Issues in Evolving GP based Classifiers for a Pattern Recognition Task

Ankur M. Teredesai
Dept. of Computer Science
Rochester Institute of Technology
Rochester, NY 14623
Email: amt@cs.rit.edu

Venu Govindaraju
CEDAR
520 Lee Entrance, Suite 202, UB Commons
Amherst, NY 14228
Email: govind@cedar.buffalo.edu

***Abstract*— This paper discusses issues when evolving Genetic Programming (GP) classifiers for a pattern recognition task such as handwritten digit recognition. Developing elegant solutions for handwritten digit classification is a challenging task. Similarly, design and training of classifiers using genetic programming is a relatively new approach in pattern recognition as compared to other traditional techniques. Several strategies for GP training are outlined and the empirical observations are reported. The issues we faced such as training time, a variety of fitness landscapes and accuracy of results are discussed. Care has been taken to test GP using a variety of parameters and on several handwritten digits datasets.**

I. Introduction

Genetic Programming acts as a useful technique for classification tasks as discussed in [1], [2], [3]. Consider the case of a simple binary classifier. Samples from two classes can be labeled as class 0 and class 1. The input to the classifier consists of a two dimensional feature vector. The classification task would be to find an optimal boundary that separates the two classes in a plane formed by the two dimensional feature vectors [4].

Now consider the same question posed in a slightly complex manner. Which of the 4 digits $\{1,4,7,9\}$ does the incoming image match the closest. A traditional classification scheme will output the class and confidence value for the incoming image to belong to each class $\{1..9\}$ and a ranking scheme will decide the result. What is called for here is not simply using the traditional classifier but building a specific classifier that focuses on features most useful in separating the numerals in the subclass set apart from each other. The subclasses in such applications can vary dynamically. Such classifiers are termed as secondary classifiers because they help classify a subset of classes with higher accuracy. Regular first stage classifiers are termed as primary classifiers or level-1 classifiers.

In order to train classifiers to perform primary and secondary classification data sets have to be organized differently. The parameters for the GP run are different in each case. The fitness functions can vary depending on the number of classes. Classification task needs to be broken down in sets of binary dichotomizers. The function sets used in the classifier development can be different. These issues are the focus of this paper.

Recognition of handwritten characters by a computer has been a topic of intense research for many years [5]. Due to its pivotal role in many applications such as postal addresses interpretation, bank checks, tax forms and census forms reading, considerable research has been conducted in the area of handwritten character and numeral recognition [6].

Due to the nature of handwritten characters, there is probably no one single method that can achieve high recognition and reliability rates for all applications. Thus in recent years, efforts have been directed toward more sophisticated considerations [6].

The need for GP-based primary and secondary classifiers can be explained with help of Table I. For every case the primary classifier was incorrect in some of the cases and a secondary classification stage would have helped classify digits more accurately. The numbers here were derived from an isolated digit set (BHA set) [7]. The top choice hit ratio is 0.9809. The (top-2-choices) hit ratio is 0.9913. For example, when discriminating '7' and '2' the features localized at the bottom right corner will be important. The objective is to use a GP based secondary classifier to focus on these important features as part of the solution tree (in tree based GP).

TABLE I
Digit occurs in top choice.

	top 1	**top 2**	**top1 + top2**	**total**
0	1640(0.987)	9(0.005)	1649(0.992)	1662
1	3017(0.994)	15(0.005)	3032(0.999)	3034
2	1998(0.965)	34(0.016)	2032(0.982)	2070
3	774(0.974)	12(0.015)	786(0.989)	795
4	2197(0.981)	31(0.014)	2228(0.995)	2240
5	747(0.975)	8(0.010)	755(0.986)	766
6	615(0.979)	4(0.006)	619(0.986)	628
7	689(0.972)	12(0.017)	701(0.989)	709
8	485(0.984)	3(0.006)	488(0.990)	493
9	480(0.978)	6(0.012)	486(0.990)	491

II. Related Work

Active character recognition [8], [9], employs an active heuristic function that adaptively determines the length of the feature vector as well as the features themselves used to classify an input pattern. The classification stage is dynamic in terms of the level of resolution. Features are hierarchically used to focus on smaller sub-images on each recursive pass till the classification process meets acceptance criteria.

