Image Recognition using Genetic Programming with Loop Structures

Fahmi Abdulhamid
School of Engineering
and Computer Science
Victoria University of Wellington
PO Box 600, Wellington, New Zealand
Email: abdulhfahm@myvuw.ac.nz

Kourosh Neshatian
School of Engineering
and Computer Science
Victoria University of Wellington
PO Box 600, Wellington, New Zealand
Email: kourosh.neshatian@ecs.vuw.ac.nz

Mengjie Zhang
School of Engineering
and Computer Science
Victoria University of Wellington
PO Box 600, Wellington, New Zealand
Email: mengjie.zhang@ecs.vuw.ac.nz

***Abstract*—Genetic Programming (GP) presents one means to automate object classification tasks in images. GP uses a process similar to biological evolution to evolve programs that are able to identify significant features and correlations in image data. Using GP with loop structures has been found to increase the classification accuracy over traditional GP without loops. We propose four new loop structures for image classification tasks. The loops provide a framework for summarizing pixel information into new image features and taking advantage of the correlation between neighboring pixels with an iteration structure. The identified features are used to distinguish between objects of different classes. The results suggest that the loops increase image classification accuracy and produce evolved programs that are smaller and more Human understandable than GP with no loops and previous work with GP using loops.**

I. Introduction

Image classification and object detection are important subjects in computer vision. Tasks such as environmental analysis, video surveillance, and image categorization all require the automatic analysis and inference of images [1] where it is not feasible for people to perform the tasks manually. Automated image classification techniques improve classification accuracy and efficiency over manual processes which are ideal for streamlining production processes and for robotics vision applications. For image classification tasks, it is often the case that locating particular regions of significance for differentiation becomes a difficult task due to the lack of domain knowledge or the natural variances between images. Machine learning processes such as Genetic Programming (GP) provide one means of automatically identifying which parts of images are useful for classification.

GP is able to analyze raw image pixel data and find correlations between pixels which may be significant for classification. GP's population based learning makes it capable of identifying correlated pixels and then improving its solutions by analyzing different combinations of pixel values. Although effective, prior knowledge of pixel correlations is often not provided to GP for image classification problems. It is understood that neighboring pixels are considered correlated but GP must undergo significant learning to identify such a correlation. Features in images span multiple pixels in size so it may not be effective to target pixels in isolation. The lack of expressiveness from targeting individual pixels and the large learning effort to correlate pixels can hinder the classification accuracy of learned solutions.

One way to improve the effectiveness of GP for image classification tasks is to incorporate loop structures to target nearby pixels that are likely to be correlated. Loop structures such as 'For' loops and 'While' loops can increase the expressiveness of learned solutions produced by GP by performing aggregate functions designed to exploit relationships within image regions [2], [3]. Image regions and aggregate functions operated upon those regions can be learned automatically. Learned solutions become easier to understand because notable regions of images are explicitly identified and the increased expressiveness of loop structures reduce the size of learned solutions. Loop structures for classification tasks have been found to increase the accuracy of learned classifiers [2], [4]. However, the range of loops used for image classification is narrow. Varying characteristics of different loop structures enhance or restrict the utility of loop structures for different types of image classification problems. There is a need to develop a variety of loop structures for examination against varying image classification problems to gain a better understanding of the utility of loops in GP.

A. Goals

This paper aims to develop an approach to the use of loop structures in GP for image recognition/classification tasks. To achieve this goal, we will develop four different loops with various considerations. Instead of using manually crafted, task specific features as inputs to the GP program classifiers, this approach will directly use the raw image pixels as inputs with the expectation that the new loop structures can automatically locate the regions (and features) of interest useful for classification. This approach will be examined and compared with the GP approach without loops and a well-known GP approach with loops (Li's Loop [2]) on three benchmark image recognition/classifications problems of varying difficulty. Specifically, we will investigate:

1) how the loop structures can be developed in GP with the considerations of image properties;

2) whether GP with the new loop structures can achieve better classification performances than GP without a loop structure and a well known GP approach with loop structure;
3) whether the loop structures in GP can automatically evolve/leave good regions/features of interest that are helpful and understandable for image classification; and
4) whether the loop structures evolved by the new approach are simpler and easier to understand than Li's loop method.

