Controlling Feature Selection in Random Forests of Decision Trees Using a Genetic Algorithm: Classification of Class I MHC Peptides

Loren Hansen[*,1,2], Ernestine A. Lee[3], Kevin Hestir[3], Lewis T. Williams[3] and David Farrelly[4]

[1]*Program in Bioinformatics, Boston University, Boston, MA 02215, USA*

[2]*Present Address: National Center for Biotechnology Information, National Library of Medicine, Bethesda, MD 20894, USA*

[3]*FivePrime Therapeutics, 1650 Owens Street, Suite 200, San Francisco, CA 94158-2216, USA*

[4]*Department of Chemistry and Biochemistry, Utah State University, USA, and Departamento de Química, Universidad Autónoma de Madrid, Cantoblanco, 28049 Madrid, Spain*

Abstract: Feature selection is an important challenge in many classification problems, especially if the number of features greatly exceeds the number of examples available. We have developed a procedure - *GenForest* - which controls feature selection in random forests of decision trees by using a genetic algorithm. This approach was tested through our entry into the Comparative Evaluation of Prediction Algorithms 2006 (CoEPrA) competition (accessible online at: http://www.coepra.org). CoEPrA was a modeling competition organized to provide an objective testing for various classification and regression algorithms *via* the process of blind prediction. In the competition *GenForest* ranked 10/23, 5/16 and 9/16 on CoEPrA classification problems 1, 3 and 4, respectively, which involved the classification of type I MHC nonapeptides i.e. peptides containing nine amino acids. These problems each involved the classification of different sets of nonapeptides. Associated with each amino acid was a set of 643 features for a total of 5787 features per peptide. The method, its application to the CoEPrA datasets, and its performance in the competition are described.

Keywords: Decision trees, random forests, feature selection, genetic algorithms, evolutionary computation.

1. INTRODUCTION

Classification problems are often stated as follows: given a set of cases known to be classified as positives or negatives, and given an associated vector of attributes or features attached to each such case, is it possible to assign unknown cases to either class based solely on knowledge of equivalent vectors of features? Random forests of decision trees (RFs) are one such means of separating classes of individuals based on knowledge of a finite number of individual features [1,2]. Herein, each set of features, taken together, will be called an "instance". The method involves supervised learning and so both a test and training set of data are needed before unknown instances can be classified. The test, training and unknown datasets are all assumed to contain entries corresponding to the same vector of features. An advantage of the method is that RFs are robust to noise and errors and so can often be used even if some of the features lie outside the bounds of some features in the training set. However, like most inductive methods, RFs may perform poorly when confronted with too many features [3-6]. Feature subset selection can therefore improve the efficiency and accuracy of classification methods.

Good feature selection algorithms can also help to identify which features are relevant in a particular context. For example, feature selection may help identify which factors are contributors to the development of a disease. Feature selection essentially reduces to finding an optimal solution in a possibly (and usually) very high-dimensional feature space. Genetic algorithms have previously been applied to feature selection, e.g., in an application using neural networks [4]. References [2,3, 5-8] provide a good overview of the literature on feature subset selection. Herein feature subset selection is done using a GA [9] in combination with RFs [1,2]. The method used falls into the category of a wrapper approach because the GA and the RFs are integrated into a single algorithm in which the GA controls feature selection on-the-fly. This is in contrast to the filter approach in which feature selection is performed independently of the classification algorithm [3,7].

The method was applied to the CoEPrA (Comparative Evaluation of Prediction Algorithms) modeling competition of 2006 [10]. This competition was developed to facilitate testing of various classification and regression algorithms using blind prediction. The datasets used in the competition are available at the CoEPrA website [10]. The datasets are related to certain class I MHC peptides which are important indicators of cellular proteomics relevant to cancer, autoimmune and viral diseases.

