Extracting Image Features for Classification By Two-Tier Genetic Programming

Harith Al-Sahaf
School of Engineering and CS
Victoria Uni. of Wellington
New Zealand
harith.alsahaf@gmail.com

Andy Song
School of CS and IT
RMIT University
Australia
andy.song@rmit.edu.au

Kourosh Neshatian
School of Engineering and CS
Victoria Uni. of Wellington
New Zealand
kourosh.neshatian@ecs.vuw.ac.nz

Mengjie Zhang
School of Engineering and CS
Victoria Uni. of Wellington
New Zealand
mengjie.zhang@ecs.vuw.ac.nz

***Abstract*—Image classification is a complex but important task especially in the areas of machine vision and image analysis such as remote sensing and face recognition. One of the challenges in image classification is finding an optimal set of features for a particular task because the choice of features has direct impact on the classification performance. However the goodness of a feature is highly problem dependent and often domain knowledge is required. To address these issues we introduce a Genetic Programming (GP) based image classification method, *Two-Tier GP*, which directly operates on raw pixels rather than features. The first tier in a classifier is for automatically defining features based on raw image input, while the second tier makes decision. Compared to conventional feature based image classification methods, Two-Tier GP achieved better accuracies on a range of different tasks. Furthermore by using the features defined by the first tier of these Two-Tier GP classifiers, conventional classification methods obtained higher accuracies than classifying on manually designed features. Analysis on evolved Two-Tier image classifiers shows that there are genuine features captured in the programs and the mechanism of achieving high accuracy can be revealed. The Two-Tier GP method has clear advantages in image classification, such as high accuracy, good interpretability and the removal of explicit feature extraction process.**

***Index Terms*—genetic programming; feature extraction; feature selection; image classification.**

I. Introduction

The importance of image classification becomes more and more apparent in the recent years with the exponential growth of image data and vision related applications, such as remote sensing, medical imaging systems and face recognition [24]. However it still remains as a complex task [12]. The common approach of image classification is utilitizing one of the conventional classification methods such as support vector machines and decision trees on extracted features, which are transformed from raw images. The dimensionality of raw image data is simply too high for most classification methods. Furthermore the spatial relationships between neighbouring pixels can not be observed by these methods since they often consider data points independent from each other. Therefore the feature extraction phase in conventional approaches is critical. The subsequent classification is directly affected by the extracted features. Poor features would not lead to accurate classification.

The challenges in feature extraction reside in several aspects. The goodness of a certain feature is often problem dependent. For a specific application, designing a totally new feature may be required. There are no universal features which can excel in all applications. Also there are numerous existing image features such as histogram features, edgeness features, texture features and features in the frequency domains. Deciding a suitable set of features is usually the focus of image classification applications. Such an exercise heavily relies on domain knowledge as the understanding of the task itself and the extensive experience on existing features in the literature are crucial. It is highly desirable to have an approach which requires less human involvement and can be applicable on wide range of problems. That is the aim of this study, which introduces a Genetic Programming (GP) based approach to automatically defining features and classifying directly on raw images rather than on manually designed features.

Fundamentally GP is a search strategy for automatic program construction. An individual of GP is effectively an executable program, which could be a program for extracting features, or a program for performing classification, or even both. The powerfulness of this GP paradigm has been shown in many complex tasks such as job scheduling, structural design, and medical diagnosis [18]. One of the advantages of GP is creativity as GP often finds excellent solutions which have never been thought by human experts. The adaptation of GP in image related tasks has been also successful, including image segmentation [17], edge detection [5], texture analysis [21], motion detection [16] and finding interest points [15]. On these tasks GP could achieve better or at least comparable performance without much domain knowledge. The advantages of GP are evident in the areas of image classification as well, for example separating haemorrhage and micro aneurysms in retina images reported by Zhang et al. [25], identifying roads and field regions from radar images reported by Bhanu and Lin [11], and classifying texture image reported by Song et. al [21]. Although most of these prior works still require human designed features, the flexibility and effectiveness of GP are clear. The above observation is the basis of our `Two-Tier` GP approach to image classification presented here. We expect GP to automatically discover genuine features which may be better than human designed features in one tier and to accurately assign class label based on these implicitly extracted features in the other tier.

A. Goals of This Study

The aim is to present a GP based image classification method which operates directly on raw pixels, and to study the features extracted implicitly inside the evolved GP classifiers. The specific questions addressed in this investigation are:

- What is the suitable GP representation for generating pixel based, not feature based, classifiers?
- Would this GP approach perform well on a range of image classification tasks especially compared with other approaches?
- Are there any genuine features automatically defined inside these classifiers by GP?
- Can we interpret the behavior of these evolved classifier to reveal the features and the decision mechanism of these classiers?