B. Organisation

In Section II a brief overview of GP and background work is given. Section III describes current loops and our new loop structures. Section IV describes the image datasets used and how experiments are carried out. In Section V results are evaluated and loop structures are discussed. Section VI concludes the paper.

II. Background

GP is a form of evolutionary machine learning that generates and 'evolves' a population of candidate programs to solve a problem. Sometimes, it can be unclear which features are best to find a solution, be it general problem solving or image recognition. GP is effective for identifying features in areas where the domain is not well understood. Below, an overview of GP is given and a brief introduction to previous work is described.

A. Overview of Genetic Programming

GP is one form of evolutionary machine learning. Given a problem, a set of function nodes (procedures such as arithmetic operators), and a set of terminal nodes (variables such as pixel values and constant numbers), GP will evolve a computer program to solve the problem through a process similar to biological evolution [5]. GP cycles through iterations called 'generations' to perform the evolution process. For each generation, the best programs from the previous generation are chosen and genetic operators such as crossover and mutation are performed to breed a new generation of programs. Each generation improves in performance until a user-defined stopping criterion has been met (perfect classification accuracy for example). When evolution concludes, the best individual is taken and applied to a testing set of unseen data to measure classification performance. For binary classification tasks, GP will evolve a program that returns a negative value for one class and a positive value for another. For image classification tasks the final output will be based on individual pixel values and their locations. Loop structures are a form of function node capable of iteration.

B. GP with Loops for Problem Solving

Much early work on loop structures for GP have been focused on problem solving tasks. For instance, a Lambda expression structure capable of implicit recursion was used in GP to solve the Even-N-Parity problem [6]. The addition of iteration through Lambda expressions allowed GP to have a notably higher probability of solving the problem than GP with no loops. In a separate study, the addition of both count-controlled and event-controlled loops were tested on the factorial problem (n!) and the artificial ant problem [3]. The GP systems with loops were able to reliably solve both problems while GP with no loops found the problems difficult to solve. The evolved solutions from GP with loops were also simpler than those generated from GP without loops which is important for analysis purposes.

C. GP with Loops for Image Classification

Loop structures have been applied to various binary image classification tasks. One previous application of loop structures in GP was the incorporation of the `for-loop-2d-rect` and `for-loop-1d` loops [2, Ch. 4]. The former loop structure is evaluated against our new loop structures in Section V. The loop structures achieved better classification accuracy than GP without loops for a simple image problem comparing boxes to circles. The learned solutions were smaller and more Human readable than those learned by GP with no loops and the probability of success for GP with loops improved upon GP with no loops. Despite the positive results, the loop structures were evaluated against only one image dataset. Further, the dataset was artificially produced and consisted of binary pixel values which may not be representative of real world problems. The datasets we use contain images with gray scale pixels of naturally occurring objects.

The application and comparison of two identical loop structures, one restricted (nesting disallowed) and the other unrestricted (nesting allowed), found that GP with loops performed notably better than GP without loops for a difficult classification problem where objects in images were shifted off-center [4]. Furthermore, the unrestricted loop structure outperformed the restricted loop structure for the difficult problem. A computer-generated character recognition problem and naturally occurring texture recognition problem were used. The evolution time of GP with loops was found to be longer than that of GP without loops. When loops were applied to a coin recognition problem, it was found that GP with loops achieved more accurate results and has less overfitting than GP with no loops [7], indicating that the solutions learned by GP with loops are more general and in turn more reliable on unseen data.

III. New Loop Structures

Four new loop structures are proposed for binary image classification problems: The *Evolvable Body Loop* (EB-Loop), *Step-Loop*, *Bar-Loop*, and *Seam-Loop*. All but the Step-Loop evaluate on 2D image data. All loops except for the EB-Loop are unrestricted, allowing nesting to occur to generate at least one argument. For comparison, GP with no loops and Li's `for-loop-2d-rect` [2, Ch. 4] (restricted loop) were implemented.

