, it can be successfully used to split the data set into regions for the experts in the ME model, thus ensuring their localization and diversity. Tang et al. [11] tried to explicitly "localize" experts by applying a cluster-based preprocessing step, headed at partitioning their input space. In particular, they used self-organizing maps (SOM) to partition the input space according to the underlying probability distribution of the data. As a result, better generalization ability together with more stability in parameter setting is achieved. Nevertheless, as they argue at the end of the paper, the proposed method has been designed for (and validated on) only binary and low dimensional problems. Wan and Bone [12] used a mixture of radial basis function networks to partition the input space into statistically correlated regions and learn the local covariance model of the data in each region. Ebrahimpour et al. [4] proposed a view-independent face recognition system using ME by manual decomposition of the face view space into specific angles (views), an expert being specialized on each view. Nevertheless, the proposed method is only efficient in 2D face recognition and, as argued by the authors, extending this approach to other classification problems and applications could be challenging and not always possible.

It is worth pointing out that, in the original formulation of the ME model, parameters are determined by maximum likelihood, which is prone to severe

overfitting –including singularities in the likelihood function. This can be particularly problematic in a complex model such as the HME, due to the relatively large number of parameters involved in defining the distributions for experts and gating networks. Indeed, there are many singularities in the likelihood function which arise whenever one of the mixture components "collapses" onto a single data point. In any case, simultaneously training gating networks and experts in an HME architecture (with the goal of obtaining sufficiently accurate classifiers with relatively optimum parameters) continues to pose a research challenge.

Recently, Armano and Hatami have proposed a classifier selection method to be used in selection-fusion strategies [2]. The method called Random Prototype-based Oracle, RPO, splits the input domain based on some prototypes selected randomly from training data and then builds a classifier on each subset. These classifiers are used in combination with an oracle that knows the area of expertise of each classifier, thus generating a miniensemble. Thanks to the random nature of this splitting procedure, miniensembles created on a specific problem differ from one run to another. Each miniensemble can be used as base classifier in any ensemble strategy to improve its accuracy without increasing the computational cost required for training. Subsequently, Armano and Hatami extended their idea using a splitting based on random prototype in the ME model [1]. In their modified ME model, called "Mixture of Random Prototype-based Experts", the input space is partitioned according to the nearest distance from randomly-chosen prototypes. This facilitates the adoption a weighting policy based on distances in both training and testing. In other words, instead of a complex gating network which needs training process to adjust its weight parameters, the proposed gating function manages both training and testing using a distance-based measure. Their experimental results highlight that removing the weight parameters from gating networks allows to decrease the number of ME parameters, thereby simplifying the search and significantly reducing the time required for training.

In this paper, we extend our recent contributions and propose a method to embed a set of random prototypes into each module of the HME model. Instead of specializing each expert on a stochastic and nested area in the feature space, ME experts focus on the centralized subspace defined by their corresponding prototypes. This allows to simplify the gating networks with simple distance-based measures, thus simplifying the structure of the modified HME model. Moreover, while increasing the individual accuracy of the ME modules used in the first layer of an HME architecture, their diversity is also expected to increase –due to the random selection of prototypes (which makes the prototypes of a module different from those used the others). Finally, the time required to train the proposed HME architecture dramatically decreases due to the large saving in the training time of each ME module.

The rest of this paper is organized as follows: in Section 2, we present the standard HME model for building classifiers. In Section 3 we briefly recall the random prototype-based splitting and then introduce the proposed hierarchical mixture of random prototype-based experts. Experimental results are reported and discussed in Section 4. Section 5 concludes the paper and briefly outlines future research directions.

2 The Hierarchical Mixture of Experts model

The HME architecture (Fig. 1) is a tree in which the gating networks lie at the nonterminal nodes and the expert networks lie at the leaves of the tree. Hence, it can be considered as an ensemble of ME modules (as shown by dashed boxes).The task of each expert is to approximate a function over a region of the input space. The task of the gating network is to assign the weights to each expert based on its contribution for each sample. Fig. 1 illustrates a mixture of four experts. In accordance with the typical terminology used for describing HME architectures: $\vec{x}$ is the input vector, $o_{ij}(\vec{x})$ is the output (expected value) of the *ij*th expert, $g_i(\vec{x})$ is the output of the top gating network denoting the prior probability for the pattern to be generated by the left or right branch of the root, and $g_{j|i}(\vec{x})$ is the output of the *i*th bottom gating network denoting the prior probability that the pattern is generated by the *ij*th expert. In addition, *t* is the target (desired output), $P_{ij}(t|\vec{x})$ is the probability associated with the *ij*th expert.

Assuming that experts are mutually exclusive, the overall probability, $P(t|\vec{x})$, and the expected value at the network output, $o(\vec{x})$, are given by:

$$P(t|\vec{x}) = \sum_i g_i(\vec{x}) \sum_j g_{j|i}(\vec{x}) P_{ij}(t|\vec{x})$$

$$o(\vec{x}) = \sum_i g_i(\vec{x}) \sum_j g_{j|i}(\vec{x}) o_{ij}(\vec{x})$$

using notations defined for the two-level depth tree shown in Fig. 1, notations that can be easily extended to larger HME networks with a binary tree architecture.

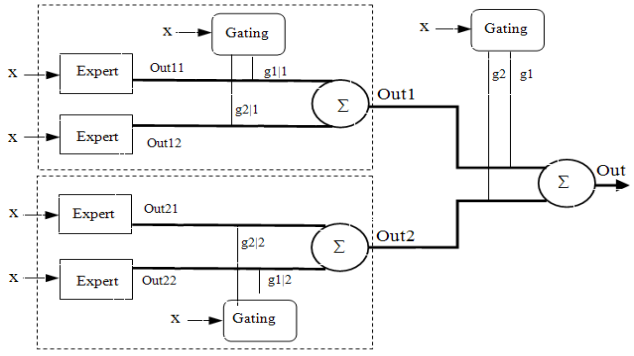


Fig. 1: Block diagram representing a two-layer HME. The generic model shown here has four experts (two ME modules, each containing two experts) and three gating networks (which act as mediators).

