International Journal of Engineering Research and Reviews (IJERR)
Vol. 2, Issue 1, pp: (16-22), Month: January-March 2014, Available at: www.researchpublish.com

A Survey of Classifier Designing Using Genetic Programming and Genetic Operators

A. M. Mansuri[1], Manish Verma[2], Pradeep Laxkar[3]

[1,2]Department of Computer Science & Engineering, MIT Mandsaur, Mandsaur, M.P., India
[3]Department of Computer Science & Engineering, ITM Universe, Vadodra, Gujarat, India

Abstract: **Classifier is a type and can own generalizations, thereby making it feasible to define generalization relationships to other classifiers. In this classifier is a redefinable element, as it is feasible to redefine nested classifiers. In the classification given a set of data representing examples of a target concept make a model to explain the concept and this model classifying future or unknown cases. Classification also estimates the accuracy of the model. This paper presents a survey of current methods for classification designs and the various existing issues. Many tasks that have been conventionally done by humans are now being passed to machines; one may therefore expect there are now an abundance of vision problems posed to computers. In this classifier system is much more than a simple expert system that can learn from experience (which in itself is an immense boon). In this paper design classifier for more than one class and for designing classifier used different operation of genetic programming. The results show that by applying different crossover together with different Mutation increase the performance of the classifier.**

Keyword: **Genetic Algorithm, classifier, data sets, Genetic Programming, Genetic Operators**

I. INTRODUCTION

Classifier is an abstract UML metaclass to support classification of instances according to their features. Classifier describes a set of instances that have common features. A feature declares a structural (properties) or behavioural (operations) characteristic of instances of classifiers. Classifier is a category of Unified Modelling Language (UML) elements that have some common features, such as attributes or methods.[1] A classifier is an abstract metaclass classification concept that serves as a mechanism to show interfaces, classes, data types and components. A classifier describes a set of instances that have common behavioural and structural features (operations and attributes, respectively). A classifier is a namespace whose members can specify a generalization hierarchy by referencing its general classifiers. A classifier is a type and can own generalizations, thereby making it possible to define generalization relationships to other classifiers. A classifier is a redefinable element, as it is possible to redefine nested classifiers. All objects that can have instances are classifiers.

A classifier system (CS) is a machine learning system that learns syntactically simple string rules, called classifiers, as introduced by Holland and Reitman [1978]. These classifiers guide the system's performance in an arbitrary environment. A classifier system derives its name from its ability to learn to classify messages from the environment into general sets and is similar to a control system in many respects. As a control system [3] uses feedback to "control" or "adapt" its output for an environment, a classifier system uses feedback to "teach" or "adapt" its classifiers for an environment. Since most environments are not guaranteed to be static and learning can never be known to be complete, the learning process may never cease. The classifier system has developed out of the merging of expert systems (as described in, Charniak and McDermott [1985]; Waterman [1986]), and genetic algorithms as originated by Holland [1975]. This synthesis has overcome the main drawback to expert systems; namely, the long task of discovering and inputting rules. Using a genetic algorithm, the CS learns the rules needed to perform in an environment, (in this current study the environment is structural shape optimization).

The development of expert systems has helped advance many fields by having computers reason more like humans. Expert systems allow the computer to use rules. Some rules are concrete facts while others are rules-of thumb (heuristics) that work in most situations, but the specific rules are still unknown for all situations. As mentioned above, an obstacle to the wider use of expert systems is the fact that all rules for the expert system must be provided by humans, and therefore have to be collected from literature and interviews. Furthermore, since the rule set is static, the system can never discover if a rule-of-thumb is non-applicable, and should consequently be eliminated or modified. Another conflict occurs when more than one rule may be applicable to a situation; all such conflicts must be foreseen or the system may halt, not knowing how to proceed. Classification typically involves the mapping of an N-dimensional feature vector to one of multiple classes. The N-dimensional feature vector is like a point in the N-dimensional feature space.

