Multiclass Classification using Genetic Programming

Hajira Jabeen, Abdul Rauf Baig and Jamil Ahmed

***Abstract*— Data classification has received increasing interest lately. It is a challenging task due to uncertainty, unpredictability and inconsistency of data. This challenge increases in the case of multi-class classification. Genetic Programming (GP) has shown promising results as an efficient and robust classification strategy. For multiclass classification, multi-tree chromosome classifiers can be used, where each tree is an arithmetic expression that discriminates between one and rest of the classes. In this paper, we have emphasized fitness of an individual tree in multi-tree classifiers which adds to the fitness of whole chromosome and results in better classifier efficiency. A series of experiments have been conducted to support the efficiency of proposed algorithm and the results have been found encouraging.**

I. Introduction

In Machine learning context, learning can be divided into two main types; unsupervised and supervised learning. Supervised learning involves the known class labels. The data attributes are divided in a way that one or more attributes represent the desired-dependent characteristic. That desired characteristic must be predicted automatically. The task of learning is to learn the behavior of occurrence of that attribute and correctly predict the presence or absence of that particular attribute. To carry out such a learning process, some sample data is provided to learn the relationships among attributes in the form of a classifier. These relationships are usually unknown and non-obvious. The data is divided into two parts; training data and test data. The training data is used to create a classifier and test data is used to check the performance of the classifier and to check its predictive accuracy. The classification is most common and well studied problem in machine learning community. This is due to the fact that the tremendous amount of data is being generated continuously and there is a need to extract relationships among this data. Moreover, these relationships are often too complex or the data may be unpredictable or uncertain. The problems make classification, a very challenging task.

Genetic programming (GP) is an evolutionary algorithm used to automatically construct computer programs. GP has been applied to a variety of problems and it has been found as a very efficient technique for classification tasks. GP is a very powerful evolutionary technique that allows flexible representations in the form of syntax trees. This allows GP to learn inherent and hidden relationships in the data without human intervention. GP based data classification offers numerous advantages, as discussed by Poli, "Genetic programming is an evolutionary computation technique that automatically solves problems without requiring the user to know or specify the form or structure of the solution in advance"[1]. GP has been successfully used to evolve classifiers of different types, this includes decision tree evolution [2], evolution of classification rules [3], [4], and evolution of mathematical expression based classifiers [5]. GP can search the space of possible classifiers resulting in various different structures with slight difference in accuracy. Interpretation of a classifier is fast and easy. Relationships important for a class can be learnt implicitly. The GP based classification is data distribution free and does not require any preprocessing of the data. However, classification using GP requires very long training time. The size of individual population member starts increasing during the evolution (bloat).

To exploit the benefits of GP for classification, several methods have been introduced in the recent years.

In this paper, we have presented an improvement of the multi-tree based classification scheme. In a combined performance view of *one versus all* classifier, accuracy of each individual classifier plays a very important role. In the scheme proposed by Kishore et al [6], there is no emphasis on best individual classifiers during the evolution phase. We search for best classifiers in each slot and create a best chromosome constituting the best classifiers. We have proven the efficiency of our approach using several datasets taken from UCI repository.

The next section of the paper gives an overview of classification schemes using GP. Section III details the proposed classification algorithm. Section VI presents the results and Section V concludes the findings of this work with some future propositions.

II. Literature Review

GP has been found efficient for the classification tasks. As, the common structure to evolve programs using genetic programming is use of syntax trees; it seems natural to evolve decision trees using GP. The decision trees are tree based classifiers that contains root and leaf nodes. A path from the root to a leaf node, represent a rule with the leaf node denoting the class or consequent of that particular rule. One such technique, to evolve decision trees using GP was

Dr. Hajira Jabeen is an Assistant Professor at Iqra University, Islamabad, Pakistan .email:hajira@iqraisb.edu.pk

Dr Abdul Rauf Baig is a professor at National University of Computer and Emerging Sciences, Islamabad, Pakistan. e-mail: rauf.baig@nu.edu.pk

Dr. Jamil Ahmed is vice president at Iqra University, Islamabad, Pakistan. email:jamil@iqraisb.edu.pk

introduced by Koza [2] 1992, almost parallel to introduction to GP[7]. Folino [8] et al proposed a parallel GP based approach, it uses the concept of cellular GP for decision tree evolution. However, the efficiency of decision trees is disturbed if the training data is too small or too large. It makes the decision trees unstable. Moreover a decision tree can become very large requiring further steps for detection and pruning of such inefficient parts.