Two training procedures are suggested in the literature [6, 7, 8] for finding optimal weight parameters of the HME architecture. The first is the standard error back-

propagation algorithm with gradient descent, whereas the second is based on the Expectation-Maximization (EM) method.

3 Hierarchical Mixture of Random Prototype-based Experts

This section presents the proposed hierarchical mixture of random prototype-based experts (HMRPE) in more detail. The key underlying idea is to randomly partition the input space of the problem into subspaces and then pushing each expert to specialize on its subspace by means of "soft" competitive learning.

3.1 RP-based splitting for HME

For each ME module of an HME architecture, the input space is partitioned according to some prototypes randomly chosen from the training set. This way, input samples can be weighted during the training and testing phases based on their distances from the selected prototypes. The main advantage of this method is that, instead of a complex gating network which must be trained concurrently with other experts, the generated gating function has no parameters (weights) to adjust –as it simply enforces a distance-based weighting policy. Fig. 2 illustrates the block diagram representation of the proposed HMRPE model on a typical binary toy problem.

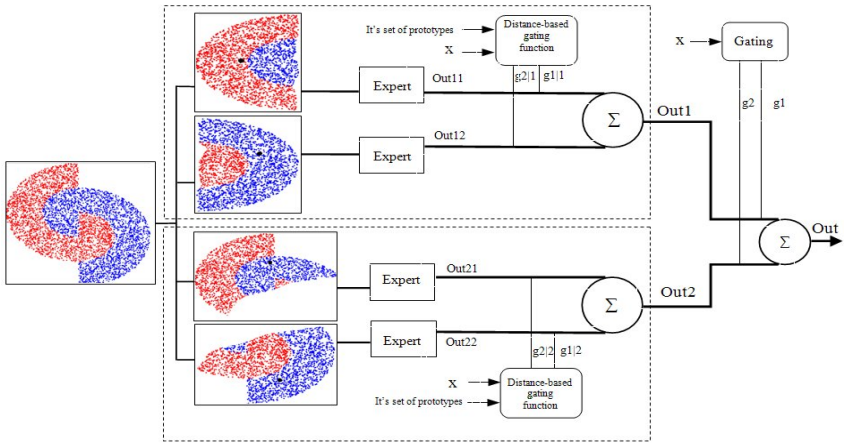


Fig. 2: Block diagram representation of the proposed HMRPE model operating on a typical binary toy problem (bold block points denote randomly-selected prototypes).

For the sake of simplicity, we first describe the random prototype-based data splitting on a synthetic two-class problem shown in Fig. 3.a. Then, we describe how the modified HME model takes advantage of this partitioning.

We use two different partitioning methods, i.e. disjoint and overlapping, shown in Fig.3.b. and 3.c., respectively. In the case of disjoint partitioning, we first measure the distance between each training sample and the prototypes, and then assign a fixed

value, η_j to the h_i of an expert proportional to these distances. h_i is an estimate of the "*a posteriori*" probability for the *i*th expert to generate the desired output o and used as the coefficient of the learning rate for updating the weight parameters of the expert. This means that the weights of the expert network whose prototype is nearest to the current input sample will be updated more than those belonging to the other experts. Similarly, in the testing phase, experts whose prototypes are nearest to the input sample will contribute to a greater extent to the final output.

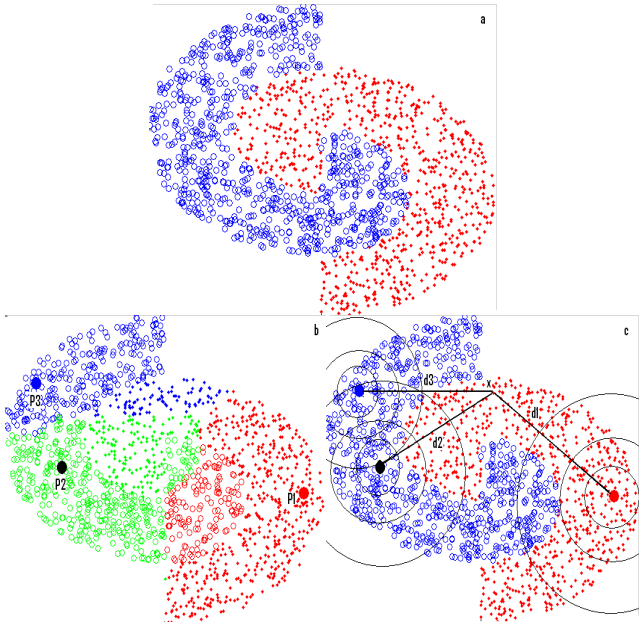


Fig.3: Partitioning of a 2-class semantic classification problem using N=3 random prototypes (bold points denote prototypes randomly selected from training data): a) original problem, b) partitioning into three disjoint regions based on the nearest distance from the prototypes, c) partitioning into three overlapping subspaces.

Unlike disjoint partitioning, where the learning rate coefficients are the same for one partition and change sharply from one to another, in the overlapping method they change smoothly –i.e., proportionally to the distance with the corresponding prototype. Similarly, the amount of d_i for the *i*th expert depends on how close the expert's prototype is to the current input sample d_i . In other words, while using the disjoint method, the degree of expertise of an expert is fixed within the corresponding partition, whereas for the overlapping method it smoothly varies with the d_i distance from the prototypes embedded by the expert itself.

It is worth pointing out that the proposed method is general enough to be applied with both standard error back-propagation and EM learning rules. Fig. 4 reports the

algorithm for training and testing a hierarchical mixture of random prototype-based experts using both disjoint and overlapping partitioning rules for any selected learning method. Let us note that the learning rules of the first-layer gating networks (gating of the ME modules) change with respect to the standard HME model, whereas the gating networks of the other layers (second, third, and so on) do not.

