Multi-class Classification using BFS Crossover in Genetic Programming

Aadil Hussain, Gaurav Sharma

Abstract— Multi-class classification is a kind of classification task which involves processing an input object and then assigning this object to one of the more than two possible classes.Crossover operation is considered to be a primary genetic operator for modifying the program structures in Genetic Programming (GP). Genetic Programming is a random process, and it does not guarantee results. Randomness is a problem that occurs during crossover operation. During the standard breeding process in Genetic Programming, crossover operation produces offspring with less than half of the fitness of their parent. Thus, it reaches the state where performance stops increasing a certain point, which ultimately leads to unsatisfactory performance of the GP. In this paper, we are proposing a special type of crossover operation named as BFS (Best First Search) crossover to improve overall performance of crossover operation. The proposed method is categorized as best first search; this ensures that it finds the optimal and complete solutions. To demonstrate our approach, we have designed a multi-class classifier using GP and tested it on various benchmark datasets. The results attained show that by applying BFS crossover together with point mutation refined the performance of classifier.

Index Terms— Best First Search, BFS Crossover, Double Tornament, Elitism, Genetic Programming, Multi-class Classifier, Point Mutation.

1 INTRODUCTION

A computational technique that automatically solves problems without requiring the user to know or specify the form or structure of the solution in advance is known as Genetic Programming (GP) [1]. GP begins with a high-level statement of what needs to be done and automatically creates a computer program to solve the problem. In Genetic Programming, we develop a population of computer programs. It means, GP stochastically transforms populations of programs into new, expectantly better, populations of programs as generation by generation.

Genetic Programming [2] is a relatively recent artificial intelligence technique which has been used for a range of tasks, including classification problems. It also used for multi-class classification purposes. Classification tasks (problems) are tasks which involve classifying some example as one of a number of classes. Each example is typically represented by a set of features (a feature vector). Finding an algorithm which can map a feature vector to a class requires the use of machine learning and search techniques. Multi-class or multi-label classification is the special case within the statistical classification of assigning one of several class labels to an object.

Genetic Programming begins with a high-level statement of the demands of a problem and tryout to generate a computer program that resolves the difficulty. The most habitual methods for the selection of the individuals in GP are "tournament selection". But in this work we have used a significant type of tournament known as Double Tournament. In a double tournament method, first we randomly choose 9 individuals from the initial population and then from those individuals we choose 5 on the basis of size and from those 5 individuals we choose 2 on the basis of fitness.

With the standard breeding process, exploring new states in the neighborhood search space of current states can be viewed as a set of random walks. If parents are randomly selected for mating, the GP algorithm will effectively act like a random (beam) search algorithm. Thus, the most crossover events in the standard breeding process produce offspring with less than half of the fitness of their parents.

An alternative, simpler, and domain independent approach to overcome this problem is to integrate the best first search technique into the breeding process to search for good offspring. This can achieve the same effect as constructive crossover operators, though at the cost of a possibly expensive local search. A path finding algorithm that makes the use of heuristics to attain its goal is called best first search. This method generates all the neighbors of a solution, selects the best one, generates all its neighbors, etc. When no better solution is found in a neighborhood, the search returns to the previous neighborhood and selects the second best, and continues until no improvements can be found. These methods are too called local search methods as they only agree with improving solutions that are in the local neighborhood of the current solution. Best first search can either be done explicitly, when the direction of the best step is determined, or implicitly, through the impact of stochastic learning processes.

2 RELATED WORK

Tackett [3] presented brood recombination operator. The brood recombination operator arbitrarily applies crossover N times to two chosen parents to make 2N offspring. After evaluating all offspring, it puts the best two into the next generation and the rejects rest of the offspring. This operator can be categorized as a partial local search operator because it searches for the best state in available states but only searches at 2N possible successor states.

