

PREDICTING STUDENT PERFORMANCE: AN APPLICATION OF DATA MINING METHODS WITH THE EDUCATIONAL WEB-BASED SYSTEM LON-CAPA

Behrouz Minaei-Bidgoli¹, Deborah A. Kashy², Gerd Kortemeyer³, William F. Punch⁴

Abstract □ Newly developed web-based educational technologies offer researchers unique opportunities to study how students learn and what approaches to learning lead to success. Web-based systems routinely collect vast quantities of data on user patterns, and data mining methods can be applied to these databases. This paper presents an approach to classifying students in order to predict their final grade based on features extracted from logged data in an education web-based system. We design, implement, and evaluate a series of pattern classifiers and compare their performance on an online course dataset. A combination of multiple classifiers leads to a significant improvement in classification performance. Furthermore, by learning an appropriate weighting of the features used via a genetic algorithm (GA), we further improve prediction accuracy. The GA is demonstrated to successfully improve the accuracy of combined classifier performance, about 10 to 12% when comparing to non-GA classifier. This method may be of considerable usefulness in identifying students at risk early, especially in very large classes, and allow the instructor to provide appropriate advising in a timely manner.

Index Terms – Data Mining, Classification, Prediction, Combination of Multiple Classifiers, Genetic Algorithm

INTRODUCTION

Many leading educational institutions are working to establish an online teaching and learning presence. Several systems with different capabilities and approaches have been developed to deliver online education in an academic setting. In particular, Michigan State University (MSU) has pioneered some of these systems to provide an infrastructure for online instruction. The research presented here was performed on a part of the latest online educational system developed at MSU, the *Learning Online Network with Computer-Assisted Personalized Approach (LON-CAPA)* [1].

Two large databases are being developed in LON-CAPA. The first contains educational resources such as web pages, demonstrations, simulations, and individualized problems designed for use on homework assignments,

quizzes, and examinations [2]. As more instructors develop educational materials for their courses to use with the LON-CAPA system, the content of this database grows.

The second database contains information about student users of LON-CAPA. This database stores a wide range of variables (to be described shortly) including when, for how long, and how many times they access each resource, the number of correct responses they give on assigned problems, their pattern of correct and incorrect responses, and so on. Needless to say, with each semester, and as more instructors adopt the LON-CAPA system, this database grows rapidly.

In this study we apply data mining methods to the LON-CAPA databases with the goals of answering the following two research questions:

- 1) Can we find *classes* of students? In other words, do there exist groups of students who use these online resources in a *similar* way? If so, can we identify that class for any individual student? With this information, can we *help* a student use the resources better, based on the usage of the resource by other students in their groups?
- 2) Can we *classify* the problems that have been used by students? If so, can we show how different types of problems impact students' achievement? Can we *help* instructors to develop the homework more effectively and efficiently?

Some research and experiments have been done to reply the second research question [3], [4]. In this paper, regarding the first research question, we hope to find similar patterns of use in the data gathered from LON-CAPA, and eventually be able to make predictions as to the most-beneficial course of studies for each learner based on their present usage. The system could then make suggestions to the learner as to how to best proceed.

DATASET, CLASS LABELS, FEATURES

As test data we selected the student and course data of an introductory physics course for scientists and engineers (PHY183), which was held at MSU in spring semester 2002. This course included 12 homework sets with a total of 184 problems, all of which were done online using LON-CAPA.

¹ Behrouz Minaei-Bidgoli, Michigan State University, Department of Computer Science, Genetic Algorithms Research and Applications Group (GARAGE), minaeibi@cse.msu.edu

² Deborah A. Kashy, Michigan State University, Department of Psychology, kashyd@msu.edu

³ Gerd Kortemeyer, Michigan State University, Division of Science and Math Education, korte@lite.msu.edu

⁴ William F. Punch, Michigan State University, Department of Computer Science and Engineering, Genetic Algorithms Research and Applications Group (GARAGE), punch@cse.msu.edu

About 261 students were initially enrolled in the course, however some of the students dropped the course after doing a couple of homework sets, so they do not have any final grades. After removing those students, there remained 227 valid samples. The final grade distribution of the students is shown in FIGURE 1.