The learning capability of the CS greatly enhances the realization of the expert system's promise. With the classifier system, one may input rules (as with an expert system) or start from random rules, or, as is likely to be done in most real world scenarios, input as many rules as possible and let the classifier system learn new ones and try to improve the entered rules. A classifier system is much more than a simple expert system that can learn from experience (which in itself is an immense boon). As the rest of this chapter will attest, a classifier system is a general machine learning system applicable to diverse environments, able to learn with incomplete information and classify the environment into hierarchies.

II. CLASSIFIER

The last few years have witnessed important new developments in the theory and practice of pattern classification. The classifier system receives information about the environment, performs internal processing and then effects the environment. It then uses feedback about the effect on the environment to learn from the experience. This arrangement has the classifier system in learning mode, because the classifier system is utilizing the feedback to learn from experience. Conversely, if no feedback is provided, the classifier system is in application mode. Application mode is utilized after sufficient learning is accomplished. The subsequent discussion up until Section 3.4 deals with the classifier system exclusively in learning mode The purpose of this survey is to offer an overview of some of these ideas of the analysis of some of the important data sets through different algorithms (i.e. Grow method, Full method etc.)A classifier system (CS) is a machine learning system that learns syntactically simple string rules, called classifiers, as introduced by Holland and Reitman [1978]. These classifiers guide the system's performance in an arbitrary environment. A classifier system derives its name from its ability to learn to classify messages from the environment into general sets and is similar to a control system in many respects. As a control system [3] uses feedback to "control" or "adapt" its output for an environment, a classifier system uses feedback to "teach" or "adapt" its classifiers for an environment. Since most environments are not guaranteed to be static and learning can never be known to be complete, the learning process may never cease. The classifier system has developed out of the merging of expert systems (as described in, Charniak and McDermott [1985]; Waterman [1986]), and genetic algorithms as originated by Holland [1975].

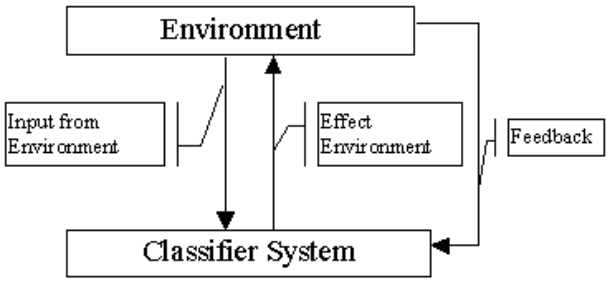

Fig. 1.Interactions between Classifier System and Environment

This synthesis has overcome the main drawback to expert systems; namely, the long task of discovering and inputting rules. The figure 1 shows the interaction between classifier and environment using a genetic algorithm, the CS learns the rules needed to perform in an environment, (in this current study the environment is structural shape optimization). To solve a complex classification problem, many researchers have resorted to ensemble methods, in which multiple classifiers are combined to achieve an accurate classification decision. For example, the Viola-Jones classifier [32] uses a cascade of classifiers, each of which focuses on different spatial and appearance patterns. Boosting [10] constructs a committee of weak classifiers, each of which focuses on different input distributions. Multiclass classification problems are very often reduced to a set of simpler (often binary) decisions, including one-vs-one [11], one-vs-all, error-correcting output codes, or tree-based approaches [11]. Intuitively, different classifiers provide different "expertise" in making certain distinctions that can inform the classification task.