A. EB-Loop

Performing good aggregate operations over loop regions is essential to derive higher-level meaning of an identified feature. When it is not fully understood which operations are useful, GP can be used to build useful loop operations. The Evolvable Body Loop (EB-Loop) is designed to analyze the performance of GP when evolving a loop body for aggregate operations. The EB-Loop structure is `(EB-Loop startPosition endPosition loopBody)`. The `loopBody` argument is a subtree that represents the evolved aggregate function. The `loopBody` argument uses two special loop terminals, `LoopCurrentValue` and `LoopResult`, to accumulate an aggregate result.

B. Step-Loop

Related object features are not guaranteed to be nearby. Useful information may be extracted by identifying if related features change together or by targeting physically large features in images. The Step-Loop is designed to find such features by targeting distant pixels. The Step-Loop structure is `(Step-Loop startIndex stepSize iterationCount loopMethod)` where iteration is akin to a for loop as illustrated in Fig.1.

C. Bar-Loop

The probability of finding good solutions increases if the search space is reduced because fewer search paths are taken. The Bar-Loop is given an orientation (vertical or horizontal) to enforce the constraint that the loop region must touch two opposing sides of an image. The constraint minimises the search space by automatically defining the length of loop regions. The Bar-Loop structure is `(Bar-Loop position thickness orientation loopMethod)` where `thickness` determines the thickness of a row or column in number of pixels. `loopMethod` determines and operation in the loop (e.g. summation).

D. Seam-Loop

The Seam-Loop is inspired by Seam-Carving [8] which attempts to find seams of low energy. The Seam-Loop finds seams of high energy (bright pixels) which may be significant for classification purposes. For performance, the Seam-Loop performs a rough version of the Seam Carving algorithm. Only vertical seams are found and the process is performed using a greedy hill-climbing algorithm without any preprocessing steps. The Seam-Loop structure is `(Seam-Loop startColumn loopMethod)` where iteration begins from the (x, y) coordinate (`startColumn`, 0) and moves downwards.

Table I describes the terminals and functions used for GP with no loops. Every GP system with loops contains a superset of the terminals and functions described in Table I. All loop arguments of type `loopMethod` are one of Summation, Standard Deviation, or Sum-Mode (the sum of all values with an equal-highest frequency) only, including Li's loop structures (which originally used Summation and Subtraction) for comparison.

Algorithm III.1: STEP-LOOP($start, step, count, data$)

$index \leftarrow start; result \leftarrow 0$
for $index \leftarrow 0$ **to** $count$
do $\begin{cases} index \leftarrow index \ \% \ data.length \\ result \leftarrow result + data[index] \\ index \leftarrow index + step \end{cases}$
return $(result)$

Fig. 1. Step-Loop Procedure

TABLE I
TERMINALS AND FUNCTIONS FOR GP WITH NO LOOPS

Node	Type	Description
Drand	Terminal	Generates a double value between 0.0-100.0 denoted by 'drand-x'.
Pixel-Value	Terminal	Generates a random position within the bounds of the image, denoted by 'pixel-value-[x,y]'. It returns the pixel value at position (x,y) and the pixel value is cast to a double.
Addition	Function	Takes two double values and returns the sum.
Subtraction	Function	Takes two double values, subtracts the second from the first and returns the result.

IV. EXPERIMENT CONFIGURATION

A. Image Datasets

Three image datasets were used for analysis: Cell, MNIST, and Face. The datasets were chosen because they are clearly different classification problems that have different characteristics which we use when comparing the new loop structures. The datasets and any modifications are described below.

1) Cell Dataset: The Microscopical Cell Image Database (Serous cytology) [9], [10] represents a collection of eighteen different types of Cells. Experiments perform a binary classification of Lymphocytes non activés and Mésothéliales. The original images are in colour and have been converted to gray scale images to simplify classification. In gray scale, each pixel is a natural number between 0 and 255 (inclusive) which allows pixels to be represented as single numbers. Images are also cropped to only include the cell by removing data not included in the nuclear segmentation of the cell. The images are cropped to avoid GP classifying cells based on artifacts outside each cell which ensures that GP systems are actually distinguishing between cells. The original cell images were of different dimensions so all images were scaled to 25×25 pixels to ensure consistency. The cells have a relatively bland internal content which means that the shape/structure of the cells play a large role in classification.