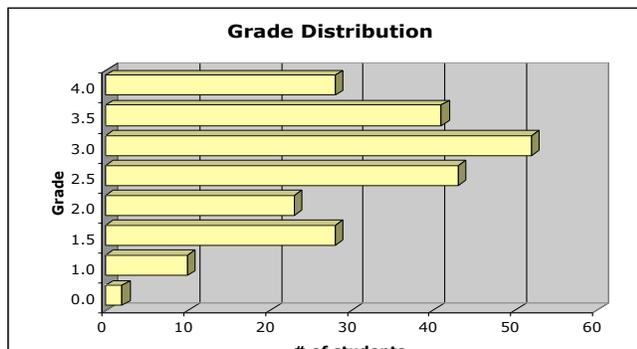


FIGURE 1
GRAPH OF DISTRIBUTION OF GRADES IN COURSE PHY183

We can group the students regarding their final grades in several ways, 3 of which are:

1. The 9 possible class labels can be the same as students' grades, as shown in table 1
2. We can group them into three classes, "high" representing grades from 3.5 to 4.0, "middle" representing grades from 2.5 to 3, and "low" representing grades less than 2.5, as shown in table 2.
3. We can also categorize students with one of two class labels: "Passed" for grades above 2.0, and "Failed" for grades less than or equal to 2.0, as shown in table 3.

TABLE 1

SELECTING 9 CLASS LABELS REGARDING TO STUDENTS' GRADES

Class	Grade	Student #	Percentage
1	0.0	2	0.9%
2	0.5	0	0.0%
3	1.0	10	4.4%
4	1.5	28	12.4%
5	2.0	23	10.1%
6	2.5	43	18.9%
7	3.0	52	22.9%
8	3.5	41	18.0%
9	4.0	28	12.4%

TABLE 2

SELECTING 3 CLASS LABELS REGARDING TO STUDENTS' GRADES

Class	Grade	Student #	Percentage
High	Grade >= 3.5	69	30.40%
Middle	2.0 < Grade < 3.5	95	41.80%
Low	Grade <= 2.0	63	27.80%

TABLE 3

SELECTING 2 CLASS LABELS REGARDING TO STUDENTS' GRADES

Class	Grade	Student #	Percentage
Passed	Grade > 2.0	164	72.2%
Failed	Grade <= 2.0	63	27.80%

We can predict that the error rate in the first class grouping should be higher than the others, because the distributions of the grades over 9 classes are so different.

Extractable Features

An essential step in doing classification is selecting the features used for classification. Below we discuss the features from LON-CAPA that were used, how they can be visualized (to help in selection) and why we normalize the data before classification.

The following features are stored by the LON-CAPA system:

1. Total number of correct answers. (Success rate)
2. Getting the problem right on the first try, vs. those with high number of tries. (Success at the first try)
3. Total number of tries for doing homework. (Number of attempts before correct answer is derived)
4. Time spent on the problem until solved (more specifically, the number of hours until correct. The difference between time of the last successful submission and the first time the problem was examined). Also, the time at which the student got the problem correct relative to the due date. Usually better students get the homework completed earlier.
5. Total time spent on the problem regardless of whether they got the correct answer or not. (Difference between time of the last submission and the first time the problem was examined).
6. Participating in the communication mechanisms, vs. those working alone. LON-CAPA provides online interaction both with other students and with the instructor. Were these used?
7. Reading the supporting material before attempting homework vs. attempting the homework first and then reading up on it.
8. Submitting a lot of attempts in a short amount of time without looking up material in between, versus those giving it one try, reading up, submitting another one, and so forth.
9. Giving up on a problem versus students who continued trying up to the deadline.
10. Time of the first log on (beginning of assignment, middle of the week, last minute) correlated with the number of tries or number of solved problems. A student who gets all correct answers will not necessarily be in the successful group if they took an average of 5 tries per problem, but it should be verified from this research.

The present classification experiment focuses on the first six features based on the PHY183 Spring 2002 class data.

CLASSIFICATION

Pattern recognition has a wide variety of applications in many different fields, such that it is not possible to come up with a single classifier that can give good results in all cases.

The optimal classifier in every case is highly dependent on the problem domain. In practice, one might come across a case where no single classifier can classify with an acceptable level of accuracy. In such cases it would be better to pool the results of different classifiers to achieve the optimal accuracy. Every classifier operates well on different aspects of the training or test feature vector. As a result, assuming appropriate conditions, combining multiple classifiers may improve classification performance when compared with any single classifier.