The passive character recognizers use methods described in [10] [5]. These classical methods employ the paradigm *One Model Fits all* [9]. They attribute their multi-resolution in feature space to the gradient, structural and concavity based features extracted. The classifier is a K-Nearest Neighbor or K-NN rule. Gradient captures the local shape of the character, structural features capture the relationships between stroke formation and concavity captures the global manner of character drawing. These recognizers assume a set of features and classification mechanism ad-hoc. They are trained using a particular dataset and are then placed in their operational environment. They are incapable of evolving over the test environment since any change in their performance can only be achieved through re-training. Some systems like adaptive resonance network learn as they perform [11].

Multiple layer perceptron model has the number of neurons, the number of hidden layers, organization of the network, training parameters, etc. selected by an expert [12]. The architecture of the recognizer should be more flexible and self organizing because learning by weight modification in a fixed network topology can limit the classifier to a locally-optimal solution. The complexity of the problem may not be scaled favorably by the complexity of the network selected. The classification standards might be met but there will always be a trade-off with learning time, network size, and such other issues. These issues and the trade-off in using GP have been adequately discussed in [2]. The solution representation in our approach is more simple compared to previously attempted methods.

The use of classifier combination for the development of accurate classifiers is attempted but optimal resource utilization and the fact that multi-resolution features can be used to design efficient combinations is not considered [13] and [14].

The relation between data mining and GP is explored in [15]. Main emphasis is placed on the classification task and optimality of features sets is not explored. Folino et al. provide an interesting cellular GP approach to classification in general but cellular GP has not been tested yet on real-world pattern recognition tasks [16]. The relation between pattern recognition and GP was recently proposed by Kishore et al [17]. Our work extends some of the methods discussed by them and suggests certain improvements in terms of fitness function selection, data set organization and implementation of multi-class classification problems. Our design of new fitness functions is more appropriate in the pattern recognition domain than the current fitness functions usually used as the defaults [1]. The issues discussed in this paper are a significant development compared to the preliminary results we reported [18]. Current fitness functions depend on the concept of high penalty for non-evolved solutions or bad solutions. These fitness functions do not relate very well with the pattern recognition domain because the penalties assigned are not a function of the ability of the solution program to focus on the regions of maximum information pertinent to the application. Data set organization into class specific training sets is a concept new to GP application domain. Many of the problems associated with stochastic learning concepts can be effectively handled using data set reorganization as we discuss later.

III. Issues in Classifier Training

A. Developing a Single Primary Classifier for n classes

Feature extraction problem is defined by Devijiver and Kittler as the problem of "extracting from the raw data the information which is most relevant for classification purposes, in the sense of minimizing the within-class pattern variability while enhancing the between-class pattern variability" [19]. Writing styles, writing equipment, resolution and image quality all play a major role in handwritten characters. To design a good feature set for machine classification with high recognition performance, many aspects should be put in consideration. This often leads to multi-resolution features for different scales of the character images and large size of feature sets being extracted. Classification methods designed for using these features will encounter difficulties in achieving high recognition rate.

Hierarchical restructuring of the image domain using Quin and Quad trees produces multi-resolution features [8], [9]. These features are used as terminals for the GP run. For a given digit image as input the output expected from a classifier is one of the 10 digit classes 0 to 9. This scheme of classifier development does not yield satisfactory results and performs rather poorly. The parameters and values used for this form of classifier development are listed in table II.

TABLE II
Parameters and Values: Evolving a single classifier for the 10 class problem.

