Missouri S&T Library and Learning Resources

Scholars' Mine

Masters Theses

Student Theses and Dissertations

Spring 2015

Rote-LCS learning classifier system for classification and prediction

Benjamin Daniels

Follow this and additional works at: https://scholarsmine.mst.edu/masters_theses

 Part of the Computer Sciences Commons, and the Operations Research, Systems Engineering and Industrial Engineering Commons

Department:

Recommended Citation

Daniels, Benjamin, "Rote-LCS learning classifier system for classification and prediction" (2015). *Masters Theses*. 7424.
https://scholarsmine.mst.edu/masters_theses/7424

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

ROTE-LCS LEARNING CLASSIFIER SYSTEM FOR CLASSIFICATION AND PREDICTION

by

BENJAMIN DANIELS

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

2015

Approved by

Dr. Steven M. Corns, Advisor

Dr. Daniel R. Tauritz, Co-Advisor

Dr. Elizabeth A. Cudney

© 2015
Benjamin Daniels
All Rights Reserved

PUBLICATION THESIS OPTION

This consists of the following articles that have been and will be submitted for publication as follows:

Pages 5-25 submitted to IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, 2012.

Pages 26-48 submitted to IEEE World Congress on Computational Intelligence, 2012.

Pages 49-65 pending submission to IEEE Transaction in Evolutionary Computation.

ABSTRACT

Machine Learning (ML) involves the use of computer algorithms to solve for approximate solutions to problems with large, complex search spaces. Such problems have no known solution method, and search spaces too large to allow brute force search to be feasible. Evolutionary algorithms (EA) are a subset of machine learning algorithms which simulate fundamental concepts of evolution. EAs do not guarantee a perfect solution, but rather facilitate convergence to a solution of which the accuracy depends on a given EA's learning architecture and the dynamics of the problem.

Learning classifier systems (LCS) are algorithms comprising a subset of EAs. The Rote-LCS is a novel Pittsburgh-style LCS for supervised learning problems. The Rote models a solution space as a hyper-rectangle, where each independent variable represents a dimension. Rote rules are formed by binary trees with logical operators (decision trees) with relational hypotheses comprising the terminal nodes. In this representation, sub-rules (minor-hypotheses) are partitions on hyper-planes, and rules (major-hypotheses) are multidimensional partitions. The Rote-LCS has exhibited very high accuracy on classification problems, particularly Boolean problems, thus far. The Rote-LCS offers an additional attribute uncommon among machine learning algorithms – human readable solutions. Despite representing a multidimensional search space, Rote solutions may be graphed as two-dimensional trees. This makes the Rote-LCS a good candidate for supervised classification problems where insight is needed into the dynamics of a problem. Solutions generated by Rote-LCS could prospectively be used by scientists to form hypotheses regarding interactions between independent variables of a given problem.

ACKNOWLEDGEMENTS

I'd like to thank my family for the financial support they provided, as well as my advisor, Dr. Corns, for sponsoring my research, even when I pursued my own projects he wasn't receiving funding for; being supported to pursue my own research, I feel, was a very significant thing for me and helped me to achieve more. I'd like to thank Dr. Tauritz for teaching me evolutionary algorithms, which has empowered me so much, and has enabled me to make some of my most important contributions and will continue to do so, I'm sure, well into the future. I'd like to thank Dr. Elizabeth Cudney for initiating the research that led to my work in machine learning and for co-authoring papers with me. I'd like to thank Dr. Chellappan for teaching me about the state of the art of anonymous network security, Dr. Ron Frank for his instruction of Genomics, and Dr. Katie Shannon for teaching me Cancer Cell Biology; without these primers I would not have been able to do the research or make advances in those fields which I care so much about. I'd like to thank the EMSE department for being a home for me.

Last and most, I would like to thank my friends who have continued to support and believe in me, as I feel that believing in others fuels them (as it has me) to do their best and can only help them to fulfill their greatest aspirations.

TABLE OF CONTENTS

Page

SECTION

LIST OF FIGURES

Page

PAPER I

PAPER II

PAPER III

LIST OF TABLES

Page

PAPER I

PAPER II

PAPER III

SECTION

1. INTRODUCTION

In the area of machine learning, learning classifier systems (LCS) are rule-based algorithms which combine reinforcement learning (for determining rule quality) with evolutionary computing (for stochastically searching rule-spaces in a directed manner) to solve a variety of prediction and classification problems. There are various representation schemes; few if any have human-readable solution outputs. A large portion of LCS algorithms, for instance, are ternary, which is not naturally human-readable. Artificial neural networks, among other types of ML, also tend to not produce human-readable solutions.

The Rote-LCS, introduced within the scope of this thesis, offers exceptional results in classification problems, while offering the potential for human-readable solutions. This characteristic may be useful in a wide variety of applications, including scientific investigation of problems in which a need exists to develop hypotheses to explain phenomena resulting from particular interactions between problem variables. The Rote is able to achieve high accuracy at low computational expense and retain human readability by evolving rule sets consisting of hypotheses which use the relational operators greater-than, and less-than or equal to (>, =<) to assess the result of an assumption about the relative value of a given independent variable of an observation (e.g. “temperature is greater than 500 degrees Fahrenheit”). These 'sentences' of relational hypotheses are able to hold relevant predictive information in a condensed form, are easy for a human to understand, and may be evaluated to true or false.

Later versions of the Rote include the use of binary trees and logical operators (tree nodes) to link the relational hypotheses (terminal nodes) to allow the evolutionary algorithm to explore relationships between independent variables for significantly increased accuracy. Binary trees may be represented two-dimensionally, and like relational sentences, are human-readable.

The Rote-LCS was originally written to solve a problem involving breast-cancer prediction. Its functionality was expanded in subsequent work to enable use of the Rote in new types of problems, and to increase its accuracy. The latest version of the Rote is able to train solutions on supervised datasets to perform classification with exceptionally high accuracy. The Rote may also be used for real-value classification and could also be used to develop AI control systems.

2. THESIS OVERVIEW

This thesis presents three incarnations of the Rote-LCS, in chronological order of development. Paper 1, accepted to 2012 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, presents the first version of Rote-LCS [1]. The Rote is used on a problem to predict breast cancer from patients' results on a series of nine cancer indicators. When trained on five hundred or more observations (patients), some positive and some negative for cancer, the Rote is able to achieve 99% accuracy or better on the test dataset. The paper was co-authored with Dr. Corns and Dr. Cudney of the Engineering Management and Systems Engineering department of Missouri University of Science and Technology.

Paper 2, accepted to 2012 IEEE Congress on Evolutionary Computation, uses the Rote on a ground-level ozone prediction problem; the Rote's functionality is expanded to use in real-value classification [2]. This paper was also co-authored with Dr. Corns and Dr. Cudney.

Paper 3, pending submission to IEEE Transactions in Evolutionary Computation, includes a major addition to the original Rote algorithm. Relational hypotheses, which make up the rule representation scheme of the original Rote-LCS, become rule segments within larger hypotheses. Binary trees and logical operators are used to link relational hypotheses. This enables the evolutionary algorithm to explore relationships between independent variables rather than considering them separately, leading to a significant increase in the accuracy of the Rote.

REFERENCES

[1] Daniels, B., Corns, C., Cudney, E., “Introduction of R-LCS and Comparative Analysis with FSC and Mahalanobis-Taguchi Method for Breast Cancer Classification.” *2012 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology.* pp. 283-289.

[2] Daniels, B., Corns, C., Cudney, E., “A Comparison of Representations for the Prediction of Ground-Level Ozone Concentration.” *2012 IEEE World Congress on Computational Intelligence.* pp. 1-8.

PAPER

I. INTRODUCTION OF R-LCS AND COMPARATIVE ANALYSIS WITH FSC AND MAHALANOBIS-TABGUCHI METHOD FOR BREAST CANCER CLASSIFICATION

Benjamin Daniels, Steven M. Corns, Elizabeth A. Cudney

Abstract – Classification for medical diagnosis is an important problem in the field of pattern recognition. We introduce a new method for classification based on repeated analysis of information tailored to small data sets – the Rote Learning Classifier System. Using the Wisconsin Breast Cancer study, this method was compared to three other methods of classification: Mahalanobis-Taguchi Systems, Finite State Classifiers, and Neural Networks. It was found that for the given data set, the Rote Learning Classifier System outperformed the other methods of classification. This new algorithm correctly classified over 92% of the data set.

I. Introduction

In the field of computational intelligence, it has been demonstrated that no problem solving method dominates all others in all problems [1]. The first method is a uniquely structured Pittsburgh learning classifier system (LCS), developed by the PI [2,3,4]. Nicknamed the ‘Rote,’ it is introduced here as the R-LCS method. This method is compared to two other methods for data classification: the Mahalanobis-Taguchi System (MTS) [5] and finite state classifiers (FSC) [6]. Both of these methods have been

successfully applied to other classification problems, and therefore are appropriate methods to compare and contrast for accuracy.

The R-LCS was developed by the PI to perform classification on data sets where a relatively small amount of training data is available. It is designed to address problems in which the input parameters consist of discrete points in one or more bounded ranges of known size. The problem input parameters for the R-LCS method consist of an arbitrary number of ranges to be evaluated for a Boolean output. The R-LCS method is particularly adept at training itself well in cases where only very small training populations exist.

The Mahalanobis-Taguchi System (MTS) was developed by Genichi Taguchi to provide a means to establish a reference group and a means to define a measure the degree of abnormality of individual observations when using the Mahalanobis distance (MD) to determine class membership [5]. Mahalanobis distance is a discriminant analysis approach that uses a distance measure based on correlations between variables and the different patterns that can be identified and analyzed with respect to a reference population. MD is used to determine the similarity of a known set of values (normal group) to that of an unknown set of values (abnormal group). It has been successfully used in a broad range of cases, largely due to its ability to identify inter-variable changes in data. Also, because the MD is measured in terms of standard deviations from the mean of the samples, it provides a statistical measure of how well an unknown sample matches a known sample set.

Finite state classifiers are Finite State Machines (FSMs) where a vote to classify the data set is performed in each state based on the data being input to drive the state machine [6, 7]. The input being analyzed by the FSM is used to drive the machine

between states, gathering information from the states as it progresses. The FSC were created using an evolutionary algorithm that treated the state descriptions as an array which was used as the genome representation for the problem. These algorithms have been shown to be useful as classifiers on Polymerase Chain Reaction (PCR) primer classification [8, 9] and image classification [10], which contain a full description of how these classifiers are constructed and implemented.

The data set used for this analysis is the Wisconsin Breast Cancer study, consisting of nine attributes and one class (positive or negative for cancer). The classification is performed based on the values of these nine separate attributes, with values ranging from 1 to 10. The nine attributes taken under consideration are:

- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape
- Marginal Adhesion
- Single Epithelial Cell Size
- Bare Nuclei
- Bland Chromatin
- Normal Nucleoli
- Mitoses

More information on these attributes can be found at the University of California Irvine Machine Learning Repository Website [11].

Evolutionary algorithms (EAs), including LCSs are generally an appropriate method for this type of problem; however, there are typically many unique ways to apply EA operators to a given problem. In the course of this paper, a new approach is taken and then compared with the use of finite-state classifiers and the Mahalanobis-Taguchi System. For additional consideration, results for a neural network approach [12] are also considered.

II. Methodology

A. Basic Approach

The representation of the R-LCS method is separated into four chromosomes to which the traditional EA operators (parent selection, recombination, mutation, and natural selection) are applied. The information contained in these four chromosomes is as follows:

- Chromosome 1 contains an integer value from 1-10 which is compared with the patient's test results. Each test will be compared to a unique location in the chromosome. These values will be compared to determine which is greater to drive the classification.
- Chromosome 2 contains a Boolean operator, or a value which can be converted into a Boolean. If true, the hypothesis is that the patient has cancer if the test score lies *above* the number in Chromosome 1, and does not if otherwise. If false, the hypothesis is that the patient has cancer if the test score lies *below* the number in Chromosome 1, and does not if otherwise.