The goal of the CoEPrA competition was to advance the development of algorithms and software for modeling chemical, biological, and medical data, with special emphasis on the prediction of physico-chemical properties and biological activities from molecular descriptors derived from the chemical structure. In addition, CoEPrA provides a reference database of modeling datasets that can be used to validate and compare new classification and regression algorithms.

*Address correspondence to this author at the National Center for Biotechnology Information, National Library of Medicine, Bethesda, MD 20894, USA; Tel: (301) 594-4013; E-mail: hansenlo@ncbi.nlm.nih.gov

1386-2073/09 $55.00+.00

In each CoEPrA task the participants received a calibration dataset and a prediction dataset. The model (classification or regression) derived from the calibration dataset was used to predict the dependent variable of the prediction dataset. These predictions were then deposited with the competition organizers before the deadline for each task. The organizers and evaluators for each CoEPrA task then evaluated all predictions and announced the rank of the participants on the CoEPrA website [10]. Application of *GenForest* to the CoEPrA classification problem No. 2 is described elsewhere [11].

2. THE COEPRA COMPETITION DATASETS

For each of the CoEPrA classification problems considered here, i.e. Nos. 1, 3 and 4 (CoEPrA-C1, C3, C4), the competition organizers [10] supplied datasets containing equal numbers of prediction (unknown) and calibration instances; each instance corresponds to a different nonapeptide, i.e. a peptide consisting of 9 amino acids. Furthermore, each amino acid in a nonapeptide is described by 643 attributes. Thus, associated with each instance are a total of $9 \times 643 = 5787$ attributes. These attributes were taken from the literature, and represent position-independent indices, i.e. an amino acid 'X' has the same value for a property 'y' irrespective of its position in the peptide chain. No global descriptors for the entire peptide were included here (such as volume, molecular weight, or solvent accessible surface area). Using only the training data provided by the organizers, the nonapeptides in the unknown and test sets were classified as being either an epitope or a non-epitope; this is a problem of considerable current interest.

These nonapeptides are known as class I MHC peptides and are important in that they serve as indicators of the cellular proteomics relevant to cancer, autoimmune and viral diseases. Class I MHC peptides typically contain 8-10 amino acids, and for any given class I molecule they possess the same or similar amino acid residues at several defined positions along the peptide as well as a hydrophobic carboxy terminus. The presence of these conserved motifs gives an individual class I MHC molecule the ability to bind to a diverse spectrum of peptides. It is the existence of these motifs that makes it possible to predict which peptides will bind to a particular class I MHC molecule. The isolation of class I MHC-peptide complexes and the subsequent purification of the peptides displayed by different cell types and different MHC halotypes has given rise to large databases that can be searched for peptides derived from pathogen proteins or are involved in the induction of autoimmune reactions. The binding of peptide epitopes (moieties recognized by the immune system) and class I MHC proteins is a key part of immune response at the cellular level. For an overview together with applications to classification see Refs. [12-20]. Therefore, the development of accurate methods to predict whether relatively short peptides bind or not, and, if possible, to predict how strongly they will bind, is an important problem in computational immunology. While a number of approaches have previously been taken to address this problem, e.g., support vector machines [13,16], the CoEPrA 2006 competition provided a forum for a head to head comparison of a wide range of methods in their ability to classify nonapeptides as epitopes or non-epitopes.

The number of prediction and calibration instances and method of prediction ranking, denoted {No. Prediction instances, No. calibration instances, Ranking method}, were as follows: C1 {89,89,AUROC then MCC}; C3 {133,133, MCC}; C4 {111,111,MCC} where AUROC = area under the receiver operator curve (ROC) and MCC = Matthews correlation coefficient [20].

In the work to be described the calibration dataset was randomly split into a set of training and test instances such that, whenever possible, there were approximately an equal number of positive (known epitope) and negative (known non-epitope) cases in each dataset.