Parameters	Values
Objective:	Evolve 0-9 classifi ers
Terminal Set:	Multi-Resolution Features
Function set:	+, -, *, DIV, SIN, COS, LOG, EXP
Population Size:	I:200, II:400, III:600
Experimental Choices For:	
Crossover probability:	80 percent
Mutation probability:	20 percent
Selection:	Tournament
Termination criterion:	$Error < 10\%$
Maximum Generations :	2000
Maximum Depth:	17
Initialization Method:	Half-and-Half
Best individual found (III:600):	65% accurate
Number of Features Used:	159 out of 512 total

Each solution tree in the GP run was setup to provide an output class in terms of the digit truth. So for an incoming image of a digit 2 the expected output was ASCII 2. The confusion index of the classes being very high the classification was not accurate. The number of samples of each class was balanced in the training set and the test set, so as to avoid bias. Failure on this account lead us to conclude that more research is needed where a higher level wrapper to GP might return multiple class values in form of top choice, second choice, etc. along with the confidence values.

B. Classification using N-Class Binary Classifiers

Smaller number of classes of characters or numerals is easier to classify. This is so because the features required may be relatively fewer and the classification decisions are sharper. Then we can decouple the problem of achieving high recognition rate from the problem of achieving high reliability. The large feature set can be utilized to get high recognition rate (high number of hits without considering thresholding). Then the resulting outputs of this first step shrink the target down to a smaller range in which fewer number of classes, for example 2, are to be classified. For each possible group of smaller classes, a subset of features is selected for a secondary classification.

To further classify among only two possible classes for the input image rejected by the first level classification, the simplest way would have been to run the first step classification routine with the original number of features for the reduced target set of two instead of ten classes. This method usually does not work since the features are designed for classification of ten classes. Features that contribute to recognition in case of one class may contribute to confusion in case of another class or a smaller set of classes. This is actually a reason why the confidence value was not high enough and the input was rejected by the first level classification.

The first task is the develop classifiers that give decision for one class. If the input feature vector belongs to class x the output is positive. (The reason we say positive and not +1 will be clear shortly). If the output is negative it signifies that the input feature vector belongs to the class σ, where $\sigma = \{y - x | y = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}\}$. The parameters and values used for this form of classifier development are listed in table III.

TABLE III
PARAMETERS AND VALUES: ONE CLASSIFIER FOR EACH CLASS.

Parameters	Values
Objective:	Evolve single digit classifi er
Terminal Set:	Multi-Resolution Features
Function set:	+, -, *, DIV
Population Size:	I:200, II:400, III:600
Crossover probability:	0.8
Mutation probability:	0.2
Selection:	Tournament selection, size 7
Termination criterion:	Error $< 10\%$
Maximum Generations :	2000
Maximum Depth:	17
Initialization Method:	Half and Half
Best Individual III:600-	96%
Number of Features Used:	115 out of 512 total per digit

The performance achieved using such classifiers was comparable to classical pattern recognition techniques like K-Nearest Neighbor, SVM and Neural Networks. The issues that we faced during training of these classifiers were mainly concerning the digit distribution in training sets. Since one classifier is supposed to handle only one digit, care should be taken that it is sufficiently represented in the training set. The training set used in this case has to be biased in a manner that around 35-50% of samples belong to the class under consideration and the rest of the classes make up the remaining 65-50%. If prior knowledge about confusion between classes is available this training set should be biased in a manner that samples from the confusing class are sufficiently represented. If for example, the digit 0 and digit 8 have a high confusion index, then while training for class 0 the training set should contain sizeable but less number of samples from class 8, so that the classifier can learn to separate between 0 and 8 effectively. A higher level interpretation of this issue can be made in terms of embedding *domain knowledge* about classes in the datasets.

Simple function set consisting of +, -, *, DIV was found most suitable. Other mathematical operands seemed to introduce fast convergence to sub-optimal solutions.