- Chromosome 3 holds the weighting factor or number of points earned towards a positive diagnosis. It is counted for a patient only if Chromosomes 1 and 2 point to a positive diagnosis.
- Chromosome 4 holds the weighting factor or number of points earned towards a negative diagnosis. It is counted for a patient only if Chromosomes 1 and 2 negate a positive diagnosis.

To better understand this approach, consider the following example in Table 1. The test results for a single patient are evaluated by the chromosomes of a single potential solution. The patient's test scores are on the left. The information in the four chromosomes determines how this data is evaluated. Finally, the results for each indicator are summed for a final determination. A negative number indicates no cancer, while a positive one indicates a positive diagnosis. To evaluate the first attribute (Clump Thickness), it is first compared to the patient's value for this indicator with the corresponding value in C1 (Chromosome 1). In this case, it is 'greater than'. Chromosome 2 has a value of false. This in conjunction with
the fact that the patient's value is greater than the C1 value means that there is a negative diagnosis on this indicator for this patient on this solution. Since the diagnosis is negative, we look at the weighting coefficient for this indicator when the diagnosis is negative. The value is 8, so -8 is the output of this indicator for the patient.

This is performed for each of the nine attributes, and then the results are summed. If the value is positive, then the prediction is positive for cancer, otherwise the prediction is negative.

B. Sensitivity Operator

The above example uses a single point which is chosen in each range, provided by Chromosome 1. However, there is one final main addition to the R-LCS method: in order to achieve more 'sensitivity,' or 'granularity,' multiple points within each range are examined simultaneously. This requires adding four more chromosomes for each increment in sensitivity; thus, the size of the rule set used for training increases by a factor equal to the sensitivity factor. For example, if the sensitivity factor is 2, then two points on each range of each indicator are chosen to evaluate. Figures 1 and 2 illustrate how this sensitivity factor is used in evaluating the attribute values. The range for clump thickness is now split twice and each of the two corresponding chromosome sets are evaluated independently. The sums of each are then added for the final diagnosis decision. For simplicity, the sensitivity factor is chosen at the onset of a simulation for the R-LCS method; all indicators of all solutions will have the same sensitivity factor. The sensitivity factor may be of any arbitrary size, but as the sensitivity factor increases, marginal improvement in EA effectiveness per increase of the sensitivity factor approaches zero.

C. Application of EA

For this method, truncation selection is used as the natural selection operator. As it is known that the selective pressure for truncation selection is extremely high, the parent selection method used in the R-LCS method was chosen to be random,

Table 1 – Example evaluation of breast cancer patient information using R-LCS. A negative sum indicates no cancer predicted.

	Patient's Test Results	C1	C2	C3	C4		Patient's Indicator Results
Clump Thickness	5	4	FALSE	7	8		-8
Uniformity of Cell Size	1	4	TRUE	1	3		-3
Uniformity of Cell Shape	1	6	FALSE	3	2		3
Marginal Adhesion	1	9	TRUE	6	7		-7
Single Epithelial Cell Size	2	8	FALSE	8	2		8
Bare Nuclei	1	1	FALSE	5	7		-7
Bland Chromatin	3	1	TRUE	5	4		5
Normal Nucleoli	1	3	FALSE	3	2		3
Mitoses	1	7	TRUE	9	2		-2
						Sum:	-8
				Result: Answer is negative, so patient does not have cancer.			

since no further selective pressure seems likely to be needed. An n-point recombination operator was selected for crossover, with the number of points selected by the user before a run. All subsequent runs will then utilize the same n-point value. Mutation occurs by randomly selecting a child, then randomly selecting a chromosome, and mutating it to a random value within the valid range of possible solutions. There will be a number of mutations equal to the value input by the user for ‘Mutation Rate’ when the simulation is started. A population size of 10 was used for the R-LCS.

D. Fitness Evaluation

Type-I (false positives) and type-II error (false negatives) are distinguished within this method. Specifically, type-II error is prioritized to decrease the likelihood that a patient with a malignant growth will be diagnosed with a benign growth. While a false positive for cancer may be stressful for the patient, it is less likely to put the patient at risk than an instance of type-II error. Type II errors were discouraged through a fitness weighting scheme. The fitness weighting criteria are as follows:

- A correct positive is worth +4
- A correct negative is worth +3
- An incorrect positive is worth -2
- An incorrect negative is worth -6

To determine fitness, each solution in the Plus (parents plus children) survival population is evaluated using data from 500 actual patients. The prediction made by the solution is compared with the real diagnosis and fitness points are awarded to solutions accordingly based on predictive accuracy.

III. Results

The R-LCS method was first tested for breast cancer prediction accuracy at various levels of its sensitivity function. It was then tested at its highest level of sensitivity and compared versus other prediction methods.

The data set used for this project is the breast cancer data from the UCI machine-learning repository, which was collected at the University of Wisconsin by W. H. Wolberg [13]. They used this data to predict whether a tissue sample taken from a patient's breast is malignant or benign. There are one class, nine numerical attributes, and 699 observations. Sixteen instances contain a single missing attribute value and were removed from the analysis [14].

Figure 1, Information contained by Chromosome 1for clump thickness, with a sensitivity factor of 1.

Figure 2, Information contained by Chromosome 1for clump thickness, with a sensitivity factor of 2.

The training data was broken into two parts: 485 for training and 199 for evaluation. The initial comparisons are conducted with previous work, so to compare the performance of the R-LCS method with the other methods it was necessary to train the data on the data set sizes of 20, 30, 50, and 100. All used data points were chosen at random from the training data set of 485. A random sample set was chosen at the beginning of each run. For each training data size, the R-LCS method was tested on 10 runs at seven different levels of sensitivity. The sensitivity levels used are 1, 2, 3, 4, 5, 7, 10, and 30. These levels are used to demonstrate the success of the R-LCS with smaller rule set sizes. A sensitivity factor of 30 is chosen arbitrarily to demonstrate relative performance on a somewhat larger rule-set size.

The R-LCS method had mediocre classification accuracy when a data set of 20 was used for sensitivities of one to four (fig. 3), with classification accuracy typically of 83-90%. When the sensitivity was increased to five or higher, the algorithm classification accuracy increased to 95% or better. As the training data set size increased, this trend continued, although the classification accuracy was always above 92% and usually above 95%. For most data set sizes, the accuracy continues to improve for three of the five data sets. This may indicate the beginning of overtraining on the data set.

The type I and type II errors were also evaluated to determine how well the algorithm classified the two possible outcomes. The type I errors were calculated by dividing the

number of false positives by the total number of biopsies that were negative (fig. 4). The type II errors were calculated by dividing the number of false positives by the number of biopsies found positive for cancer (fig. 5). Comparing the results, it can be seen that there is a distinct bias against type II errors, with only about one percent of errors falling in this category for all positive biopsy results.

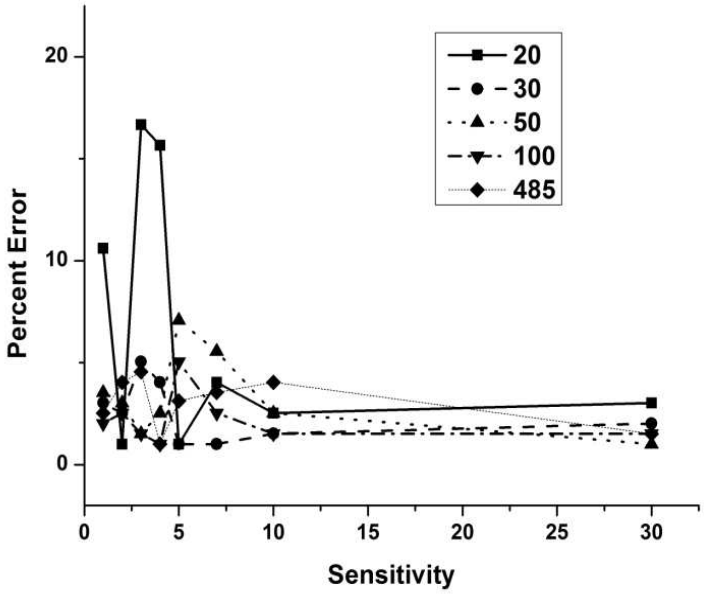

Figure 3, total number of errors vs. sensitivity for the R-LCS approach with training set sizes of 20, 30, 50, 100, and 500.

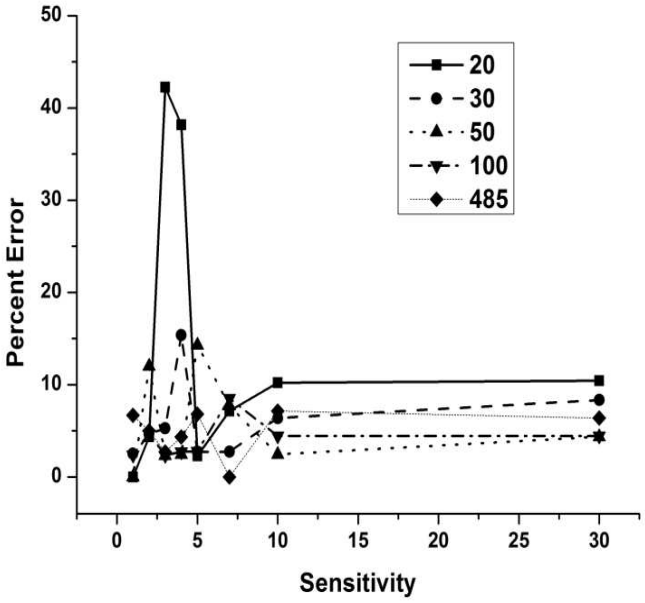

Figure 4, Percentage of type I errors vs. sensitivity for the R-LCS approach with training set sizes of 20, 30, 50, 100, and 500.

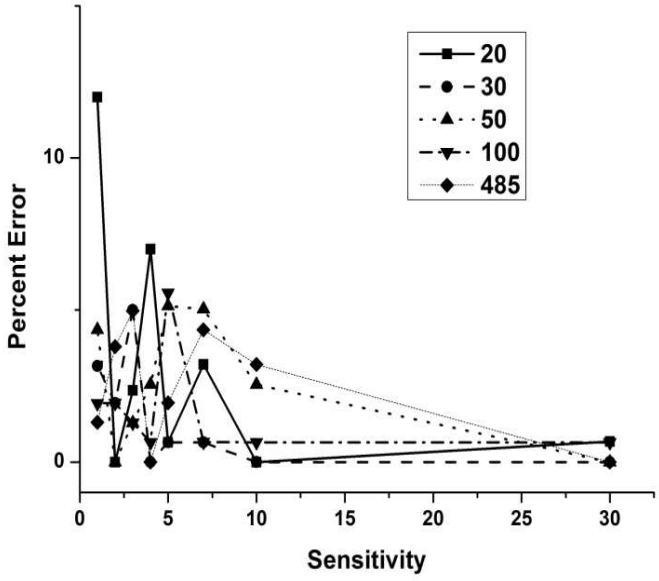

Figure 5, Percentage of type II errors vs. sensitivity for the R-LCS approach with training set sizes of 20, 30, 50, 100, and 500.

The best performing sensitivity level was then compared against Mahalanobis-Taguchi System and the neural net methods. The information for this comparison was extracted from previous work [14] in which the Mahalanobis-Taguchi system was compared to a neural network approach. Data from exactly ten runs was collected and averaged, and the results plotted and compared with ten runs of the neural network and ten applications of the Mahalanobis-Taguchi system. Figure six shows the comparison between these methods. The data collected from each of the experiments here are given in tables 2, 3, and 4.