The rest of this paper is organized as follows: Section II briefly discusses the relevant prior work. Section III presents the Two-Tier GP methodology. Section IV shows four image classification tasks and their corresponding features which are designed manually. Section V reports the experiments along with the results. Section VI is the analysis on some of the evolved GP classifiers. Section VII concludes this investigation.

II. Background

As a domain independent method, GP has been adapted extensively in classification including image classification [4, 3, 2]. GP can not only generate classifiers but also evolve feature extraction methods [23, 8, 7]. For example the features for fault detection evolved by GP outperformed the features designed by domain experts [6]. Additionally these features cost less to compute. The GP generated features for edge detection reported by Zhang and Rockett performed better than the classical Canny algorithm [26]. Similarly in the work of Lam and Ciesielski, the texture features generated by GP were similar or slightly better in comparison with manually constructed texture features[9].

It is expected that the GP classifiers from the Two-Tier approach could perform both feature extraction and classification. There are similar works existed in the literature, such as a two-stage GP scheme introduced by Oechsle and Clark[14]. They have two separate stages in the system. The first is for feature extraction while the second is for classification. Although both stages are GP-based, it is different from our approach because two GP programs are involved in their system and human intervention is required to reformulate the extracted features so that the two programs can be integrated together. Our aim is to generate one program for both tasks. This approach appeared in the early work of Atkins et al. [1] where GP classifiers with three tiers: *image filtering tier*, *feature extraction tier* and *classification tier*, were evolved. Their evolved classifiers could achieve similar performance compared to the features designed manually. Our study is the extension of an work, as the three-tier approach has some limitations which we aim to avoid in this two-tier approach.

III. Two-Tier GP

The structure of `Two-Tier` GP is presented in this section. The constructs of an individual in Two-Tier GP are shown in Figure 1. There are two types of functions, classification functions (CF) and aggregation functions (AggF), which form the two tiers, the *Classification* tier and the *Aggregation* tier respectively.

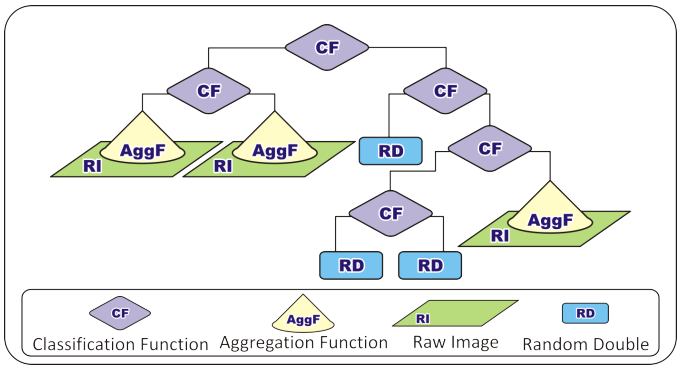


Fig. 1: Tree Structure in Two-Tier GP

The aggregation tier does not permit any functions to be attached to it. The only kind of children to this tier are raw image terminals (RI). Each RI terminal feeds a 2-D array of pixels to its parent aggregation function. Additionally no aggregation function can be the root node. So the aggregation tier always locates at the bottom level of a tree, just above RI terminals. These constrains do not apply on classification functions which can be nested and can be placed as the root node. Moreover a CF node can take randomly generated doubles (RD) as its terminals. A tree with just classification tier and no aggregation functions is grammatically correct. However it can not produce good classification hence will be eliminated from the population due to its low fitness.

Each aggregation function (AggF) takes an image as its input and produces a numeric value as its output. It is actually transforming a matrix of raw pixels into a single value. This dimensionality reduction process can be viewed as feature extraction although the exact behavior of the aggregation function is not predefined, but automatically generated by the evolution. All the aggregation functions on the same tree take the same image as their inputs, so the images under different AggF nodes in Figure 1 are identical. However their behaviors will be different. They are expected to locate different regions of the same image and capture different characteristics of the image to facilitate classification in the classification tier.

Each classification function operates on double numbers which may come from AggF nodes or CF nodes attached to the function, or RD terminals. The return value from the root CF node is the decision from this GP classifier. The image received from the bottom RI nodes will be labeled as one class if the return value from the root is positive, or as the other class if the return value is zero or negative. Such a decision boundary is used because our study here only focuses on

TABLE I: Aggregation Functions

Function	P 1 (Image)	P 2 (X)	P 3 (Y)	**P 4 (Shape)**	P 5 (Size S)	Return
$AggMean$	Image	Integer	Integer	*Enum*	Integer	Double
$AggMed$	Image	Integer	Integer	*Enum*	Integer	Double
$AggStDev$	Image	Integer	Integer	*Enum*	Integer	Double
$AggMax$	Image	Integer	Integer	*Enum*	Integer	Double
$AggMin$	Image	Integer	Integer	*Enum*	Integer	Double

binary classification. Multi-class classification can be adapted by changing this decision mechanism or introducing binary decomposition.