Another observation was made with respect to the expected output from the solution function. Initial attempts to converge proved unsuccessful when the output values were +1 for the digit class x and -1 for the class σ. Experimentation with the values greater than equal to +1 for class x and less than equal to -1 for class σ showed better results. The best separation between classes was accorded by values greater than or equal to +10 for class x and less than or equal to -10 for class σ. Some input feature vectors were classified with output values between +10 to -10 and were placed in the reject class for this experiment. The width of between class boundary seems to play an important role in classifier training. This shows that well separated boundaries play the same role as thresholds play in neural network based classifiers.

C. Building Pairwise Discriminatory Classifiers

For each pair of possible classes, we want to get rid of the redundant features that confuse the recognizer and retain only those features which reveal maximal separability between the two classes [13]. We build recognizers as pairwise discriminatory classifiers with each giving decision for a subclass of only two classes(2 numerals). They will take the inputs rejected by the first classifier and further classify between the first two choices returned.

To build each pairwise discriminatory classifier, GP selects a subset of feature positions – key feature positions in the feature vectors of the two classes. In order to understand the features selected by GP as key features, the distributions of features in the feature vectors are computed for each class. For example, Figure 1 is the distribution of features for numeral 2. For each pair of classes, the differences of the distributions at each feature positions are also computed, figure 2 gives the distributions of the differences for the features of numeral 2 and numeral 7.

This problem of determining the key features can be rendered as an optimization problem in terms of selecting

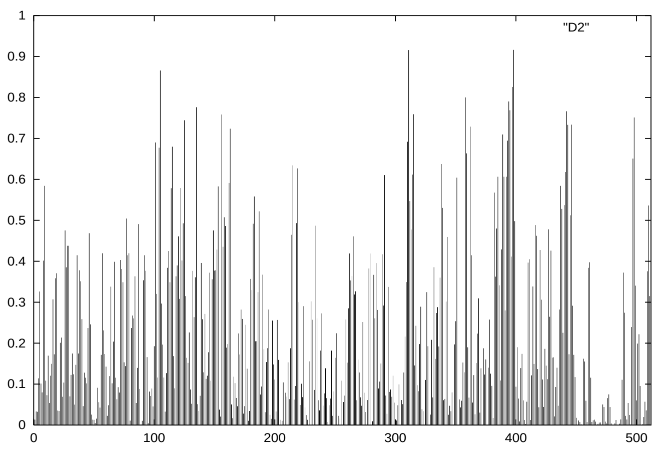

Fig. 1. Feature distributions of numeral 2. The x-axis represents features 1-512 and Y-axis represents the probability distribution over the entire dataset.

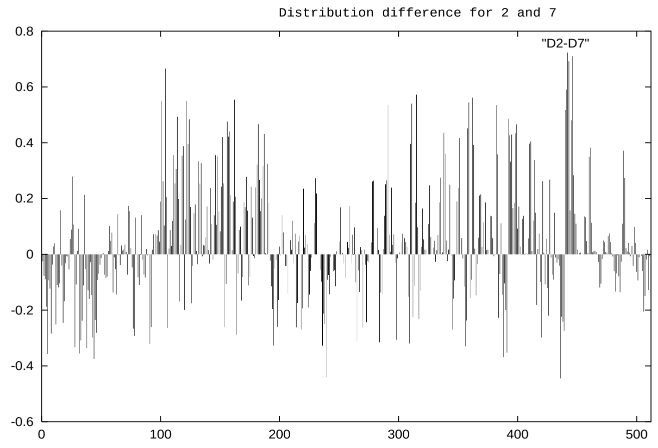

Fig. 2. Differences in Feature distributions of numeral 2 and numeral 7.

those positions in the feature vectors that present optimum separation between classes with respect to their distributions for those classes. Such a problem can be effectively handled with help of GP [1].

D. Fitness Functions

Fitness functions should reflect the overall goal of the GP run. Minimization of the fitness over several iterations should produce good solutions in form of accurate classifiers. In pattern recognition the fitness of an individual should be proportional to the number of patterns it can classify correctly. Such fitness functions should levy high penalty on erroneous solutions and should reward good solutions. There are several fitness equations we suggest.