III. GENETIC PROGRAMMING

Genetic programming (GP) in which typically chromosome is encoded as a parse tree. Indeed, GP has been used previously to optimize feature extraction and selection stages. The field of evolutionary computation (EC) has a long history keeping a variety of alternate methods and approaches to problem solving. Some important approaches that have

been applied to problems based on Genetic Algorithms (GA) classifier systems evolutionary strategies and evolutionary programming . Both GP and GA are being used for image feature extraction, selection, and classifiers optimization. However, in recent the field of Genetic Programming (GP) has emerged as an effective means for evolving solutions to problems.GP can represent solution in the form of computer programs. GP program trees are constructed through a set of primitive components. The internal nodes of the tree are called functions, while the leaf nodes of the tree, which take no arguments, are called terminals. The terminals comprise the input for the program. Together the set of functions and terminals combines to give the set of possible components that can be used by GP to construct programs An initial population of individuals is generated and tested against the problem at hand. An objective fitness value is assigned to individuals based upon their ability to solve the problem, and then fitness proportionate selection is used to select which individuals will pass their genetic material into the next generation using genetic operators like crossover, mutation, and reproduction operations. GP is a new and fast developing method for automatic learning, where evolutionary methods are used to search for a computer program that can solve a task. The powerful evolutionary search and expressive computer program representation make GP an important research area. The below figure 1.2 shows the representation of nodes of a tree that shows that a root node of tree and the components node of a tree, the component node are in different nodes but the parent node is root node.

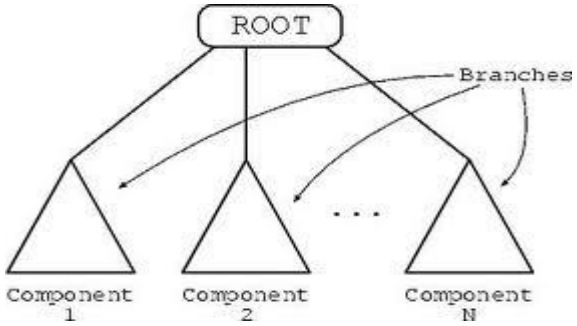

Fig. 2 Tree representation in GP

GP has been used to evolve many types of structure such as decision trees and classification rule sets. The numerical expression classifier has also been developed recently, and has been seen to be applicable to a wide range of problems. Each numeric expression classifier program typically returns as its output a single floating-point value, which is a high level representation of the feature inputs. Steps of GP for evaluating the programs:

1. Create initial random population of programs.
2. Evaluate the fitness of each program using training data.
3. Modify population of programs to get new generation, using genetic operators -reproduction, crossover and mutation.
4. If a good program is found or prespecified number of generations gets completed, finish, else go to step 2.

III. REPRESENTATION & SURVEYS OF PROGRAMS

GP evolves computer programs, traditionally represented in memory as tree structures. The problem of classification is about guessing or predicting the unknown class of an observation. An Observation is often a collection of numerical and/or categorical measurements represented. Today the word 'survey' is used most often to describe a method of gathering information from a sample of individuals. This 'sample' is usually just a fraction of the population being studied.

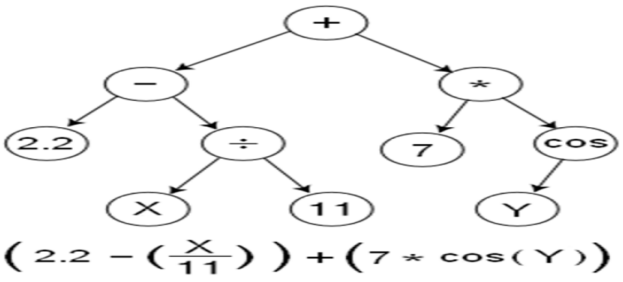

Fig. 3. Representation of program in GP

Figure 3 represent of programs how the programs is represent in GP. All members of the population are studied; surveys gather information from only a portion of a population of interest-the size of the sample depending on the purpose of the study. In this work the survey of the different data sets and population of the programs and calculate the fitness of the programs. In a good survey, the sample that has been studied represents the target population..In this research work the survey of the program and calculating the size and depth of the program by using the different operators of GP .The program represent the parent node and child node are created and the parent/child is selected/rejected on the basis of fitness, size and depth limit.