2) MNIST Dataset: The MNIST Database of Handwritten Digits [11] is a collection of images representing hand written numbers from 0 to 9 (inclusive). All images are 28×28 pixels. Experiments perform a binary classification between the "6" and "9" digits. All images are gray scale and numbers are centered on each image. Because the actual digits are filled with a solid colour, GP systems must classify digits based on

TABLE II
GP EXPERIMENT VARIABLES

Variable	Value
Experiment runs	30
Generations	50
Population size	1024
Crossover (%)	70
Mutation (%)	28
Elitism (%)	2
Termination	0% error or generation limit reached
Training Set size	500 (250 class 1 and 250 class 2)
Testing Set size	500 (250 class 1 and 250 class 2)
Tree Depth	Min: 1 Max: 8

shape alone.

3) Face Dataset: The Face dataset [12] contains a collection of images which are either a face or an unknown object. Faces occupy the entire space of the image and hair is not displayed (though facial hair is). Experiments perform a binary classification between "Face" and "Not Face" images. All images are in gray scale and 19×19 pixels. Faces have well known significant regions that can be used for differentiation: Eyes, noes, and mouth. Because the border of the face is not clearly displayed in images, GP systems must perform classification based on the content of the face, not the shape of the head.

B. GP Parameter Settings

Each dataset consists of 1000 images: 500 for training and 500 for test. Experiments perform binary classification between two classes. The training and test sets contain an equal split of both classifications. The GP settings are described in Table II. The GP settings are similar to those used by Li [2, Ch. 4] in order to facilitate a comparison with the new loop structures. Each GP system was run 30 independent times in order to produce accurate results. Every run used the same training and testing sets with only variability in the seeds chosen for GP terminals which require a random number. Strongly Typed GP using ECJ [13] was used to ensure the proper construction of programs.

V. RESULTS AND DISCUSSIONS

Table III displays the results of the GP systems against the three image classification problems. Training and testing set mean classification accuracies with a + or − superscript indicate statistical significance of $p < 0.05$ when compared to GP with no loops for a higher or lower value respectfully. Results are discussed in the following sections. In most cases, the best program evolved with loops achieved better performance than GP without loops, particularly on the test set.

A. Cell Classification Results

It is clear in Table III that each GP system with loops achieved a higher accuracy in the testing set than GP with no loops. Furthermore, GP without loops achieved the greatest reduction in performance between the training set and testing set of 3.81%, indicating that GP with loops is able to generalize more effectively on image data than GP without loops. Li's loop achieved one of the best classification accuracies on the testing set. Li's loop structure requires a start coordinate and an end coordinate in order to target significant regions in images. Although more precise regions may be able to produce more significant results by better targeting correlated features, the added complexity of the loop structure increases the search space which may have reduced the quality of the loop regions learned. The Bar-Loop and Seam-Loop produced the best results (marginally better than Li's loop) while the Step-Loop and the EB-Loop found the classification task more difficult.

To understand why the evolved loops can perform well, Fig.2 shows the learned loop regions of the best solutions for the Cell dataset. Loop regions are marked in yellow. Green is used for clarity to differentiate multiple overlapping loop regions. The Bar-Loop achieved a good result by evolving two large loop regions. The first (in yellow) targets the bottom half of the cell which is extracting features from the internal content and size of the cell while the second (in green) uses a pixel-value to determine the thickness of the Bar. A black pixel value due to a small cell will result in a narrow horizontal line that may likely produce a small value (Fig.2i) while a gray pixel will produce a thicker bar that would produce a larger value (Fig.2d). The best solution from Li's loop produced two smaller loops. Despite the use of smaller loop regions that may be more capable of targeting specific features, the size and position of the loops may be sensitive to variations in cell size and shape within a single class. The increased sensitivity may have lead to a poorer classification accuracy than the Bar-Loop. The large loop region is targeting cell internal content while the smaller loop is targeting cell size. The EB-Loop only produced two valid loop structures, both targeting the top of the cell to detect cell size.