3. METHODS

Our initial application of RFs was done mainly on a proprietary dataset consisting of 7500 proteins which were to be divided into secreted (positive) and non-secreted (negative) classes. Associated with each protein were 75 different physical and chemical attributes (features) developed in-house at FivePrime Therapeutics[1]. These features were based on such quantities as charge density, hydrophobicity, molecular weight, isoelectric point, amino acid type, etc. *GenForest* was developed to reduce the number of features from 75 to 13 and these 13 led to classification with a sensitivity of 0.95 and a specificity of 0.89 (unpublished). This provided the starting point for the further development and subsequent application of *GenForest* to the CoEPrA datasets.

The *GenForest* algorithm consists of a RF algorithm [2] and a GA [7,12]; the latter controls and optimizes the subset of attributes sent to the RF. Initially, the calibration dataset is split randomly into test and training data. To initialize the GA the first generation of individuals are created by picking random subsets of attributes and training the RF on those attributes. The fitness of an individual is then determined by the performance of the RF on the test data. Once an initial population of individuals is generated the GA procedure is then used to evolve a new generation of individuals and the process is repeated for typically 40 generations. In this way the GA progressively iterates onto a near optimal set of attributes.

3.1. The Random Forest of Decision Trees

The RF algorithm was originally developed by Breiman [1]. The method requires a training set of known positives and negatives. A forest of random trees is then generated which perfectly classifies the negatives and positives. Presented with a new instance to classify, each individual tree in the forest would do poorly because it over fits the training data. However, since over-fitting is random across the forest, it turns out that the consensus of all the trees (the tree-vote) is a very good classifier. Suppose we have a dataset consisting of known positives and negatives which must be large enough, both in terms of number of attributes and total number of instances, so that the RF can be trained and tested. We conceptualize this data as N + 1 columns; the first column being the classifier which is typically a 0 (negative) or a 1 (positive) while the remaining N columns are the attributes or features. In our application each column of attributes is normalized to lie in the range (−1, 1). A particular random tree is constructed by recursively splitting the training data

[1] E.A. Lee, K. Hestir and L. T. Williams (unpublished)

on randomly chosen hyperplanes in the N −dimensional attribute space. The splitting continues until there is a single data point on a branch where the tree terminates in a leaf that is classified as positive or negative according to this single data point. The tree is thus defined by the randomly chosen hyperplanes and the positive or negative categories of each leaf. It classifies a new data point by sending the point to a leaf and giving the same positive or negative category as the leaf. In this way a random tree is similar to a nearest neighbor classifier in that points are given the same category as the training data point they are closest to when distance is measured by the tree. Notice that a single tree will perfectly classify the training data so any noise in this data is also represented in the tree. This makes a single tree a bad classifier but combining the random noise representation for all the trees into a collective prediction (the tree-vote) averages out this noise and improves the overall predictive accuracy. In general: the more trees in the forest, the more accurate the resulting prediction. The tree-vote is the average score over all of the trees in the forest and necessarily takes on a value between 0 and 1. A tree-vote of 0 represents a negative classification while a tree-vote of 1 is positive. Clearly, some cut-off has to be chosen in the range (0,1) above (below) which individuals are classified as positives (negatives). Examination of ROC curves provides a useful way to determine this cut-off in a given application.

3.2. The Genetic Algorithm

Genetic algorithms have proved to be extremely useful and efficient in optimization problems when the dimensionality of the problem is large [9]. In this case a GA is used to extract feature subsets that best classify input vectors of data for the CoEPrA-C2 datasets [10].

The problem is represented as follows. Each possible subset of attributes corresponds to an "individual" or "chromosome" in the GA algorithm and is represented by a string of bits of length N ; for example in CoEPrA-C1, C3 and C4 there are 5787 attributes and so N = 5787 in each case. Each chromosome is a bit-string of length 5787 with binary representation 1 or 0; a 1 means that particular attribute is included in the subset of attributes to be supplied to the RF while a 0 means that it is excluded. Note that the term "individual" is different from "instance". An individual can be thought of as a template which selects or unselects particular attributes from the 5787 attributes associated with each instance. The initial set of individuals is generated randomly with the caveat that each attribute is required to be represented at least once in the initial pool. The pool of individuals which constitutes the first generation is then evolved using the GA which consists of a number of operations.