IV. TRAINING, TESTING AND VALIDATION SETS

In machine learning, the aim is often to evaluate how good the learning method is compared to previous methods. In this case, only a portion of the entire data set, called the training set, is used to train the algorithm. The rest of the data set, called the test set, is used to evaluate how good the method is on unseen data (data it wasn't trained on). The training set may be further divided into a training and a validation part, in order to control over-fitting. As training progresses, a learning algorithm will fit the data in the training set increasingly well. At some stage, the ability of the algorithm to generalize to the test set may suffer, in what is termed over-fitting or over-training. The purpose of the validation set is to control over-fitting. The performance of the learning method on the validation set is used as a barometer for the method's performance on the test set.

A. *Data Sets*

A dataset (or data set) is a collection of data. Most commonly a dataset corresponds to the contents of a single database table, or a single statistical data matrix, where each column of the table represents a particular variable, and each row corresponds to a given member of the dataset in question. The dataset lists values for each of the variables, such as height and weight of an object, for each member of the dataset. Each value is known as a datum. The dataset may comprise data for one or more members, corresponding to the number of rows. The term dataset may also be used more loosely, to refer to the data in a collection of closely related tables, corresponding to a particular experiment or event. The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. For the different classifier system we have been used different data sets like (Breast Cancer, Car Evaluation etc.)

V. THE GENETIC OPERATIORS APPLIED TO DATA SEETS

There are two classes of new genetic operator which we have used for classification allow changes to be made to program structure. Both are variants of the standard GP operators.

- Mutation: The argument with an inclusion factor of zero is replaced by a randomly generated subtree. This operation is shown in figure.

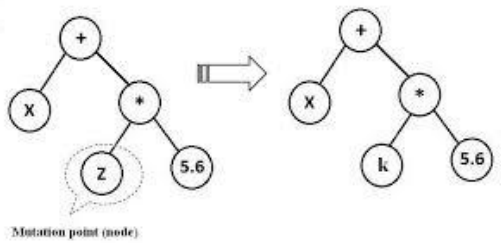

Fig. 4 Mutation operation in GP

- Crossover: The argument with an inclusion factor of zero is replaced by a randomly selected subtree from the population. This operation is shown in figure.

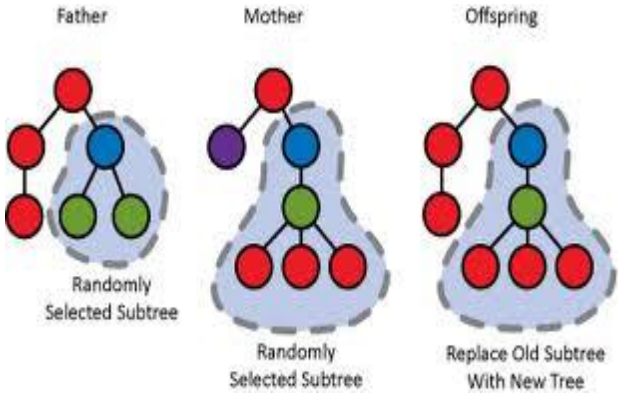

Fig.5. Crossover operation in GP

VI. CONCLUSION AND RESULTS

Classifier system is an adaptive system that classifies all training samples.The main goal of this paper is to investigate a novel approach to design multiclass classification in Genetic Programming (GP).The below figure shows the different data sets like Breast Cancer, Car Evaluation classification results in our work. The figure 5 and figure 6 shows the result of the classification of the Breast Cancer, Car Evaluation data. And shows that when we increase the no. of training sample the testing error reduces. In this work, a new Genetic Programming classification system with a dynamic class projection was implemented and tested In this populations of non-linear transformations are evolved to transform the input training data to be classified to a new one-dimensional space with a maximum discrimination between the projected classes. The classification task becomes much easier with the transformed data and the new testing samples are then transformed with the generated transformation and assigned to their corresponding class using a simple search
Algorithm. Some result of this research work is show below figures. Table 1-3 shows the parameter of this work which used for this research work and other tables for the data sets and its attributes.