The first fitness function we suggest is for multi-class classifiers in form of fitness
$F1 = N_b P_b + N_x P_x + P_c$
where
N_b= Number of training set samples on the boundary
N_x= Number of training set samples belonging to class x incorrectly classified.
P_b= Penalty associated with boundary classification.
P_x= Penalty associated with incorrect class classification.
P_c= Penalty associated with non evolving solution criteria being met.

Non-Evolving solution criteria is said to be met if the solution for the multi-class classifier fails to classify at least one sample from each class under consideration.

For single class classifiers the fitness function takes the form
$F2 = N_b P_b + N_x P_x + N_\sigma P_\sigma$
where
N_σ= Number of training set samples belonging to class σ incorrectly classified.
P_σ= Penalty associated with incorrect σ classification.

The third kind of fitness function is derived from F2 but incorporates an adjustment to the raw fitness if the solution does not classify a certain minimum number of samples from the class x. This modification enables the GP run to quickly come out of the local-minima that is reached if the training set is not well balanced as discussed in the first issue.
$F3 = N_b P_b + N_x P_x + N_\sigma P_\sigma + P_j$
where
C_j= The criterion deciding minimum number of samples of class x that must be targeted by every solution in order to maintain potency in the population.
P_j= Penalty associated with not meeting C_j.

The value of C_j must be decided before we begin the run and should usually fall in the range of 1-5% of the number of samples belonging to class x in the training set.

E. Role of Key Features

Given sufficient resources, any classification task is possible with a high accuracy, but to achieve a particular task given finite resources, the problem is to utilize these resources intelligently. Cognitive studies in human vision associate multi-resolution features with high recognition accuracy. We show that classifier development using separability optimization is very similar to emulation of human cognition. The identification of key features leads to optimal resource utilization by the classifier. Evolving such classifiers is one of the issues in this discussion because if GP based classifiers are not optimal in terms of feature space the curse of dimensionality can become rather severe. The resources required for classification can be identified in terms of amount of time required to develop a recognizer, amount of processing power required and the number and kind of features extracted. Our methodology is termed as active based on the premise that once the complexity of a classification task is known an intelligent recognizer should incrementally increase the resources needed for classification. Machine learning based classifiers often lack the ability to discern and strive to utilize complete information available from a digit image for all the classes under consideration. How

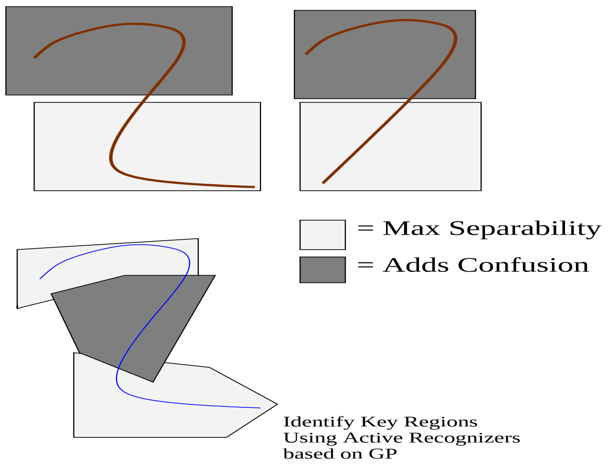

Fig. 3. Identifying zones for maximum separability between digit classes.

does the human eye distinguish effectively between different digits? Is there any relation between the way we extract features visually to the way we have such advanced cognition? Our belief is that in order to develop advanced classifiers two aspects are extremely important.

1) Multi-resolution features
2) Maximal separability

Multi-resolution features accord a multi-level view of the image domain as discussed in [8]. The features extracted in this manner break up the image domain into a hierarchical feature space. It is possible to focus in on various regions of the image in an iterative manner level by level in this feature space. Several features from different levels and regions combine be combined to form many information enriched zones. For example, the central region in multi-resolution space plus the start and end points of the digit image can combine to provide one information enriched zone. If we assume that such zones are the basis of human perception accuracy, then these zones are responsible for maximal separability 3.