3.2.1. Application of the GA Operators

Each individual is used to train a forest of decision trees. Then a fitness score is assigned to each individual based on how well the corresponding tree classifier classified the test dataset. A number of different fitness functions were employed initially, including; area under the resulting ROC curve, accuracy, Mathew's correlation coefficient (MCC), and error rate. The fitness function, *f*, that was finally arrived at was the following;

$$f = 1/(X - X_0) + 1/X_1 \quad (1)$$

where X_0 is the average of tree-votes for (known) non-epitopes (i.e. instances known to be classified correctly as 0) and X_1 is the average of the tree-votes values for (known) epitopes (i.e. instances known to be classified correctly as 1) in the test dataset. So, for example, perfect classification would yield $X_0 = 0$ and $X_1 = 1$ and a fitness, $f = 2$ which is the global minimum of the fitness function. Once the fitness has been calculated for all individuals, parent individuals are selected to undergo crossover with a probability proportional to their fitness.

The choice of fitness function in eq. (1) has some advantages over the perhaps more obvious choice of ROC area, especially when the test dataset contains a small number of instances. For example, it is possible that two different individuals might lead to almost perfect classification with, say, $X_0 = 0.2$ and $X_1 = 0.8$ for the first individual and $X_0 = 0.4$ and $X_1 = 0.6$ for the second. Depending on the specifics, these two individuals would have very similar or possibly identical ROC curve areas. However, it is clear that the first individual is classifying with a higher level of confidence than is the second. The fitness function, *f*, in eq. (1) attempts to take into account the level of confidence of the classification. In addition, it is simpler to evaluate eq. (1) as compared to the area under an ROC curve, a procedure which may also suffer from numerical error.

To produce the next generation of individuals, crossover was performed on two parent individuals. One-point crossover was used with the crossover point selected at random. To generate a child, data from the two parent-individuals were swapped. For example if a crossover point of 100 were selected then child No. 1 would be composed of the bit string consisting of positions 1-100 of parent' No.1 (P_1) and positions 101 − N of P_2. Parents are selected for crossover until a suitable number of individuals are produced for the next generation. In this case we used a population size of 12. Elitism was employed which means the best solution from each generation was copied unchanged to the next generation.

Mutation was included to prevent the GA from becoming trapped at a local minimum. In our implementation a feature is selected at random and the bit associated with it is then switched. Mutation is performed after selection and crossover with a mutation rate of 5%. In general 25 trees were used in the RF. To initialize the procedure 2-25 features are selected randomly. After initialization the number of features selected by the GA is not constrained. However, it does depend to some extent on the initial size range used. For example, picking 10-1000 features randomly at initialization would likely lead to individuals containing more features. For the parameters used an individual typically consisted of ≈ 20 − 30 features. These parameters were chosen because of the computational effort involved - in principle, larger values for each parameter should improve the results.

3.2.2. Termination

Due to the stochastic nature of the GA and the large dimension of the attribute space, only rarely will the GA converge to the identical solution in each run. This is the case even if *GenForest* is run using the same training and test sets but is initialized using a different random number seed. To address this problem *GenForest* was run multiple times

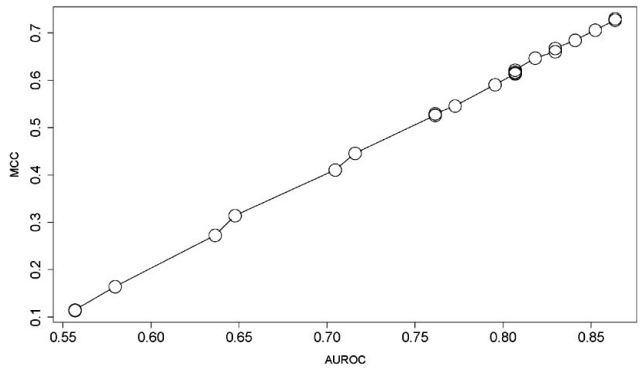

Fig. (1). Summary of the results for the entries into CoEPrA classification problem No. 1. AUROC = area under the ROC curve and MCC = Matthews Correlation Coefficient. The line follows the ranking order with top ranking lying in the top right-hand corner of the plot.

