VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

A Genetic Programming Approach to Binary Image Classification

Yi Sian Lim

Supervisors: Mengjie Zhang, Bing Xue, Ying Bi

Submitted in partial fulfilment of the requirements for
Bachelor of Science with Honours in Computer Science.

Abstract

In recent years, image classification has received substantial attention from researchers all over the world but remains a challenging problem. It is a widely used task in artificial intelligence for many applications such as social media platforms, automobile industry and medical diagnosis. Genetic Programming (GP) has been applied to this area with promising results. High-level features extracted using methods such as Local Binary Patterns (LBP) and Histogram of Orientated Gradients (HoG) are commonly used for image classification. This project aims to investigate novel approaches to using GP with uniform LBP and an improved variant of LBP, known as Local Quinary Pattern (LQP) to extract/construct high-level features to effectively perform image classification respectively. Some high-performing GP individuals are analysed to interpret how GP can be effectively be used with high-level features.

Acknowledgments

Thank you to my supervisors: Mengjie Zhang, for his excellent advice, Bing Xue, for her constructive feedback and Ying Bi, for her ongoing guidance to put me on the right track.

My appreciation also goes to my friends and family, who supported me throughout this year and made this journey a fulfilling one.

Contents

Chapter 1

Introduction

Computer vision is a field of study that aims to develop robust computational models that has a similar or better capability than humans to interpret and extract information from images or videos [10]. Image classification is an important task in computer vision with a wide range of applications such as battlefield analyses in the military, robot navigation in robotics and breast density classification in medical diagnoses [45].

Image classification refers to the ability of a computer to identify several variables such as people, objects or buildings by analysing an image. While humans can identify images with relative ease, it is far too labour-intensive to analyse a large number of images manually. Image classification remains a difficult problem in computer science and machine learning due to the large variations in images.

1.1 Motivations

The objective of this project is to develop new techniques in a novel way with existing techniques for image classification. It aims to investigate the use of genetic programming (GP) to perform feature extraction and construction to create advanced high-level features for classifying images.

Feature extraction is a major step in image classification. It derives values intended to be informative and non-redundant in order to facilitate subsequent learning. It is a special form of dimensionality reduction to obtain the most relevant information from a raw image and represent that information in a lower dimensional space. The primary goal of feature extraction is to extract a set of features to maximize the recognition rate of a learning system. A number of widely used feature extraction methods are Local Binary Patterns (LBP) [31], Histogram of Orientated Gradients (HOG) [14] and Grey-Level Co-occurence Matrix (GLCM) [23].

Evolutionary computation is a general problem solving technique based on simulated evolution. It typically creates a population of individuals and evolve these individuals to create a new population. This technique is able to search for the best solution from a set of solutions through a number of generations without any human intervention. GP is one of the most commonly used evolutionary computation technique on image analysis [57]. In literature, GP has shown high performance for optimizing and classifying related problems [33] [9] [3].

Since it was introduced in the 1990s, GP has been applied in various fields of image analysis tasks such as edge detection [22], object tracking [47], segmentation [44] and classification [9]. Combining GP with feature extraction methods has the potential to effectively perform extract high-level features for image classification. However, existing works on using image-related operators/descriptors in GP for feature extraction and image classifications are limited.

LBP is a simple gray-scale descriptor that has been combined with GP for image classification in [9] and [46]. LBP, when combined with GP, was able to achieve a high accuracy compared to other state-of-art image classification methods. LBP have inspired many variants which are widely considered the state-of-art among texture descriptors such as local ternary pattern (LTP) [59] and local quinary pattern (LQP) [37]. The results of LQP in [37] has shown high performance, thus it is able to find a reliable set of features. Employing LQP and LBP in GP has the potential to allow it to evolve high-level image features from raw images.

While GP, LBP and LQP have been used extensively for various image classification tasks, literature combining LBP or LQP as feature extractors with GP is limited. Using LBP and LQP by itself to extract features across a whole image can be limited, when only parts of an image may produce meaningful features. Combining GP with these effective descriptors has the potential to boost image classification performance.