Maximal separability is a term we use to quantify the differences between various digits. The images of digits '2' and '7' differ most at the lower left quarter of the image provided the images are placed in a bounding box and are slant corrected. Any classifier that uses this information about these digits would provide good separability. Now let us remove the assumption that the images are bounded by the bounding box and are slant corrected. In this case the problem of identification of regions that provide high separability becomes a difficult problem. Since multi-resolution features are highly tolerant to scaling and transformations, they can be used effectively to provide separability. The process of identification of such feature subsets is attempted by using evolutionary methods like genetic programming [20].

IV. Experiments

We tested our system on various datasets including the NIST handwritten digit sets, and a more noisy CEDAR-Digit data set. The NIST data set consists of 159228 images in the training set, and 53300 images in the test set. The CEDAR-Digit data set is smaller in size with 7245 images in the training set and 2711 in the test set. This set consists of more noisy data with images that were incorrectly recognized or rejected even by the current recognizers based on K-Nearest Neighbor rule and neural networks. A sample of such images is seen in Figure 4.

For a view on such input images as those accepted incorrectly by the secondary classifier and which did not have the truth in the top 2 choices of the primary classifier see Figure 4 and Figure 5.

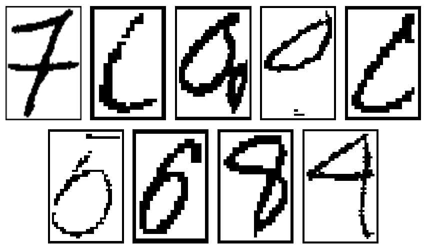

Fig. 4. Images with truth not being the top choice of the primary classifi cation.

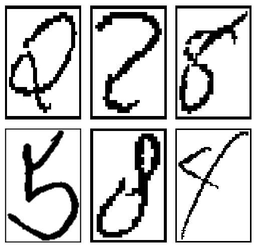

Fig. 5. Images with error: rejected from the primary classifi ers but correctly accepted by the secondary pairwise discriminatory classifi ers.

Let us first present an analysis for a conventional passive classifier like GSC which is based on K-NN rule and has been is use for more than a decade in our systems. When set of 12242 handwritten digit images were used as testing data: 540 images were rejected when a thresholding value of 0.4 is used on the confidence value of top choices (the confidence value ranges from 0 to 0.6 for the GSC). Among the accepted (12242-540) images, 60 are incorrect.

Accept rate = (12242-540)/12242 = 95.6 %

Error rate = 60/(12242-540) = 0.51%

Feature vectors are represented as a two dimensional array where the first index represents the level of the feature and the second index represents the actual feature. For example F17 would indicate that this is a the 7th feature extracted at level 1. Detailed discussion of the Hierarchical feature extraction scheme was presented in [9]. It is sufficient for the current discussion to note that assume that multi-resolution features were extracted from the image and were input to the classifier. The results we obtained using our GP based methodology are presented in Table IV. A comparative performance evaluation with other techniques for handwritten digit classification was

TABLE IV

RESULTS OBTAINED DURING A TYPICAL TEST RUN.

Class	Images	Correct	Incorrect	Mis-fired true	Correct (%)
0	5511	5373	138	20	97.5
1	5822	5665	157	30	97.3
2	5308	5143	165	12	96.9
3	5470	5311	159	17	97.1
4	5118	4949	169	15	96.7
5	4773	4639	134	12	97.2
6	5352	5207	145	22	97.3
7	5516	5317	199	28	96.4
8	5263	5047	216	11	95.9
9	5167	5043	124	34	97.6

recently reported by Teredesai and Govindaraju [21]. Some sample classifier solutions for different digits are also presented below:

TABLE V

A PART OF THE SOLUTION FOR DIGIT 3

generation: 5 nodes : 20 depth : 5 hits : 5328/5470
accuracy : 97.1
Part of the solution tree : (+ (* (cos (- F04 F198))
(+ (cos (sin F02))
(exp F04)))
(* (cos (- F04 F198))
(cos (sin F02))))

Some of the features used in the first solution are **F04, F198, F02** suggesting that this implementation was able to recognize 97.1% of the presented test patterns using only some of 9*31 = 279 total features. Another interesting fact is that the solution is very simple in expression and suggests that sub-image 0 and sub-image 19 accorded maximum separability in this case.