A. Function Set

The function set for Two-Tier GP includes classification functions and aggregation functions. Their details are given below.

The five aggregation functions included in the function set of Two-Tier GP are listed in Table I. Each one has five input parameters. The first parameter is the original input image for classification. The second and third are X, Y coordinates in integers. The fourth is an enumerated value which indicates a shape. The last is also integer which is a size S. For an input image, these functions sample a region of size S in the shape specified in the 4th parameter, at the position X, Y. Then they return the mean, the median, the standard deviation, the maximum and the minimum of the sample region respectively. The return of these functions is a double number.

The parameter values X, Y, S are input from corresponding terminals. The smallest possible value for size S is 3. A sub-image smaller than 3×3 can not sufficiently represent the entire image. To avoid out-of-range errors, the maximum for size S is $min(Image_{width}, Image_{height})$ while that for X, Y is $Image_{width}$ and $Image_{height}$ respectively. The sampling window will be truncated if it exceeds the image boundaries.

The fourth parameter, *Shape*, randomly provides an enumerated value from five choices "square", "column", "row", "circle" and "rectangle". The flexibility in sampling shapes would increase the expressiveness of the classifiers and increase the chance of finding prominent features from the images. When the shape is "column", then the function samples a column of pixels as a vertical 1-D array of which the length is S. For a shape "row", a function would sample a horizontal 1-D array of length S to perform the corresponding calculation. The thickness of sub-images either in "row" or "column" is just 1. One might think that would not be enough to catch important characteristics of input images. However there will be likely not just one but multitude of these functions in one classifier. They operate together, so we expect a collection of lines, or a mix of lines with other shapes would be better in capturing the most important regions.

When the shape is "circle", then the function will take X and Y as the center, using the *Bresenham* circle algorithm to generate a circle of which the diameter is S. For shape "rectangle", the sampling window starts as X and Y as its top-left corner. In this case, the size S is unused. Instead two extra values will be generated to determine the *width* and *height* of the window. The calculations of mean, median and three other values are then based on the pixels under the circle or the rectangle but within the image boundaries.

These aggregation functions are similar to feature extraction operations in the literature. Functions like *mean* and *standard deviation* are used widely such as [19, 14, 25]. The task of GP is rather to find the best regions with an optimal size, and the right operations on these regions so the prominent characteristics can be captured.

TABLE II: Classification Functions

Function	Input Parameters	Return
$+$	Double, Double	Double
$-$	Double, Double	Double
$\times$	Double, Double	Double
$\div$	Double, Double	Double
IF	Double, Double, Double	Double

The classification functions are shown in Table II, which contain four arithmetic operators, taking two double numbers as inputs, and one conditional operator IF taking three doubles input. All of them produce a double number as the output. The division operator $\div$ is protected and will output zero if the denominator is zero. The IF function passes the second input as its output if its first input is negative. Otherwise, the third input will be taken as the output. This choice of operators is similar to that in the related works [25, 21, 17, 1].

B. Terminal Set

The terminal set in conventional GP methods for image classification usually just includes a random number generator and nodes for receiving feature values as the inputs. However the terminal set in our methodology is a little more complicated due to the Two-Tier program structure. They are shown in Table III. The "RD" terminal is the usual random number generator. It is the terminal for classification functions. The other terminals are all for the aggregation functions described in the previous sub-section.

TABLE III: Terminals Set

$Terminal$	Type	Description
RD	Double	Randomly generate a number in between [0,1]
$Image$	Image	2D-array containing raw image pixel values
$Size$	Integer	Size of the sampling window
X, Y	Integer	Returns a value as the coordinate
$Shape$	Enum	Returns one from {square, row, column,circle, rectangle}

The *Image* terminal is the input of GP classifiers, an image in raw pixels. Terminal *Size* returns a value in between 3 and the minimum of image width and height, $[3, min(Image_{width}, Image_{height})]$, to specify the size of a sampling region under an aggregation function. Terminals X, Y are responsible to generate random integers as

coordinates for the aggregation functions. They can not go beyond the image boundaries. The last terminal *Shape* has five possible return values: "square", "column", "row", "circle" and "rectangle". The aggregation function attached to this terminal will behave differently according to the return value from this terminal.