Random Prototype-based HME

INITIALIZING:

- LS = Learning Set, TS = Testting Set
- $\psi = \{\varepsilon_i \,|\, i = 1,2,\ldots,N\}$
- strategy $= \{\text{static}, \text{dynamic}\}$
- $E = \{\eta_j \in (0,1) \,|\, j = 1,2,\ldots,N\}$ such that:

$$\eta_k \le \eta_{k+1}; k = 1,2,\ldots,N-1 \text{ and } |E| = \sum_j \eta_j = 1$$

TRAINING:

For jth ME module $\in$ HME Do:

For each $x \in LS$ Do:

- choose randomly N prototypes from the LS; $P_j = \{p_{ij} \in LS \,|\, i = 1,2,\ldots,N\}$;
- $D(x) = \{d_{ij}(x) \,|\, i = 1,2,\ldots,N\}$ where

$$d_{ij}(x) = \|x - p_{ij}\| \text{ and } \|.\| \text{ is any distance metric (e.g. Euclidean)}$$

- $H(x) = \{h_{ij}(x) \,|\, i = 1,2,\ldots,N\}$ where

$h_i(x)$ represents the expected capacity of ε_i to deal with the given input x

$$[\text{strategy} = \text{static}]: h_{ij}(x) = \eta_k \text{ where } K = \text{Rank}(\varepsilon_{ij}, D(x))^*$$

$$[\text{strategy=dynamic}]: h_{ij}(x) = 1 - \frac{d_{ij}}{|D(x)|} \text{ where } |D(x)| = \sum_k d_k(x)$$

- update each expert ε_{ij} $(i = 1,2,\ldots,N)$ according to the standard learning rule for ME

TESTING:

For jth ME module $\in$ HME Do:

Given an $x \in TS$ Do:

- $D(x) = \{d_{ij}(x) \,|\, i = 1,2,\ldots,N\}$
- $G(x) = \{g_{ij}(x) \,|\, i = 1,2,\ldots,N\}$ where

$$[\text{strategy=static}]: g_{ij}(x) = \eta_k \text{ where } k = \text{Rank}(\varepsilon_{ij}, D(x))^*$$

$$[\text{strategy=dynamic}]: g_{ij}(x) = 1 - \frac{d_{ij}}{|D(x)|} \text{ where } |D(x)| = \sum_k d_k(x)$$

- calculate the overall output:

$$o_k(x) = \sum_{i=1}^{N} g_{ij}(x).o(x, W_{ij})$$

- select the class label c_l such that

$$L = \arg\max_k (o_k(x))$$

$^*k = \text{Rank}(\varepsilon_i, D(x))$ returns the rank of expert ε_i (i.e. a number in $[1, N]$) according to the distance $D(x)$ evaluated on the input x (the lovest the distance, the highest the ranking)

Fig.4: Hierarchical Mixture of Random Prototype-based Experts Algorithm.

3.2 Why does HMRPE work?

Notwithstanding the big of empirical studies proving that diversity and individual accuracy of ensemble members are two primary factors that affect the overall

classification accuracy, theoretical studies clearly show that they are not independent [9]. Hence, the success of the proposed HMRPE approach can be attributed to three factors as follows:

1. Splitting the input space into *N* centralized parts makes the subproblems easier to learn for the expert network. As a consequence, the individual accuracy of the ensemble members is expected to be better than, or at least not worse than, the one exhibited by sets of experts specialized over the nested and stochastic subspaces. It is worth noting that, although expected, higher individual accuracy is not guaranteed by any means, since it depends on the complexity of classification boundaries, on the adopted learning algorithm, as well as on the position of the selected prototypes. Fig. 5 compares the regions of expertise of an ME module embedded in the both standard HME and HMRPE on a four-class toy problem. The first figure from the left in the top, shows the original problem and the next three figures are nested areas of expertise of three experts in the standard ME module. At the bottom, splitting of the problem using three random prototypes is shown in the first figure. The next three figures show the centralized areas of expertise of three experts in the proposed HMRPE module.

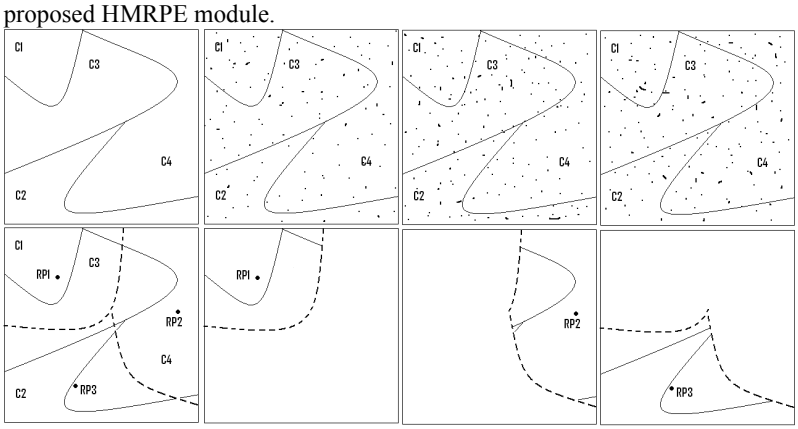


Fig.5: Comparison between the standard HME model and the HMRPE model for a 4-class toy problem.

2. Since each ME module embedded in the HMRPE architecture has its own set of prototypes (which are different from those embedded by the other ME modules), experts have been specialized on very different data subsets – thus enforcing diversity.
3. The accuracy of HME classifier is affected by the performance of both experts and gating networks. Accordingly, resulting misclassifications in this model derive from two sources: (a) the gating networks are unable to correctly estimate the probability for a given input sample and (b) local experts do not learn their subtask perfectly. Since simple distance rules used by the gating function are more robust with respect to errors in determining the area of expertise of an expert, errors in the proposed HMRPE model are mainly

limited to the error made by the expert networks –thus improving the overall accuracy of the overall classifier.

4 Experimental Results

Some of the UCI machine learning data sets [10] have been used to check the validity of the proposed method. These data sets include real-world and synthetic problems, with variable characteristics, previously investigated by other researchers. Table 1 shows the selected datasets with more detail.

Table1. The main characteristics of the selected UCI datasets.

