Euclidean Space Data Projection Classifier with Cartesian Genetic Programming (CGP)

WK Wong[1], Gopal Lenin[1], Terence Tan[1], Chekima Ali[2]
[1]*Curtin University, Miri, Malaysia.*
[2]*Universiti Malaysia Sabah (UMS), Malaysia.*
Weikitt.w@curtin.edu.my

***Abstract*—Most evolutionary based classifiers are built based on generated rules sets that categorize the data into respective classes. This research work is a preliminary work which proposes an evolutionary-based classifier using a simplified Cartesian Genetic Programming (CGP) evolutionary algorithm. Instead on using evolutionary generated rule sets, the CGP generates i) a reference coordinate ii) projection functions to project data into a new 3 Dimensional Euclidean space. Subsequently, a distance boundary function of the new projected data to the reference coordinates is applied to classify the data into their respective classes. The evolutionary algorithm is based on a simplified CGP Algorithm using a 1+4 evolutionary strategy. The data projection functions were evolved using CGP for 1000 generations before stopping to extract the best functions. The Classifier was tested using three PROBEN 1 benchmarking datasets which are the PIMA Indians diabetes dataset, Heart Disease dataset and Wisconsin Breast Cancer (WBC) Dataset based on 10 fold cross validation dataset partitioning. Testing results showed that data projection function generated competitive results classification rates: Cancer dataset (97.71%), PIMA Indians dataset (77.92%) and heart disease (85.86%).**

***Index Terms*—Cartesian Genetic Programming (CGP); Evolutionary-based Classifier; Clustering.**

I. INTRODUCTION

Evolutionary-based classifiers have been applied in various methods. Although Artificial Neural Network (ANN) and Support Vector Machines (SVM) have dominated the classification algorithm research, the full potential of evolutionary classifier can be further explored. Most evolutionary classifier uses a set of Bayesian rules to classify into either of the classes. The advantage of rule-based classifier is that it is easily readable and transferable to other platform as it is not as complex compared to 'black box' classifier such as Back propagation Neural Network. Rule-based classifier could (but not necessarily) be lighter in terms of computation power consumption as compared to SVM or Neural Network although training using Evolutionary Strategy (ES) would take more time to generate the rule functions. The rule-based classifier can also be highly complex equation but also as simple a function with two operands dependant on the data set. Thus, this makes it ideal for application in embedded systems power due to portability, readability of classifier rules and possibly lower computational cost.

An ideal classifier algorithm would also be required to have feature pruning capabilities to eliminate noisy and non-discriminating features. This may be a universal advantage for most evolutionary rule-based classifier as ES generates the functions based on selected features. The objective of this paper is to introduce and evaluate a new type of feature creation and projection mechanism for binary classification using Cartesian Genetic Programming (CGP) by projecting the feature data into new Euclidian space and classify based on Euclidean distance from selected point.

This paper will be organized as follows: Section 2 of this paper will discuss some of the research work relevant to this research. Section 3 will briefly explain the algorithms and parameters setting. The results of training and testing on benchmarking datasets are shown and discussed in Section 4. Lastly, the conclusion and future work are discussed in Section 5.

II. LITERATURE REVIEW

In this section, several related works and the fundamental concepts of these methods will be briefly discussed. CGP is a form of Genetic Programming or evolution based generation of computer programs/ functions which uses a directed graph to represent a program/function. Each node in the program represents a module of the program/function. CGP was initially proposed to be a solution finding algorithm for electronic circuit in [1] but was later expanded to other problem [2]. Variations of CGP such as the SMCGP [3] and Embedded CGP [4] were later proposed showing faster solution finding on benchmark problems. Due to the capability of CGP to be able to generate complex function, it has been applied to generate Artificial Neural Networks (ANN) [5] and other classification functions using generated rule sets. CGP may be used to generate complex functions by mutating the chromosome and therefore suitable to be applied in generating complex classification rules. Unlike conventional genetic algorithm, CGP does not apply crossover functions in as they were shown not contributing to solution finding. Hence, only mutation function was applied, most commonly a 1+ n mutation strategy as discussed in [1].

Various Feature dimension reduction algorithms have been proposed in various research works. Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Multidimensional Scaling (MDS) are several methods that are often applied for feature dimension reduction. MDS is a type of non-linear data projection in which the distance of the original high dimension feature space is preserved. PCA is a linear projection method that finds orthogonal combination of input feature space which accounts for most variation in the data.