Furthermore, most of the existing methods involve the use of image operators using predetermined parameters. These image operators are not tuned with a range of parameter values to adapt its operation to work with a wide range of problems. As a consequence, the image operators may end up using parameters that are unsuitable for the problem. However, manually tuning of the parameters requires a significant amount of time and requires expert knowledge, which is not always available. Therefore, employing GP in order to evolve the parameters used in these image operators can aid in choosing parameters that are more suited for different data sets.

1.2 Goals

In this project, the goal is to develop new GP-based methods that incorporate LBP and LQP operators to improve upon the accuracy of binary image classification. Binary classification was chosen as a starting task of the project in order to evaluate the effectiveness of the proposed methods. The solution could be extended in future to deal with multi-class image classification. The specific objectives for the project include:

1. Developing new GP methods with a new program structure, a new function set and a new terminal set to identify regions of interest, extract useful features, construct powerful high-level features, and classify the images. Two potential methods will be developed which extract features using LQP and LBP in the first and second method respectively.

2. Investigating whether the proposed methods can outperform the GP and non-GP methods on a range of datasets of varying difficulties.

3. Visualizing and interpreting the features that have been automatically extracted and constructed by the new GP methods.

1.3 Major Contributions

This project contains a number of major contributions:

- This work shows that using texture descriptors such as LQP and LBP operators to extract and construct high-level features for image classification shows promising results. This proves that textural information are effective features for many image classification problems, including non-texture datasets. The method using uniform LBP was able to achieve comparable average testing accuracy to general and GP based classification methods. While the method did not always outperform all GP methods, they achieved faster training time than the best performing GP classifier.

- This work shows that GP programs are intelligible and interpretable, which is an important goal in data mining. This was achieved by visualising the evolved programs as tree structures, and representing the nodes of the trees as the intermediary steps of the algorithm. This is an advantage over general classifiers such as Random Forest or AdaBoost where it is difficult and sometimes impossible to visualise each step of the algorithm due to its extreme complexity.

- This work shows how GP can be used for simultaneously selecting regions on interests, extracting good features and performing classification by using a single evolved program that combines local binary pattern features and low-level features. The automatically evolved programs were able to achieve comparable average testing accuracy to general and GP based classification methods.

1.4 Report Organisation

Chapter 2 presents an overview of machine learning, evolutionary computation and image classification, background to the work, including a general overview of feature extraction operations (LBP and LQP) and related works. Chapter 3 presents the proposed LQP-GP method, results and analysis of the new GP approach using LQP as well as datasets and baseline methods used in this work. Chapter 4 presents the proposed uLBP-GP method. Chapter 5 concludes the report and discusses future work.

Chapter 2

Background and Related Work

An overview of machine learning, evolutionary computation and genetic programming that form the basis of this work are discussed. Related works in image classification are also discussed, and the limitations of existing works are summarised to form the motivations of this project.

2.1 Machine Learning

Machine learning [7] is a field in Artifical Intelligence (AI) that solves a problem by learning from past experiences or input data, in order to make predictions on unseen data. Machine learning often uses algorithms that involve building and refining a model by optimizing the corresponding parameters using the input data.

Machine learning techniques can be categorised in one of the three following categories, supervised learning, unsupervised learning and reinforcement learning.

2.1.1 Supervised Learning

Supervised learning trains a program on a pre-defined set of labelled training examples and the expected output of the training example is known. For example, using a Decision Tree or Support Vector Machine classifier to classify whether a patient has cancer or not, given a dataset containing information on the patient's health.

Supervised learning algorithms such as Random Forest [10] and AdaBoost [27] have been employed to perform image classification tasks with good results. Both Random Forest and AdaBoost are ensemble learning techniques. Random Forest is a classifier that consists of a number of trees, in which each tree is grown using randomization [10]. Each individual tree in the random forest determines the class label and the class label with the highest number of votes becomes the model's final prediction. AdaBoost works in a similar manner by classifying datasets using a weighted sum of all base classifiers which represents the model's final prediction. [27].