Problem	# Train	# Test	# Attributes	# Classes
Iris	150	-	4	3
Satimage	4435	2000	36	6
Pendigits	7494	3498	16	10
Letter	20000	-	16	26
Vowel	990	-	11	11
Segment	210	2100	19	7
Glass	214	-	9	7
Yeast	1484	-	8	10

We used 10-fold cross validation to ensure statistical significance while evaluating the accuracy of classifiers. To build the standard HME and the proposed HMRPE models, we used a Multi-Layer Perceptron (MLP) architecture with one hidden layer, trained with the back-propagation learning rule [5]. To determine the best value for the *N* number of partitions, which is equal to the number of experts, we varied it from 2 to 10 for each dataset. We also varied the number of hidden neurons in expert networks to experimentally find the optimal architecture of the MLP experts for each problem. The results of these experiments (shown in Table 2) highlight that the proposed method outperforms the standard HME model for all selected datasets –no matter whether disjoint or overlapping partitions are adopted.

Table 2. The mean and standard deviation of accuracy of the HME vs. the proposed HMRPE on the selected UCI datasets (in percentage).

	Iris	Sat.	Pen.	Lett	Vow.	Seg.	Gla.	Yeast
Standard HME	88.7± 0.59	88.6± 1.05	88.0± 0.43	71.5± 0.94	60.9± 1.55	79.0± 0.85	72.1± 1.75	50.6± 2.22
Disjoint partition	89.3± 0.42	90.9± 0.80	89.5± 0.44	73.0± 0.73	63.8± 1.00	82.2± 0.68	75.8± 1.77	52.7± 1.56
Overlapping partition	89.6± 0.40	91.1± 0.78	90.2± 0.31	73.8± 0.82	64.4± 1.21	82.9± 0.79	76.9± 1.44	54.0± 1.45

The time required for training the different datasets is shown in Table 3 for further comparison. Table 3 highlights that the training time of the proposed method is considerably shorter than the standard version. Simulations are performed using an Intel CPU with 2.83GHz and 4GB RAM memory. Note that the results presented here which compare standard HME and the proposed method, use the same parameters and architecture.

Table 3. Training time of the HME vs. HMRPE (seconds).

	Iris	Sat.	Pen.	Lett.	Vow.	Seg.	Gla.	Yeast
Standard HME	84	324	451	604	99	71	44	67
HMRPE	57	198	311	451	63	53	30	42

5 Conclusions and future directions

A modified version of the popular HME algorithm is presented here. Unlike the standard HME, which specializes expert networks on nested and stochastic regions of the input space, the proposed method partitions the sample space into subspaces based on similarities with randomly-selected prototypes. This strategy enables to define a simple rule for the gating network for both training and testing. As shown by experimental results, despite its simplicity, the proposed method improves the accuracy of the standard HME model and reduces the training time.

Future works focus on defining a procedure for automatically determining the number of optimal experts for each problem without resorting to complex preprocessing and time consuming methods. Adapting this method to simple distance-based classifiers instead of neural networks is another interesting future research direction, aimed at reducing the training time of the overall network while maintaining high accuracy.

We are also currently making some experiments on heuristics able to help in the process of partitioning the input space instead of using random prototypes. Furthermore, as the proposed approach is not limited to neural networks, it would be interesting to investigate the behavior and performance of other learners in the proposed HMRPE architecture.

References

1. Armano, G. and Hatami, N., Mixture of Random Prototype-based Local Experts, HAIS (1), LNAI 6076 , 2010: 548-556.
2. Armano, G. and Hatami, N., Random Prototype-based Oracle for selection-fusion ensembles, ICPR 2010, 20th International Conference on Pattern Recognition, Istanbul (in press).
3. Avnimelech, R., Intrator, N., Boosted mixture of experts: an ensemble learning scheme, Neural Comput. 11 (2) (1999) 483–497.

4. Ebrahimpour, R., Kabir, E., Yousefi, M.R., Teacher-directed learning in view-independent face recognition with mixture of experts using overlapping eigenspaces, Computer Vision and Image Understanding 111 (2008) 195–206.
5. Haykin, S., 1999. Neural Networks: A Comprehensive Foundation, Prentice Hall, 2nd Edition.
6. Jacobs R, Jordan M, Barto A (1991) Task decomposition through competition in a modular connectionist architecture: the what and where vision tasks. Tech rep. University of Massachusetts, Amherst, MA.
7. Jacobs R, Jordan M, Nowlan S, Hinton G (1991) Adaptive mixtures of local experts. Neural Computation 3:79–87.
8. Jordan, M. I., Jacobs, R. A.: Hierarchical mixtures of experts and the EM algorithm Neural Comp. 6 (1994), 181–214.
9. Kuncheva, L.I., Combining Pattern Classifiers: Methods and Algorithms, Wiley, 2004.
10. Murphy P.M. and Aha, D.W. UCI Repository of Machine Learning Databases, Dept. of Information and Computer Science, Univ. of California, Irvine, 1994.
11. Tang, B., Heywood, M., Shepherd, M., Input partitioning to mixture of experts, in: International Joint Conference on Neural Networks, 2002, pp. 227–232.
12. Wan, E., Bone, D., Interpolating earth-science data using RBF networks and mixtures of experts, in: NIPS, 1996, pp. 988–994.
13. Waterhouse, S., Cook, G., Ensemble methods for phoneme classification, in: M. Mozer, J. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems, vol. 9, The MIT Press, Cambridge, MA, 1997, pp. 800–806.

Three Data Partitioning Strategies for Building Local Classifiers: an experiment

Indrė Žliobaitė

Vilnius University and Eindhoven University of Technology
zliobaite@gmail.com

Abstract. Divide-and-conquer approach has been recognized in multiple classifier systems, aiming to utilize local expertise of individual classifiers. In this study we experimentally investigate three strategies to build local classifiers, based on different sampling for training routines. The first two are based on clustering the training data and building a separate classifier for each cluster or a combination. The third one divides the training set based on a selected feature and builds a separate classifier for each subset. Experiments are carried out on simulated and real datasets. We report improvement in final classification accuracy as a result of combining the three strategies.