Tsakonas [9] used grammar based GP to evolved intelligent structures. Freitas [10] introduced a framework for classification using SQL queries. SQL based encoding enables faster and parallel execution and offered scalability and privacy.

The classification problems in the real world are usually multi class classification problems. On the other hand, most of the machine-learning-classifiers are binary in nature. This increases the need to efficiently classify multi-class data. Some intelligent methods are desired to use binary classifiers for multi-class classification. One of the well known methods is one-versus-all method.

The method proposed by Kishore et al [11] decomposes an 'n' class classification problem into n binary classification problems. For each problem, a genetic programming classifier expression GPCE is evolved. One GPCE is able to differentiate between one class and the remaining classes. The system requires multiple evolution phases depending upon the number of classes present in the training data. Another feature of the proposed approach is the conflicts among multiple classifiers. Although a conflict resolution operation is also presented but it decreases the overall accuracy of the system.

Loveard[12][13][14] has proposed various classification strategies using GP. The method for nominal attribute classification involves execution branching and other transforms nominal value to binary values. Some other methods for multiclass classification include a decomposition method, similar to Kishore et al. Range selection, where real value ranges are selected to represent different classes I n the data. These ranges include static range selection and dynamic range selection where range is adapted during the training process. The other methods are class enumeration and evidence accumulation.

Muni at al [6] proposed a novel method for multiclass classification which includes a multi-tree representation. The method is efficient in the manner, that it requires only one evolution phase of GP, to create a classifier for all the classes. Some other noticeable features of this approach is A new notion of unfitness of trees, a new crossover operator that takes care of multiple classifiers. However, the working of the resulting classifier is similar to binary decomposition method and the method suffers from conflicts among multiple classifiers in a single chromosome. A conflict resolution method is also proposed to overcome the limitation, but it adds an overhead of extra computation.

Multi-objective optimization methods have been used to evolve classifiers using GP. This is known has MOGP. The two goals are, usually, accuracy and size of classifiers. However these goals are kept generic and any two goals can used to create classifiers using GP.

Besides above mentioned methods many others have also been proposed for data classification using GP. GP suffers from a downside of unproductive code increase (bloat) during evolution. This amplifies the program density during the evolution process without helpful raise in fitness. This increase in density must be tackled clearly by placing a bound on the upper limit of tree depth or nodes of the tree.

III. Proposed Classification Algorithm

The algorithm used for evolving classification trees for Multi-Class classification is presented. One of the specialties of this algorithm is the Multi-Tree representation that makes it different from other GP Multi-Class classification algorithms.

A. Classification Algorithm

The classification algorithm is summarized in Figure 1. The number of classes is determined from the input dataset to set the count of trees in the evolving chromosome. Initial trees are created using ramped half and half method. Chromosomes for evolution are populated using these initial trees, where each chromosome has trees equal to the number of classes present in the training data. We have used incremental learning as proposed by the authors for efficient classifier evolution. Initially a part of data is used as training data for certain number of generations (HalfGen) which is incrementally increased during evolution until whole data is utilized as training data. Two different fitness values have been used for full and incremental mode. Fitness 1 is fitness function used during the incremental learning. And fitness 2 is the fitness function used during full learning. Three evolutionary operators are used crossover, mutation and reproduction. The chromosomes for crossover are selected using tournament selection, chromosomes for mutation are selected randomly and chromosomes for reproduction are chosen using proportionate selection.

Termination condition (term cond) is either completion of generations or a classifier having 100 % accuracy

The best chromosome is saved and returned at the end of evolutionary process. Now we will explain the details of algorithm with proposed modifications.

A. Chromosome Representation

The multi-tree representation of a chromosome for the classification is used. There are as many trees in a chromosome as the classes present in the data. Therefore, each tree position represents a class label and only one tree is supposed to output value greater than or equal to zero

The whole output of a chromosome is a vector of real values, if all values are negative or more than one value

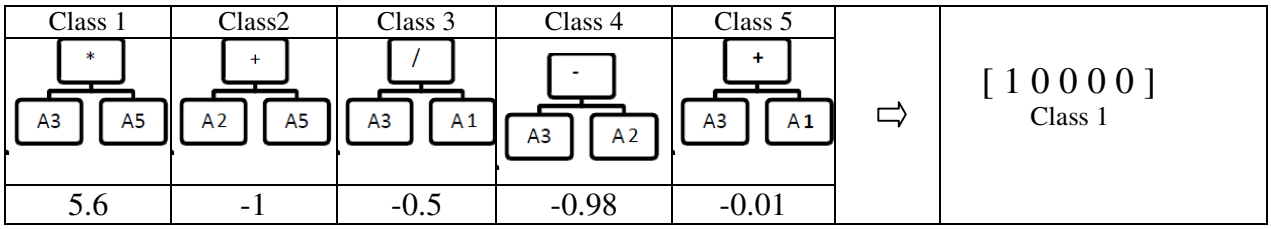

Figure 1: A chromosome for 5 class classification problem and its output for an arbitrary instance

is positive the chromosome cannot decide between the class labels and that condition is named 'don't know' condition.