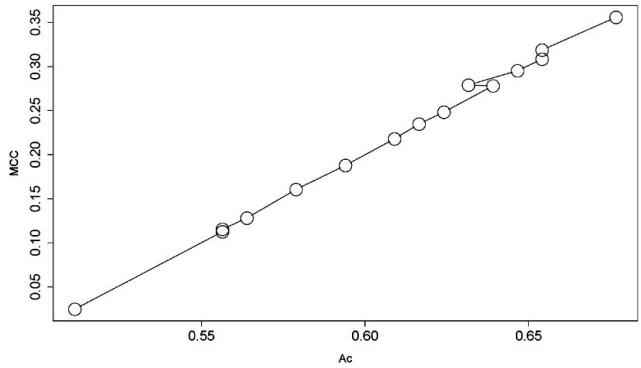

Fig. (2). Summary of the results for the entries into CoEPrA classification problem No. 3. Ac = accuracy and MCC = Matthews Correlation Coefficient. The line follows the ranking order with top ranking lying in the top right-hand corner of the plot.

on different initial splits of training and test data. To arrive at the final subset of optimal features the frequency of occurrence (FOO) of each attribute was computed across and all the independent runs of *GenForest*. Specifically, for problem C1 we started with a known data set containing 89 instances. The dataset was then split randomly into 60 instances to be used as training data and 29 instances to be used as test data. After 40 generations are evolved the most fit individual found by *GenForest* is then saved and the process repeated using (i) a different randomly selected set of 60 training cases and 29 test cases and (ii) a different seed. This entire process is repeated 1820 times and a FOO assigned to each attribute. A similar procedure was adopted for datasets C3 and C4 except that; 90 training and 43 test instances were used for problem C3 and; 80 training and 31 test instances were used for problem C4. To obtain the final subset of attributes, the attributes that are selected the most often across all random splits of the data are collected and combined into a final, super-individual. The cut-off for including attributes in the super-individual was obtained by optimizing its fitness using the training dataset. The super-individual is then used to classify the un-known dataset. This approach has the advantage of considerably reducing over-fitting and also produces a much greater consensus in attribute subsets across consecutive runs of the algorithm.

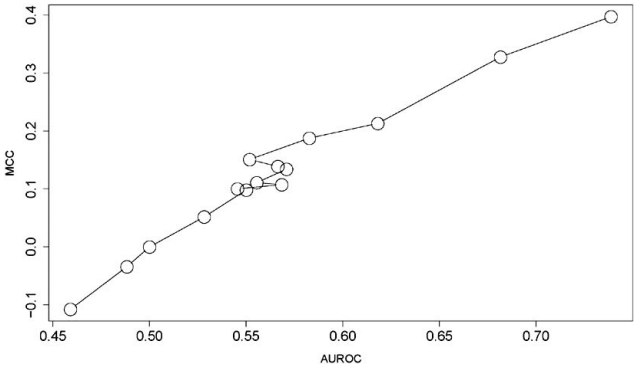

Fig. (3). Summary of the results for the entries into CoEPrA classification problem No. 4. See Fig. (**2**) for notations.