These feature reduction methods are usually combined with clustering algorithms or Bayesian algorithms that classifies the projected data sets in either of the class. The

research work in [6] applied MDS with genetic algorithm for projection onto a lower dimension data. For classification purposes, [7] applied MDS with SVM as the classification algorithm. The results yielded feasible solutions using benchmark datasets.

A similar concept of using a spherical method to bound data was developed in [8] as Support vector Data Descriptor (SVDD). The method is known to be a single class training set, i.e. only the target set is required for training. This is highly useful for dataset with highly unbalanced dataset or only the target class is available. The algorithm defers from SVM where a plane is used to separate the target and outlier class. In SVDD, a spherical boundary is used to classify the classes of data.

III. METHODOLOGY

Three datasets are used as benchmark test for the evolutionary classifier which is the cancer datasets, heart disease dataset, and Pima Indian Diabetes dataset. These benchmark datasets are usually used to gauge performance of classifiers specifically neural networks. The details can be found in [20]. The datasets are tested on 10 fold cross validation which means 90% of the data are applied for training procedure and 10% is reserved for testing. The details of the classifier are shown in Table 1. As shown, the dataset contains missing values. The missing values are replaced with 0. Various research works have excluded the missing value instances from training and testing. However, for this research work, all the instances will be included to test the robustness of the algorithm with regards to noisy data. The values are normalized using min-max normalization where values are normalized to [0, 1]

Table 1
Details of benchmark test sets

Dataset	No. of Features	Instances	Missing values
Cancer	9	699	Yes
Diabetes	8	768	Yes
Heart Disease	35	920	Yes

Similar to conventional CGP, the algorithm does not include crossover and only involves the 1+4 mutation strategy. The equations for each dimension were generated based on 100 nodes where each node represents two inputs and a function operator. The key parameters are shown in Table 2.

Table 2
Key parameters in CGP

Parameters	Value
Maximum generations	1,000
Crossover	No
Mutation	1+4 Strategy
Chromosome value	[1,300]
Chromosome length	1507

The chromosome segment is represented by the chromosome vector, $v \in [1,300]$. For each instances of the data set, a new projected coordinate is generated (x_m,y_m,z_m) are generated by decoding from the chromosome. The cluster centroid point (x_t,y_t,z_t) is generated from the chromosome, v segments as shown in Equation 2 to 5. The first 300 segments of v represent the function and the two input. The subsequent 200 segments of the chromosome vector v are decoded into weightage coefficient that is multiplied to the input. Table 3 show the list of functions and their respective operands where w1 and w2 represents the weightage to the dataset input.

$$x_m = function\ (v_{1,\ldots,}v_{500,}\ f_{1,\ldots}f_n) \quad (1)$$

$$x_t = v_{501} \quad (2)$$

$$y_m = function\ (v_{503,\ldots,}v_{1003,}\ f_{1,\ldots}f_n) \quad (3)$$

$$y_t = v_{1004} \quad (4)$$

$$z_m = function\ (v_{1006,\ldots,}v_{1506,}\ f_{1,\ldots}f_n) \quad (5)$$

$$y_t = v_{1507} \quad (6)$$

$$O_x = v_{502} \quad (7)$$

$$O_y = v_{1005} \quad (8)$$

$$O_z = v_{1507} \quad (9)$$

where f_n represents the normalized dataset values, (x_t,y_t,z_t) represents the cluster centroid, (x_m,y_m,z_m) represents the projected coordinates and $O_x\,O_y\,O_z$ represents the selection of the output from the node chain in the respective dimension.

A node of 100 in CGP represents the length of allowable function equation. Similar to the original CGP proposed in [1], three types of inputs were decoded as operand to the functions as shown in Table 3. The three types of inputs are constants, dataset values and output from previous nodes. The types of inputs are decoded from the chromosome vector values by using look up approach in which certain range corresponds to either of the input types. Figure 1 shows the functions generated from a section of the node chain for a single dimension. $Node_k$ is a section of the equation where the operands are the n^{th} feature value in a dataset instance $\langle f1 \quad \ldots\ldots \quad fn\rangle$. The output of $Node_k$ is fed back to $Node_{k+1}$ as second input after multiplying with the second weightage coefficient, w_2. O_x which is decoded from the original chromosome vector v decides the segments of the node chain as the output.

The chromosome vector integer value is decoded into the function, various nodes output reference using the directed cyclic graph system where the values are continuously subtracted with a threshold until a valid range is obtained.