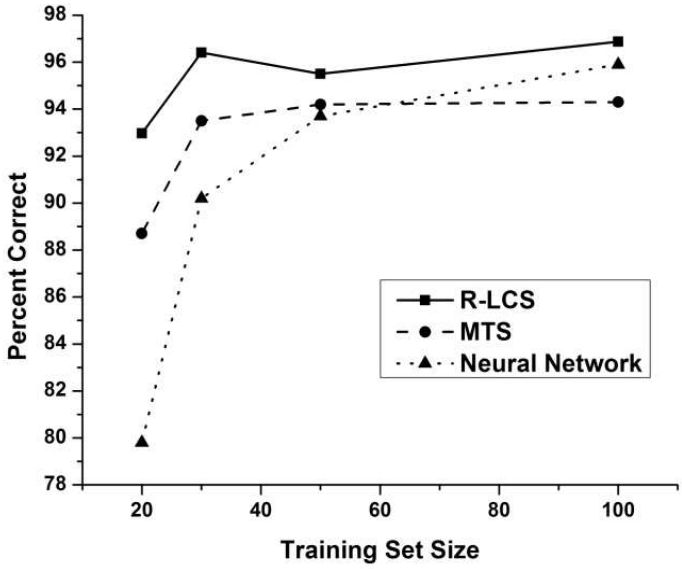

Figure 6, Comparison of correct classifications vs. training data size for R-LCS, MTS, and NN.

Table 2 – Results for Mahalanobis - Taguchi System [11]

	data20	data30	data50	data100
1	0.946	0.936	0.925	0.962
2	0.846	0.955	0.962	0.961
3	0.928	0.933	0.958	0.909
4	0.911	0.949	0.946	0.949
5	0.838	0.946	0.934	0.950
6	0.794	0.896	0.929	0.924
7	0.880	0.956	0.953	0.936
8	0.895	0.941	0.920	0.933
9	0.918	0.915	0.949	0.947
10	0.920	0.921	0.944	0.959
MD	0.887	0.935	0.942	0.943

Table 3 – Results from Neural Network Classification [11]

	data20	data30	data50	data100
1	0.848	0.966	0.905	0.966
2	0.947	0.870	0.971	0.937
3	0.854	0.968	0.958	0.968
4	0.821	0.902	0.944	0.941
5	0.857	0.933	0.944	0.968
6	0.650	0.937	0.941	0.963
7	0.637	0.952	0.900	0.956
8	0.799	0.811	0.952	0.955
9	0.683	0.838	0.896	0.968
10	0.880	0.843	0.961	0.965
NN	0.798	0.902	0.937	0.959

Table 4 – Results for the R-LCS method

	data20	data30	data50	data100
1	0.974	0.924	0.936	0.985
2	0.947	0.990	0.952	0.985
3	0.842	0.947	0.900	0.990
4	0.995	0.985	0.969	0.985
5	0.856	0.963	0.990	0.936
6	0.888	0.930	0.990	0.952
7	0.849	0.952	0.888	0.936
8	0.990	1.000	0.958	0.969
9	0.969	0.979	0.947	0.963
10	0.906	0.909	0.990	0.974
RLCS	0.921	0.958	0.952	0.967

The next step was to compare the R-LCS method with a finite-state classifier (FSC) method. The FSC used for the comparison were the best classifier from 100 different runs, and was constructed in a similar manner to that used in previous work [10]. Each state had am attribute number, a real value assigned to it, and a comparison operator. When entering the state, the value stored in the state was compared to the value of the attribute specified in the state. Based on whether the statement was true or false, a response instructed the state incremented, decremented or in some cases ignored a running counter. When the information was fully evaluated, a positive result indicated the growth was malignant and a negative valued indicated benign. 64 states were used for each of the FSC, and they were evolved using a generational algorithm using two point crossover and a mutation operator that changed the initial state (10%), the comparison operator (10%), the transition destination (20%), the response (10%), or the value used to make the comparison (50%). This algorithm was run for 1000 generations using training data set sizes of 20, 30, 50, 100, and 485.The best run of 10 runs with a sensitivity

coefficient of 30 from the R-LCS method was compared. The two EAs were compared on training data set size of 20, 30, 50, 100, and 485. Sets of these sizes were chosen to demonstrate general performance on small and moderately sized training sets; results from these were compared with a set of 485, which was the largest available. Figure 7 shows that the R-LCS method had a better success rate on classification of the data for all sizes of training data. The R-LCS method also demonstrates negligible overtraining on this data set.

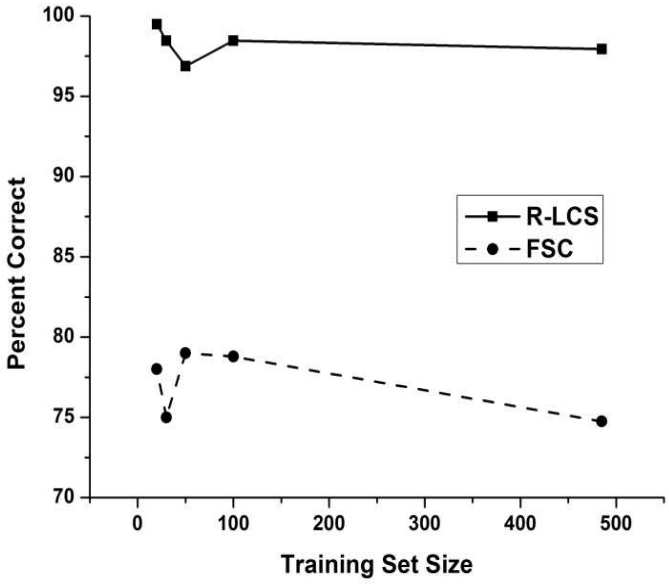

Figure 7, Comparison of correct classifications vs. training data size for R-LCS, and FSC

IV. Discussion

A. Sensitivity Factor and Performance

As expected, higher levels of sensitivity performed better in preliminary testing before comparison with other methods. As the sensitivity increased to 10, error distribution between the different population sizes stabilized and generally decreased. Since a sensitivity factor of 30 seemed to be the highest performing, it was chosen to be

compared with the other methods. It performed well at smaller population sizes versus the Mahalanobis-Taguchi System and neural network methods, and slightly better at larger population sizes. The R-LCS method dominated the other methods on all data set sizes.

It is interesting to note that overall performance of the R-LCS method actually decreased slightly as population size increased, and then began to increase again. It is possible that this is due to randomness in chosen population members. It is also possible that this may be due to some effect of the high sensitivity level used with the R-LCS method for the comparison. Additional experiments with a larger number of runs will be necessary to evaluate this fully.

Although a sensitivity factor of 30 was the highest used, the sensitivity factor for the R-LCS method may be set arbitrarily high. Results have shown that as the sensitivity factor is increased, general accuracy of the EA is also increased, at all training sizes. The downside is computation time; the EA requires more computation time as the sensitivity factor (and thus rule set size) is increased. There was no study done to determine what this rate of increase in computation time is, or whether it is linear or exponential. Also, the marginal benefit yielded by an increase of the sensitivity factor should eventually approach zero. In terms of general computation time, the Mahalanobis-Taguchi System has the lowest; it is virtually instantaneous. The R-LCS method had much better computation times than the FSC, which required a wall clock time of approximately one hour.

B. Further Study

In order to further investigate the performance of the R-LCS method with larger training populations, the R-LCS method was trained with a population size of 485 observations, the maximum possible. At this level, all sensitivity levels discovered better solutions. The best solution found had a 99% success rate, with zero instances of type-II error, and only two instances of type-I error after testing on 199 observations.

C. Possible Sources of Error in Comparison

The first possible source of error comes from the fact that sample size (ten runs) used to compare the Mahalanobis-Taguchi System, neural network method, and the R-LCS method is quite small. For this paper, it was necessary to use ten runs because only ten runs were performed for the Mahalanobis-Taguchi System and neural network prediction methods. As a rule of thumb, it is typically advisable to have at least 30 samples for comparisons of the mean between different populations. It may also be possible that variation from the sampling of the data sets has introduced some uncertainty in the results, as there is no available data on the particular portion of the data set the Mahalonbis-Taguchi System and the neural network approaches used.

Another possible source of error is due to the distinction which the R-LCS method makes between type-I and type-II error. Unlike the other methods to which the R-LCS method was compared, the R-LCS method penalizes type-II error significantly. This may lead to more total errors in a solution if it is necessary to have more instances of type-I error in order to avoid type-II errors with a greater success rate.

One final possible source of error in comparison with the Mahalanobis-Taguchi and neural net methods is in the size of the pool of training observations used for creating a

training population versus that used for testing. Although the same data was used by all methods compared, the other methods may have allocated the use of data differently. For example, the Mahalanobis-Taguchi System analyzes the data directly and does not have a training step.

V. Conclusions

The R-LCS method was extremely successful at solving this problem, and may be capable of regularly out-performing other known methods at solving range searching problems. The R-LCS method is very efficient in terms of computation time, with very low training time compared with the FSC. The R-LCS method dominated all other prediction methods at all levels of population size.

Repetition within the algorithm on the attributes of the data set allows the algorithm more opportunity to fine-tune the weighting of the different attributes. This allows for a gradual build up of weights and makes it more likely that complimentary combinations of information can be found. A well-known disadvantage to the application of LCS in general is the possibility of overtraining. However, in this test case no evidence of overtraining was apparent.

The outcome of the R-LCS method seems promising; it is important to acknowledge, however, that the data set size in this case was relatively small for an LCS application. An LCS approach in this case is still justifiable; nine variables with ten possible states each provides a search space with 10^9 (one billion) possible combinations, which is generally too large to warrant a heuristic approach. Future applications of the R-LCS method for breast cancer could include the development of a nine dimensional response

surface that may be used to indicate combinations of variables that indicate positive results for malignant growths.

It will likely be useful to look for other applications where the R-LCS method could be used. It may be beneficial to investigate further the use of different levels of sensitivity in the training of the R-LCS method, or to further refine the technique. It would also be advantageous to apply this method to other data sets to gain more information on the benefits of the weighting scheme and to explore any potential areas where overtraining would be a serious issue.

The problem solving techniques which the R-LCS method was compared against are generally good at a wide variety of problems. It could be valuable to compare the R-LCS method against other problem solving techniques which are specially formulated for range searching.

VI. References

[1] Wolpert, D.H.; Macready, W.G., "No free lunch theorems for optimization," *Evolutionary Computation, IEEE Transactions on*, vol.1, no.1, pp.67-82, Apr 1997.

[2] R. J. Urbanowicz and J. H. Moore, "Learning classifier systems: a complete introduction, review, and roadmap", Journal of Artificial Evolution and Applications, vol. 2009, pp. 1-25, January 2009.

[3] J. Bacardit, "Pittsburgh genetics-based machine learning in the data mining era: representations, generalization, and run-time", Ph. D. dissertation, Ramon Llull University, 2004.

[4] S. F. Smith: "A Learning System Based on Genetic Adaptive Algorithms", PhD thesis, University of Pittsburgh, 1980.

[5] Taguchi, G and R. Jugulum, “New Trends in Multivariate Diagnosis”, *Indian Journal of Statistics*, 62, Series B, 2 233-248 (2000).

[6] Stanley, E. A., Ashlock, D. A. and Tesfatsion, L., “Iterated prisoner's dilemma with choice and refusal,” *Artificial Life III*, vol. 17 of Santa Fe Institute Studies in the Sciences of Complexity, edited by Christopher Langton, Reading, 1994, Addison-Wesley, pp. 131-176.

[7] Fogel, D. B., “Evolving behaviors in the iterated prisoner’s dilemma”. *Evolutionary Computation*, Vol. 1, No. 1, 1993.

[8] Ashlock, D. A., Bryden, K. M., Corns, S. M., Schnable, P. S., and Wen, T.J., “Training Finite State Classifiers to Improve PCR Primer Design,” 10th Annual AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY, 2004.

[9] Yadav, S. and Corns, S.M., “Improved PCR Design for Mouse DNA by Training Finite State Machines,” *2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 249-254, 2010.

[10] Cudney, E.A.; Corns, S.M.; , "A comparison of Finite State Classifier and Mahalanobis-Taguchi System for multivariate pattern recognition in skin cancer detection," *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2011 IEEE Symposium on* , vol., no., pp.1-7, 11-15 April 2011

[11] University of California Irvine Machine Learning Repository, Wisconsin Breast Cancer Study <http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29, last accessed 12/3/2011.