C. Fitness Measure

The fitness in the case of image classification is quite straightforward, simply the classification accuracy as shown in Equation 1:

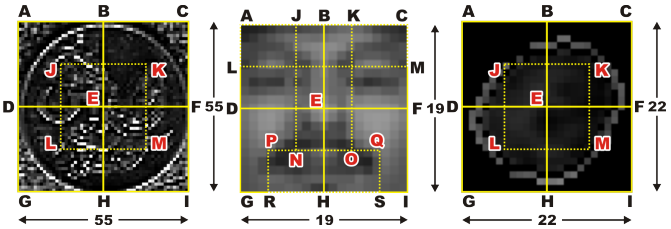
$$Fitness = Classfication\ Accuracy = \frac{TP + TN}{TOTAL} \times 100\% \quad (1)$$

where TP is the total number of True Positives, the positive examples been classified as positive; TN is the total number of True Negatives, the negative examples been correctly classified; $TOTAL$ is the total number of examples in the dataset.

IV. Image Classification Tasks

Four sets of image classification tasks are introduced. They are described here in the order of problem difficulties. A group of manually defined features for each task are also presented.

a) Coins: The first task *Coins* is relatively easy[20], which is to differentiate heads from tails. It contains 384 grey-scale images of size 55×55. The coins in both classes appear in different orientations. So a successful classifier can not be too specific to a particular coin position. Figure 2(a) shows the predefined features for this task. Coin images are divided into small regions, and ten features can be extracted accordingly. They are the *Mean* and *Standard Deviation* of pixel intensities for the four quadrants ABED, BCFE, DEHG, and EFIH; and the central square JKML.



(a) Coin Image (b) Face Image (c) Cell Image

Fig. 2: Pre-defined Features for Coin, Face and Cell Images

b) Face Detection: The second task is to separate face images and non-face images from MIT *Faces* dataset [22], which contains 60000 examples of size 19×19. This is considered a problem of medium difficulty compared to the next two images sets as the face images are relatively similar to each other. For example, dark regions are always present around areas of the eyes, but not on non-face images. The pre-defined features for this case are designed based on the work of [2]. As shown in Figure 2(b), an image is divided into seven regions. They are the quadrants ABED, BCFE, DEHG and EFIH; and three specific areas ACML, JKON and PQSR which represent the eyes, the nose, and the mouth. The fourteen features are the *Mean* and *Standard Deviation* of these seven areas.

c) Cell Recognition: The more difficult task is *Microscopic Cells* [10], which is to classify between two classes: *Lymphocytes non activés* and *Mésothéliales*. The original dataset consists of 3900 color images of eighteen classes, manually categorized by domain experts. There are in total 1297 instances which have been converted into grey-scale images of size 22×22 pixels. The manually designed features for the cell problem are similar to these for the coins dataset. Ten features (*Mean* and *Standard Deviation*) have been extracted from five different areas as shown in Figure 2(c): the quadrants ABED, BCFE, DEHG and EFIH; and the central square area JKLM.

d) Pedestrian Detection: The most difficult task among these four is the pedestrian detection problem [13], because both classes, pedestrian images and non-pedestrian images, contain large variations. This dataset consists of 10002 grey-scale examples of size 18×36. The domain-specific features are based on the work of [1]. There are in total 22 features (*Mean* and *Standard Deviation*) extracted from eleven regions as shown in Figure 3. These regions are: the octets ABED, BCFE, DEHG, EFIH, GHKJ, HILK, JKMN and KLON; and three middle areas PQSR, RSUT and TUWV which roughly cover the head, the torso and the legs of a pedestrian.

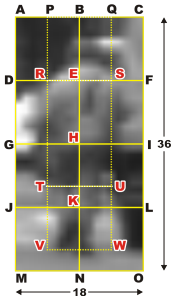


Fig. 3: Pre-defined Features for Pedestrian Detection

V. Experiments and Results

For each of the task mentioned above, we split the dataset into two halves. One for training and one for test. The sizes of these datasets are listed in Table IV. The columns Pos and Neg show how many positive examples and negative examples each dataset has. There is no class imbalance between the positives and the negatives to bias the final result.

TABLE IV: Datasets for the Four Tasks

	Training Set			Test Set		
	Pos	Neg	Total	Pos	Neg	Total
Coins	96	96	192	96	96	192
Faces	1500	1500	3000	1500	1500	3000
Blood Cells	349	299	648	349	300	649
Pedestrians	2501	2501	5001	2500	2500	5001

A. Methods for Comparison

To evaluate the effectiveness of the *Two-Tier GP* (2TGP) and to validate the features generated by this approach, we