Table 1. Parameter Used

Parameter	Values
Probability of Crossover Operation	85%
Probability of Mutation Operation	12%
Population Size	100-400
Number of Generations	5-50

Table 2. Car Evaluation Data Set

Data Set Characteristics:	Multivariate	**Number of Instances:**	1728
Attribute Characteristics:	Categorical	**Number of Attributes:**	6
Associated Tasks:	Classification	**Missing Values**	No

Table 3. Breast Cancer Data Set

Data Set Characteristics:	Multivariate	**Number of Instances:**	286
Attribute Characteristics:	Categorical	**Number of Attributes:**	9
Associated Tasks:	Classification	**Missing Values**	Yes

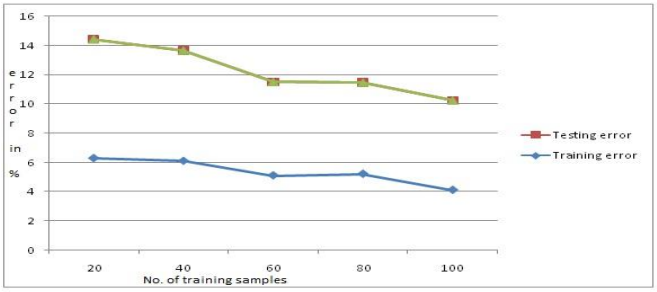

Fig.5. Numbers of training samples versus percentage error

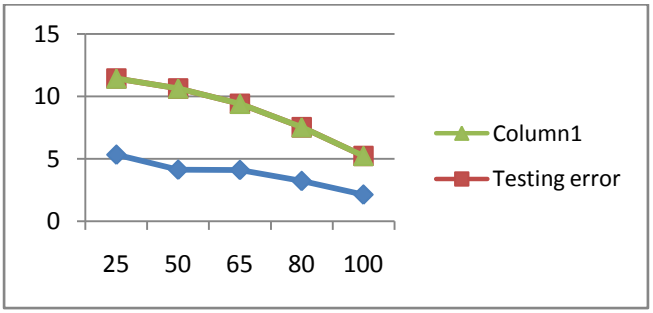

Fig.6. Numbers of training samples versus percentage error

REFERENCES

[1] *J. Aczel and J. Daroczy, On measures of information and their characterizations, New York: Academic, 1975*.
[2] R. C. Casey and G. Nagy, "Decision tree design using a probabilistic model," IEEE Trans. Inform. Theory vol. IT-30, 93-99 (1984).
[3] R. L. P. Chang, "Application of fuzzy decision techniques to pattern recognition and curve fitting," Ph.D. Thesis, Dept. of EECS, Princeton Univ., 1976.
[4] R. L. P. Chang and T. Pavlidis, "Fuzzy decision tree algorithms," *IEEE Trans Syst. Man Cybernet.*, vol. SMC-7, 28-35 (1977).
[5] P. A. Chou and R. M. Gray, "On decision trees for pattern recognition," IEEE Symposium on Information Theory, Ann Arbor MI., 69, 1986.
[6] M. Chou, T. Lookabaugh, and R. M. Gray, " Optimal pruning with applications to tree structured source coding and modeling," *IEEE Trans. Inform. Theory*, vol. IT-35, 299-315 (1989).
[7] G. R. Dattatreya and L. N. Kanal, "Adaptive pattern recognition with random costs and its application to decision trees," IEEE Trans. Syst. Man Cybernet., SMC-16, No.2, 208-218 (1986).
[8] http://archive.ics.uci.edu/ml/datasets.html?
[9] W. Lee, and S. Stolfo. "A Framework for Constructing Features and Models for Intrusion Detection Systems", Information and System Security, 2000, Vol. 3, No. 4, pp. 227–261.
[10] M. Sabhnani, and G. Serpen, "Application of Machine Learning Algorithms to KDD Intrusion Detection
[11] Dataset within Misuse Detection Context", in Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications (MLMTA 2003), Las Vegas, NV, June 2003, pp. 209-215.
[12] A. Küçükyılmaz, Pattern Classification: A Survey and Comparison. Department of Computer Engineering, Bilkent University, Ankara, Turkey. April 2005.
[13] A. K. Jain, R. P.W. Duin, and J. C. Mao, "Statistical pattern recognition: a review", IEEE Transactions on Pattern Analysis and Machine Intelligence, January 2000, Vol. 22. No. 1, pp. 4-37.
[14] R. Agarwal, and M. V. Joshi, PNrule: A New Framework for Learning Classifier Models in Data