[12] Jugulum, R., and Monplaisir, L. “Comparison between Mahalanobis-Taguchi-System and Artificial Neural Networks”, *Journal of Quality Engineering Society*, Vol. 10, No.1, pp.60-73, 2002.

[13] Cudney E. A., Hong J., Jugulum R., Paryani K., Ragsdell R. M., Taguchi G. “An Evaluation of Mahalanobis-Taguchi System and Neural Network for Multivariate Pattern Recognition”, *Journal of Industrial and Systems Engineering*, Vol. 1, No. 2, pp 139-150, 2007.

[14] Carlos Andres Pena-Reyes, Moshe Sipperemail. A fuzzy-genetic approach to breast cancer diagnosis Artificial Intelligence in Medicine Volume 17, Issue 2 pages 131-155, October 1999.

II. A COMPARISON OF REPRESENTATIONS FOR THE PREDICTION OF GROUND-LEVEL OZONE CONCENTRATION

Benjamin Daniels, Steven Corns, Elizabeth Cudney

Abstract – This work presents a comparison of methods to predict ground-level ozone to highlight differences in the ability of the algorithms and to compare their performance to an established signal to noise based prediction method. Existing data related to weather conditions and ground-level ozone was divided into a training set and a test set. Three algorithms were trained using the training set to create predictors, which were then analyzed with the test set, and then compared to the Taguchi Method to determine performance. It was found that the newly introduced Rote-EA performed well on this problem, predictors using the Taguchi method had a smaller deviation from actual results. This indicates an additional factor other than the level of correlation in the data that dictates how well these predictors perform on classification problems.

Keywords-component; evolutionary computation, predictor, classifier

I. Introduction

For this analysis, a case study for ozone concentration is used to investigate the ability of three different methods to predict ground-level ozone: a standard evolutionary algorithm (EA), a graph based evolution algorithm, and a rote evolutionary algorithm. These results were compared to the Taguchi Method (T-method); a design of experiments approach originally used to reduce variation in production. Comparing these evolutionary

computation methods to the T-method gives an indication of how well these stochastic methods can identify the correlations found with the T-method. All of these methods were applied to make predictions of ground-level ozone concentrations based on information from seven monitoring sites that gathered thirteen different data points from the Dallas-Fort Worth area, aggregated between January 1, 2005 and December 31, 2007. This area was selected since it does not currently meet the EPA's satisfactory level of ozone pollution [1, 2]. It has the 12th highest ozone air pollution in the nation, which is likely a result of nearby facilities such as a material incineration plant, concrete installations, and other local industrial. In addition, an increasing number of automobiles used by commuters is likely a major contributor of this pollutant [3].

Ground-level ozone is one of the most common human health hazards that is directly associated with human activity. This pollutant is formed when volatile organic compounds (VOCs) and nitric oxides react in the presence of sunlight [1]. High levels of this pollutant are hazardous to the human health and harmful to the environment. About 50% of all anthropogenic NOx emissions result from motor vehicles [4], making it an even greater concern in areas highly travelled by these vehicles. At elevated levels of ozone, an individual can experience effects ranging from airway irritation to permanent lung damage, with a person with average sensitivity to ground-level ozone having adverse reactions to concentrations as low as 0.08 parts per million (ppm). An accurate method to predict daily ozone levels would be a valuable tool for assessing this risk to public health [5].

The ozone prediction problem involves a set of independent variable parameters, or indicators, which consists of information that is believed to be useful in achieving good predictability. For ozone prediction, these indicators are shown in Table 1.

Table 1 – Variables used to determine ground level ozone concentrations

Variable	**Units**
Independent	
Maximum wind gust	Miles per hour
Nitric Oxide	Parts per billion
Outdoor Temperature	Degrees Fahrenheit
Resultant wind direction	Degrees
Resultant speed	Miles per hour
Solar radiation	Langley's per minute
Standard Deviation of horizontal wind direction	Degrees
Wind Speed	Miles per hour
Month	Month of the Year
Day	Day of the Month
Year	Year of the Study
Weekday	Day of the week
Dependent	
Ozone	Parts per billion

The baseline algorithm for this study is a simple evolutionary algorithm [6]. This was represented as a pool of solutions that were created uniformly at random and allowed to interact with one another through recombination and mutation operators. To gain a better comparison a graph based evolutionary algorithm (GBEAs) was also used to determine any effects diversity preservation had on the solution quality [7]. These GBEAs mimic

geographic boundaries that restrict mating within an evolving population, which has been found beneficial to preserving solution diversity [8]. This is done by placing solutions at the vertices of a graph and proceeding with a steady state mating scheme [9], allowing mating to occur only between vertices with a common edge. For this experiment a cycle graph was selected, as it had been found to provide larger population diversity than other GBEAs previously used [8]. The results of these experiments were then compared to the T-Method and another evolutionary algorithm, the RoteEA.

The RoteEA is an Evolutionary Algorithm method originally developed to predict instances of breast cancer among patients being tested for the disease. As applied in this problem, each predictive result was either positive or negative; an implementation in which the RoteEA was found to be highly effective. In this study, the RoteEA is tested on a problem where each predictive result consists of a point on an unbounded range.

Using the RoteEA, solutions are evolved which enable prediction of the ozone level based on the indicators in Table 1. These indicators are not necessarily the only factors which affect ground-level ozone concentrations, and therefore any solution evolved to predict ozone based on these indicators will be limited by the information on hand.

II. Methodology

A. Standard Evolutionary Algorithm and GBEA

This approach used a standard evolutionary algorithm with no special accommodations for training on the data. Each population member is a string of eight doubles which were used as coefficients for the eight variables used to predict ozone. The eight variables and the ozone level coinciding with those values were read into the EA and normalized. A population of 512 solutions was generated randomly, with each

coefficient initialized to a value from zero to one. Mating was performed using a steady state mating process with two point crossover at a 100% crossover rate. In addition, a mutation operator was used that added or subtracted from 0 to 0.1 to the value of a coefficient selected uniformly at random. The mutation rate applied was 80%. 100 trials were conducted with the standard evolutionary algorithm, with a stopping criteria of 200,000 mating events.

To investigate the use of diversity preservation, a graph based evolutionary algorithm was applied to the problem. GBEAs use graphs to impose an artificial geography on the population of evolving solutions. For this work, the cycle graph was used as it provided the most diversity preservation and therefore the most contrast to the standard evolutionary algorithm. All of the parameters used in the standard evolutionary algorithm were applied to the GBEA, with the mating restrictions imposed by the graph the only difference between the standard evolutionary algorithm. More information on GBEAs can be found in [Bryden, 2006].

B. RoteEA

1) Background of the RoteEA: The first application of the RoteEA used a particular methodology to attempt to find relationships between various indicators by evolving as a solution a set of rules in terms of 1) Direction and 2) Significance.

If one were to visualize how a RoteEA-evolved solution works, it could be seen as operating like a pressure gauge. Direction is the direction of pressure (positive or negative) for each rule within the solution. The RoteEA method differs significantly from finite state classifiers (FSCs) in its method of mapping Direction. With FSCs, each individual FSC maps a given input to one of many possible outputs; many FSCs are

typically used in tandem to create a map. By contrast, Direction in the RoteEA method, (which may be likened to an FSC output) is Boolean; it has only two possibilities. In theory, this decreases complexity while still allowing for versatility in evolving highly fit solutions.

Significance is a weighting factor that is applied to Direction to determine its overall importance in the solution with the objective of finding good predictive results. In other words, Significance is the magnitude of pressure in the given Direction for a given rule.

The input from each indicator is considered multiple times, the number of which is determined by a ‘Sensitivity Factor’, which is an input parameter given to the RoteEA before it begins to evolve solutions. The Sensitivity Factor is static and is the same for all indicators. A solution evolved by the RoteEA is simply a set of rules. The number of rules is equal to

$$S * I \quad (1)$$

Where S is the Sensitivity Factor, and

I is the number of indicators

Every rule consists of a Direction and a Significance. Four chromosomes are used to hold this information; the method of constructing these chromosomes is given in the next section. When a solution is evaluated using the required indicator parameters, the result is a ‘pressure’ which is either positive or negative, and has a certain magnitude. The final pressure value itself is the final result which is then evaluated for fitness by comparing it

with the value for the dependent variable. It is found by summing the pressure values from the evaluations of each rule in the solution.

For the purposes of this paper, the original RoteEA which was used to predict cancer has been altered in one significant way: instead of using the 'pressure value' to represent whether the patient was positive or negative for cancer, the pressure value is instead mapped onto a space which represents ground-level ozone in parts-per-billion (ppb). With the RoteEA, the evolved solution contains information of the actual relationships between the indicator variables in a way which allows for extrapolation, to an extent, into territory for which the RoteEA was not fully trained. This should allow the RoteEA to find solutions for the dependent variable (ozone level in ppb) without bounds (for instance, the highest ozone ppb in training data may be 65, however, solutions evolved by the RoteEA should still be capable of predicting ozone levels in cases where the ppb should be higher than 65; ozone levels are allowed to be arbitrarily high).

2) Structure of Chromosomes: The genetic information, or 'DNA,' of the solution set of the RoteEA is separated into four chromosomes to which the traditional EA operators (parent selection, recombination, mutation, and natural selection) are applied. Before the EA begins evolving solutions, the data for all chromosomes is randomized. chromosomes 1 and 2 contain the information about the Direction of each rule; chromosomes 3 and 4 contain information about the Significance. Remember:

- Each rule is evaluated only for a single indicator
- Each indicator typically has several rules; the number of rules per indicator is equal to the Sensitivity Factor.
- Each rule contains information from each of the four chromosomes

Before the EA begins evolving solutions, the test data is scanned to find the highest and lowest values for each indicator. This is used to determine the bounds for chromosome 1.

The information contained in the four chromosomes consists of the following:

- Chromosome 1 contains a number between the highest and lowest values for the indicator found while scanning training data. When testing a solution or evaluating it for fitness, the number for Chromosome 1 for a given rule is compared with the value in the indicator for that rule.
- Chromosome 2 contains a Boolean operator, or a value which can be converted into a Boolean. If true, the hypothesis is that the pressure is positive if the indicator score lies *above* the number in Chromosome 1, and does not if otherwise. If false, the hypothesis is that the pressure is positive if the indicator score lies *below* the number in Chromosome 1, and does not if otherwise.
- Chromosome 3 holds the weighting factor (Significance), or number of points earned towards a positive pressure. It is counted only if Chromosomes 1 and 2 point in the positive Direction.
- Chromosome 4 holds the weighting factor (Significance), or number of points earned towards a negative pressure. It is counted only if Chromosomes 1 and 2 negate a positive pressure Direction.

3) Application of EA: For this method, truncation selection is used as the natural selection operator. As it is known that the selective pressure for truncation selection is extremely high, the parent selection method used in Method A was chosen to be random, since no further selective pressure was likely to be needed.

This EA uses an n-point recombination operator; the number of points may be selected by the user before a run; all solutions of all runs will then utilize the same n-point value.

Mutation occurs by randomly selecting a child, then randomly selecting a chromosome, and then mutating it to a random value within the valid range of possible solutions. There will be a number of mutations equal to the value input by the user for 'Mutation Rate' at the onset of a simulation.

4) Fitness Evaluation: The fitness evaluation consists of summing the pressures for each instance in which the pressure Direction was positive, summing the pressures for each instance in which the pressure Direction was negative, adding these together to get the predicted result, and then taking the absolute value of the predicted result minus the actual:

$$D = |G - A| \qquad (2)$$

Where D is the distance between actual ground-level ozone concentration and the estimate

A is the actual ground-level ozone concentration

G is the estimated ground-level ozone concentration

D is averaged over all used training-data members and multiplied by (-1) to find the final fitness score for the RoteEA-evolved solution.