TABLE VI: Experiment Results

	Coin Classification		Face Detection		Cell Classification		Pedestrian Detection	
	Training%	Test%	Training%	Test%	Training%	Test%	Training%	Test%
2TGP	100.0	100.0	97.93	97.27	96.14	97.69	90.34	92.28
FeEx+GP	100.0	100.0	91.67	94.03	89.35	91.37	86.58	87.66
FeEx+Naïve Bayes	100.0	100.0	87.53	91.17	77.01	88.30	87.48	66.62
FeEx+Decision Tree	100.0	100.0	99.13	95.67	95.37	85.36	98.96	68.41
FeEx+SVM	100.0	100.0	90.00	86.20	87.50	88.60	90.40	68.03
2TFe+Naïve Bayes	100.0	100.0	92.73	94.86	93.52	94.92	87.50	73.71
2TFe+Decision Tree	100.0	99.47	99.50	96.50	97.84	95.38	99.22	87.94
2TFe+SVM	100.0	100.0	97.66	95.10	94.44	95.69	93.48	78.76

TABLE V: GP Run Time Parameters

Parameter	Value
Generations	50
Population Size	1024
Crossover Rate	0.80
Mutation Rate	0.19
Elitism Rate	0.01
Tree Depth	2-10
Selection Type	Tournament
Tournament Size	7

performed the following comparisons on the four classification tasks.

- The *"FeEx+GP"* method [4], which is applying GP on the pre-defined features described in Section IV. GP operates on feature vectors, not on raw images in this case.

- The *"FeEx+[conv_method]"* approach, which takes the conventional way of image classification, applying a classification method on the pre-defined features. The classification methods include Naïve Bayes, Decision Trees and SVM (Support Vector Machines) as they are arguably the most popular choices in the field of classification.

- The *"2TFe+ [conv_method]"* approach, which is to validate the features embedded in Two-Tier classifiers. The classification nodes in these evolved classifiers were removed and the aggregation nodes were applied onto images to generate features. Then Naïve Bayes, Decision Tree and SVM were applied on these feature vectors.

B. GP Run Time Parameters

Both the Two-Tier GP and the FeEx+GP approaches involve GP runs. For the comparison purpose all the run time parameters for them were identical. They are listed in Table V. The initial populations are created by ramped half-and-half method. Each evolutionary process stops at the maximum generation 50 unless a perfect classifier with accuracy 100% is found. Every GP run was repeated 30 times.

C. Results and Discussions

The experimental results from all the experiments are presented in Table VI. These are the mean accuracies of the corresponding 30 runs. The training and test accuracies for each classification task are listed in one column. There are eight methods involved in our experiments, including the Two-Tier GP method, GP classification based on pre-defined features, three conventional classification methods on pre-defined features and on features discovered by Two-Tier GP. The accuracies obtained by one method on these tasks are shown in one row in the table.

From the results, we can see that the performance of every method degrades from the coin classification to pedestrian detection. That indicates the increase of difficulty level among these four tasks. On the easiest problem, coin classification, all methods achieved perfect or near perfect results. On the face detection problem, Two-Tier GP was the best performer. The conventional classification methods, especially SVM and Naïve Bayes, were not that accurate. Such observation is also true in the case of cell classification and pedestrian detection.

Comparing the Two-Tier GP with other approaches, it consistently reached the highest accuracy on the test set. The absence of pre-defined features did not damage its performance. On the contrary operating on raw pixels gives GP more flexibility in terms of constructing meaningful features and classifiers. This is evident by the results from "FeEx+GP"'. The GP components in "FeEx+GP" and in "2TGP" are very similar. However the former built classifiers on given features and the accuracies were lower than their counterpart from "2TGP" on the three relatively hard tasks, as its performance was hindered by these features.

In term of the capability of building classifiers on these pre-defined features, GP is at least comparable to these popular conventional methods, and arguably better on cell and pedestrian images ("FeEx+GP" vs "FeEx+{Naïve Bayes or Decision Tree or SVM})". This result is consistent with the finding from other studies: "GP-based classifiers generally compare quite well with the ones induced by other algorithms" (VII. B [4]).

The more interesting results are the accuracies from the three conventional methods on features that are defined by these aggregation functions in the best 2TGP classifiers (see the last three "2TFe" rows in Table VI). On all the tasks except the easy coins problem, all three methods received a significant increase in performance compared to their corresponding accuracies on the pre-defined features, both on training and on test. This suggests that those manually defined features are not the best for these tasks. The Two-Tier GP was able

to construct even better features, possibly by defining more meaningful regions or more meaningful aggregation functions or the combination of these two.

Despite the improvement introduced by "2TFe", these three conventional methods were still less accurate than "2TGP". This indicates that the classification functions on these Two-Tier GP classifiers are meaningful. They could better combine feature values returned from these aggregation functions into a more prominent decision variable, so the accuracies were higher than those from the three conventional methods on the same features.

VI. Program Analysis

Interpretability of evolved solution is a well known issue in GP. Here we aim to analyze some of the classifiers generated from the experiments in Section V, to understand their behavior and reveal the features defined by the aggregation functions since these features appeared better than the pre-defined features in the comparison.