1 Introduction

In supervised learning tasks local models tailored to a particular subgroup are often applied [4,7,8,10,12]. The intuition behind such multiple classifier systems is to maintain explicit specialization of individual classifiers.

Consider a task of evaluating scientific research proposals for funding. Given a research proposal in accordion music we would like to assign it to an expert in music rather than biology.

There are two main design issues to be determined when constructing such systems of local classifiers. The first is how to partition the training data to form groups of local expertise. In the research funding example, the submission tracks might be formed in different ways: based on the area (biology, music, literature, physics), theoretical versus industrial projects, based on the size of a requested budget or combination of all criteria. Thus the issue is how to train the local experts to achieve local specialization (directed diversity).

The second design issue is how to determine to which local classifier to assign the incoming new data. In the research proposal example, suppose there is only the proposal text available, no leading document. A secretary scans through all the incoming proposals to determine whether it is biology or music. Thus the issue is how to associate an incoming instance with a particular group of expertise.

A popular design of such systems is to cluster the training data and train a separate classifier on each cluster. Then the incoming data is assigned for decision making based on cluster membership (e.g. [4, 8, 12, 13]). Depending on

the data, this approach might lead to rather uniform class membership within one cluster and raise challenges for individual classifier training. To overcome this we introduce one modification and one alternative partitioning technique.

We experimentally analyze the three strategies for data partitioning to build local classifiers. The effects to the final classification accuracy are demonstrated using a simulated dataset and six real datasets. We report improvement in final classification accuracy as a result of combining the three strategies.

The remainder of the paper is organized as follows. In Section 2 we present three alternatives for training local classifiers. In Section 3 we illustrate the alternatives by analyzing toy dataset example. In Section 4 we present and analyze experimental results using six real datasets. In Section 5 we conclude.

2 Alternatives for Building Local Classifiers

There are two directions for designing multiple classifier systems with local expertise. The first is to generate a pool of classifiers using randomized choices of training set for each individual classifier and then ensure the diversity by the fusion rules (e.g. bagging [2], see also a review [3]).

Our focus is on the second type, which partitions the dataset in a directed manner to form local groups of similar data (e.g. clustering approaches taken in [4, 8, 12]). We call this *a directed diversity*. The input space can be divided into regions of competence either based on features or instances or both. We analyze three alternatives for partitioning of the training data, two instance based partitioning and one partitioning in feature space.

2.1 Instance Based Partitioning

A popular way to partition the data for building local experts is clustering [4, 8, 12]. Let $X \in \Re^p$ be the input space, partitioned into k subsets $R_1, R_2, \ldots, R_k$ using a selected clustering algorithm. An ensemble of classifiers $\mathcal{L} = L_1, L_2, \ldots, L_k$ is trained using the corresponding subset for each classifier: $L_i = f(R_i)$. Note that unlabeled data is used for partitioning, while for training local classifiers the class labels are needed as well. We refer to this technique as **CLU**.

There are two main design issues to decide for CLU: to choose the number of clusters k as close as possible to the nature of the data and choose an algorithm (and distance metric) for clustering.

Back to the research proposal example, assume we have a pile of historical not annotated documents, but we know (domain knowledge) that the proposals came from three departments: biology, physics and music. Thus we probably choose to cluster all the documents into $k = 3$ clusters. If we do not have the background knowledge, we need to guess k.

After the local classifiers are trained, the question is how to assign new incoming instances. In CLU it is typically assigned to the closest cluster center. Apparently, the same distance metric as in the original clustering shall be used.

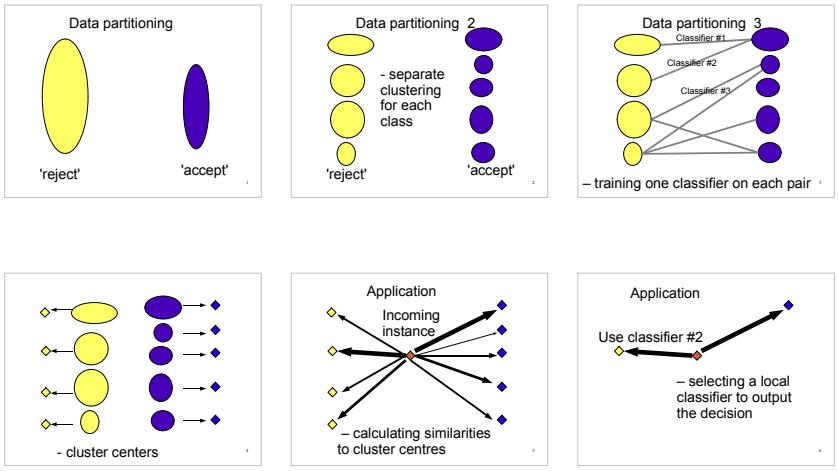


Fig. 1. Illustration of CL2 partitioning.

In the proposal example the ultimate task is to classify the new incoming document into 'accept' and 'reject'. Given a new incoming document we do not know its label, but we can extract the content features the same way we did for the historical data. Based on the features we can assign the document to one of the existing clusters. Based on the cluster assignment we apply a local expert to output the decision 'accept' or 'reject'.

Clustering seems to be a reasonable approach to build local classifiers. However, there is a problem related to class membership. The clusters might capture the actual class membership ('accept' or 'reject'). This results in rather homogeneous clusters in terms of class labels (all 'accepts' in one cluster, 'rejects' in other). It might pose a problem for training a classifier, as there would be not enough examples from other classes for training.