- *Aadil Hussain is currently pursuing Master of Engineering in Computer Science & Engineering in RGPV, India, PH-+91-9770977123. E-mail: aadil_hussain77@yahoo.com*
- *Gaurav Sharma is currently Assistant professor in Computer Science & Engineering in RGPV, India. E-mail: er.gaurav622@gmail.com*

Purohit et al. [4] introduce a new type of crossover known as FEDS (Fitness, Elitism, Depth limit & Size) crossover. In which they select the best two parent individuals from a double tournament method for the crossover operation. From first parent, a subtree is selected and placed at two different positions in the second parent to generate two children. Similarly, from second parent a subtree is selected and placed in the first parent. In this way four individuals are generated from two parents and calculate the fitness, elitism, depth limit and size of the generated children. Two children which have the best result are transferred to the next generation and if the children does not have the better fitness than the parent/child will be retained to the next generation with a 0.5 probability. The FEDS crossover operator can be categorized as a partial local search operator because it looks for the best state in available states but only looks at four possible successor states.

Lang [5] presented another type of crossover operation is called a headless chicken crossover operator. This crossover operation is applied to a selected program P and a new randomly produced program R. This operator continuously generates offspring from P by replacing a sub-tree of P with a replaced sub-tree from R until it finds an offspring with greater or equal fitness to P's. This crossover operator can be classified as a hill-climbing local search. Hill climbing search is only a partial local search because it randomly searches for a state better than or equal to the current state and stops once it finds such a state rather than searching at all possible successors.

A context-aware crossover operator is presented by Majeed and Ryan [6], that determines all possible contexts in one parent for a randomly-chosen sub-tree from the other parent, then evaluates each of them. The context that produces a program with the highest fitness is used and, then the program produced is transferred into the next generation. Different selection methods are used to select parents in different Fitness problems. The authors claimed that the context-aware crossover operator reduces code growth in most of their experiments, improves the fitness of programs, and produces significantly smaller individuals in most cases. This operator can be also categorized as a partial local search operator.

Bleuler et al. [7] introduced a variant of the tournament selection method known as Double Tournament method. The double tournament method is similar to a multi objective approach to bloat, however the objectives of fitness and size are treated separately. Hence, there are two tournaments: one based on parsimony, that generates an initial set of winners, and a subsequent tournament which chooses a subset of those winners based on fitness.

Thomas Helmuth et al. [8], presented another variant of the tournament selection method known as size-based tournament. Here nodes are selected based on a tournament, from that the largest participating sub-tree is selected. Size-based tournaments distinguish between inner nodes of different sizes, whereas Koza 90/10 treats all internal nodes evenly. Thus, the size-based tournament improves performance with no increases in code growth as compared to Koza 90/10 and unbiased selection methods.

The remainder of this paper is structured as follows: Section III describes the proposed work. Section IV presents and analyzes experimental results. Section V draws conclusions.

3 PROPOSED WORK

We have designed a multi-class classifier using BFS crossover and point mutation technique to demonstrate our methodology.

3.1 Initialization

In GP, firstly an initial population has been generated for performing operations on it. The important components for generating initial population are terminal set and function set. The terminal set and function set are the alphabet of the programs to be made. The terminal set contains the variables and constants of the programs. The functions are numerous mathematical functions, such as addition, subtraction, division, multiplication and additional more complex functions. Each tree of the population is generated randomly using the function set F that contains arithmetic functions and the terminal set T containing feature variables and constants. The function set F and terminal set T used are as follows:

F = {+, -, *, %} and

T = {feature variables, R}.

Where R contains randomly generated constant from [0.0 to 10.0].

The three usual tree-generation algorithms are Grow, Full, and Ramped Half-and-Half. In this paper, individual's trees are initially generated by the Ramped Half-and-Half algorithm.

3.2 Fitness Assessment

Once the initial random population has been produced, the individuals require to be evaluated for their fitness. This is a problem definite issue that has to answer the question "how good (or bad) is this individual?" The fitness assess is the prime method for communicating the high-level statement of the problem's requirements to the Genetic Programming system. If initial population generation views as defining the search space for the problem, then view the fitness assessment step as specifying the search's desired direction. The fitness assess is the means of comparing that one individual candidate is better than another.