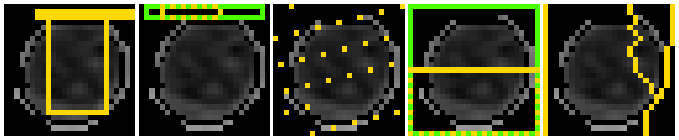
Of all the GP systems, the Seam-Loop achieved the best result on the Cell dataset. The Seam-Loop used pixel values to determine the start position of seams based on the cell while the Seam Carving algorithm automatically targeted the border of the cell, which may be significant for classification. The reduced search space due to the simplicity of the loop structure coupled with the particular characteristics of the cell image may have significantly contributed to the Seam-Loop achieving the overall highest accuracy on the test set. The Step-Loop clearly iterated over much of the cell to detect changes in cell internal content and cell size. Due to the relatively bland internal content of cell images, the Step-Loop had used a large step size to target many areas across cells rather than localized regions. Notwithstanding, the difficulty for the Step-Loop to target localized regions may be a factor that contributed to a lower best result than the other loop structures.

B. MNIST Classification Results

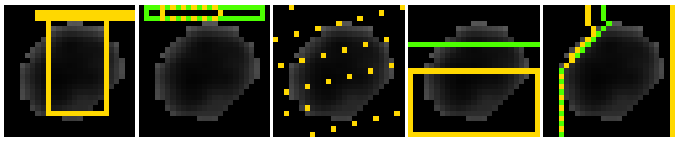
Li's loop achieved the best average accuracy for the MNIST dataset followed closely by the Bar-Loop and the Step-Loop. It is apparent that due to the relatively simple nature of the

TABLE III
GP RESULTS FOR IMAGE DATASETS.

Data Set	Fitness Set	Variables	No-Loop	Li	EB-Loop	Step-Loop	Bar-Loop	Seam-Loop
Cell	Training	Mean (%)	96.00	96.10	94.49^{-}	95.70	96.27	97.43^{+}
		Std. Dev. (%)	0.63	0.62	1.74	1.37	0.80	0.56
		Best (%)	97.60	97.20	96.80	97.80	97.80	98.40
		Duration (s)	13.63	48.43	238.03	335.37	977.13	177.53
	Test	Mean (%)	92.19	94.32^{+}	93.54^{+}	93.13^{+}	94.91^{+}	94.84^{+}
		Std. Dev. (%)	1.45	0.69	1.85	1.77	1.15	0.62
		Best (%)	94.60	95.60	96.00	95.60	97.20	96.40
MNIST	Training	Mean (%)	99.31	99.57^{+}	98.71^{-}	99.43	99.47^{+}	98.56^{-}
		Std. Dev. (%)	0.30	0.14	0.81	0.17	0.19	0.79
		Best (%)	100.00	100.00	99.60	99.80	100.00	99.80
		Duration (s)	14.20	56.13	203.03	419.53	1112.97	129.13
	Test	Mean (%)	96.54	99.03^{+}	97.95^{+}	98.55^{+}	98.93^{+}	95.70^{-}
		Std. Dev. (%)	1.12	0.31	1.19	0.75	0.35	1.14
		Best (%)	98.60	99.60	99.60	99.60	99.40	97.80
Face	Training	Mean (%)	96.67	96.06	92.32^{-}	95.91^{-}	96.51	95.76^{-}
		Std. Dev. (%)	0.96	1.89	3.41	1.69	1.61	1.98
		Best (%)	98.00	99.40	96.80	98.40	98.60	99.00
		Duration (s)	14.00	29.77	139.00	293.93	524.63	72.83
	Test	Mean (%)	91.41	91.45	86.95^{-}	91.82	92.95^{+}	91.27
		Std. Dev. (%)	1.33	2.58	3.70	2.58	2.64	2.34
		Best (%)	93.80	96.60	92.20	96.40	96.60	95.00

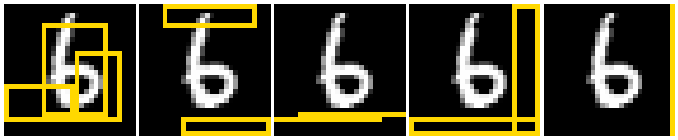


(a) Li (b) EB-loop (c) Step-Loop (d) Bar-Loop (e) Seam-Loop

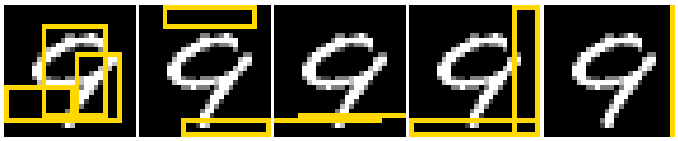


(f) Li (g) EB-Loop (h) Step-Loop (i) Bar-Loop (j) Seam-Loop

Fig. 2. Example evolved loops of the best solutions for the Cell dataset. The top row are class 1 images and the bottom row are class 2 images.