The scope of this survey is restricted to comparing some popular non-parametric pattern classifiers and a single parametric pattern classifier according to the error estimate. Six different classifiers using the LON-CAPA datasets are compared in this study. The classifiers used in this study include *Quadratic Bayesian classifier*, *1-nearest neighbor (1-NN)*, *k-nearest neighbor (k-NN)*, *Parzen-window*, *multi-layer perceptron (MLP)*, and *Decision Tree*.⁵ These classifiers are some of the common classifiers used in most practical classification problems. After some preprocessing operations were made on the dataset, the error rate of each classifier is reported. Finally, to improve performance, a combination of classifiers is presented.

Normalization

Having assumed in Bayesian and Parzen-window classifiers that the features are normally distributed, it is necessary that the data for each feature be normalized. This ensures that each feature has the same weight in the decision process. Assuming that the given data is Gaussian distributed, this normalization is performed using the mean and standard deviation of the training data. In order to normalize the training data, it is necessary first to calculate the sample mean \bar{x} , and the standard deviation σ of each feature, or column, in this dataset, and then normalize the data using the equation(1).

$$x_i = \frac{x_i - \bar{x}}{\sigma} \quad (1)$$

This ensures that each feature of the training dataset has a normal distribution with a mean of zero and a standard deviation of one. In addition, the kNN method requires normalization of all features into the same range. However, we should be cautious in using the normalization before considering its effect on classifiers' performances.

Combination of Multiple Classifiers (CMC)

By combining multiple classifiers we hope to improve classifier performance. There are different ways one can think of combining classifiers:

- The simplest way is to find the overall error rate of the classifiers and choose the one which has the least error rate on the given dataset. This is called an *offline CMC*.

This may not really seem to be a CMC; however, in general, it has a better performance than individual classifiers.

- The second method, which is called *online CMC*, uses all the classifiers followed by a vote. The class getting the *maximum votes* from the individual classifiers will be assigned to the test sample. This method intuitively seems to be better than the previous one. However, when tried on some cases of our dataset, the results were not better than the best result in previous method. So, we changed the rule of majority vote from "*getting more than 50% votes*" to "*getting more than 75% votes*". This resulted in a significant improvement over offline CMC.

Using the second method, we showed in [18] that CMC can achieve a significant accuracy improvement in all three cases of 2, 3, and 9-classes for the PHY183 data, in which we are going to use GA to optimize the CMC performance.

MAP THE PROBLEM TO GENETIC ALGORITHM

Genetic Algorithms have been shown to be an effective tool to use in data mining and pattern recognition. [11], [14], [10], [21], [20], [17], [8]. An important aspect of GAs in a learning context is their use in pattern recognition. There are two different approaches to applying GA in pattern recognition:

- Apply a GA directly as a classifier. Bandyopadhyay and Murthy in [7] applied GA to find the decision boundary in N dimensional feature space.
- Use a GA as an optimization tool for resetting the parameters in other classifiers. Most applications of GAs in pattern recognition optimize some parameters in the classification process. Many researchers have used GAs in feature selection [6], [13], [16]. GAs has been applied to find an optimal set of feature weights that improve classification accuracy. First, a traditional feature extraction method such as Principal Component Analysis (PCA) is applied, and then a classifier such as k-NN is used to calculate the fitness function for GA [21], [5]. Combination of classifiers is another area that GAs have been used to optimize. Kuncheva and Jain in [11] used a GA for selecting the features as well as selecting the types of individual classifiers in their design of a Classifier Fusion System. GA is also used in selecting the prototypes in the case-based classification [23].

In this paper we will focus on the second approach and use a GA to optimize a combination of classifiers. Our objective is to *predict* the students' final grades based on their web-use features, which are extracted from the homework data. We design, implement, and evaluate a series of pattern classifiers with various parameters in order to

⁵ The first five classifiers are coded in MATLAB™ 6.0, and for the decision tree classifiers we have used some available software packages such as C5.0, CART, QUEST, and CRUISE.

compare their performance on a dataset from LON-CAPA. Error rates for the individual classifiers, their combination and the GA optimized combination are presented.