In the multi-class classification, the fitness function evaluates how well it performs the classification task.

$$\text{Fitness} = \frac{\text{Number of samples correctly classified}}{\text{Total number of samples}}$$

3.3 Selection

In this paper, we select the parent for BFS crossover on the basis of double tournament selection method. Randomly parent is selected for point mutation and roulette wheel selection method is used for reproduction.

3.4 Algorithm for Double Tournamnet

1) First we randomly select 9 individuals from the population.

2) Now on the basis of the size we eliminate 4 individuals from 9 and the individuals with smaller size are selected.

3) Finally from these 5 individuals we select 2 individual with higher fitness.

3.5 BFS Crossover

In BFS crossover we applied a best first search to improve the overall crossover operation. Firstly 9 individuals are randomly selected from the population for the double tournament. In double tournament, from those 9 individuals we select 5 on the basis of size and from those 5 individuals we select 2 on the basis of fitness. The best two parent individuals of the tournament are chosen for the BFS crossover operation. In BFS crossover operation, we generate all the possible children from the parent and check the fitness of all the generated children. Two children with the highest fitness are chosen. Now we apply the elitism, fitness of the chosen children is compared with the parent. If the fitness of the chosen children is better than the parent, children are transferred to the next generation. If we don't find the children better than the parent, then we transfer the parent to the next generation. This method is better than previous methods in finding a solution, because this method has removed the randomness from the crossover operation. There is no need to find the crossover point for performing the crossover. We are generating all the possible combination and finding the best offspring among them. This ensures that, we transfer the optimal offspring to the next generation in every stage.

3.6 Algorithm for BFS Crossover

1) Randomly select individuals from the population for double tournament selection.

2) Select the best two individuals of the double tournament for the BFS crossover operation and call them P_1 and P_2.

3) Swap the sub trees of P_1 and P_2 at different positions and generate all possible children say Ch_1, Ch_2,...., Ch_n.

4) Check the fitness, select the best two children from Ch_1, Ch_2,...., Ch_n on the basis of fitness, the children with the highest fitness are chosen(Ch_1 and Ch_3, say).

5) Now we apply elitism

If ft (P_1 and P_2) < ft (Ch_1 and Ch_3) then
transfer Ch_1 and Ch_3 to the next generation.
else
transfer P_1 and P_2 to the next generation.

3.7 Point Mutation

In point mutation technique, replace the randomly selected function/terminal node of the parent with some randomly generated function/terminal in order to provide some diversity among the individuals. Point mutation also does not cause the problem of bloat because the overall size of the generated children will remain the same as that of the parent.

3.8 Algorithm for Point Mutation

1) Select an individual randomly from the population for the mutation operation.

2) Randomly select the function/terminal node of the parent to replace.

3) Replace the selected function/terminal node of the parent with some randomly generated function/terminal.

3.9 GP Algorithm with BFS Crossover

1) GP begins with a randomly generated population of solutions of size N.

2) A fitness value is assigned to each solution of the population.

3) A genetic operator is selected probabilistically.

4) If it is the reproduction operator, then an individual is selected (we use roulette wheel selection method) from the current population and it is copied into the new population. Reproduction replicates the principle of natural selection and survival of the fittest.

5) If it is the crossover operator, then we apply the BFS Crossover.

6) If the selected operator is mutation, we apply a point mutation.

7) Continue 3. Until the new population get solutions.

8) This completes one generation.

9) Step 3 to 7are repeated till a desired solution is achieved. Otherwise, terminate the GP operation after a predefined number of generations.

4 Experimental Results

In this section, proposed GP based multi-class classifier is tested. We have used Java 6.0 as a front end tool to implement above proposed method. We have used three benchmark data sets IRIS, WBC and BUPA for testing proposed method. Table 1 gives a brief description about all the datasets used. All the datasets are taken from the UCI (UC Irvine) Machine Learning repository [9].

4.1 Datasets

Three datasets are described in this section.