(a) Li (b) EB-Loop (c) Step-Loop (d) Bar-Loop (e) Seam-Loop



(f) Li (g) EB-Loop (h) Step-Loop (i) Bar-Loop (j) Seam-Loop

Fig. 3. Example evolved loops of the best solutions for the MNIST dataset. The top row are class 1 images (6s) and the bottom row are class 2 images (9s).

images (white object on a black background), GP with loops had little variation in classification accuracy as evidenced by a small standard deviation. Li's loop and the Bar-Loop were the most stable GP systems for this problem while the complexity of the EB-Loop and the unpredictability of the Seam-Loop lead to higher variances in classification results.

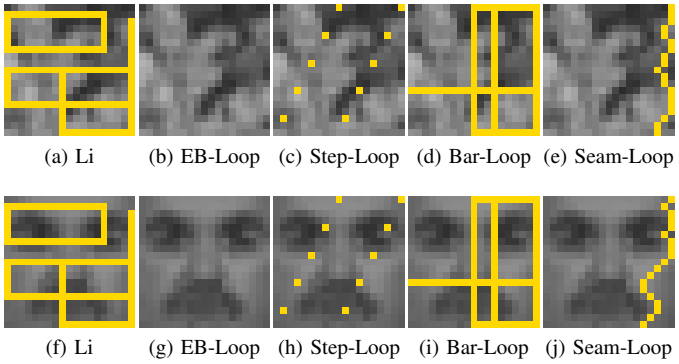
Concerning the loop regions of the best solutions for the MNIST dataset (Fig.3), it is important to note that each best solution, except for the Seam-Loop, performed equally well. Li's loop identified regions within the body of digits. The left loop region is targeting the area under the head of the nine which may be empty for a nine but non-empty for a six. The right loop is targeting the body of the six and the right vertical stroke of the nine for differentiation. The center loop may be analyzing the mass of the digit which may be taking advantage of peculiarities in the drawing of six and nine digits. Li's loop and the Seam-Loop produced the least intuitive solutions while the remaining loops produced easily understandable solutions.

It is clear that the EB-Loop had targeted the top 'stalk' of the six and the bottom stalk of the nine which are both unique. The Step-Loop produced two loops (the overlapping loop is not shown) to target the stalk of the nine only. A step of one ensures that the narrow stalk is not missed. The Bar-Loop also targeted the stalk of the nine, but also the right of the image which may be detecting the right vertical stroke of the nine or the base of the six.

It is clear that the new loop structures had evolved relatively simple solutions that provide insight into which regions of images are significant for classification. The clarity of solutions may be helpful for analysis of data where the domain may not be well understood.

C. Face Classification Results

The Bar-Loop performed the best for the Face classification problem with a tie for second place with Li's loop, the Step-Loop, and the Seam-Loop. It is clear in Fig.4 that Li's loop



(a) Li (b) EB-Loop (c) Step-Loop (d) Bar-Loop (e) Seam-Loop

(f) Li (g) EB-Loop (h) Step-Loop (i) Bar-Loop (j) Seam-Loop

Fig. 4. Example evolved loops for the best solutions for the Face data set. The top row are class 1 images and the bottom row are class 2 images.

produced more complex results than the remaining loops with the Step-Loop and Bar-Loop producing simpler results. Li's loop targeted the significant portions of the face, the eyes, nose, and mouth to achieve good classification results. The Bar-Loop produced a better result by targeting only one side of the face to produce more specific loop regions as faces are nearly symmetrical. The EB-Loop did not learn any valid loop structures and hence it achieved the worst classification results because no iteration could be performed. The unpredictability of the Seam-Loop lead to a poorer best result, especially with regard to the fact that the horizontal alignment of features in faces could not be extracted by the vertical-only Seam-Loop.