C. Taguchi Method

The T-method is used to calculate the Signal-to-Noise (S/N) ratio for the overall prediction of the dataset. One of the main benefits of the T-Method is that it can be used to predict results based on a relatively small set of input data. While there are several other methods to make such predictions, most require much larger data sets to reach an acceptable solution. For many studies this becomes a limiting factor when these large data sets are not always available to perform these analyses, and so the T-Method is an ideal choice as it can identify correlations with relatively little data. The T-method examines the variation of the data points within the data set to calculate a signal to noise ratio (S/N) associated with the overall prediction of a data set. This property is used in situations where weather conditions often display high variability and the data is limited, the T-method proves efficient as it can be applied on a limited dataset.

The T-Method as it is applied here is an implementation of the Taguchi System of Quality Engineering (TSQE). This methodology was introduced to eliminate variation during product design and manufacturing through the use of four steps: 1) product parameter design, 2) tolerance design, 3) process parameter design and 4) on-line quality control. The method is carried out in three stages. First the objective is defined, then the available feasible solutions are explored, and the process is then completed with the selection of the best alternative to meet the design objective, using the S/N ratio as a measure of fidelity [10]. Depending on the quality characteristics used, different formulations of the S/N ratio may be used, such as: 1) normal-the-best (NTB), 2) smaller-the-better (STB), and 3) larger-the-better (LTB). For the development of technological systems and components the dynamic S/N ratio is typically used, in part because it allows

designers to better assess the stability of the system. Additional types of S/N ratios can be found in publications by Taguchi [11], Phadke [12], Fowlkes & Creveling [10], and Taguchi & Jugulum [13].

The T-method requires the selection of a unit space (or normal space) in which all members of the group must have the same or similar output. This multivariate output data is partitioned into two different classes that are then used to perform the analysis. The first class is referred to as the unit group and the second class is the signal group. The average output value of the unit group and the average output value of each variable are subtracted from each member of the signal group. Using the relationship between the variable values and the output of the signals, the slope, β, and S/N ratio, η, are calculated to obtain an overall estimate of the true output value for each signal member. The slope, β, is the sensitivity of the output with respect to the explanatory (independent) variable. The present study is limited to applications having only one response (dependent) variable and several explanatory (independent) variables. In addition, this work assumes that the explanatory variable is selected based on expert knowledge and understanding the nature of the problem. Previous applications of the T-method include the prediction of the food self-sufficiency ratio and the prediction of the total precipitation based on historical data [14, 15].

III. Results

The standard evolutionary algorithm and GBEA were evolved using 899 data sets and then evaluated using the remaining 195 data sets. The deviation of the prediction varied greatly over the evaluation data set, but the mean deviation was fairly consistent across all 100 trials for the two methods. For the first twenty data sets, the standard evolutionary

algorithm (Figure 1) had a large variation in the results, which continued for the entire 195 data sets, resulting in an average deviation of 15.579 from the true ozone level. The GBEA solution fared better than the standard evolutionary algorithm (Figure 2). The trend tracked the actual ground-level ozone better than the standard evolutionary algorithm, but still had a significant deviation from the actual ozone levels, with an average deviation of 7.709.

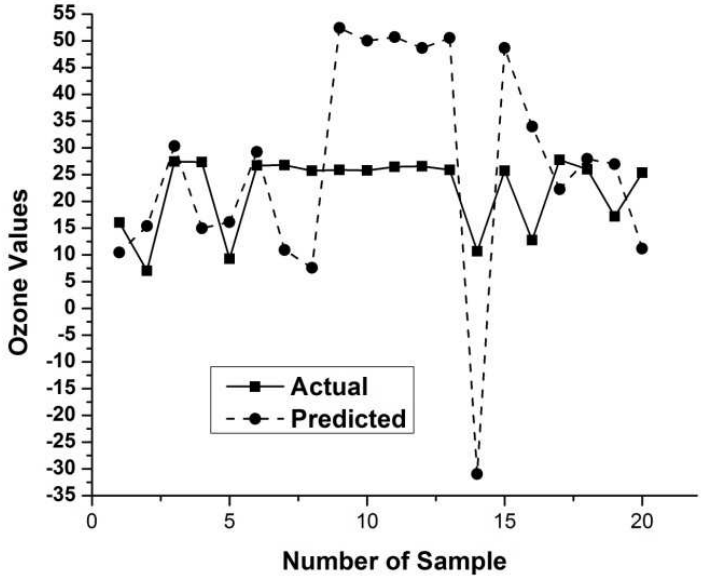

Figure 1, Standard EA results showing actual and predicted ground level ozone for 20 sample data sets.

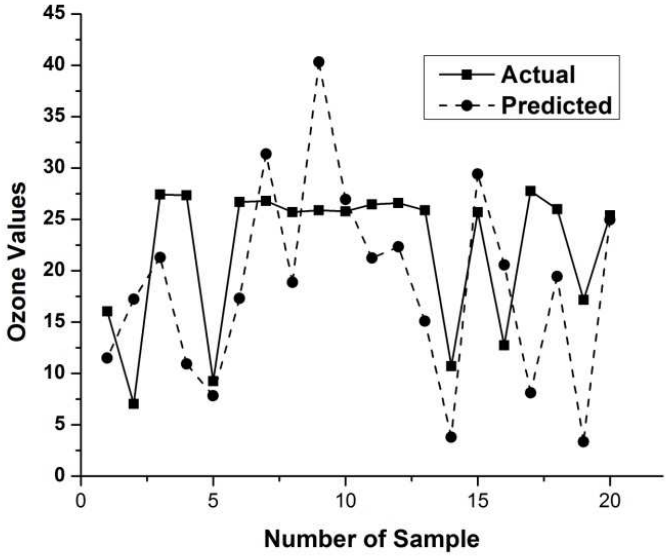

Figure 2, GBEA results showing actual and predicted ground level ozone for 20 sample data sets.

The Rote EA was initially tested over 11 runs, each with a different combination of parameter values between two parameters:

1) The number of data points used in training out of a total possible of 899

2) The Sensitivity Factor

A table containing the different combinations, as well as the respective results of each run in terms of mean deviation of predicted vs. actual, is shown in Table 2.

Table 2 – Sensitivity factors, training set size, and predictor results.

	Sensitivity Factor	Number Trained	Mean Deviation
1	32	899	6.06
2	128	899	5.74
3	32	50	7.12
4	32	100	6.58
5	128	50	6.71
6	64	100	6.34
7	128	100	6.57
8	32	500	7.2
9	64	500	5.85
10	128	500	5.9
11	128	50	6.92

The RoteEA was evaluated over 30 runs using a training set of 100 data points. For each run, the mutation rate was 10, and the Sensitivity Factor was also 10. The RoteEA was evaluated based on the average difference between actual and predicted values. Actual versus predicted values for ground-level ozone are provided in Figure 3.

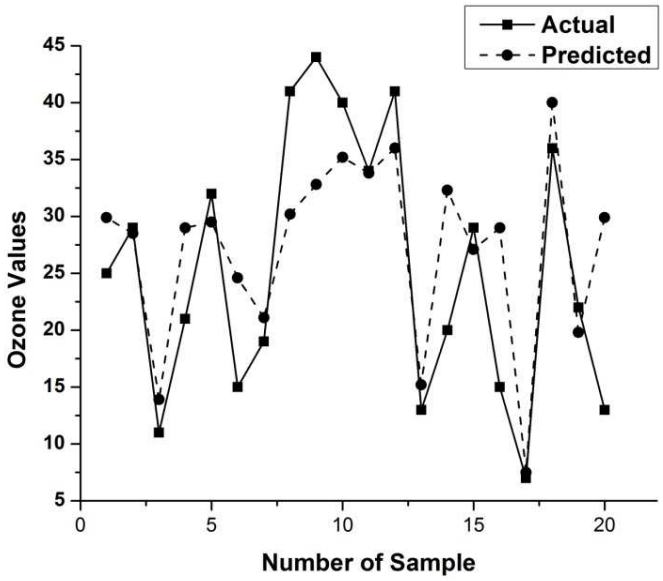

Figure 3, Rote-EA results showing actual and predicted ground level ozone for 20 sample data sets.

Of the 30 runs, the best performing exhibited an average deviation of predicted from actual values of 5.825. The worst run showed an average deviation of 8.815, and the average deviation for all runs was 7.203.

To forecast the ozone concentration using the T-method, twenty samples from the ozone concentration data are selected at random to construct a base system. This base system will then be used to predict the values of ozone concentrations, while the remaining data will be used as a validation data set. The unit space is first defined to provide the reference data that will be used to forecast the output. The average of all the data points available is calculated to determine this unit space, as the unit space should capture all of the data and these are distributed around this mean value. In addition, calculating the mean of the data set provides the user information on how the data points will be distributed around the mean. Before the results of the T-Method can be used to forecast the unknown values of ozone concentrations it must be validated using the remaining values of the ozone concentration from the data set. The known ozone concentration data is selected as the signal space. Each unit space value is then subtracted from the signal data to obtain a new standardized dataset. This method of standardization is used to show the deviation of the signal data is from the unit space (signal-to-noise ratio) [15].

The average output response value is denoted by M_0, and the average values of input variables are denoted as $x_{01}, x_{02}, x_{03},\ldots, x_{0k}$. The total number of signal members is n. This case study involves twenty signal members, representing the twenty data samples. Therefore, M_0 is the average value of twenty response values $M_1, M_2,\ldots, M_{20}$. Similarly, $x_{01}, x_{02}, x_{03},\ldots, x_{0k}$ corresponds to the average values of $(x_{11}, x_{12}, x_{13},\ldots, x_{1k})$,…, $(x_{i1}, x_{i2},$

$x_{i3},\ldots, x_{ik}$), where *k* varies from one to thirteen to represent the independent variable signals in the unit space.

The sensitivity between one input variable and the output response is denoted using β, and the corresponding signal-to-noise ratio is denoted η. The calculations outlined by Taguchi and Jugulum (2002) are given in Equations 3-8:

$$\beta_1 = \frac{M_1 x_{11} + M_2 x_{21} + \ldots + M_n x_{n1}}{r} \quad (3)$$

$$r = M_1^2 + M_2^2 + \ldots + M_n^2 \quad (4)$$

$$\eta_1 = \begin{cases} 0 \\ \dfrac{S_\beta - V_e}{r(V_e)} \end{cases}$$

$$(S_\beta > V_e) \quad (5)$$

Where,

$$S_T = x_{11}^2 + x_{21}^2 + \ldots + x_{in}^2 \quad (6)$$

$$S_\beta = \frac{(M_1 x_{11} + M_2 x_{21} + \ldots + M_1 x_{n1})^2}{r} \quad (7)$$

$$Ve = \frac{(S_T - S_\beta)}{n-1} \quad (8)$$

Using the Equations 3 through 8, β and η values are calculated for each value of *x*. Using the individual S/N ratio, each individual prediction can be calculated as follows:

$$\hat{M}_i = \frac{\eta_1 \times \frac{x_{i1}}{\beta_1} + \ldots + \eta_k \times \frac{x_{ik}}{\beta_k}}{\eta_1 + \eta_2 + \ldots + \eta_k} \quad (i=1, 2,...n) \qquad (9)$$

The signal values (M_i) and the predicted values ($\hat{M}_i$) can then be tabulated and compared to show the prediction accuracy for these individual. The calculation for overall prediction and the S/N ratio for the entire system are calculated as shown in Equations 10-15 (Taguchi and Jugulum, 2002).

$$S_T = (\hat{M}_1)^2 + (\hat{M}_2)^2 + \ldots + (\hat{M}_n)^2 \quad (f=l) \qquad (10)$$

$$L = M_1\hat{M}_1 + M_2\hat{M}_2 + \ldots + M_n\hat{M}_n \qquad (11)$$

$$S_\beta = \frac{L^2}{r} \quad (f = 1) \qquad (12)$$

$$S_e = S_T - S_\beta \qquad (13)$$

$$V_e = \frac{S_e}{n - 1} \qquad (14)$$

$$\eta = 10 \log \frac{(S_\beta - V_e)}{r(V_e)} \qquad (15)$$

The S/N ratio calculated in equation 15 gives an indication of the robustness of the system being analyzed. After the method is validated with the data set of known ozone concentration, the forecasting procedure is applied to calculate the values of the unknown ozone concentrations. In this case, the signal space is the data containing the values of the factors ($x_{0,1} - x_{0,k}$) affecting the unknown ozone concentrations. The sum of the mean

square error is used to obtain the base system that will be used to predict the unknown values of ozone concentrations (Parthiv et al., 2009). The mean square error for each combination of variables with the lowest value obtained is selected for the base system. This is selected as it represents the lowest variation obtained in the base system, giving a stable base system to allow for more accurate predictions. Seventy-five sample data entries were analyzed and the results were used to forecast future ozone concentrations based on the system generated with the unit space and signal space. The signal data containing the values of known ozone concentrations are taken into account for this prediction. The results between the actual and predicted data of the signal data are shown in Figure 4. This shows the validation of the data for the known values and the accuracy of the T-method for selecting the base system.