TABLE VI

A PART OF THE SOLUTION FOR DIGIT 5

generation: 17 nodes : 12 depth : 5 hits : 4639/4773
accuracy :97.2
Part of the solution tree : (+ (exp (sin (exp (* F217 F36))))
(exp (exp (* F78 F35))))

The second solution is also relatively simple in expression. One observation from the second solution is that the solution space has a lot of variation in terms of features suggesting that GP tries to focus on the key areas of separability and embeds them in individual solutions.

Consistent convergence towards a good solution can be observed from fig 6. Over the number of generations in a given GP run this figure shows the fitness of individuals reducing as the accuracy of the classifier increases. The error rate decreases in proportion to fitness after the run has stabilized over initial generations. The spike in the error curve during the initial populations is suggestive of the amount of variability accorded by GP to discover accurate classifiers in initial generations.

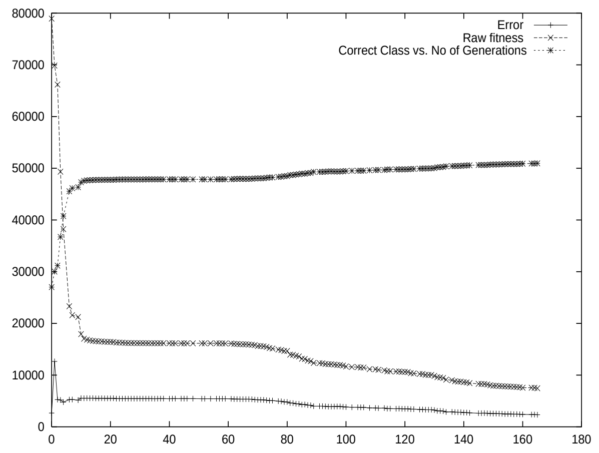

Fig. 6. Plot showing error, fitness and accuracy over number of generations of the GP run.

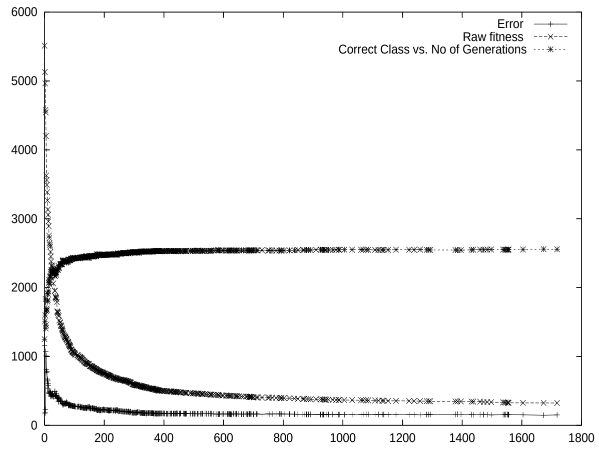

Fig. 7. Plot showing error, fitness and accuracy over number of generations for digit 0 where data set is re-organized.

The plot in fig 7 shows how data set re-organization plays a role in greater accuracy and faster convergence. Over the same parameters in a run, the observed error rate stabilizes much earlier compared to original data set.

V. CONCLUSION

In this paper several issues in classifier training using genetic programming were explored. Some of the key observations made are: a) data set re-organization plays an important part in the GP run, in terms of faster convergence, b) GP based classification using multi-resolution feature vectors is convergent, c) effective use of breaking the n-class classification task in n-binary classifiers (dichotomizers) is the best approach when using GP and d) fitness proportionate classification demands that good fitness functions should be designed in the future to drive the training procedure more effectively. Some possible fitness functions are presented for generic use in such pattern recognition tasks.