4. RESULTS

The complete results of applying *GenForest* as well as the methods used by the other entrants to the competition may

Table 1. Selected Results from the CoEPrA 2006 Competition for the CoEPrA-C1 Dataset

Rank/23	TP	TN	FP	FN	Se	Sp	Ac	MCC	AUROC
1	40	36	8	4	0.9091	0.8182	0.8636	0.7303	0.8636
10 (*GenForest*)	34	37	7	10	0.7727	0.8409	0.8068	0.6151	0.8068
23	25	24	20	19	0.5682	0.5455	0.5568	0.1137	0.5568

The full results may be found on the CoEPrA 2006 homepage together with descriptions of the various methods used [10]. TP = number of true positives; TN = number of true negatives; FP = number of false positives; FN = number of false negatives; Se = selectivity; Sp = specificity; Ac = accuracy; MCC = Matthews correlation coefficient; AUROC = area under ROC curve.

Table 2. Selected Results from the CoEPrA 2006 Competition for the CoEPrA-C3 Dataset. See Table 1 for Notations

Rank/16	TP	TN	FP	FN	Se	Sp	Ac	MCC
1	50	40	26	17	0.7463	0.6061	0.6767	0.3560
5 (*GenForest*)	32	52	14	35	0.4776	0.7879	0.6316	0.2791
16	27	41	25	40	0.4030	0.6212	0.5113	0.0248

Table 3. Selected Results from the CoEPrA 2006 Competition for the CoEPrA-C4 Dataset. See Table 1 for Notations

Rank/16	TP	TN	FP	FN	Se	Sp	Ac	MCC	AUROC
1	13	73	19	6	0.6842	0.7935	0.7748	0.3972	0.7388
9 (*GenForest*)	9	61	31	10	0.4737	0.6630	0.6306	0.1073	0.5684
16	16	7	85	3	0.8421	0.0761	0.2072	-0.1076	0.4591