D. Comparison of Loop Structures

It is clear that Li's loop structure performs well when compared to GP with no loops and our proposed loop structures, but the learned solutions are generally not as simple and Human readable as those produced by the Bar-Loop and the Step-Loop. Through restricting the search space by applying constraints to the Bar-Loop, GP was able to more quickly find good image regions that when looped over produce significant results for classification. It is also apparent that the Step-Loop is adaptable to various types of images because it is able to vary the step size in order to target nearby or distant features and is able to alter its direction to capture variances in images that occur on particular axis. The Seam-Loop, while able to produce good results for training data by automatically targeting significant portions of images, does not generalize as well on unseen data when compared to the remaining loop structures because the path of the Seam differs from picture to picture. The learned solutions are also less understandable than the remaining loop structures.

The EB-Loop was designed to learn complex loop bodies in order to make better use of identified regions of significance in images. But the difficulty of evolving valid loop structures as well as the large size of the loop structure are limiting factors when considering finite restrictions on the number of generations and GP tree depth. It is clear that the new loop structures all increase the evolution time when compared to GP with Li's loop and GP with no loop, but with the benefit of being able to produce simpler and more accurate classifiers.

VI. Conclusion

In this paper, we have proposed four new loop structures for image classification tasks and compared the new loop structures with GP with no loops and GP with Li's loop structures on three image classification problems. Two of the new loop structures, the Bar-Loop and the Step-Loop, have been able to achieve good results, especially with regard to the Bar-Loop achieving the best average testing accuracies for two out of the three classification problems. The Step-Loop and Bar-Loop were also able to produce program solutions that were simpler and more Human readable than those learned by Li's loop, and also those from GP with no loops. We have found that the complexity of a loop structure can negatively impact classification performance as evidenced by the EB-Loop.

It is clear that GP with loops can achieve better results than GP with no loops for binary image classification problems but more research must be undertaken to better understand how the different properties of loop structures impact image classification performance. Research to increase the efficiency of GP to learn solutions with loop structures is also desirable to ensure the wider application of loop structures.

References

[1] R. Poli, W. B. Langdon, and N. F. McPhee, *A Field Guide to Genetic Programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`, 2008.

[2] X. Li, "Utilising restricted for-loops in genetic programming," Ph.D. dissertation, School of Computer Science and Information Technology, RMIT University, February 2007.

[3] G. Chen and M. Zhang, "Evolving while-loop structures in genetic programming for factorial and ant problems," in *AI 2005: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3809, pp. 1079–1085.

[4] J. Larres, M. Zhang, and W. Browne, "Using unrestricted loops in genetic programming for image classification," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, July 2010, pp. 1–8.

[5] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[6] T. Yu and C. Clack, "Recursion, lambda-abstractions and genetic programming," in *Late Breaking Papers at EuroGP'98: The First European Workshop on Genetic Programming*, R. Poli, W. B. Langdon, M. Schoenauer, T. Fogarty, and W. Banzhaf, Eds. CSRP-98-10, The University of Birmingham, UK, 1998, pp. 26–30.

[7] G. Wijesinghe and V. Ciesielski, "Using restricted loops in genetic programming for image classification," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, September 2007, pp. 4569 –4576.

[8] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *ACM SIGGRAPH 2007 papers*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007.

[9] O. Lezoray, A. Elmoataz, and H. Cardot, "A color object recognition scheme: Application to cellular sorting," *Machine Vision and Applications*, vol. 14, pp. 166–171, 2003.

[10] O. Lezoray. (2011, July) List of image databases. [Online]. Available: http://users.info.unicaen.fr/~lezoray/Databases.php

[11] Y. LeCun and C. Cortes. The MNIST database of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[12] M. cbcl. (2000) Face data. [Online]. Available: http://cbcl.mit.edu/software-datasets/FaceData2.html

[13] S. Luke. (2011, July) ECJ 20: A java-based evolutionary computation research system. [Online]. Available: http://cs.gmu.edu/~eclab/projects/ecj/