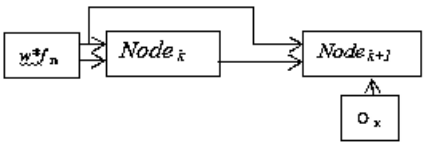

Figure 1: Equation generation from various node chain

The CGP equations project the features into new features spaces. A spherical boundary is introduced from the three dimensional coordinate (x_t,y_t,z_t). If the projected features lie within the boundary (0.5 from the cluster centroid), it will be classified as target. The Euclidean distance from the cluster centroid coordinate (x_t,y_t,z_t) is expressed in Equation 10.

$$d_m = \sqrt{(x_m - x_t)^2 + (y_m - y_t)^2 + (z_m - z_t)^2} \quad (10)$$

where x_m, y_m, z_m, are the generated values from the CGP generated projection functions and x_t, y_t, z_t are the selected reference coordinate. d_m represents the distance of new projected coordinates of instances m in the normalized dataset values.

The outlier/target class is determined by the following, if (d_m>=0.5), then the features will be classified as the target class. Else, the features will be classified as outlier class.
Table 3 shows the functions, function arity and the equivalent function number that may be applied to each node as previously depicted in Figure 1. As shown, 12 function operators were introduced and their respective integer representation. For the unary operation, the second integer (int2) will be ignored. 12 integers represent the operators as shown in the third column of Table 3. For unary functions, only the first input is considered.

Table 3
Function Operator

Function	Arity	Integer representation
(w_1*int1)+(w_{2*}int2)	Binary	1
(w_1*int1)-(w_{2*}int2)	Binary	2
(w_1*int1)*(w_{2*}int2)	Binary	3
(w_1*int1)/(w_{2*}int2)	Binary	4
Exp (w_1*int1)	Unary	5
-(w_1*int1)	Unary	6
(w_1*int1)^2	Unary	7
$\sqrt{(w1 * \text{int1})}$	Unary	8
Cos(w_1*int1)	Unary	9
Sin (w_1*int1)	Unary	10
Min(w_1*int1, w_{2*}int2)	Unary	11
Max(w_1*int1, w_{2*}int2)	Unary	12

w_1 and w_2 are the multipliers to the input (int1 and int2)

The mutation of the CGP is based on a simplified mutation scheme. The CGP algorithm decides which segment of the chromosome vector *v* to be mutated based generation of enabler vector *v* in which *v1* is a randomly generated precision numbers [0,1] of the same length as chromosome vector *v*. The chromosome segment *v* is only replaced with a random integer value [1,300] if the enabler *v1* segment is above a specified range. In all the testing for this research, a value threshold value of 0.95 is determined to ensure minimal mutation. Figure 2 shows the overall pseudocode for the evolutionary algorithm using the 1+4 mutation strategy.

-Start with initial values of 1 for all vector segment *v*
- initialize Previous_best=1
Repeat for 1:1000
-Generate enabler vector v1
-mutate using 1+4 mutation strategy
-for i=1:1507, if (v1 (i)>=0.95, replace all alternative solutions v(i) with rand(1,300).
-Decode and calculate the score value based on CGP
The solution with highest score=current_best
-if (current_best>previous best solution)
Previous_best<-Current_best
end

Figure 2: Pseudocode for the proposed CGP

The fitness score for optimisation is the sensitivity *specificity ratio as shown in Equation 11.

$$Score = \frac{TP}{(TN + TP + FP + FN)} * \frac{TN}{(TN + TP + FP + FN)} \quad (11)$$

where TP is true positive sets, TN is true negative sets, FP is false positive and FN is the false negative sets.

IV. RESULTS

The projection and classification are performed using ten-fold cross validation (90% of the instances for training and 10% of the instances for testing). The algorithm stops running and assumes the best solution after 1000 generations. The CGP is ran for ten times and the average is the average for all the testing /training classification rates. Table 4 shows the classification rate of the algorithm on the three benchmark datasets. It is noteworthy that the partitioning of the testing and training dataset is different in some of the research work and thus a direct comparison might not be applicable. Some research works eliminate instances with the missing value. The average values are from acquired by calculating the mean of the recognition rate from 10 set of optimisation for each data set.