Figure 4 shows an evolved program that has scored 100% accuracy on both of the training and test sets on the coin images. It is quite small as there are only two aggregation functions. The first (AggStDev) defines a 16×16 square region at the position $(29, 32)$, and the second (AggMean) defines a 4×25 rectangle at the position $(3, 22)$. These two regions are marked on a Tail image (framed in blue) and a Head image (framed in red) in the figure. Effectively this program is

$$StDev(Region_1) - Mean(Region_2)$$

This is the difference between the standard deviation of the first region and the mean of the second region. This classifier is actually comparing the middle area with the edge on the coin images. The center areas of head and tails are indeed quite different. This could be a defining characteristic to distinguish these two classes. This feature is not affected by coin rotations.

Tracing the return values from the two aggregation functions on the tail image and the head image, we can see how this classifier reaches its decision. In the case of Tail (the blue numbers), the standard deviation of center square is much larger than the mean of the edge, so a positive value (35.81) is returned. In the case of Head (the red numbers), this difference between center than edge gives a negative return (-9.4). Hence by taking zero as the decision boundary, this classifier can separate tails and head.

Similar to Figure 4, a 2TGP classifier for face detection is displayed in Figure 5 along with one example of non-face image, one example of face image and their corresponding values propagated on the tree. This program scored 93.63% and 92.17% on the training and test set respectively. Note this program is not the most accurate 2TGP classifier because the more accurate programs are large in size. So this small program is selected to facilitate the analysis.

There are three aggregation functions in this face detection program. Hence three regions have been identified which are all squares (with size 3, 3 and 12 respectively). It can expressed as

$$0.876 - Mean(Sq_1) + Mean(Sq_2) + StDev(Sq_3)$$

Two squares are around the area of the right eye. The standard deviation on the big square is seemingly able to capture the distinct feature of a face. The return value from it on a face is much larger than that on a non-face image (71.75 vs. 28.42). Consequently the final output for face is positive (2.18) while that for non-face is negative (-49.81). This classifier can differentiate two classes using zero as the decision boundary.

Similarly one example from the pedestrian detection problem is shown in Figure 6. To facilitate the analysis we again did not select the most accurate classifier which was much bigger. The performance of this program is reasonable, 88.26% and 88.68% on the training and test sets respectively.

This classifier has three aggregation functions. It can expressed as

$$0.412 \times Mean(Row_1) - Min(Row_2) + Min(Rect_3)$$

The first two regions are bottom rows while the third region is a rectangle covering the torso area. The separation in return values for pedestrian and no pedestrian is also clear. One is positive (31.31) while the other class produces a negative number (-40.46). In short we can see that the aggregation functions do provide distinctive feature values for different classes, and the classification functions can combine these feature values to form a meaningful decision mechanism.

VII. Conclusions

This paper studied the Two-Tier GP methodology for image classification, which is based on raw pixels rather than on pre-defined features. A classifier in Two-Tier GP has a tier of aggregation functions which are to transform an image into a single numeric value, and a tier of classification functions which are to transform the outputs of aggregation functions into a class label. One classifier may have multiple aggregation nodes. Each node identifies a region in one of the five shapes and calculates an aggregation value over this region by one of the five functions. This approach has been compared with the conventional approach: extracting predefined features then applying one classification method on the features. The classifiers in comparison include Naïve Bayes, Decision Tree, SVM and GP itself. The results show that the Two-Tier representation is valid and can outperform all these four classifiers on most of the image tasks used in this study.

Furthermore, to verify the features automatically defined in the aggregation nodes, we applied these features to Naïve Bayes, Decision Tree and SVM which achieved better accuracies compared with pre-defined features. This suggests that the features found by Two-Tier GP classifiers are genuine and they are better than the manually designed features used in this study. Although neither human intervention nor domain knowledge is involved, these programs are still able to find out prominent regions or features. In a way implicit and effective feature discovery is achieved through this Two-Tier GP method.