1) **IRIS Dataset:** This is the well-known Anderson's Iris dataset. It contains a set of 150 measurements in four dimensions taken of Iris flowers of three different species or classes. The classes are Iris Setosa, Iris Versicolour and Iris Virginica. The four features are sepal length, sepal width, petal length, and petal width. The

data set contains 50 instances of each of the three classes.

2) **Wisconsin Breast Cancer (WBC):** This data set has 683 data points distributed in two classes. Each data point is represented by nine attributes. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe the characteristics of the cell nuclei present in the image.

3) **BUPA Liver Disorders (BUPA):** It consists of 345 data points in six dimensions distributed into two classes on liver disorders. The first 5 variables are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption.

TABLE 1
DATASETS

Name of Dataset	Number of Classes	Number of features	Size of datasets and class wise distribution
IRIS	3	4	150(=50+50+50)
WBC	2	9	683(=444+239)
BUPA	2	6	345(=145+200)

4.2 Parameters

In this approach, we used the tree structure to represent genetic programs [1]. The ramped half-and-half method [1] used in generating programs in the initial population. The parameter values used in the GP system for the three data sets are shown in Table 2. Evolution is terminated at maximum generation unless a successful solution is found, in which case the evolution is terminated early.

TABLE 2
COMMON PARAMETERS FOR ALL DATASETS

Parameters	Values
Probability of crossover operation	85%
Probability of reproduction operation	5%
Probability of mutation operation	10%
Population Size	500 – 1000
Minimum Tree Depth	2
Maximum Tree Depth	6
Number of Generation	10 – 50
Double Tournament Size X , Y	9 , 5
Non – Terminal Probability	90%

4.3 Results

Iris, WBC and BUPA datasets are tested using 10-fold-cross-validation method for the proposed method. We have compared the outcome of BFS crossover method with the FEDS crossover method [4] shown in Table 3.

TABLE 3
COMPARISON OF FEDS CROSSOVER WITH BFS CROSSOVER METHOD

Datasets	FEDS Crossover Method		BFS Crossover Method	
	Training Accuracy	Generalization Accuracy	Training Accuracy	Generalization Accuracy
IRIS	92.12%	90.59%	96.33%	98.0%
WBC	85.66%	83.64%	95.375%	96.4883%
BUPA	71.78%	66.89%	70.0%	79.09091%

Figure 1 shows the comparison of FEDS crossover with the BFS crossover operation.