[15] Mining, Technical Report TR 00-015, Department of Computer Science, University of Minnesota, 2000.
[16] AGNELLI, D., BOLLINI, A., AND LOMBARDI, L. Image lassification: an evolutionary approach. Pattern Recognition Letters 23, 1-3 (2002), 303–309.
[17] ALLWEIN, E. L., SCHAPIRE, R. E., AND SINGER, Y. Reducing multiclass to binary: A unifying approach for margin classifiers. In Proc. 17th International Conf. onMachine Learning (2000),Morgan Kaufmann, San Francisco, CA, pp. 9–16.
[18] ANDRE, D. Learning and upgrading rules for an OCR system using genetic programming. In Proceedings of the 1994 IEEE World Congress on Computational Intelligence (Orlando, Florida, USA, 27-29 June 1994), IEEE Press.
[19] BANZHAF, W., NORDIN, P., KELLER, R. E., AND FRANCONE, F. D. Genetic Programming˜: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications. dpunkt – Verlag f ¨ur digitale Technologie GbmH and Morgan Kaufmann Publishers, Inc., Heidelberg and San Francisco CA, resp., 1998.
[20] BERANEK, B., AND MONTANA, D. J. BBN technical report 7866: Strongly typed genetic programming, June 28 1994.
[21] BRAMEIER, M., AND BANZHAF, W. A comparison of linear genetic programming and neural networks in medical data mining. IEEE Transactions on Evolutionary Computation 5, 1 (2001), 17–26.
[22] BRAMEIER, M., AND BANZHAF, W. Neutral variations cause bloat in linear GP. In Proceedings of the Sixth European Conference on Genetic Programming (EuroGP-2003) (Essex, UK, 2003),
[23] E. C. C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, Ed., vol. 2610 of LNCS, Springer Verlag, pp. 286–296.
[24] CASTILLO, P. A., ARENAS, M. G., MERELO, M., ROMERO, G., RATEB, F., AND PRIETO, P. Comparing hybrid systems to design and optimize artificial neural networks. In Genetic Programming 7th European Conference, EuroGP 2004, Proceedings (Coimbra, Portugal, 5- 7 Apr. 2004),
[25] M. Keijzer, U.-M. O'Reilly, S. M. Lucas, E. Costa, and T. Soule, Eds., vol. 3003 of LNCS, Springer-Verlag, pp. 240–249.
[26] DIETTERICH, T. G., AND BAKIRI, G. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2 (1995), 263–286.
[27] De Leeuw Edith, E. D. (2005). To mix or not to mix data collection modes in surveys. Journal of Official Statistics, 21(2), 233–255.
[28] J. P. Marney, D. Miller, C. Fyfe, and H. F. E. Tarbert. Risk adjusted returns to technical trading rules: a genetic programming approach. In 7th International Conference of Society of Computational Economics, Yale, 28-29 June 2001.
[29] Massey, J. A. Clark, and S. Stepney. Evolution of a human-competitive quantumfourier transform algorithm using genetic programming. In H.-G. Beyer, et al., editors,GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation, volume 2, pages 1657–1663, Washington DC, USA, 25-29 June 2005.ACM Press. ISBN 1-59593-010-8.