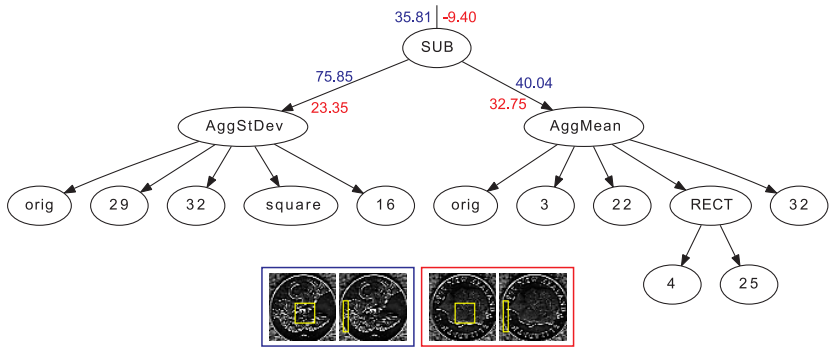


Fig. 4: An Example of 2TGP Classifier for the Coin Problem

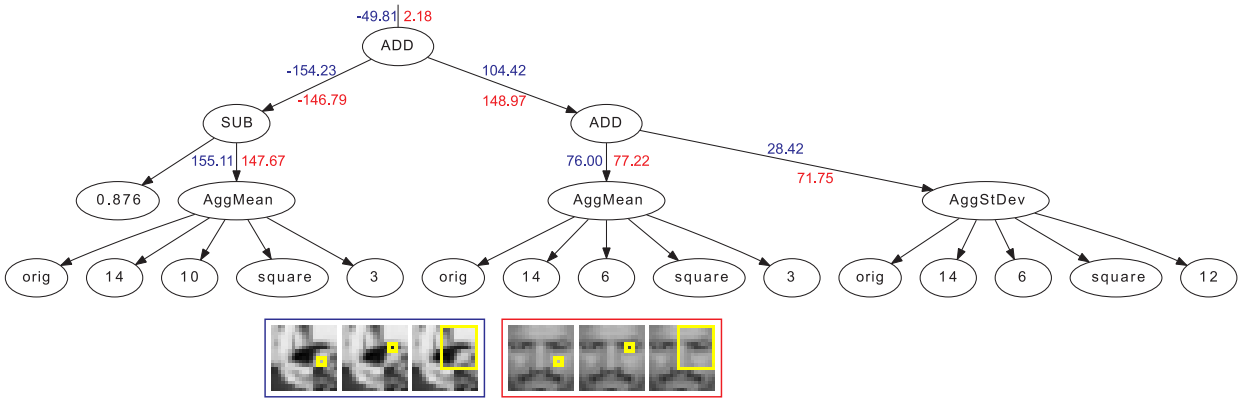


Fig. 5: An Example of 2TGP Classifier for Face Detection

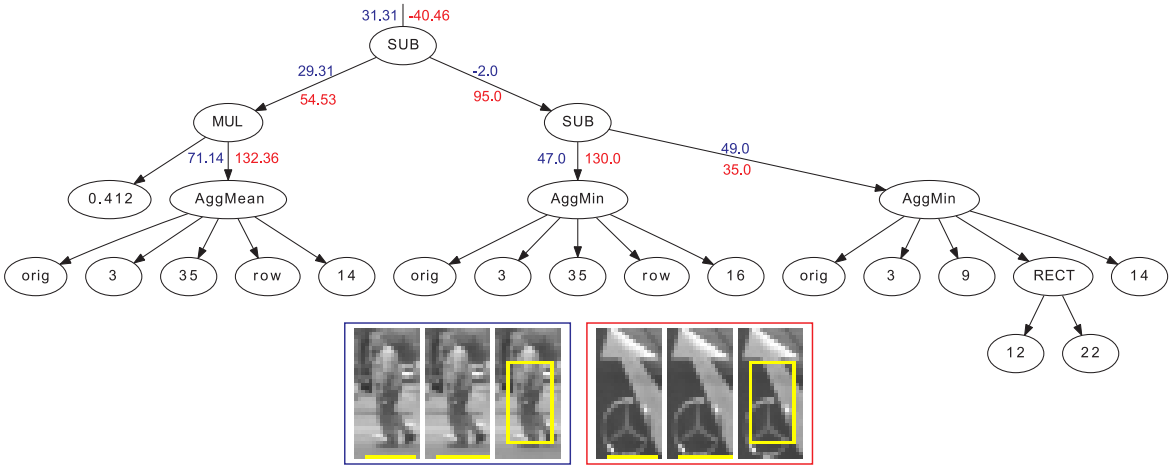


Fig. 6: An Example of 2TGP Classifier for Pedestrian Detection

The evolved Two-Tier image classifiers were further analyzed to reveal their behavoirs. Some of the programs are small in size and their decision making mechanism are explainable. We can see that their success is not by chance. There are distinctive regions identified by these classifiers. Based on these regions the classifiers produce positive values for one class and negative values for the other class. Good interpretability can be another advantage of the Two-Tier image classification approach.

In the near future we will incorporate program simplification to further reduce the tree size for complex classifiers. More interesting patterns may be revealed by such analysis. This would help us in solving some more challenge problems. Moreover new functions will be proposed to enhance the applicability of this method.

References

[1] Daniel Atkins, Kourosh Neshatian, and Mengjie Zhang. A domain independent genetic programming approach to automatic feature extraction for image classification. In *IEEE Congress on Evolutionary Computation*, pages 238–245, 2011.