We propose an alternative to overcome capturing class discriminative information. The idea is to cluster the classes separately and then make all possible combinations of the clusters taking one cluster from each class. Let $X^{(1)}, X^{(2)}, \ldots, X^{(c)}$ be a partitioning of the historical data based on the class membership, where c is the number of the classes. First we cluster each $X^{(i)}$ to obtain k_i subsets $R_1^{(i)}, R_2^{(i)}, \ldots, R_{k1}^{(i)}$. Then we train an ensemble, consisting of $k_1 \times k_2 \times \ldots \times k_i$ classifiers: $R_{i1}^{(1)} \cup \ldots \cup R_{ic}^{(c)} \rightarrow L_{i1i2\ldots ic}$. This allows to handle the problem of class imbalance within one cluster. Note that subsets $R_1^{(1)}, R_2^{(1)}, \ldots, R_{k1}^{(1)}, \ldots, R_1^{(c)}, R_2^{(c)}, \ldots, R_{k1}^{(c)}$ do not intersect and together form an input data X. We refer to this technique as **CL2**.

Differently from CLU, in CL2 the data is partitioned within each class separately. This way class discriminatory information does not affect the partitioning.

We assign a new incoming instance for a decision making the same as in CLU, based on the proximity to the cluster centers. However in CL2 it is assigned to

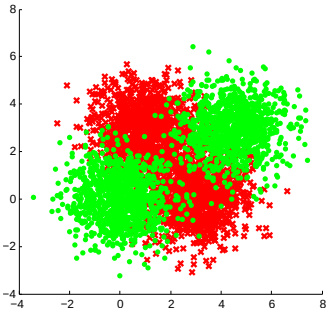


Fig. 2. Modeling dataset.

c nearest clusters (one from each class), since naturally we do not know the true class membership of the incoming instance.

CL2 approach requires the number of clusters $k_1, k_2, \ldots, k_c$ as parameters.

The intuition relates to the same proposal example. The 'rejected' proposals in biology might be the most similar in content to the 'accepted' proposals in physics. Thus we are interested to learn the peculiarities to distinguish between the two. So we cluster first all the 'accepted' documents. Then we cluster all the 'rejected' documents independently. Then build a local classifier using the two closest clusters as a training set. We illustrate CL2 in Figure 1.

2.2 One Feature Based Partitioning

We propose an alternative to clustering based partitioning (**FEA**). The data can be partitioned based on one selected feature, like bread slicing. Assume we would like to get k subsets of the training data. We select a feature x_s, where $s \in p$ (feature space) and split it into k equal intervals. Let $\delta_k = \frac{\max x_s - \min x_s}{k}$. Then the i^{th} interval is $r_i : [\min x_s + (i-1)\delta_k, \min x_s + i\delta_k)$[1]. The historical data is partitioned into subsets $R_1^{(F)}, R_2^{(F)}, \ldots, R_k^{(F)}$, where $X^j \in R_i^{(F)}$ if $x_s^j \in r_i$. Then a local classifier is trained on each subset. A new incoming instance will be assigned to a classifier based on the value of its x_s feature the same way.

In the research proposal example, assume there are experts, which are good in evaluating short and concise proposals and another experts, which are good in long detailed proposals. In such case the subsets might be formed based on the number of words in the document, having it as a feature in the feature space.

There are two parameters to be specified for FEA: k and x_i. The selection of x_i is to be based on the domain expertise or visual inspection of the data.

3 Analysis of the Toy Dataset

We construct a toy example to explore CLU, CL2 and FEA partitioning techniques. We generate the following heterogeneous dataset in 2D. Four data subset

[1] The value $\min x_s + k\delta_k = \max x_s$ for the last interval r_k is inclusive.

centers are fixed at $(0, 0)$, $(4.5, 3)$, $(1, 3)$, $(3, 0.1)$. Two of them we label as 'class 1', the other two as 'class 2'. The data is illustrated in Figure 2. We generate 5000 normally distributed instances in a subset.

It can be seen that the data is linearly inseparable. However, a multiple classifier system, consisting of locally specialized linear classifiers could do it.

3.1 Testing Scenario

Testing set. We generate an independent testing set using the same distribution as for training. We generate a testing set of 20000 instances in total.

Performance metrics. We compare the alternative techniques based on testing error as an indicator of generalization performance. The prediction error is used, where outputs are not rounded (an experimental design choice, no specific reason). A standard error is calculated assuming Binomial distribution over testing errors $SE = \sqrt{\frac{E \times (1-E)}{N}}$, where E is the observed error and N is the testing sample size.

Alternative techniques. We experimentally compare the performance of CLU, CL2 and FEA. For an illustration of the three techniques on the modeling data see Figure 3. '$\times$' indicates an instance under consideration. The data subsets from which an expert is built to be used to this instances are plotted in (a) for CLU, in (b) for CL2 and in (c) for FEA.

In addition to the there discussed techniques, we include the averaging technique into the experiments. We will refer to it as **MMM**.

Let $\hat{Y}_{CLU}$, $\hat{Y}_{CL2}$ and $\hat{Y}_{FEA}$ be the labels output by the three respective techniques. Then $\hat{Y}_{MMM} = \frac{\hat{Y}_{CLU}+\hat{Y}_{CL2}+\hat{Y}_{FEA}}{3}$. The intuition is to combine the diverse views on the same instance obtained by the respective methods.

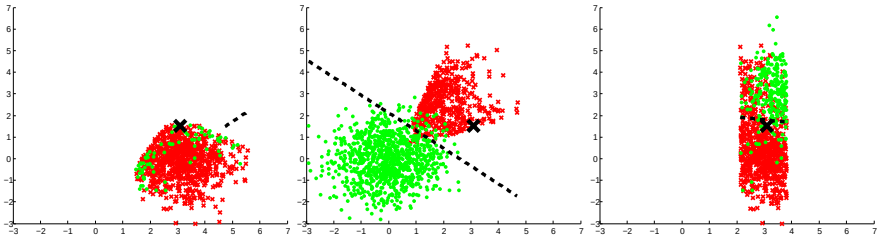
It can be viewed as an ensemble of ensembles, where classifier selection is used in the first level and then classifier fusion is applied on top. This combination is similar to random oracle [11]. The principal difference is that random oracle uses the same partitioning technique for all miniensembles. In this experimental study we investigate the performance of alternative partitioning techniques rather than ensemble building techniques, thus we do not include oracles in the experiments.