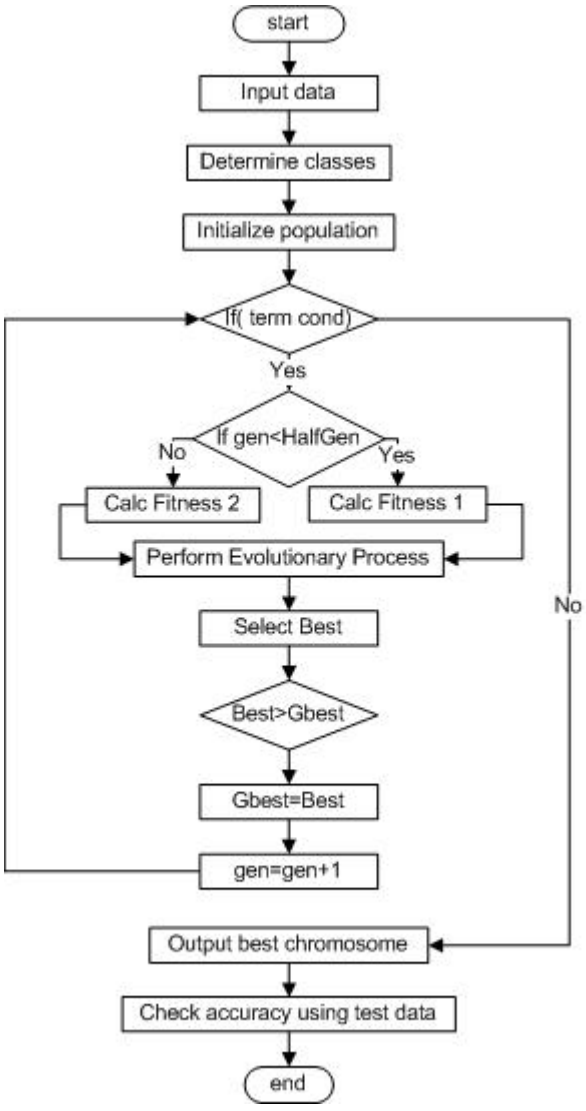

Figure 2 Classification Algorithm

B. Individual tree fitness

We have Introduced a new criteria of fitness of each tree present in the amalgamated classifier. We calculate the classification accuracy of each tree and assign it as the fitness of that tree. The trees are selected for mutation or crossover operator based on inverse of this fitness.

C. Fitness Function

Different fitness functions were used during incremental learning and regular learning.

1. During incremental learning (Fitness 1)

This fitness function averages the number of correct classifications of all trees for a given sample and sums them for all training samples.

2. After stepwise learning (Fitness 2)

This function emphasizes only correctly classified sample, in which whole chromosome output a correct response. i.e one tree should output a value greater than or equal to zero and all others should output a value less than zero. Overall correct classifications are summed and divided by total number of training samples

D. Smaller Tree Elitism

We have introduced a new notion of tree elitism, where best tree for each class is selected while fitness calculation in each generation. If two trees have same fitness , the tree of smaller size is preferred. One the trees for each class is selected these trees are combined to form a new chromosome and made a part of new generation. This process is similar to elitism but instead of making a fitter chromosome part of new generation, we create a fitter chromosome from fitter individual trees and pass it to the next generation.

IV. RESULTS

A. Experimental Setup

To test the classification accuracy of proposed algorithm,

Table 1 GP Parameters used for Classification

GP parameters	
Population	600
Crossover Rate	0.75
Mutation rate	0.25
Reproduction Rate	0.25
Selection for cross over	Tournament selection with size 7
Selection for mutation	Random
Selection for reproduction	Fitness Proportionate selection
Mutation type	Point Mutation
Initialization method	Ramped half and half method with initial depth 6

we have taken the datasets from UCI repository. These data sets are Iris, Bupa, Wine, Glass and Wisconsin breast cancer. These datasets are numerical data sets with different number of classes and attributes. This is done to prove the robustness of the proposed algorithm..The Table 1 lists the GP parameters used for the experimentation. All the parameter used for GP classification algorithm have been kept same as Kishore et all [11]. The datasets used for experimentation are mentioned in Table 2.