Optimizing the CMC Using a GA

We used GAToolBox⁶ for MATLAB to implement a GA to optimize classification performance. Our goal is to find a population of best weights for every feature vector, which minimize the classification error rate. The feature vector for our predictors are the set of six variables for every student: Success rate, Success at the first try, Number of attempts before correct answer is derived, the time at which the student got the problem correct relative to the due date, total time spent on the problem, and the number of online interactions of the student both with other students and with the instructor.

We randomly initialized a population of six dimensional weight vectors with values between 0 and 1, corresponding to the feature vector and experimented with different number of population sizes. We found good results using a population with 200 individuals. The GA Toolbox supports binary, integer, real-valued and floating-point chromosome representations. We used the simple genetic algorithm (SGA), which is described by Goldberg in [9]. The SGA uses common GA operators to find a population of solutions which optimize the fitness values.

Fitness Function

During the reproduction phase, each individual is assigned a fitness value derived from its raw performance measure given by the objective function. This value is used in the selection to bias towards more fit individuals. Highly fit individuals, relative to the whole population, have a high probability of being selected for mating whereas less fit individuals have a correspondingly low probability of being selected. The error rate is measured in each round of cross validation by dividing “the total number of misclassified examples” into “total number of test examples”. Therefore, our *fitness function* measures the error rate achieved by CMC and our objective would be to maximize this performance (minimize the error rate).

EXPERIMENT RESULTS

Without using GA, the overall results of classifiers’ performance on our dataset are shown in the Table 4. Regarding individual classifiers, for the case of 2-classes, kNN has the best performance with 82.3% accuracy. In the case of 3-classes and 9-classes, CART has the best accuracy of about 60% in 3-classes and 43% in 9-Classes. However, considering the combination of non-tree-based classifiers, the CMC has the best performance in all three cases. That is, it achieved 86.8% accuracy in the case of 2-Classes, 71% in the case of 3-Classes, and 51% in the case of 9-Classes.

⁶ Downloaded from <http://www.shef.ac.uk/~gaipp/ga-toolbox/>

TABLE 4

COMPARING THE ERROR RATE OF ALL CLASSIFIERS ON PHY183 DATASET IN THE CASES OF 2-CLASSES, 3-CLASSES, AND 9-CLASSES, USING 10-FOLD CROSS VALIDATION, WITHOUT GA

		Performance %		
Classifier		2-Classes	3-Classes	9-Classes
Tree Classifier	C5.0	80.3	56.8	25.6
	CART	81.5	59.9	33.1
	QUEST	80.5	57.1	20.0
	CRUISE	81.0	54.9	22.9
Non-tree Classifier	Bayes	76.4	48.6	23.0
	1NN	76.8	50.5	29.0
	kNN	82.3	50.4	28.5
	Parzen	75.0	48.1	21.5
	MLP	79.5	50.9	-
	CMC	86.8	70.9	51.0

For GA optimization, we used 200 individuals in our population, running the GA for 500 generations. We ran the program 10 times and show the averages in table 5. In every run, 500 \times 200 times fitness function are used in the 10-fold cross validation to measure the average performance of CMC. Thus every classifier is called 3 \times 10⁶ times for the case of 2-classes, 3-classes and 9-classes. The time overhead for fitness evaluation is therefore a critical issue. Since using the MLP in this process takes about 2 minutes while the other four non-tree classifiers (Bayes, 1NN, 3NN, and Parzen window) take only 3 seconds collectively, we omitted the MLP from our classifiers group so we could obtain the results in a reasonable time.

TABLE 5

COMPARING THE CMC PERFORMANCE ON PHY183 DATASET USING GA AND WITHOUT GA IN THE CASES OF 2-CLASSES, 3-CLASSES, AND 9-CLASSES, 95% CONFIDENCE INTERVAL.

		Performance %		
Classifier		2-Classes	3-Classes	9-Classes
CMC of 4 Classifiers without GA		83.87 \pm 1.73	61.86 \pm 2.16	49.74 \pm 1.86
GA Optimized CMC, Mean individual		94.09 \pm 2.84	72.13 \pm 0.39	62.25 \pm 0.63
Improvement		10.22 \pm 1.92	10.26 \pm 1.84	12.51 \pm 1.75

The results in Table 5 represent the mean performance with a two-tailed t-test at a 95% confidence interval. For the improvement of the GA over non-GA result, a P-value indicating the probability of the Null-Hypothesis (There is no improvement) is also given, showing the significance of the GA optimization. All have $p < 0.000$, indicating significant improvement. Therefore, using GA, we observe more than a 10% mean individual performance improvement and about 12% to 15% mean individual performance improvement. Figure 2 shows the graph of average mean individual performance improvement.