Results on the Benchmark test (Table 5 to 7) showed the testing results were competitive as compared to the other benchmarking results. Among the three benchmarking datasets, the heart disease dataset contained the highest number of missing data but gave the best results as compared to the others. However, the average testing for this dataset is at 80.212% which showed that there is there is high variance among the ten sets of function generations (solution finding).

Table 4
Results of training and testing

Dataset	Best/average training and testing rates (%)			
	Best training	Best testing	Average Training	Average testing
Cancer	96.97	97.71	96.19	94.85
Diabetes	80.36	77.92	73.45	71.07
Heart Disease	77.53	85.86	76.56	80.21

Benchmarking is performed by comparing the results with other algorithms. Table 5 to 7 shows the benchmarking with other research work.

Table 5
Benchmarking of current work with other research work on Wisconsin Breast Cancer dataset

Method	Accuracy %
Radial Basis Function Networks [10]	49.8
Probabilistic Neural Networks [10]	49.8
ANN (Back Propagation) [10]	51.9
Recurrent Neural Network [10]	52.7
Competitive Neural network [10]	74.5
Support Vector Machine [11] [1]	96.9
Memetic Pareto ANN [12]	98.1
ANN (Back Propagation) [12]	98.1
Genetic Programming [13] [2]	98.2
Support Vector Machine [14] [2]	98.4
MT-CGP [9]	99.3
Data projection using CGP(best)	97.71

Table 6
Benchmarking of current work with other research work on Pima Indians Diabetes dataset

Method	Accuracy %
Self-generating Neural Tree (SGNT) [15]	68.6
Learning Vector Quantization (LVQ) [15]	69.3
1-Nearest Neighbor [15]	69.8
Self-generating Prototypes (SGP2) [15]	71.9
Linear Genetic Programming [16]	72.2
k-Nearest Neighbor [15]	72.4
Self-generating Prototypes (SGP1) [15]	72.9
Gaussian mixture models [15]	72.9
Neural Network [16]	75.9
Infix Form Genetic Programming [17] [3]	77.6

Method	Accuracy %
Support Vector Machine [18] [2]	77.6
Principal Curve Classifier [19]	78.2
MT-CGP [9]	79.2
Data projection using CGP (best)	77.92

Table 7
Benchmarking of current work with other research work on Heart disease dataset

Method	Accuracy %
Neural Network [20]	80.3
Linear GP [16]	81.3
Support Vector Machine [18]	83.2
Infix Form GP [17]	84.0
MT-CGP [9]	85.3
Data projection using CGP (best)	85.86

1. Results are based on leave-one-out validation
2. Results are based on ten fold validation
3. This paper does not use the pre-defined training and validation split

V. CONCLUSION AND FUTURE WORK

The proposed algorithm produced feasible results that are comparable with other algorithms as shown in the benchmarking results. More research can be applied to improve the classification by changing the various range of the chromosome. More exploration can be applied to optimise the classification algorithm by applying different chromosome range, initial value and different evolutionary algorithms such as ant colony optimisation, Cuckoo Search and even Gradient Descent.

The research in this work can be extended by projection in multiple dimensions (hyper dimensions). This may contribute to better classification but could also cause a bloating problem (solutions that are overly or unnecessarily complex). Increasing the number of nodes will provide more parameters for tuning in the classification development and can be extended to a probabilistic output rather than just a discrete output. It is noteworthy to highlight that increasing the dimensions of data projection also increases complexity in the solution searching and ultimately it's a trade-off situation to get the optimal number of dimensions.

It is also desirable to explore various classification method apart from the bounding and clustering method. This research work can be extended to using a linear function to separate the classes and may even be extended to multiclass classification.

In conclusion, this preliminary work on this classifier based on CGP data projection showed that the method is worth further exploration based on results in Table 4 to 7. The preliminary results achieved using a simplified CGP without the application of only mutating the active nodes. More complex versions of CGP such as ECGP and SMCGP can also be tested with more datasets.

REFERENCES

[1] J. F. Miller, P. Thomson, and T. C. Fogarty (1997). "Designing Electronic Circuits Using Evolutionary Algorithms, Arithmetic Circuits: A Case Study," In D. Quagliarella, J. Periaux, C. Poloni and G. Winter (eds.), *Genetic Algorithms and Evolution Strategies in Engineering and Computer Science: Recent Advancements and Industrial Applications*, Chapter 6. Wiley.

[2] J. F. Miller (1999). "An Empirical Study of The Efficiency of Learning Boolean Functions using a Cartesian Genetic Programming Approach", In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO99)*, 1135-1142. Morgan Kaufmann.