In addition, we include two benchmark techniques. With the first we will test the benefit of specialization. Here the instances to train the local classifiers are assigned at random instead of employing a directed procedure. In this case the classifiers are expected to have no specialization, since it will be trained on a random subset of the historical data. We refer to this technique as **RSS**.

The second benchmark technique does not do any partitioning of the data at all, it just trains a single classifier on all the historical data. We will test the effect of smaller training sample size to the accuracy with it. We call it **ALL**.

Finally, we include a baseline technique, which assigns all the labels according to the highest prior probability. We refer to is as **NAY**.

The number of clusters. In the toy dataset four clusters can be visually identified. We investigate the strategies of building local classifiers under two scenarios: A the number of subclasses fits correctly (in our toy dataset $k = 4$)



(a) (b) (c)

Fig. 3. Illustration of data partitioning: (a) CLU, (b) CL2 and (c) FEA.

and B the number of classes is incorrect (we use $k = 9$). That means for CLU, FEA and RSS we test $k = 4$ and $k = 9$ respectively. For CL2 we test $k_1 = k_2 = 2$ and $k_1 = k_2 = 3$. The latter case leads to $k = 3 \times 3 = 9$ local experts for CL2.

Base classifier. We choose logistic regression as the base classifier. The motivation for this choice is twofold. First, the weights can be interpreted as the importance score of each feature. Thus it is popular in application tasks, such as credit scoring, where heterogeneity of the data is an issue and local expertise models intuitively make sense. Second, it is a parametric classifier, thus rearranging the training subsets change the local expert rules significantly.

We already mentioned that data partitioning techniques are likely to distort prior probabilities of the classes within each subset as compared to the whole set of historical data. Logistic regression uses prior information in training, thus *a correction for priors* is required. We use a prior correction for rare events [9]. The regression coefficients are statistically consistent estimates, while the correction for an intercept is as follows: $\beta_0 = \hat{\beta}_0 - log\left[\left(\frac{1-\tau}{\tau}\right)\left(\frac{\pi}{1-\pi}\right)\right]$, where $\hat{\beta}_0$ is an intercept estimated from the training sample, τ is the population prior for the 'class 1' and π is the sample prior for the 'class 1'.

3.2 Results

In Table 1 we provide the testing errors of the discussed strategies. The best results indicated in **bold**. In both cases, when the number of subsets fits the underlying data and when it does not MMM outperforms the baseline strategies by a large margin. In case the number of clusters is correct, CLU has comparable performance, however it fails here if k is incorrectly set.

Interestingly, we observe an improvement in the performance of FEA when the number of subsets is increased to $k = 9$. This can be explained by the nature of the modeling data. It is not linearly separable, thus the slices of the data selected by FEA are not separable as well, see Figure 3(c). But the smaller the slices in this case, the more linearly separable are the subsets.

Table 1. Testing erors using the modeling dataset.

	A: $k = 4$	B: $k = 9$
	error st. err.	error st. err.
CLU	10.6% (±0.2)	12.1% (±0.2)
CL2	13.9% (±0.2)	14.3% (±0.2)
FEA	17.1% (±0.3)	14.1% (±0.2)
MMM	**10.5**% (±0.2)	**10.5**% (±0.2)
RSS	49.7% (±0.4)	49.8% (±0.4)
ALL	49.7% (±0.4)	49.8% (±0.4)
NAY	50.0% (±0.7)	50.0% (±0.7)

4 Experiments with Real Data

We analyze the performance of the three techniques using six real datasets.

4.1 Datasets

The characteristics of the six datasets used in the experiments are presented in Table 2. All datasets present a binary classification task for simplicity and computational issues; however, the tested methods are not restricted to the binary tasks. For Shuttle data we aggregated the classes into binary task (class 1 against all the others). In Marketing data we transformed the categorical features to numerical by expanding the feature space. We constructed Chess dataset[2] using the statistics from Chess.com, the task is to predict the outcome of a game given the players and game setup characteristics. Elec2 dataset is known to be non stationary. In these settings non stationarity is expected to be handled directly by local learners. Data comes from a mix of sources, since the datasets at hand by the author were selected to be included into the experiments.

Table 2. Datasets.

	size	dimensionality	class balance	source
cred	1000	23	70% − 30%	(German credit) [1]
shut	43500	9	22% − 78%	(Shuttle) [1]
spam	4601	57	39% − 61%	(Spam) [6]
marc	8993	48	47% − 53%	(Marketing) [6]
elec	44235	7	43% − 57%	(Elec2) [5]
chess	503	8	39% − 61%	our collection

4.2 Implementation details

Training-testing sets. Holdout testing procedure is used. Each dataset is split into two equal parts at random, one is used for training, the other for testing.

[2] The dataset is available at http://sites.google.com/site/zliobaite/resources-1 .

Parameters and experimental choices. The number of partitions was fixed to $k = 4$ in all partitioning techniques (CLU, CL2, FEA, RSS). We choose to use simple and intuitive k-means clustering algorithm.

For FEA we choose the partitioning feature to be the first feature in a row having 4 or more distinct values. The selected feature is different for each dataset, but the procedure for choosing it is the same for all.

4.3 Experimental goals

The goal of these experiments is to compare the classification accuracies when using the three partitioning techniques (CLU, CL2, FEA) and a combination of those (MMM) and analyze the underlying properties of the data leading to these accuracies.

We aim to be able to assign the credits for a better accuracy. Thus, two benchmarks (ALL, RSS) and a baseline (NAY) are included for control reasons. We look at the following guidelines for control:

- We expect the partitioning techniques (CLU, CL2, FEA) to do better than no partitioning (ALL) in order for partitioning to make sense.
- We expect random partitioning (RSS) not to do much worse than no partitioning (ALL). Much worse accuracy would indicate the problems due to small sample size within each partitioning.
- NAY gives the error of classification by the largest prior, given equal costs of mistakes, we expect all intelligent predictors to do better.

In addition, we aim to analyze the effect of directed diversity for the accuracy of MMM, achieved by the presented techniques for building local classifiers.

We present two sets of the experimental results. First we present and discuss the accuracies of each method. Second, we analyze the relationship between outputs of the four techniques (CLU, CL2, FEA, MMM).