FIGURE 2

CHART OF COMPARING CMC AVERAGE PERFORMANCE, USING GA AND WITHOUT GA.

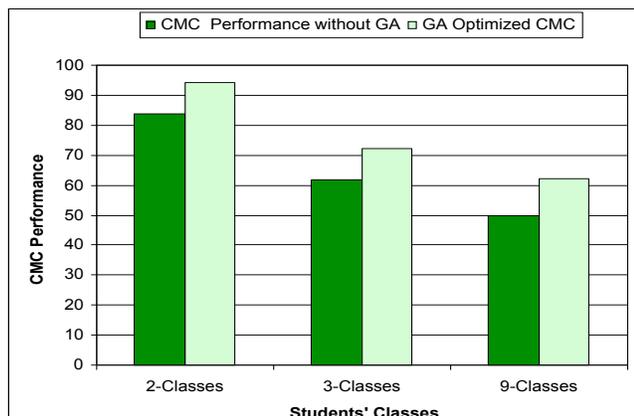
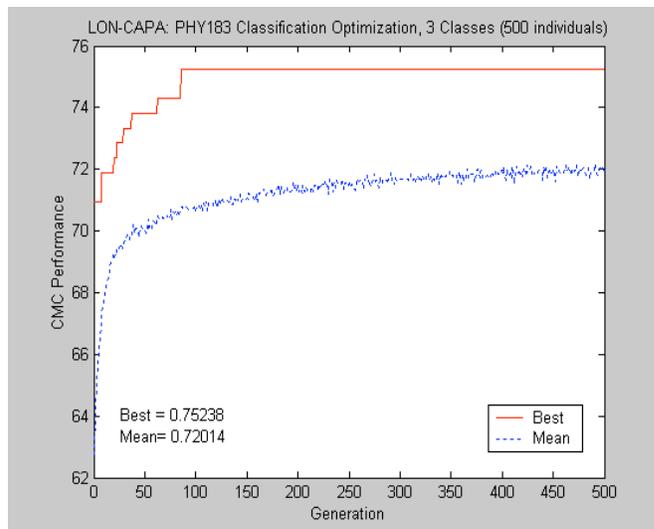
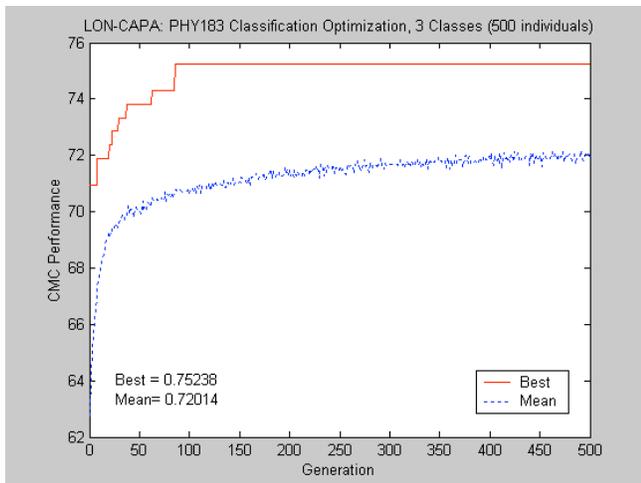
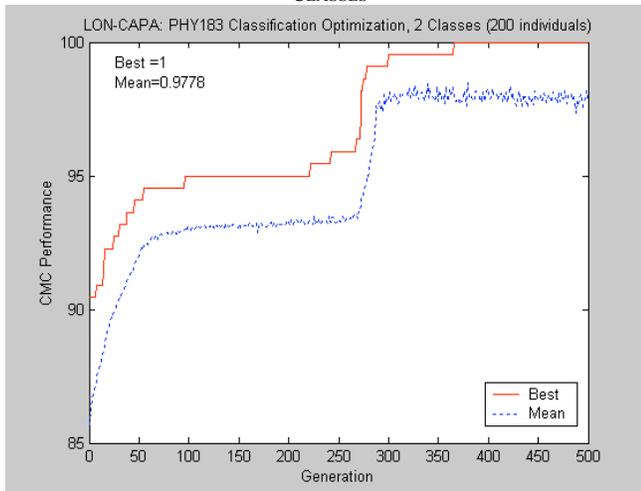


Figure 3 shows the best result of the ten runs over our dataset. These charts represent the population mean, the best individual at each generation and the best value yielded by the run.