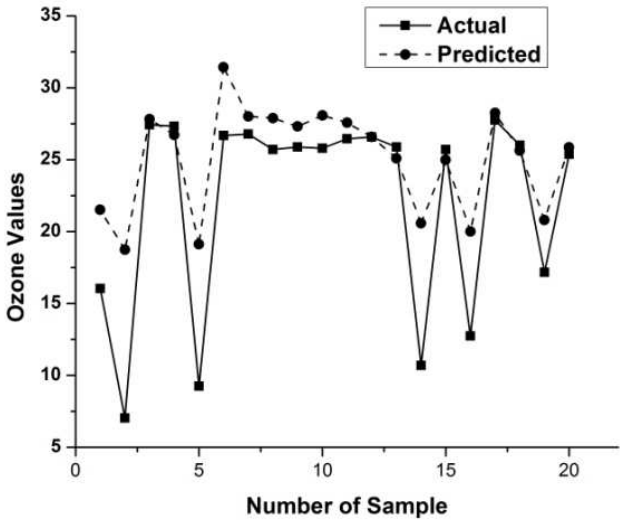

Figure 4, T-Method results showing actual and predicted ground level ozone for 20 sample data sets.

IV. Conclusions

A comparison of the three evolutionary computation algorithms shows that the representation of the problem has a large impact on the quality of the predictor found.

The average deviation for twenty evaluations of each method is shown in Table 3. For the standard evolutionary algorithm, the predictors found performed very poorly, with most predictions deviating more than 100% from the actual ozone values. The use of a diversity preserving graph improved the accuracy of the predictors significantly, as did the use of the Rote-EA.

Table 3 – Standard Error

	T-Method	SGA	Cycle GBEA	RoteEA
Standard Error	1.949	15.579	7.709	7.203

The RoteEA seems to benefit from a high value for the sensitivity factor; it seems also to benefit from a greater number of trained data points. Both of these results were expected. The higher the sensitivity factor, the more rules are available to be trained per solution. The higher the number of data points to train on, the more opportunities the RoteEA has to refine its solutions. Each of these parameters, however, causes longer computation times if high values are chosen, and diminishing marginal returns are experienced.

The RoteEA does not require a high sensitivity factor or large training data sets to be effective, as demonstrated by Figures 4 and 5. Figure 4 demonstrates the ability of the RoteEA to capture relationships; none of the 11 runs differ vastly from the others despite large differences in the number of rules and the number of data points used to train. Figure 5 demonstrates the ability of the RoteEA to consistently train on small data sets and few rules.

This work represents the first time the RoteEA was applied to a problem of which the dependent variable lies on an unbounded range. As a new problem-solving method, the RoteEA demonstrated solid performance on a bounded problem; however, a weakness of the RoteEA was also discovered. Although the RoteEA seems adept at finding rules which describe relationships between independent variables and the dependent variable individually, the RoteEA in its current form is not able to explore the interrelationships between the independent variables of a problem; a significant disadvantage.

These evolutionary computation problems were compared to an existing prediction method that has been proven to perform well with a small number of variables and amount of training data. One of the concerns when using the Rote-EA is with overtraining the algorithm on the initial data set, a problem not encountered in the T-method. The algorithm did perform much better than the standard evolutionary algorithm, but did not perform as well as the more established T-method. Previous studies have shown that EC methods can sometimes outperform statistical models similar to the T-method [16], although this particular data set favored the T-method. This does indicate that as the T-method performs well on all correlated data, other factors affect the ability of evolved predictors to accurately perform on some data sets. This could be due to the cyclic nature of the collected data causing additional overtraining problems when applying a learning algorithm.

Based on past work, the overall result of the RoteEA method is similar to what one might expect if a system of finite state classifiers (FSCs) was evolved in a way which put a particular weight upon each individual FSC. However, the RoteEA seems to have some

advantages over FSCs. The RoteEA is believed to be able to build solutions in a less complex manner, thereby saving computation time.

This problem may be overcome by 'squaring' the RoteEA. Squaring the RoteEA entails utilizing a new set of chromosomes which act upon the previously existing ones, treating them as independent variables. In this way, the RoteEA should be able to develop rules which describe the cross-links between independent variables and better map the problem space. This should lead to the evolution of significantly better solutions, while only doubling the size of the existing RoteEA. The RoteEA is very lean in its design and exhibits short computation times; this change should therefore prove to be an inexpensive trade-off. In future work, a Rote-Squared EA may be introduced which will nullify the vulnerabilities of the current version.

V. References

[1] Ozone: Basic Information. (2008). *Air and Radiation*. Retrieved December 16, 2008, from Environmental Protection Agency Web site: http://epa.gov/air/ozonepollution/basic.html

[2] Ozone and Your Patients' Health: Training for Health Care Providers. (2007). *Air and Radiation*. Retrieved December 16, 2008, from Environmental Protection Agency Web site: http://epa.gov/air/oaqps/eog/ozonehealth/.

[3] Regional Ozone Information. (2008). *Transportation*. Retrieved December 16, 2008, from North Central Texas Council of Governments Web site: http://www.nctcog.org/trans/air/ozone/.

[4] Delmas, R., Circa, D., and Jambert, C. (1997) “Global Inventory of NOx Sources”, Nutrient Cycling in Agroecosystems, 48:51-60, Kluwer Academic Publishers.

[5] Durai, S. (2009). “Using Classification and Regression Trees to Predict Ozone Concentration”, Proceedings of the 2009 Industrial Engineering Research Conference.

[6] D. Ashlock, *Optimization and Modeling with Evolutionary Computation,* S pringer-Verlag, *2006.*

[7] Bryden, K.M., Ashlock, D. A., Corns, S. M., and Willson, S. J., "Graph Based Evolutionary Algorithms", *IEEE Transactions on Evolutionary Computations*, Vol. 10:5, pp. 550-567, October, 2006.

[8] S. M. Corns, D. A. Ashlock, D.S. McCorkle and K. M. Bryden, “Improving Design Diversity Using Graph Based Evolutionary Algorithms,” Proceedings of the 2006 IEEE World Congress on Computational Intelligence, pages 1037-1043, 2006.

[9] Syswerda, G., 1991, “A Study of Reproduction in Generational and Steady State Genetic Algorithms,” *Foundations of Genetic Algorithms*, pp. 94-101.

[10] Fowlkes, W. Y. and Creveling, C. M. (1995). *Engineering Methods for Robust Product Design*, Addison Wesley.

[11] Taguchi, G. (1986). Introduction to Quality Engineering: Designing Quality into Products and Processes, Asian Productivity Organization.

[12] Phadke, M. S. (1989). Quality Engineering Using Robust Design, Prentice Hall.

[13] Taguchi, G. and Jugulum, R. (2002). The Mahalanobis- Taguchi Strategy – A Pattern Technology System, John Wiley & Sons Inc.

[14] Parthiv, S., Cudney, E., Jikar, V., Drain, D., Jugulum, R., Ragsdell, K., and Shibano, H. (2008). "Predicting Annual Precipitation using T-Method", *Concurrent Engineering and Research Applications.*

[15] Parthiv, S., Cudney, E., Jikar, V. (2009). "Predicting Gas Mileage using T-Method", *2009 SAE World Congress*.

[16] Cudney, E. and Corns, S., "A Comparison of Finite State Classifier and Mahalanobis-Taguchi System for Multivariate Pattern Recognition in Skin Cancer Detection," 2011 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology.

III. USE OF DECISION TREES TO MODEL COMPLEX VARIABLE INTERACTIONS TO IMPROVE ROTE-LCS ACCURACY ON CLASSIFICATION PROBLEMS

Benjamin Daniels

Abstract – The Rote-4 Learning Classifier System (Rote-4 LCS) expands on earlier versions of the Rote-LCS algorithm by linking Rote rules (consisting of relational hypotheses) by logical operators within a binary tree. The result is a rule-set consisting of one or more 'major-hypotheses', each of which consist of a binary tree and in turn are comprised of 'minor hypotheses', each of which consist of a terminal node of the binary tree. The result is a significant expansion of the accuracy of the Rote-LCS for classification problems. An added benefit of the representation method is human-readability; it is easier to understand relationships and rules represented in tree form than other common representation schemes in the field of machine learning.

I. Introduction

The Rote-LCS is a novel representation among learning classifier systems (LCS), a subset of machine learning algorithms. The Rote has been modified several times since its inception, improving its accuracy and increasing its versatility.

Rote4-LCS evolves rule sets, where each rule, or 'major-hypothesis' is a decision tree (binary tree with logical operators for tree nodes) with rule segments at the terminal nodes, each of which is a relational statement referred to as a 'minor-hypothesis'. Each

solution consists of one or more rules, which, when evaluated, contributes a value which may be different if the rule is evaluated to true rather than false; these values are summed to produce a prediction or classification.

Unlike previous versions of Rote-LCS implementation [7, 8], Rote4-LCS is able to consider complex relationships between independent variables due to the inclusion of binary trees. Relational sentences convey a great deal of information in a relatively compact form. In addition, relational operators work well within binary tree structures because they may be evaluated to 'true' or 'false'. The result is a representation scheme which treats the solution space as a hyper-rectangle, where each independent variable represents one dimension. Each minor-hypothesis is a partition on a hyper-plane, and each rule is a multi-dimensional partition within the solution space. Unlike most or all other hyper-dimensional representation methods, however, Rote-LCS solutions may be represented in a two-dimensional graph as a tree of relational statements, allowing it to be human-readable [1, 2]. This is especially useful for applications, where a hypothesis must be derived concerning interactions between problem variables; one example of this is mapping of the carbon cycle.

The Rote-LCS is extremely competitive for classification problems, particularly Boolean problems. The Rote may also be useful for repetitive learning applications, such as those required by some AI control systems. In this paper, Rote4-LCS methodology is compared with earlier versions of the Rote in classification and real-value classification.

II. Background

The Rote-LCS is a supervised Pittsburgh learning classifier system (LCS) [3]. Learning classifier systems are machine learning algorithms intended for use in classification, reinforcement learning, and function approximation [4]. LCS algorithms are a subset of genetic algorithms (GA), which originated in 1975 with work by Dr. John Holland, and expanded upon by Dr. David Goldberg and others [5,6].

The addition of 'decision trees' of binary trees with tree nodes consisting of logical operators is partly inspired by earlier work of others, particularly Koza, involving the evolution of decision trees within a GA [7,8]. In this paper, decision trees are used to link rules such as those used by earlier research on the Rote. Rote4-LCS is partly inspired by Genetic Programming [5, 6] particularly the concept, representation, and methodology of evolving binary trees. A departure from earlier LCS methodology is the use of binary tree representation to create complex hypotheses from logical and relational operators. Unlike many prior supervised learning methods, the Rote attempts to represent relationships between independent variables in a way that could yield useful insight into the problem itself. By contrast, the majority of LCS methods use a ternary representation scheme, where binary inputs are compared to ternary rules. This is disadvantageous for at least two reasons; firstly, a mapping scheme must be created to enable this approach. Also, solutions will not be human-readable. Other common machine learning algorithms, such as neural nets and finite state machines also do not produce human-readable solutions.