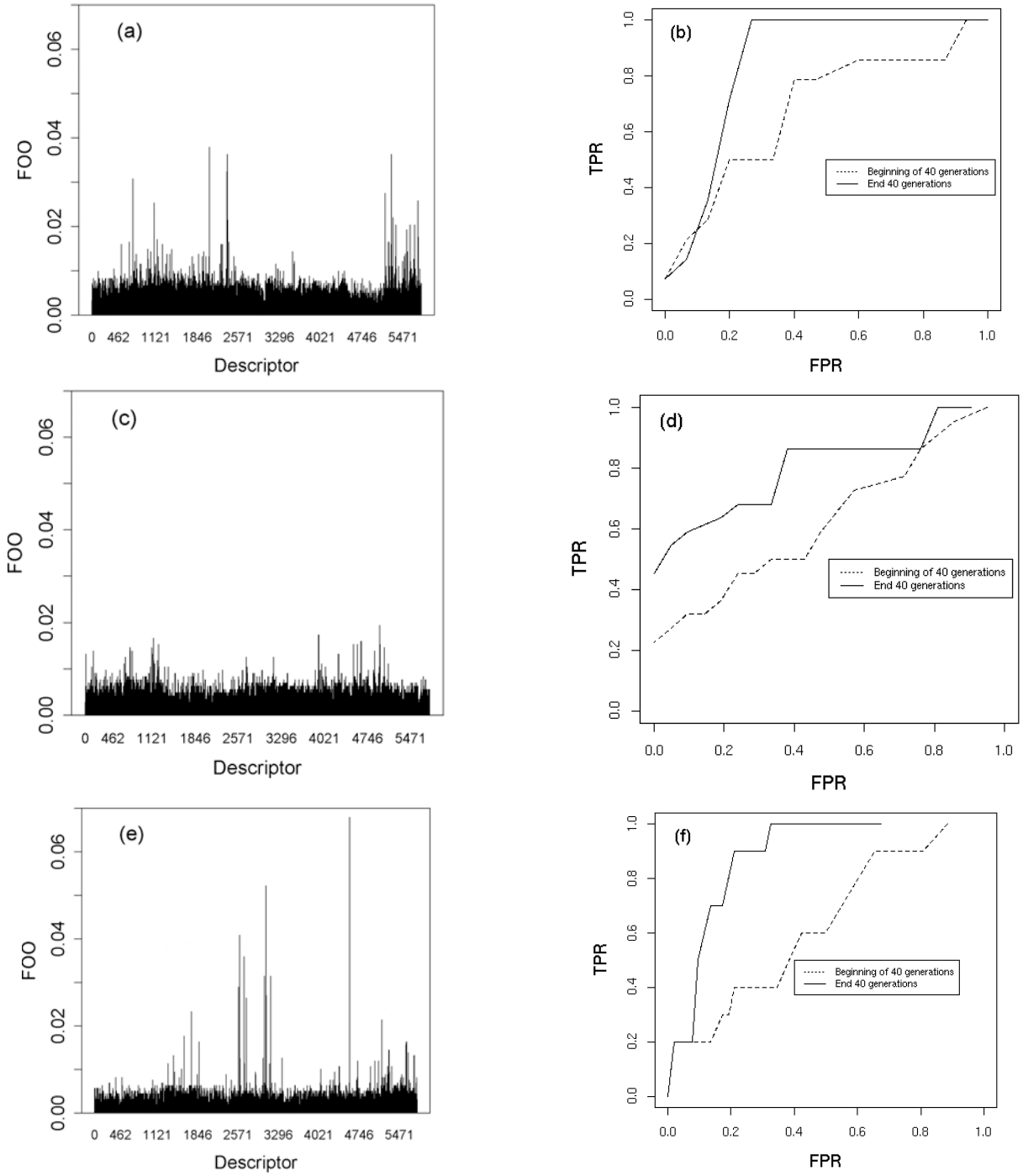

Fig. (4). Attribute frequency of occurrence (FOO) plots (left-hand column) and ROC (right-hand column) curves in the *GenForest* procedure as applied to the CoEPrA-C1 (**a** and **b**), C-3 (**c** and **d**), and C-4 (**e** and **f**) datasets. TPR = true positive rate and FPR = false positive rate. The FOO histograms were compiled during the application of *GenForest* to each of the three datasets over 40 generations and ≈ 1500 − 1800 runs depending on the problem. The ROC plots compare typical initial (dashed line) and final ROC curves after 40 generations (solid line).

be found at the CoEPrA 2006 competition home page [10]. Figs. (**1-3**) summarize these results for the datasets C1, C3 and C4. Tables **1-3** compare *GenForest* to the best and worst entries for each dataset. Fig. (**4**) provides examples taken from each classification problem showing how the ROC curve improves over the 40 generations used in *GenForest*. The ROC curves were generated from the calibration data. Despite the FOO histogram for the C4 dataset being the cleanest, *GenForest* performed significantly worse on this problem compare to problems C1, C2 [8] and C3. In the calibration dataset for C4 there are only 19 examples of positive instances. Hence random noise will play a larger role in classifying positives. This is because the number of negatives vastly outnumbers the number of positives. This is apparent in Table **3** where the method classifies 61/71 negative instances correctly but only 9/40 positives correctly. Incidentally, this explains why the ROC for the C4 calibration data in Fig. (**4f**) appears to be quite good whereas results for the unknown data are much poorer. Because the calibration dataset is skewed so strongly towards negative instances, *GenForest* does a good job of classifying them; however, even though it performs poorly on the positive instances, there are so few of them in the dataset that the ROC quality is not severely degraded. This is not the case in the unknown dataset.

5. CONCLUSIONS

The datasets used to test *GenForest* are of intrinsic interest because they are related to class I MHC peptides which serve as significant indicators of cellular proteomics related to various cancers, autoimmune and viral diseases.