4.4 Results

In Table 3 the testing errors of the alternative partitioning techniques are compared. Clustering techniques and feature based split do not show significant effect on accuracies individually. However, blending all three (MMM) leads to significant improvement in accuracy and dominates in five out of six datasets.

RSS performs worse than ALL in all six datasets, which points out the effect of training set size (smaller in RSS) to the final accuracy. If we partition input space into non overlapping regions to gain on specialization of classifiers, naturally we loose on sample size. However, as seen from the table, here these deterioration effects are not that drastic. Smaller sample size partially explains not outstanding performance of the individual partitioning methods (CLU, CL2,FEA).

Averaging over three methods leads to significant improvement in accuracy, thus we look at how diversified are the individual outputs. In Figure 4 we picture simple pairwise correlations between classifier outputs. Black corresponds to

Table 3. Testing errors using the real datasets.

	cred	shut	marc	spam	elec	chess
testing errors						
CLU	31.8%	3.9%	30.5%	11.0%	29.3%	22.2%
CL2	32.8%	2.2%	32.1%	11.6%	32.5%	27.0%
FEA	28.7%	**1.6**%	28.8%	11.5%	32.4%	28.3%
MMM	**28.2**%	2.1%	**24.8**%	**8.4**%	**24.7**%	**19.3**%
RSS	32.7%	5.3%	34.5%	11.5%	32.1%	27.9%
ALL	31.2%	5.4%	32.2%	11.7%	32.7%	26.9%
NAY	31.0%	21.4%	46.5%	38.7%	42.6%	42.1%
standard error	(±2.0)	(±0.1)	(±0.7)	(±0.7)	(±0.3)	(±3.0)

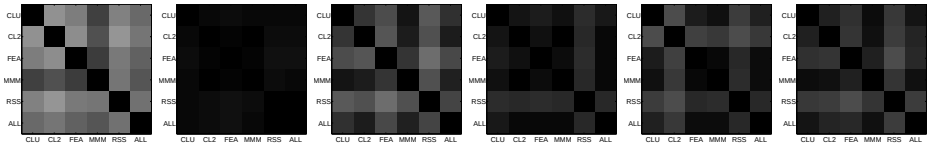


Fig. 4. Correlations of the classifier outputs.

perfect correlation (1) and while denotes independence (0). 'Spam' and 'shut' are nearly black because overall accuracy on these datasets is higher.

In the experiment we used fixed number of clusters ($k = 4$). In order to see the effect of local specialization we look at the sensitivity of the results to the number of clusters. For that test we choose the two largest datasets 'shut' and 'elec' to have sufficient number of instances for large number of partitionings. In Figure 5 we plot the relationship between the number of clusters and testing error for the comparable methods.

The performance of the partitioning methods is improving with increasing number of clusters and then stabilizes, especially in 'shut'. The trends indicate that given large enough datasets, directed classifier specialization is beneficial to the final accuracy.

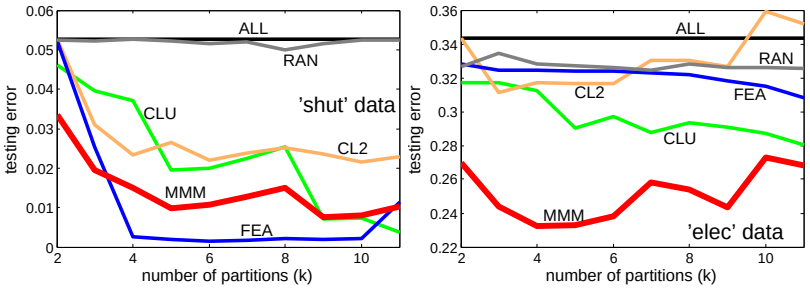


Fig. 5. Sensitivity of testing accuracy to the number of clusters: (a) 'shut', (b) 'elec'.

5 Conclusion

We experimentally investigated three approaches of building local classifiers. Each partitioning technique individually often does not give significant improvement in accuracy as compared to a single classifier. However, blending the three techniques demonstrated significant improvement in accuracy as compared to the baseline and benchmarks.

This approach can be viewed as two level ensemble. The first level uses deterministic classifier selection from a pool of local classifiers. The second level averages over individual classifiers selected from different partitioning of the input space.

A natural and interesting extension of this study would be to look at partitioning techniques, not limited to distance in space as clustering. Developing meta features for subsets of instances is one example.

Acknowledgements We thank dr. Tomas Krilavičius for the comments, which contributed to improving presentation of the results.

References

1. D.J. Newman A. Asuncion. UCI machine learning repository, 2007.
2. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
3. G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5 – 20, 2005.
4. D. Frosyniotis, A. Stafylopatis, and A. Likas. A divide-and-conquer method for multi-net classifiers. *Pattern Analysis and Applications*, 6(1):32–40, 2003.
5. M. Harries. Splice-2 comparative evaluation : Electricity pricing. technical report UNSW-CSE -TR-9905, Artificial Intelligence Group, School of Computer Science and Engineering, The University of New South Wales, Sidney, 1999.
6. T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2005.
7. R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87, 1991.
8. I. Katakis, G. Tsoumakas, and I.Vlahavas. Tracking recurring contexts using ensemble classifiers: an application to email filtering. *Knowledge and Information Systems*, 22(3):371–391, 2009.
9. G. King and L. Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2):137–163, 2001.
10. L. Kuncheva. Clustering-and-selection model for classifier combination. In *Proc. of the 4th Int. Conf. on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, volume 1, pages 185–188, 2000.
11. L. I. Kuncheva and J. J. Rodriguez. Classifier ensembles with a random linear oracle. *IEEE Trans. on Knowl. and Data Eng.*, 19(4):500–508, 2007.
12. M. Lim and S. Sohn. Cluster-based dynamic scoring model. *Expert Systems with Applications*, 32(2):427–431, 2007.
13. R. Liu and B. Yuan. Multiple classifiers combination by clustering and selection. *Information Fusion*, 2(3):163–168, 2001.