Rote4-LCS is the fourth incarnation of the Rote. The first version was a classifier using only relational operators, the second used the same representation for real-value

classification, and the third included the use of the AND operator to build relationships between independent variables.

III. Methodology

A. Solution Representation

Solutions are trained via an evolutionary process with the traditional operators of an evolutionary algorithm: parent selection, recombination, mutation, and natural (survivor) selection. Each solution consists of a set of rules; the number of rules for a given solution may increase or decrease over the course of evolution.

Each rule is evaluated on each observation of a dataset during training; each rule of a solution is evaluated independently of the other rules of a solution. Each rule represents a hypotheses, which may be proven true or false for a given observation. A different value, or weight, will be assessed for a given rule depending on whether it has been evaluated to true or false. The results of each rule of a solution are summed, providing a prediction or classification (depending on the problem type) for the given observation. The distance between the prediction result and the dependent value of the given observation is used to assess a fitness score for the observation. The summed fitness over all observations in a dataset is used to provide a fitness score for a given solution during the training process.

Each rule in a solution is comprised of a set of seven chromosomes. A chromosome is defined here as a rule component which contains information that is directly altered by recombination and mutation during evolution. Rules also contain other components which are altered indirectly by evolution in response to changes in the chromosomes; this information is required for implementation of the method but is not significant to the

learning architecture theory described in this section.

The seven chromosomes of the solution architecture are designed to enable exploration of relationships between independent variables and the dependent variable using binary trees, logical operators, and relational operators. Each rule represents a 'major hypothesis' which consists of one or more 'minor hypotheses', which will be referred to henceforth as 'rule segments'. Each rule segment uses the relational operators (>=) and (<) to evaluate individual independent variables of an observation. Rule segments comprise the leaves (terminal nodes) of a binary tree which uses the logical operators AND, OR, XOR, and NOT to evaluate the result of the major hypothesis.

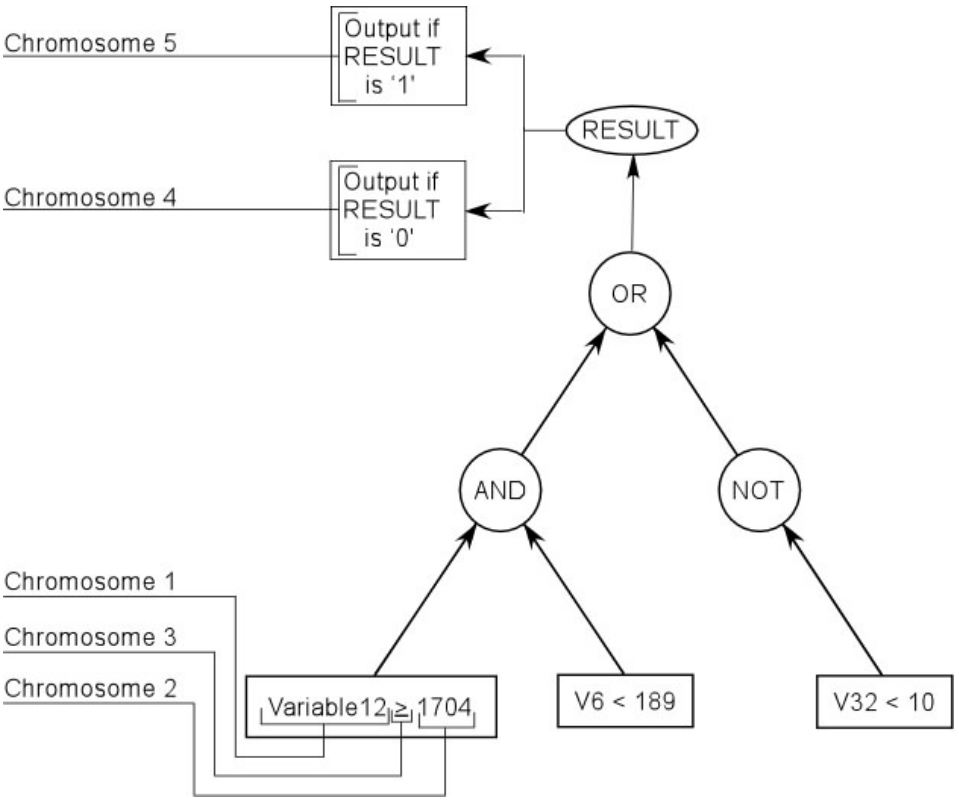

Figure 1, A hypothetical rule with a trunk length of three is shown. All values depicted are arbitrary. Chromosomes 1, 2, and 3 comprise rule segments. Chromosomes 4 and 5 hold weighted values associated with rule output. Chromosomes 6 and 7, not explicitly depicted, hold information associated with the nodes and edges of the graph, respectively.

As shown in Figure 1, above, the first three chromosomes contain all information relevant for rule segments. The first chromosome stores the independent variable to use for each respective leaf. The second chromosome holds a value for evaluating the variable of the rule segment. The third holds a Boolean value which determines whether the relational operator will be (>=) or (<). Chromosome 6 holds the information for tree nodes (but not terminal nodes); it contains the identity of the logical operator for each tree node. Chromosome 7 holds the node linkage information for the tree. Chromosomes 4 and 5 provide a weight to be output by the rule, depending on whether the major hypothesis is proven or disproven.

B. Initialization and Parsimony Control

Initialization begins with the random generation of a randomly selected number of rules; the default initial bounds, originating from informal experimentation, are between 5 and 30 rules for each population member. Each rule consists of a randomly generated tree with default trunk length randomly selected between 1 and 6 (default initial values originate from informal experimentation with parsimony control requirements for acceptable run times on a modern PC).

Before leaves are generated, the minimum and maximum bounds are found for Chromosome 2. This is done by going through all variables of all observations in the training file to find the highest and lowest values for each variable; these are used for the minimum and maximum bounds when generating (or mutating) the second chromosome of a given rule segment.

The minimum and maximum bounds for Chromosomes 4 and 5 values depends on the number of significant digits for the dependent (predicted) variable in problems involving prediction of a number (for instance, ground level ozone ppm). As a rule of thumb, the range for Chromosomes 4 and 5 is -100:100, with an order of magnitude added for every significant digit after the first four significant digits of the dependent variable; for instance, four significant digits in the dependent variable yields a range of -100:100, and eight yields -1000000:1000000 for Chromosomes 4 and 5. In classification problems, the default minimum and maximum values for Chromosome 4 and 5 are -100 and 100, respectively.

Parsimony control occurs in two ways: 1) providing a fitness subsidy to solutions with relatively lesser rule (binary tree) sizes, and 2) restricting the number of rules per solution to a user defined range with a specific minimum and maximum. The number of rules can have a tendency to grow quickly on some problems -- particularly number generation problems -- using the Rote4-LCS method. Because parsimony control spans the entire solution, not just individual rules, subsidizing smaller trees results in pressure to have fewer rules. In classification problems, this may actually result in fewer rules than is optimal. In the course of informal experimentation, no need for absolute restriction on tree size was discovered; providing fitness subsidy for smaller trees seemed sufficient to control tree size for all problems investigated.

C. Design of Evolutionary Operators

After informal experimentation, it was determined that the Rote seemed to perform

well with relatively light to moderate selective pressure for parent selection. Default parent selection for the Rote is fitness proportionate with replacement. Random selection with replacement also performed well in informal tests for parent selection.

Recombination methodology involves pooling all unique rules of two parent solutions, randomly determining the number of these rules to include in a child solution, and randomly selecting those rules. Individual rules are not recombined; rules are kept intact from the contributing parents. No method utilizing individual rule recombination was tested; reasons for this will be addressed in the discussion section.

The first step in mutation involves randomly selecting a number of mutations to perform on the population of child solutions. The maximum number of mutations which may be performed per generation is a user input, though a good default, established through informal experimentation, is a number of mutations roughly equal to the number of children per generation. Once a given child is randomly selected for mutation, one of four mutation possibilities will be chosen from, with equal probability:

- Rule addition
- Rule deletion
- Alteration of a leaf or rule weight
- Mutation of a tree

Rule addition and deletion are straightforward; addition results in the random generation of a new tree and leaves. The new tree, like initialized trees, will have a trunk length between 1 and 6. Deletion consists of the removal of one of the rules of the

solution. If alteration is chosen, then one of the first five chromosomes will be chosen for mutation. Alteration of a leaf involves a mutation in Chromosome 1, 2, or 3 of a randomly selected rule segment of a randomly selected rule; alteration of a rule weight involves a mutation in Chromosome 4 or 5 of the selected rule. The chromosome chosen for mutation will be a weighted probability: 3/11 chance of mutation for each of the chromosomes 1-3 and a 1/11 chance of mutation for each of the chromosomes 4 and 5. These weights were not justified experimentally, but were not chosen arbitrarily; it seems pertinent to mutate terminal nodes more often than trees, as they are more abundant.

Tree mutation is ‘subtree’ mutation [11, 12]. If this type of mutation occurs, an entire subtree is randomly generated and appended at a randomly chosen node of the original tree; any existing subtree at that node is deleted. The subtree will have a trunk length between 1 and 5 levels. The terminal nodes of the subtree are randomly generated.

Rote4-LCS has been tested informally with truncation and tournament survivor selection methods. Informal testing has demonstrated convergence on each to virtually identical fitness values, but with faster convergence for tournament selection. Informal testing for tournament selection seems to reveal a near-optimal number of tournament participants to be approximately equivalent to half of the child population.

D. Implementation of Chromosomes

Chromosomes are stored in a series of multidimensional jagged arrays of integers, where each dimension represents a 'domain,' or information type.

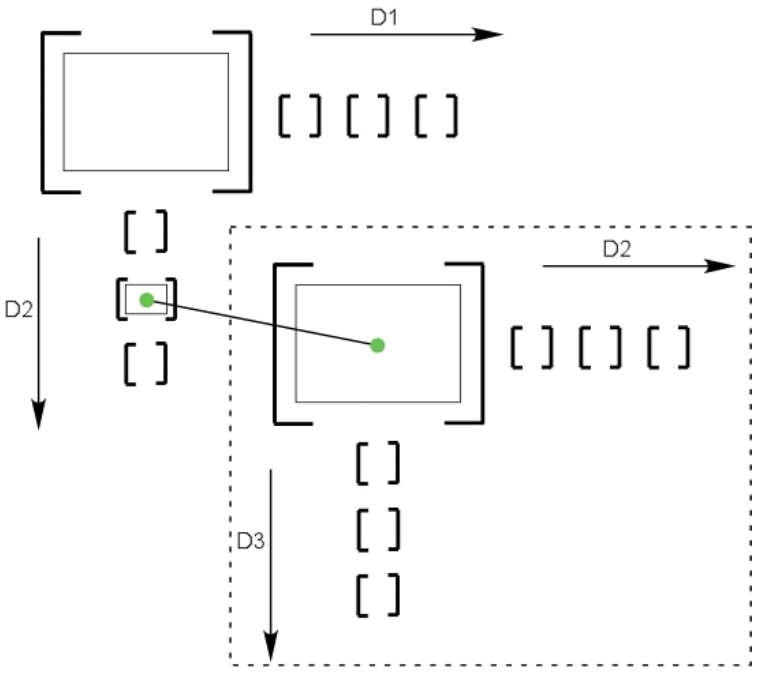

Figure 2, All chromosomes store information in the form of multidimensional arrays of integers. As an example for Chromosome 3, D1 represents a rule, D2 represents a rule segment of D1, and D3 represents a classification.

There are five domains over the seven chromosomes: Rule, Rule Segment, Trunk Level, Branch Location, and Classification. The first four of these are relatively self-explanatory; the last is used in classification problems where more than two possible classifications exist. For any other problem type, only the zeroth place of this domain is utilized.