Future work on building more optimal pair wise discriminatory classifiers is under consideration. The problems associated with noise in feature vectors and its implication on training issue is also subject of further exploration. A more comprehensive evaluation with other handwritten digit classification techniques is also required.

REFERENCES

[1] J. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[2] J. Koza, F. Bennett, D. Andre, and M. Keane. *Genetic Programming III*. Morgan Kaufmann Publishers, 1999.

[3] K. Kinnear Jr. *Advances in Genetic Programming*. The MIT Press, 1994.

[4] R. Duda, P. Hart, and D. Stork. *Pattern Classifi cation*. John Wiley and Sons Inc., 605 third Avenue, New york, NY 10158, 2 edition, 2001.

[5] C. Suen. Character recognition by computer and applications. In T.Y.Young and K.S. Fu, editors, *Handbook of Patter Recognition and Image Processing*, pages 569–589. Academic Press, 1986.

[6] G. Kim and V. Govindaraju. Bankcheck recognition using cross validation between legal and courtesy amount. Technical report, Automatic Bankcheck Processing, World Scientifi c, 1997.

[7] K. Yamamoto, H. Yamada, and T. Saito. Current state of recognition method for Japanese characters and database for research of handprinted character recognition. In S. Impedovo and J. Simon, editors, *From pixels to features III: Frontiers in Handwriting Recognition*, pages 250–260. Elsevier Science Publishers, 1992.

[8] J. Park and V. Govindaraju. Active character recognition using a^**-like* algorithm. In *Proceedings of Computer Vision and Pattern Recognition*, 2000.

[9] J. Park, V. Govindaraju, and S. Srihari. *OCR* in a hierarchical feature space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):400–407, April 2000.

[10] J. Favata and G. Srikantan. A multiple feature/resolution approach to handprinted digit and character recognition. *International Journal of Imageing Systems and Technology*, 7:304–311, 1996.

[11] G. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. In *Computer Vision, Graphics, and Image Processing*, pages 35–115, 1987.

[12] R. Duda and P. Hart. *Pattern Classifi caiton and Scene analysis*. Wiley International, 1973.

[13] T. Ho, J. Hull, and S. Srihari. Desicion combination in multiple classifi er systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):66–75, 1994.

[14] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifi ers and their applications to handwritting recognition. *IEEE Systems, Man, and Cybernetics*, 22(3):418–435, 1992.

[15] A. Freitas. A genetic programming framework for two data mining tasks: Classifi cation and generalized rule induction. In J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 96–101, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.

[16] Folino G., Pizzuti C., and Spezzano G. A cellular genetic programming approach to classifi cation. In *Proc. of GECCO 99 Genetic and Evolutionary Computation Conference,*, pages 1015–1026. Morgan Kaufmann Publishers, S. Francisco, CA, July 1999.

[17] J. Kishore, L. Patnaik, V. Mani, and V. Agrawal. Application of genetic programming for multicategory pattern classifi cation. *IEEE Transactions on Evolutionary Computation*, 4(3):242–257, September 2000.

[18] A. Teredesai, J. Park, and V. Govindaraju. Active handwritten character recognition using genetic programming. In *EuroGP2001 Proceedings*, 2001.

[19] P. Devijiver and J. Kittler. *Pattern Recognition: A statistical approach*. Prentice-Hall, London, 1982.

[20] V. Govindaraju, Z. Shi, and A. Teredesai. Secondary classifi cation using key features. In P. Kantor, D. Lopresti, and J. Zhou, editors, *Document Recognition and Retrieval VIII, Proceedings of the SPIE*, volume 4307, pages 272–279, 2001.

[21] A. Teredesai and V. Govindaraju. Active digit classifi ers: A separability optimization approach to emulate cognition. In L. Spitz K. Tombre and C. Fuh, editors, *Proceedings of The Sixth International Conference on Document Analysis and Recognition*, pages 401–406, September 2001.