Common stopping criteria are reaching a predefined accuracy reached on the training set, training all instances trained or elapsing a certain amount of computational time. The trained model is then applied to the unseen instances in the test set. The accuracy of the model on the test set determines the predictive ability of the model on future data, and hence gives a evaluation of its performance and effectiveness.

However, a common problem in supervised learning is overfitting, a situation where a model corresponds too closely or exactly to the training set, resulting in poor performance on the test set. One way to combat this issue is to introduce a validation set. After each iteration of the training process, the performance of the model is evaluated on the validation set. When the performance starts to deteriorate, it indicates that the model is over-fitting and training should be stopped. Therefore, using a validation set allows test accuracy to be improved by preventing the model to be overly specific on the training set.

2.1.2 Unsupervised learning

Unsupervised learning trains a program on a set of unlabelled training examples and the expected output is unknown. Unsupervised learning [7] finds regularities or natural partitions on the training set in order to group the data. For example, K-means clustering is an unsupervised learning approach that is commonly used for document clustering, which involves grouping a set of documents into groups based on their content similarities.

Unsupervised learning is able determine unknown patterns in the training set, which is highly useful in the business field in order to explore new ventures or understand financial drivers of businesses based on current trends.

2.1.3 Reinforcement learning

Reinforcement learning operates by maximizing a reward in a particular solution. It approaches a problem by learning the behavior through trial-and-error interactions in a dynamic environment [26]. By maximizing a numerical reward signal, it allows the learning system to learn which actions yield the most reward by trying them out. An example is solving different games at super-human performance such Tic-Tac-Toe where the reward is how close the agent is to creating a horizontal, vertical or diagonal connection. [48]

There are two main strategies for solving reinforcement learning problems - search in the space of behaviors to find the behavior with the best performance in the environment and use statistical techniques and dynamic programming methods to estimate the utility of taking actions in states of the world. The first strategy has been taken by work in genetic algorithms and genetic programming [25].

Reinforcement learning differs from supervised learning in several ways. The most crucial difference is that there are no representations of input and class labels. After an action is chosen, the agent is told the immediate rewards and the subsequent state, instead of the action that would have benefited the agent the most.

2.2 Evolutionary Computation

Evolutionary computation (EC) is a field in AI that involves the study of algorithms inspired by biological evolutionary principles. EC is often applied to problems with large search space. The common underlying idea behind all EC techniques operate iteratively to refine candidate solutions to a problem in each step in order to find the optimal solution. An objective function is used as an abstract fitness measure.

EC techniques typically includes Evolutionary Algorithms [19], Swarm Intelligence [28] and other algorithms. Evolutionary Algorithms (EA) are based on Darwinian principles of natural selection [18] such as reproduction, mutation and crossover.

The fittest candidates are chosen based on the result computed by the fitness function, to seed the next generation by applying genetic operators such as mutation, crossover and reproduction. The process repeats until a given criterion is reached (e.g. a specific number of iterations have been executed or a good solution has been found). The general scheme of EA can be given as follows [18]:

```
Initialize population with random individuals (candidate solutions)
Evaluate the fitness all individuals using a suitable objective function
while stopping criterion is not reached DO
    Select parent from parent population
    Create offspring using genetic operators on parent
    Evaluate the fitness of the offspring
    Replace less fit parents by some offsprings
END
```

Common EC techniques are Particle Swarm Optimisation (PSO) [17] and Genetic Algorithms (GA) [24]. PSO is a swarm intelligence technique that replicates the behaviour of swarming animals, such as birds, by using a number of particles which are moved around the search space. Each particle is a candidate solution and the solution is encoded within the particle. The position of the particle changes after each iteration based on its best known position and the swarm's best known position. On the other hand, GA is an EA algorithm that have similar encoding methods as PSO, but uses the biological evolutionary principles discussed above to iteratively refined their pool of candidate solutions.