E. Problem Type Configuration

Problem types are treated similarly, with a few differences in fitness calculation in the fitness function. For classification problems, negative and positive scores from all rules are simply added for all classifications and the classification with the highest score is chosen. For real-value classification, rule score is normalized against a finite range that suits the dependent variable. This range is calculated by taking the highest and lowest

value of the dependent variable over all observations and multiplying by the constant 1.25 (to enable extrapolation in the case of extreme test observations).

For classification problems, fitness is simply a sum of correctly classified observations. For real-value classification, fitness is calculated by measuring the distance between actual and predicted values and averaging this score over all observations. The maximum fitness score possible for real-value classification is zero, which would indicate no distance between actual and predicted values over all observations. The maximum fitness score for classification problems is equal to the number of observations.

IV. Results

A. Experimental Setup

Rote4-LCS was tested on both classification and real-value classification problems. Three datasets were used; two of these have been used to evaluate previous versions of the Rote [7, 8].

For all tests, tournament selection was used on a generational population of 10 members and 20 children. Mutation, survivor selection, and other parameters were confined to 'best mode' values as prescribed in Section III.

As discussed in Section III, classification and real-value classification problems have different reactions to tree subsidization; real-value classification problems often incur bloat, and classification problems may have the opposite problem. To investigate this further, eight experiments were performed; two using the breast cancer dataset used in [7], three using the ground-level ozone dataset used in [8], and three with an MMA

(mixed martial arts) dataset. The MMA dataset was obtained via a web-scraping application tailored to scrape three MMA websites for statistics of MMA contenders and match history. The MMA dataset was created exclusively for testing Rote4-LCS for classification.

The eight experiments each compared Rote4-LCS performance at different configurations of rule parsimony, where different ranges were utilized to disambiguate Rote performance expectation with regard to rule restriction on classification and real-value classification problems.

B. Results

Table 1 – Fitness scores are shown for five runs over eight experiments on three datasets

	Max Possible		Score 1	Score 2	Score 3	Score 4	Score 5		Average	Accuracy
Cancer Lower Bound 10	197		193	196	192	194	194		193.8	0.983756
Cancer No Restriction	197		193	195	195	196	192		194.2	0.985787
MMA Lower Bound 10	293		244	248	254	255	251		250.4	0.854608
MMA Lower Bound 20	293		240	243	250	256	248		247.4	0.844369
MMA No Restriction	293		249	241	250	##	247		246.75	0.84215
Ozone No Restriction	0		-1751	-2879	-3070	-1198			-2224.5	
Ozone Upper Bound 75	0		-1314	-1805	-1977	-2085			-1795.25	
Ozone Upper Bound 125	0		-592	-2630	-3796	-2132			-2287.5	

From the table shown, the best score for Rote-4 on the breast cancer dataset was 99.5%, with an average score of 98.5% for the five runs, on the configuration with a lower bound of 10 rules. The best score on the MMA dataset was 87%, with an average of 85.5% over the five runs, on the configuration with a lower bound of 10 rules. The best score on the ozone dataset, a real-valued classification problem, was a deviation of 5.92.

These scores are contrasted with Rote-1 and Rote-2, over the breast cancer and ozone datasets, respectively, as well as the results of Rote-1 on the MMA dataset tested

informally (results not shown). The best score achieved by Rote-1 on the breast cancer dataset was 99%, with an average of 96.7% over 10 runs [7]. The best score achieved by Rote-2 on the ozone dataset was 5.825; the worst run showed an average deviation of 8.815, and the average deviation for all runs was 7.203. This is contrasted with the Taguchi Method, which achieved an average deviation of 1.9 [8]. The best score achieved on the MMA dataset by Rote-1 in the course of informal testing was 75%.

V. Discussion

A. Review

Earlier versions of the Rote-LCS method did not use binary trees to link rule segments. Rules comprised simple relational statements, as with rule segments in Rote-4-LCS. Theoretically, this limited the ability of the method to explore complex relationships between the independent variables of a problem. Rote4-LCS methodology attempts to expand on previous work by linking rule segments using logical operators to build 'hypotheses' consisting of complex relational statements.

Based on results provided in this paper, as well as results from informal testing and comparison with previous work, Rote-4-LCS dominates earlier versions of the Rote and other algorithms in Boolean classification, and is dominated by earlier versions and by other algorithms in real-value classification. Formal and informal testing shows a wide gap between training and testing results for Rote4-LCS in real-value classification. Some degree of overtraining is apparent, as some solutions generated earlier within a run often perform better than later ones.

Though Rote-4 does not appear to be a good match for real-value classification in its current configuration, it remains undefeated by other algorithms in the benchmark Boolean classification problems attempted. Formal experiment appears to demonstrate that rule restriction may not be necessary for classification problems.

B. Future Work

A great deal of work remains for the Rote. This includes an improvement to increase the Rote's accuracy in real-valued classification problems. Four additional chromosomes will be added to enable each rule to project a range – weighted with a particular level of certainty – predicted to contain a good result. Each Rote rule will provide a predictive statement containing a relational operator, a weight, and an upper or lower bound (depending on the identity of the relational operator). An example rule may be such as "The desired value is greater than 24, with a certainty of 305." This places a weight of 305 on each discrete value from 25 to the upper bound. After all rules are evaluated, the center of gravity will be found for all overlapping ranges; this point will become the predicted result. This is in contrast to the current representation scheme for real-valued problems, which envisions the prediction range as a pressure gauge; each Rote rule either adds or subtracts a pressure quantity (weight) for a final pressure sum. The bounds of the gauge is normalized against the bounds of the range of possible real-valued answers of the problem; the final prediction is therefore a ratio with the pressure sum.

Also of important note is the 'Rote-Annex', a method which intends to use the Rote for evolving materials and systems by first modeling their characteristics using the Rote, and then predicting improvements based on the model. One example utility is the design

of composites. Composite materials manifest various properties based on their input materials and fabrication process. To maximize manifestation of useful properties while minimizing laboratory experimentation, a four-part process would involve: 1) many initial observations consisting of random input material combinations and variations in fabrication method, 2) generation of a model of material properties based on the resulting data, 3) optimization of desired material properties based on the model, and 4) an iterative approach in which successful improvements are interspersed with random changes, re-modeling, and re-optimization as the result is refined until a defined threshold of declining marginal gain is reached.

VI. Conclusions

Rote4-LCS uses binary trees and logical operators to link relational statements used in earlier versions of the Rote. Rote4-LCS appears to be extremely competitive in classification problems, remaining undefeated on the datasets used to benchmark classification performance on Boolean problems.

Rote4-LCS currently appears less useful for real-value classification problems, and may suffer from significant overtraining issues for this problem type. This issue, however, will be addressed in future work, as discussed in Section IV.B.

VII. References

[1] Loiacono, D., Marelli, A., Lanzi, P. “Support vector regression for classifier prediction.” *GECCO 2007: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 1806-1813.

[2] Marshall, J. R., Kovacs, T. “A representational ecology for learning classifier systems.” *Proceedings of the 2006 Genetic and Evolutionary Computation Conference* pp. 1529-1536.

[3] Smith, S. “A learning system based on genetic adaptive algorithms.” Ph.D. thesis, University of Pittsburgh, Pittsburgh, PA, USA, 1980.

[4] Urbanowics, R., Moore, J. “Learning Classifier Systems: A Complete Introduction, Review, and Roadmap.” *Journal of Artificial Evolution and Applications.* June, 2006.

[5] Man, K., Tang, K. Kwong, S. “Genetic Algorithms: Concepts and Applications.” *IEEE Transactions on Industrial Electronics*. Vol. 43. No. 5, October, 1996.

[6] Booker, L.B. Goldberg, D. E. Holland, J. H. “Classifier Systems and Genetic Algorithms.” *Artificial Intelligence – AI.* , Vol. 40, No. 1-3, pp. 235-282, 1989

[7] Daniels, B., Corns, C., Cudney, E., “Introduction of R-LCS and Comparative Analysis with FSC and Mahalanobis-Taguchi Method for Breast Cancer Classification.” *2012 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology.* pp. 283-289.

[8] Daniels, B., Corns, C., Cudney, E., “A Comparison of Representations for the Prediction." *2012 IEEE World Congress on Computational Intelligence.* pp. 1-8.

[9] Koza, J. R. “Hierarhical Genetic Algorithms Operating on Populations of Computer Programs.” *Proceedings of the 11th Joint Conf. on Genetic Algorithms.* Volume I. pp. 768-774. 1989.

[10] Koza, J. R. “Concept Formation and Decision Tree Induction using the Genetic Programming Paradigm.” *Parallel Problem Solving from Nature.* Vol. 496. Pp. 124-128. 1991.

[11] Luke, S., Spector, L. “A Comparison of Crossover and Mutation in Genetic Programming.” *Genetic Programming 1997: Proceedings of the Second Annual Conference.* 1997.

[12] Koza, J.R., “Genetic Programming: On the Programming of Computers by Means of Natural Selection.” The MIT Press. 1992. p. 106.

SECTION

3. CONCLUSION

Rote-LCS is a unique learning classifier system with exceptional results on classification problems (particularly Boolean problems). and probable applicability to at least some reinforcement learning problems. It further has a rare advantage among learning classifiers and machine learning algorithms in general -- a representation scheme that is able to offer human readability. Rote-LCS is intended only for problems for which classifiers are applicable, particularly supervised learning classification problems. The Rote thus far does not contain functionality for unsupervised learning. The Rote has only had mediocre success on real-valued classification, although a modification will be applied in future work which is believed to be likely to fix this issue.

The first incarnation of the Rote-LCS established the basic rule representation methodology used throughout all subsequent versions of the Rote. Relational hypotheses are used to create partitions which define solutions within a hyper-rectangular search space, where each independent variable represents a dimension within the search space. Better than 99% accuracy is achieved on the best solutions on a breast-cancer dataset, dominating the other algorithms benchmarked against.

The second version of the Rote-LCS expanded the Rote's functionality to include real-valued problems, from only Boolean problems. Although the Rote-LCS was defeated by one of the benchmarked algorithms on a ground-level ozone prediction dataset, it implied its applicability to a wider range of problems than originally anticipated.

The latest version of the Rote-LCS includes the use of binary trees and logical operators (decision trees) to significantly expand the accuracy of the Rote by allowing

search of relationships between independent variables. This can also be imagined as an alteration to the way partitions are formed in the multi-dimensional search space; rather than considering each dimension and partition individually, partitions themselves are now multi-dimensional. The use of trees to link relational statements maintains the option for human-readable solution outputs; while it is not possible to visualize hyper-dimensional spaces, partitions may be represented two-dimensionally as trees which are not difficult to comprehend and may be represented visually.

Future work planned for Rote-LCS includes an improvement to increase the Rote's accuracy in real-valued classification problems; this will involve the addition of four additional chromosomes to enable each rule to project a range – weighted with a particular level of certainty – predicted to contain a good result. Further work will be done to produce the Rote-Annex, a method by which Rote modeling will enable reduction of lab work and experimentation for various applications, such as that of composite material creation. Also, work is planned to benchmark the Rote-LCS against other algorithms on various problems to determine how best to apply the Rote for maximum societal benefit.

VITA

Benjamin Daniels was born in Houston, Missouri, United States of America. He received his primary education in Raymondville, Missouri, and his high school education in Houston, Missouri. He received his Bachelor of Science in Engineering Management at Missouri University of Science and Technology (Missouri S&T), in Rolla, Missouri in 2008. He performed independent work in algorithmic market trading and alternative energy research, making discoveries which he would build upon when he returned to the Engineering Management and Systems Engineering department at Missouri S&T to pursue a Master's degree in Systems Engineering.

Working as a graduate researcher, Benjamin Daniels has made advances in the fields of anonymous network security, computational bioinformatics, and machine learning, publishing papers at IEEE conferences, and completing research for several journal articles. In May 2015, he received his Master's degree in Systems Engineering from Missouri S&T.