FIGURE 3

GRAPH OF GA OPTIMIZED CMC PERFORMANCE IN THE CASE OF 2, 3, AND 9-CLASSES



Finally, we can examine the individuals (weights) for features by which we obtained the improved results. This feature weighting indicates the importance of each feature for making the required classification. In most cases the results are similar to Multiple Linear Regressions or tree-based software that use statistical methods to measure feature importance. Table 6 shows the importance of the six features in the 3-classes case using the Entropy splitting criterion. Based on entropy, a statistical property called *information gain* measures how well a given feature separates the training examples in relation to their target classes. Entropy characterizes *impurity* of an arbitrary collection of examples S at a specific node N . In [9] the impurity of a node N is denoted by $i(N)$ such that:

$$Entropy(S) = i(N) = - \sum_j P(\varpi_j) \log_2 P(\varpi_j) \quad (3)$$

where $P(\varpi_j)$ is the fraction of examples at node N that go to category ϖ_j .

TABLE 6

FEATURE IMPORTANCE IN 3-CLASSES USING ENTROPY CRITERION

Feature	Importance %
Total_Correct_Answers	100.00
Total_Number_of_Tries	58.61
First_Got_Correct	27.70
Time_Spent_to_Solve	24.60
Total_Time_Spent	24.47
Communication	9.21

The GA results also show that the “Total number of correct answers” and the “Total number of tries” are the most important features for the classification. The second column in table 6 shows the percentage of feature importance.

As a result, having the information generated through our experiment the instructor would be able to identify students at risk early, especially in very large classes, and allow the instructor to provide appropriate advising in a timely manner.

CONCLUSIONS AND FUTURE WORK

Four classifiers were used to segregate the students. A combination of multiple classifiers leads to a significant accuracy improvement in all 3 cases. Weighing the features and using a genetic algorithm to minimize the error rate improves the prediction accuracy at least 10% in the all cases of 2, 3 and 9-Classes. In cases where the number of features is low, the feature weighting worked much better than feature selection. The successful optimization of student classification in all three cases demonstrates the merits of using the LON-CAPA data to *predict* the students' final grades based on their features, which are extracted from the homework data.

We are going to gather more sample data by combining one course data during several semesters to avoid overfitting in the case of 9-Classes. We also try to find the paths that students usually choose to solve the different types of the problems from activity log to extract more relevant features. We also want to apply Evolutionary Algorithms to find Association Rules and Dependency among the groups of problems (*Mathematical, Optional Response, Numerical, Java Applet, and so forth*) of LON-CAPA homework data sets.

As more and more students enter the online learning environment, databases concerning student access and study patterns will grow. In this paper we have shown that data mining efforts can be useful in predicting student outcomes. We hope to refine our techniques so that the information generated by data mining can be usefully applied by instructors to increase student learning.

ACKNOWLEDGMENT

This work was partially supported by the National Science Foundation under ITR 0085921, with additional support by the Andrew W. Mellon and Alfred P. Sloan foundations.