2.2.1 Genetic Programming

Genetic programming (GP) uses Darwinian natural selection to evolve computer programs to perform a given task. GP is a subarea of EA that will be the focus of this project due to its flexible representation and interpretability of the solution/model. GP typically have individuals represented as a tree based structure, commonly referred to as programs. The resulting programs are often hierarchical. For example, a mathematical function 2 - (x * 8) can be represented as a tree structure as shown in Figure 2.1.

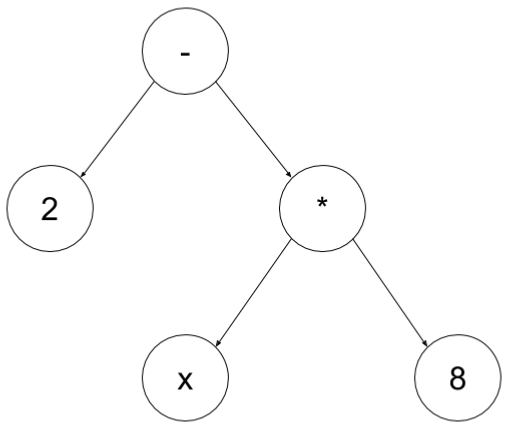

Figure 2.1: Mathematical function of $2 - (x * 8)$ represented as a tree structure.

Before applying GP to a problem, there are five preparatory steps to carry out [30]. They involve determining the terminal set, the function set, an objective function for the purpose of fitness measure, the parameters for controlling and the criterion for terminating a run. The terminals in the terminal set are usually inputs to the program and the functions in the function set should be able to accept one or more values as its arguments that may be returned by any function in the function set or assumed by any terminals in the terminal set. A major consideration is that the function set and the terminal set must satisfy both closure (any function can accept the output of any other function or terminal) and sufficiency (a combination of functions and terminals can solve the problem) property [30].

Another major step of GP is designing an appropriate objective function for fitness measure. Each individual is evaluated using the fitness measure to determine its performance in the problem environment. Often, the fitness is measured by the discrepancy between the results produced by the individuals and the desired results. For example, a fitness measure for a suitable classification problem would be the percentage of instances correctly classified, where 100 is the most ideal and 0 is the worst.

The primary parameters for controlling a run of genetic programming includes the population size and the number of generations. The termination criterion frequently used is the maximum number of generations reached.

Strongly Typed Genetic Programming

Strongly typed genetic programming (STGP) was proposed by Montana [36] places restrictions on how functions and terminals may be combined. STGP differs from standard GP in which the data type constraints are enforced for each argument of the function. This allows STGP to handle a combination of data types to generate more high-level programs as more complex functions can be used. For instance, in an image classification problem, a function may take one child of type *image* and output a processed *image*.

2.3 Computer Vision

Computer vision is a field that works to automate computers to see, identify and process images the same way the human brain does. However, enabling computers to differentiate between images of different objects is a difficult task, as images are simply represented as pixels.

Computer vision consists of many tasks such as image classification, object recognition and object detection [49]. Image classification aims to assign a label to the image as a whole, object recognition aims to determine all of the objects within an image and object detection is similar to object recognition with the difference that it also finds the position of the objects within the image. [13].

This project focuses on image classification due to its wide use in various real-world applications.

2.3.1 Image Classification

Image classification [35] is a task in Computer Vision which involves assigning an image with a class label based on the content in the image. For instance, determining if a given image contains a face or not, which is an example of binary classification. Another type of image classification is called multi-class image classification, which differs from binary classification in which the problem have at least three or more classes to be labelled from. For instance, determining if a set of fruit image is a "strawberry", "tomato" or "orange" based on what fruit was in the image.

While humans are able to identify images with ease, it can be challenging for computer programs to achieve similar performance due to several factors such as differences in contrast, brightness and presence of noises across images A common approach to address these issues is to perform feature extraction.