[2] Urvesh Bhowan, Mengjie Zhang, and Mark Johnston. Genetic programming for classification with unbalanced data. In Anna Isabel Esparcia-Alcázar, Anikó Ekárt, Sara Silva, Stephen Dignum, and A. Sima Etaner-Uyar, editors, *EuroGP*, volume 6021 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2010.

[3] J. A. dos Santos, Cristiano D. Ferreira, Ricardo da Silva Torres, Marcos André Gonçalves, and Rubens A. C. Lamparelli. A relevance feedback method based on genetic programming for classification of remote sensing images. *Inf. Sci.*, 181(13):2671–2684, 2011.

[4] Pedro G. Espejo, Sebastián Ventura, and Francisco Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(2):121–144, 2010.

[5] Wenlong Fu, Mark Johnston, and Mengjie Zhang. Genetic programming for edge detection: A global approach. In Alice E. Smith, editor, *Proceedings of the 2011 IEEE Congress on Evolutionary Computation*, pages 254–261, New Orleans, USA, 5-8 June 2011. IEEE Computational Intelligence Society, IEEE Press.

[6] Hong Guo, L. B. Jack, and Asoke K. Nandi. Feature generation using genetic programming with application to fault classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(1):89–99, 2005.

[7] Hong Guo and Asoke K. Nandi. Breast cancer diagnosis using genetic programming generated feature. *Pattern Recognition*, 39(5):980–987, 2006.

[8] Taras Kowaliw, Wolfgang Banzhaf, Nawwaf N. Kharma, and Simon Harding. Evolving novel image features using genetic programming-based image transforms. In *IEEE Congress on Evolutionary Computation*, pages 2502–2507, 2009.

[9] Brian T. Lam and Victor Ciesielski. Discovery of human-competitive image texture feature extraction programs using genetic programming. In *GECCO (2)*, pages 1114–1125, 2004.

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

[11] Yingqiang Lin and Bir Bhanu. Object detection via feature synthesis using MDL-based genetic programming. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 35(3):538–547, June 2005.

[12] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing.*, 28:823–870, January 2007.

[13] Stefan Munder and Dariu M. Gavrila. An experimental study on pedestrian classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(11):1863–1868, 2006.

[14] Olly Oechsle and Adrian F. Clark. Feature extraction and classification by genetic programming. In *ICVS*, pages 131–140, 2008.

[15] Gustavo Olague and Leonardo Trujillo. Evolutionary-computer-assisted design of image operators that detect interest points using genetic programming. *Image Vision Comput.*, 29(7):484–498, 2011.

[16] Brian Pinto and Andy Song. Detecting motion from noisy scenes using genetic programming. In *Proceeding of the 24th International Conference Image and Vision Computing New Zealand, IVCNZ '09*, pages 322–327, Wellington, 23-25 November 2009. IEEE.

[17] Riccardo Poli. Genetic programming for feature detection and image segmentation. Technical report, The University of Birmingham, School of Computer Science, The University of Birmingham, Edgbaston, Birmingham B 15 2TT, UK, 2000. R.Poli@cs. bham. ac. uk.

[18] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A Field Guide to Genetic Programming*. lulu.com, 2008.

[19] Will R. Smart and Mengjie Zhang. Classification strategies for image classification in genetic programming. In *Proceeding of Image and Vision Computing Conference*, pages 402–407, 2003.

[20] William D. Smart and Mengjie Zhang. Using genetic programming for multiclass classification by simultaneously solving component binary classification problems. In *EuroGP*, pages 227–239, 2005.

[21] Andy Song and Vic Ciesielski. Texture segmentation by genetic programming. *Evolutionary Computation*, 16(4):461–481, Winter 2008.

[22] Kah Sung. *Learning and Example Selection For Object and Pattern Detection*. PhD thesis, Massachusetts Institute of Technology, 1995.

[23] Walter Alden Tackett. Genetic programming for feature discovery and image discrimination. In *ICGA*, pages 303–311, 1993.

[24] Roberto A. Vázquez, Humberto Sossa, and Beatriz A. Garro. An evolutionary feature-based visual attention model applied to face recognition. In *Proceedings of the 5th international conference on Hybrid Artificial Intelligence Systems - Volume Part I*, HAIS'10, pages 376–384, Berlin, Heidelberg, 2010. Springer-Verlag.

[25] Mengjie Zhang, Victor B. Ciesielski, and Peter Andreae. A domain independent window approach to multiclass object detection using genetic programming. *EURASIP Journal on Applied Signal Processing*, (8):841–859, 2003.

[26] Yang Zhang and Peter Rockett. Evolving optimal feature extraction using multi-objective genetic programming: a methodology and preliminary study on edge detection. In Hans-Georg Beyer and Una-May O'Reilly, editors, *GECCO*, pages 795–802. ACM, 2005.