REFERENCES

- [1] Kortemeyer, G., Bauer, W., Kashy, D. A., Kashy, E., & Speier, C., "The LearningOnline Network with CAPA Initiative", *Proceedings of the Frontiers in Education conference*, 2001. See also: <http://www.lon-capa.org>
- [2] Kashy, D. A., Albertelli, G., Ashkenazi, G., Kashy E. Ng, H. K., & Thoennessen, M., "Individualized interactive exercises: A promising role for network technology", *Proceedings of the Frontiers in Education conference*, 2001.
- [3] Albertelli, G., Minaei-Bidgoli, B., Punch, W.F., Kortemeyer, G., & Kashy, E., "Concept Feedback In Computer-Assisted Assignments", *Proceedings of the Frontiers in Education conference*, 2002.
- [4] Kashy, D. A., Albertelli, G., Kashy, E. & Thoennessen, M. "Teaching with ALN Technology: Benefits and Costs", *Journal of Engineering Education*, 90, 499-506, 2001.
- [5] Punch, W.F., Pei, M., Chia-Shun, L., Goodman, E.D., Hovland, P., and Enbody R. "Further research on Feature Selection and Classification Using Genetic Algorithms", In 5th International Conference on Genetic Algorithm, Champaign IL, pp 557-564, 1993.
- [6] Bala J., De Jong K., Huang J., Vafaie H., and Wechsler H. "Using learning to facilitate the evolution of features for recognizing visual concepts". *Evolutionary Computation* 4(3) - Special Issue on Evolution, Learning, and Instinct: 100 years of the Baldwin Effect. 1997.
- [7] Bandyopadhyay, S., and Muthy, C.A. "Pattern Classification Using Genetic Algorithms", *Pattern Recognition Letters*, Vol. 16, 1995, pp. 801-808.
- [8] De Jong K.A., Spears W.M. and Gordon D.F. (1993). "Using genetic algorithms for concept learning". *Machine Learning* 13, 161-188, 1993.
- [9] Duda, R.O., Hart, P.E., and Stork, D.G. *Pattern Classification*. 2nd Edition, John Wiley & Sons, Inc., New York NY. 2001.
- [10] Falkenauer E. *Genetic Algorithms and Grouping Problems*. John Wiley & Sons, 1998.
- [11] Freitas, A.A. "A survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery". See: www.pgia.pucpr.br/~alex/papers. To appear in: A. Ghosh and S. Tsutsui. (Eds.) *Advances in Evolutionary Computation*. Springer-Verlag, 2002.
- [12] Goldberg, D.E. "Genetic Algorithms in Search, Optimization, and Machine Learning", MA, Addison-Wesley. 1989.
- [13] Guerra-Salcedo C. and Whitley D. "Feature Selection mechanisms for ensemble creation: a genetic search perspective". *Freitas AA (Ed.) Data Mining with Evolutionary Algorithms: Research Directions – Papers from the AAAI Workshop*, 13-17. Technical Report WS-99-06. AAAI Press, 1999.
- [14] Jain, A. K.; Zongker, D. "Feature Selection: Evaluation, Application, and Small Sample Performance", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 2, February 1997.
- [15] Kuncheva, L.I., and Jain, L.C., "Designing Classifier Fusion Systems by Genetic Algorithms", *IEEE Transaction on Evolutionary Computation*, Vol. 33 2000, pp 351-373.
- [16] Martin-Bautista MJ and Vila MA. "A survey of genetic feature selection in mining issues". *Proceeding Congress on Evolutionary Computation (CEC-99)*, 1314-1321. Washington D.C., July (1999).
- [17] Michalewicz Z. "Genetic Algorithms + Data Structures = Evolution Programs". 3rd Ed. Springer-Verlag, 1996.
- [18] Minaei-Bidgoli, B., and Punch W.F.; "Using Genetic Algorithms for Data Mining Optimization in an Educational Web-based System". to be appear in *Proceeding of Genetic and Evolutionary Computation Conference*, July 2003.
- [19] Muhlenbein and Schlierkamp-Voosen D., "Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization", *Evolutionary Computation*, Vol. 1, No. 1, 1993, pp. 25-49.
- [20] Park Y and Song M. "A genetic algorithm for clustering problems. Genetic Programming" *Proceeding of 3rd Annual Conference*, 568-575. Morgan Kaufmann, 1998.
- [21] Pei, M., Goodman, E.D., and Punch, W.F. "Pattern Discovery from Data Using Genetic Algorithms", *Proceeding of 1st Pacific-Asia Conference Knowledge Discovery & Data Mining (PAKDD-97)*. Feb. 1997.
- [22] Pei, M., Punch, W.F., and Goodman, E.D. "Feature Extraction Using Genetic Algorithms", *Proceeding of International Symposium on Intelligent Data Engineering and Learning '98 (IDEAL '98)*, Hong Kong, Oct. 1998.
- [23] Skalak D. B. "Using a Genetic Algorithm to Learn Prototypes for Case Retrieval and Classification". *Proceeding of the AAAI-93 Case-Based Reasoning Workshop*, pp. 64-69, 1994.