Elsewhere [11] we have reported our application of *GenForest* to CoEPrA classification problem No. 2 where it scored 5/19. While *GenForest* did not provide the best classification in any of the four classification problems in the competition, it generally provided an acceptable and consistent level of classification. The worst performance observed was on Problem C4. We attribute this to the fact that the calibration dataset for this problem contained only 19 positive instances and 92 negative instances. Provided that roughly comparable numbers of positive and negative calibration instances are available, we conclude that *GenForest* can provide a fairly robust classification procedure. In particular, it is most useful when the number of attributes available for classification greatly exceeds the number of available instances. In addition, the method is easy to parallelize. Possible future improvements include (i) increasing the size of the forest from 25 to, say, 100 trees; (ii) increasing the number of features used to initialize the GA; (iii) allowing the number of attributes in the subset selected to itself be a variable in the fitness function; (iv) modifying the fitness function to try to improve the selectivity and specificity independently. In addition, optimizing the cut-off for including attributes in the super-individual would probably improve the overall performance.

ACKNOWLEDGEMENTS

The work was supported, in part, by the National Science Foundation though grants 0202185 and 0718547 to Utah State University and by MEC-Spain, under contract - SAB 2006-0086.

REFERENCES

[1] Breiman, L. *Mach. Learn.*, **2001**, *45*, 5.
[2] Breiman, L.; Friedman; J. H.; Olshen, R.A.; Stone, C.J. *Classification and Regression Trees*; Wadsworth International Group: Belmot, CA, **1984**.
[3] Almuallim, H.; Diettrich; T.G. *Artif. Intell.*, **1994**, *69*, 279.
[4] Brill, F.; Brown, D.; Martin, W. *IEEE Trans. Neural Netw.*, **1992**, *3*, 324.
[5] Salzberg, S. *Mach. Learn.*, **1993**, *10*, 57.
[6] Yang, J. H.; Honavar, V. *IEEE Intell. Syst. Appl.*, **1998**, *13*, 44.
[7] Riedesel, H.; Kolbeck, B.; Schmetzer, O.; Knapp, E. W. *Genome Inform.*, **2004**, *15*, 198.
[8] Wuju, L.; Momiao, X. *Bioinformatics*, **2002**, *18*, 325-326.
[9] Goldberg, E. *Genetic Algorithms in Search, Optimization & Machine Learning*; Addison Wesley: Reading, MA, **1989**.
[10] CoEPrA 2006 competition, **2006**, http://www.coepra.org.
[11] Lee, E. A.; Hansen, L.; Hestir, K.; Williams, L. T.; Farrelly, D. *Protein Pept. Lett.*, **2008**, (in press).
[12] Goldsby, R. A., Kindt, T. J.; Osborne, B. A. *Immunology*; 4th ed., Chaps. 7-10, W. H. Freeman & Co.: New York, NY, **2000**.
[13] Zhao, Y. D.; Pinilla, C.; Valmori, D.; Martin, R.; Simon, R.. *Bioinformatics*, **2003**, 19, 1978.
[14] Admon, A.; Barnea, E.; Ziv, T. *Mol. Cell. Proteom.*, **2003**, *2*, 388.
[15] Honeyman, M. C.; Brusic, N.; Stone, L.; Harrison, L. C. *Nat. Biotech.*, **1998**, 16, 966.
[16] Liu, W.; Meng, X. S.; Xu, Q. Q.; Flower, D. R.; Li, T. B. *BMC Bioinformatics*, **2006**, *7*, 182.
[17] Tong, J. C.; Tan, T. W.; Ranganathan, S. *Brief Bioinform.*, **2007**, *8*, 96.
[18] Dotchinova, I. A.; Flower, D. R.; *J. Chem. Inf. Mod.*, **2007**, *47*, 234.
[19] Hattotuwagama, C. K.; Guan, P. P.; Dotchinova, I. A.; Flower, D. R. *Org. Bio. Chem.*, **2004**, *2*, 3274.
[20] Matthews, B. W. *Biochim. Biophys. Acta,* **1975**, *405*, 442.

Received: December 6, 2008 **Revised: February 6, 2009** **Accepted: February 10, 2009**