The results for the proposed modified multi-tree algorithm are presented in the following table. We can observe that the proposed modification has yielded better results for all the data sets.

Table 2 DataSets

Dataset	Attributes	Instances
IRIS	4	150
WDBC	13	699
HABER	3	306
WINE	13	178
BUPA	6	345
HABER	3	306

This is aligned with the fact that each individual classifier has a strong impact on the final amalgamated classifier's accuracy.

Table 3 Comparison of proposed modification

Dataset	Muni et all [6]	Proposed
IRIS	98	98.1
WDBC	80.6	81.2
HABER	49.2	52.1
WINE	74.5	76.3
BUPA	56.7	57.7
HABER	Muni et al	Proposed

V. CONCLUSIONS

In this paper we have proposed a modified multi-tree evolution using Genetic Programming. The proposed modification emphasizes the fitter building blocks for the eventual amalgamated classifier. This enhances the performance or classification accuracy of classifier. The proposed modification has yielded better results over five UCI ML data sets, proving its efficiency.

The future works includes investigation of better fitness function and some mechanism to reduce the complexity of classifiers during evolution to enhance their efficiency in terms of classification accuracy.

VI. REFERENCES

[1] R Poli, W B Langdon, and N F McPhee, *A Field Guide to Genetic Programming*., 2008.

[2] J R Koza, "Concept formation and decision tree induction using the genetic programming paradigm," in *Parallel Problem Solving from Nature,Proceeding of first workshop,Lecture Notes in Computer Science*, 1991.

[3] A P Engelbrecht, L Schoeman, and S Rouwhorst, "A Building Block Approach to Genetic Programming for Rule Discovery, in Data Mining: A Heuristic Approach," in *Data Mining*.: Idea Group Publishing, pp. 175-189.

[4] A A Freitas, "A Genetic Programming Framework For Two Data Mining Tasks : Classification And Generalized Rule Induction," in *Genetic Programming*, CA , USA, 1997, pp. 96-101.

[5] W Smart and M Zhang, "Multiclass Object Classification using Genetic Programming," *Lecture notes in computer science*, pp. 367–376, 2004.

[6] D P Muni, N R Pal, and J Das, "A Novel Approach To Design Classifiers Using GP," *IEEE Transactions of Evolutionary Computation*, 2004.

[7] J R Koza, "Genetic Programming: On the Programming of Computers by Means of Natural Selection," in *MA*, Cambridge, 1992.

[8] G Folino, C Pizzuti, and G Spezzano, "Parallel Genetic Programming for Decision Tree Induction," in *Proceedings of the 13th International Conference on Tools with Artificial Intelligence*, Dallas, TX USA, 2001.

[9] A Tsakonas, "A Comparison of Classification Accuracy of Four Genetic Programming-Evolved Intelligent Structures," *Information Sciences*, pp. 691-724, 2006.

[10] A A Freitas, "A Genetic Programming Framework for Two Data Mining Tasks : Classification and Generalized Rule Induction," in *Genetic Programming*, CA , USA, 1997, pp. 96-101.

[11] J K Kishore, L M Patnaik, A Mani, and V K Agrawal, "Application of Genetic Programming for Multicategory Pattern Classification," *IEEE transactions on Eolutionary Computation*, 2000.

[12] T Loveard, "Genetic Programming Methods for Classification Problems," Department of Computer Science, RMI, PhD Thesis 2003.

[13] T Loveard and V Ciesielski, "Employing nominal attributes in classification using genetic programming," in *4th Aisa pacific conference on simulated evolution and learning*, singapore, 2002, pp. 487-491.

[14] T Loveard and V Ciesielski, "Representing Classification Problems in Genetic Programming," in *IEEE Congress on Evolutionary Computation*, 2001, pp. 1070-1077.

[15] J Eggermont, "Evolving Fuzzy Decision Trees for Data Classification," in *Proceedings of the 14th Belgium Netherlands Artificial Intelligence Conference*, 2002.

[16] M Zhang and V Ciesielski, "Genetic Programming For Multiple Class object Detection," in *Proceedings of the 12th Australian Joint Conference on Artificial Intelligence*, Australia, 1999, pp. 180–192.

[17] C C Bojarczuk, H S Lopes, and A A Freitas, "Genetic Programming for Knowledge Discovery in Chest-Pain Diagnosis," *IEEE Engineering in Medicine and Biology Magazine*, pp. 38-44, 2000.

[18] M Zhang and W Smart, "Genetic Programming with Gradient Descent Search for Multiclass Object Classification," in *7th European Conference on Genetic Programming, EuroGP*, 2004, pp. 399-408.