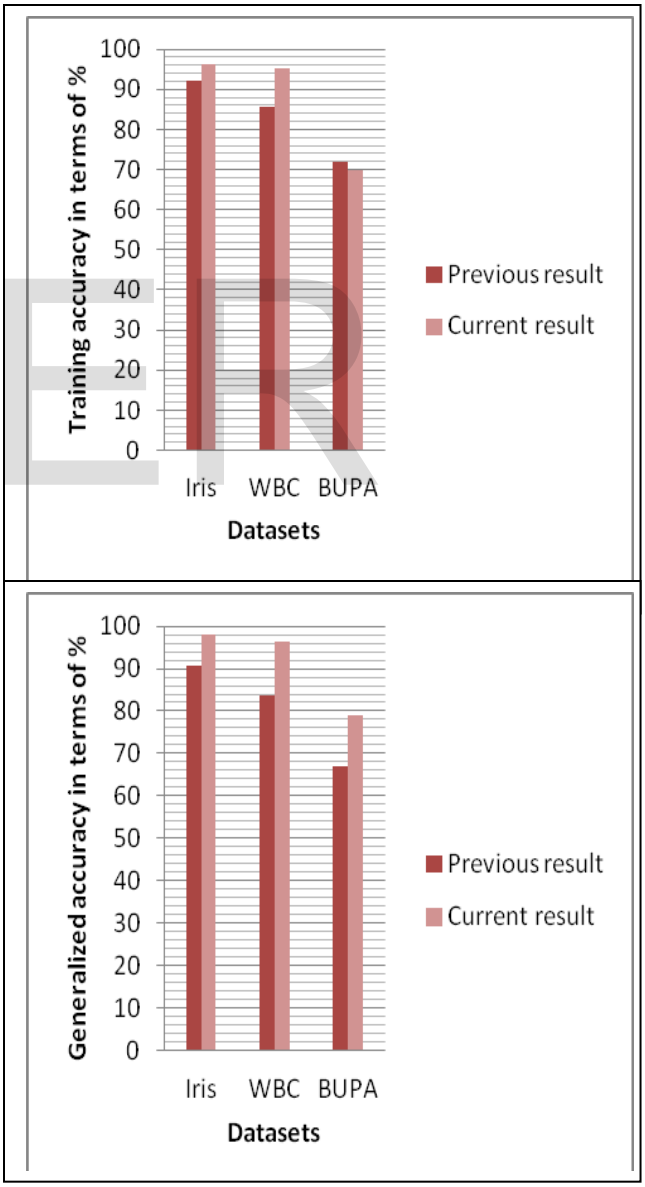

Fig. 1. Comparison of FEDS crossover with BFS crossover operation.

It is found that BFS crossover method outperforms FEDS crossover method. BFS crossover method improves the accuracy of the classifier with a fair amount. The conventional crossover and mutation technique can be destructive. Rather than generating a good tree with better fitness, the conventional methods can produce the trees with lesser fitness than parents.

In this proposed method we apply the elitism technique which will compare the parent fitness with the child and if the fitness of the parent is better than the child then we transfer the parent to the next generation.

5 Conclusion

In this paper, we proposed a BFS crossover operator to increase the performance of GP. In BFS crossover, we have used best first search technique to find the best individuals among all possible generated children. Double tournament has used to provide selection pressure in choosing parent for BFS crossover. We have also used point mutation to provide diversity among individuals. The goal was successfully achieved by developing a new crossover operator with best first search. To describe usefulness of our approach, we have examined and compared BFS crossover method with FEDS crossover method on three different benchmark datasets.

References

[1] Riccardo Poli, W. William B. Langdon, Nicholas F. McPhee 2008. "A Field Guide to Genetic Programming". ISBN 978-1-4092-0073-4. March-2008.

[2] John R. Koza. "Genetic Programming". MIT Press, Campridge, Massachusetts, ISBN 0-262-1170-5, 1992.

[3] W. A. Tackett. "Recombination, selection, and the genetic construction of computer programs". PhD thesis, University of Southern California, Los Angeles,CA, USA, 1994.

[4] Anuradha Purohit, Arpit Bhardwaj, Aruna Tiwari, Narendra S. Choudhari. "Removing code bloating in crossover operation in genetic programming". IEEE-International Conference on Recent Trends in Information Technology, ICRTIT 2011 978-1-4577-0590-8/11/$26.00 ©2011 IEEE MIT, Anna University, Chennai. June 3-5, 2011.

[5] K. J. Lang. "Hill climbing beats genetic search on a boolean circuit synthesis of Koza's". In Proceedings of the Twelfth International Conference on Machine Learning. Morgan Kaufmann. 1995.

[6] H. Majeed and C. Ryan. "A less destructive, context-aware crossover operator for GP". In P. Collet and et al, editors, Proceedings of EuroGP 2006, volume 3905 of LNCS, pages 36–48. Springer-Verlag, 2006.

[7] S. Bleuler, M. Brack,L. Thiele, and E.Zitzler, "Multiobjective Genetic Programming : Reducing bloat using SPEA2," in Proc. 2001 Congr .Evol. Comput. (CEC '01), Piscataway, NJ: IEEE Press, pp. 536-543. 2007.

[8] Thomas Helmuth, Lee Spector and Brian Martin, "Size-Based Tournaments for Node Selection": GECCO'11, Dublin, Ireland. July 12–16, 2011.

[9] Asuncion, A & Newman, D.J. "UCI Learning Repository", CA: University of California, Department of Information and Computer Science. 2007.