[3] Harding S. L., Miller J. F. Banzhaf W., "Developments in Cartesian Genetic Programming: Self-modifying CGP", *Genetic Programming and Evolvable Machines*, Vol. 11 (3/4) (2010) pp 397-439.

[4] J. A. Walker and J. F. Miller, "Evolution and Acquisition of Modules in Cartesian Genetic Programming," In *Proc. of the 7th European Conference on Genetic Programming*, Volume 3003 of LNCS, pages 187–197, Coimbra, Portugal, 5-7 Apr. 2004. Springer-Verlag.

[5] Maryam Mahsal Khan, Arbab Masood Ahmad, Gul Muhammad Khan, Julian F. Miller, "Fast Learning Neural Networks Using Cartesian Genetic Programming", *Neurocomputing* 121 (2013) 274–289

[6] Dzidolikaitė, Agnė. "Genetic Algorithms for Multidimensional Scaling." *Science–Future of Lithuania/Mokslas–Lietuvos Ateitis* 7.3 (2015): 275-279.

[7] X. l. Wang, L. y. Zhang, C. w. Dong and X. p. Rui, "A Multi-Dimensional Visualization Method Combining MDS and SVM," *2012 8th International Conference on Natural Computation*, Chongqing, 2012, pp. 436-439

[8] Tax, David MJ, and Robert PW Duin. "Support Vector Data Description." *Machine learning* 54.1 (2004): 45-66.

[9] Harding, Simon, et al. "Mt-cgp: Mixed Type Cartesian Genetic Programming." *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*. ACM, 2012.

[10] R. Janghel, A. Shukla, R. Tiwari, and R. Kala. "Intelligent Decision Support System for Breast Cancer". In Y. Tan, Y. Shi, and K. Tan, editors, *Advances in Swarm Intelligence*, Volume 6146 of Lecture Notes in Computer Science, pages 351–358. Springer Berlin / Heidelberg, 2010.

[11] H. X. Liu, R. S. Zhang, F. Luan, X. J. Yao, M. C. Liu, Z. D. Hu, and B. T. Fan. "Diagnosing Breast Cancer Based On Support Vector Machines," *Journal of Chemical Information and Computer Sciences*, 43(3):900–907, 2003.

[12] H.A. Abbass. "An Evolutionary Artificial Neural Networks Approach for Breast Cancer Diagnosis," *Artificial Intelligence in Medicine*, 25:265–281, 2002.

[13] P.-F. Guo, P. Bhattacharya, and N. Kharma. "Automated Synthesis of Feature Functions for Pattern Detection," In *Electrical and Computer Engineering (CCECE)*, 2010 23rd Canadian Conference on, Page 1 – 4, May 2010.

[14] P. Hao, L. Tsai, and M. Lin. "A New Support Vector Classification Algorithm with Parametric-Margin Model," In *Neural Networks*, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, pages 420–425. IEEE, 2008.

[15] H. A. Fayed, S. Hashem, and A. F. Atiya. "Self-Generating Prototypes for Pattern Classification. Pattern Recognition," Pages 1498–1509, 2007.

[16] M. Brameier and W. Banzhaf. "A Comparison of Linear Genetic Programming and Neural Networks in Medical Data Mining," *Evolutionary Computation, IEEE Transactions on*, 5(1):17 –26, feb 2001.

[17] M. Oltean and C. Grosan. "Solving Classification Problems Using Infix Form Genetic Programming," In Programming, *The 5th International Symposium on Intelligent Data Analysis*, M. Berthold (et al), (Editors), LNCS 2810, Pages 242–252. Springer, 2003.

[18] P. Hao, L. Tsai, and M. Lin. "A New Support Vector Classification Algorithm with Parametric-Margin Model," In *Neural Networks*, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, Pages 420–425. IEEE, 2008.

[19] K. Chang and J. Ghosh. "Principal Curve Classifier-A Nonlinear Approach to Pattern Classification," In *Neural Networks Proceedings*, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on, Volume 1, Pages 695 –700 vol.1, May 1998.

[20] L. Prechelt. PROBEN1, "A Set of Benchmarks and Benchmarking Rules for Neural Network Training Algorithms," *Technical Report 21/94*, Fakult¨at fur ¨ Informatik, Universit¨at Karlsruhe, D-76128 Karlsruhe, Germany, Sep 1994.