2.3.2 Feature Extraction

Feature extraction is a procedure which derives features that are intended to be informative and improve the view representation of the image in order to facilitate the subsequent learning. It works by reducing the number of data required to describe a large set of data. The output from feature extraction is then used as inputs for classifiers.

Two feature extraction techniques are primarily used in this project, Local Binary Pattern and Local Quinary Pattern.

Local Binary Pattern

LBP is a textural image descriptor proposed by Ojala et al. [41] that can define the local spatial structure and the local contrast of the image. LBP compares a pixel with its surrounding neighbors in an image by thresholding the neighbor into a binary pattern. First, a set of neighboring pixels are considered for each pixel in the image. The intensity of the central pixel is then compared with the intensity of the neighboring pixels. If the intensity of the neighboring pixel is larger than the central pixel, then the value for that neighbor in the corresponding extracted binary pattern is 1, otherwise it is 0. Finally, a binary weighted sum of the values is obtained, resulting in the LBP value. The equations to calculate the LBP value is shown below.

$$LBP_{P,R} = \sum_{p=0}^{P-1} f(v_p - v_c)2^p \tag{2.1}$$

$$f(x) = \begin{cases} 1, & \text{if } v_p - v_c > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

where P denotes the number of neighbors of the central pixel c, v_c denotes the intensity of the central pixel and v_p denotes the intensity of the p^{th} neighbor. The $f(x)$ function is used to calculate the binary values and is calculated according to Equation (2.2). A more detailed example of calculating LBP value is shown in the figure below.

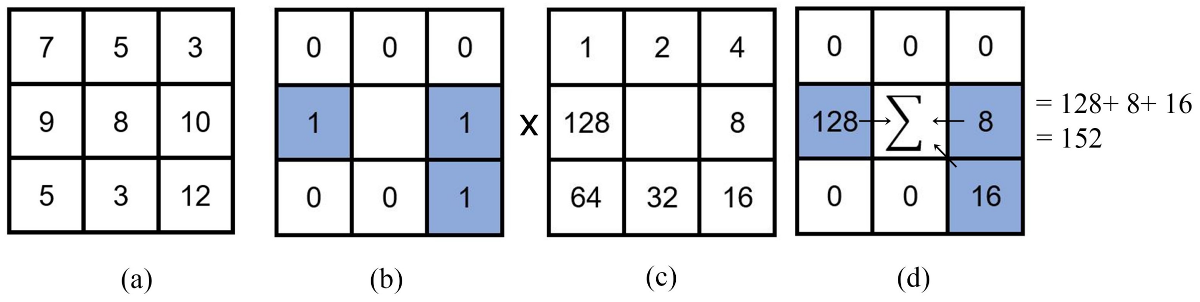

Figure 2.2: Illustration of the main LBP steps.

Fig. 2.2(a) displays a sample of a square neighborhood in the size of 3 by 3. The number of neighbors for this neighborhood is 8. The intensity of the neighbor pixel are thresholded by the intensity of the central pixel according to Equation (2.2) to extract the binary pattern in Fig. 2.2(b). The binary pattern is multiplied by the weights given to the corresponding pixels Fig. 2.2(c) resulting in the values shown in Fig. 2.2(d). Finally, the values of the 8 pixels are summed to obtain the LBP value. The LBP value for the given example is $128 + 8 + 16 = 152$. After the LBP values are calculated for all pixels of the image, the histogram of all the LBP values is computed. The bins of histogram are used as features.

Later on, the LBP descriptor was extended to consider circular neighborhoods, called circular LBP. Circular LBP uses circular neighborhoods which allow any radius and number of pixels in the neighbourhood by interpolating the pixel values using bilinear interpolation [1]. When using circular LBP, Equation 2.1 still remains. If the coordinates of v_c is $(0, 0)$, then coordinates of v_p are $(Rcos(2\pi p/P), Rsin(2\pi p/P)$ where R denotes the value of the radius, p denotes the p^{th} neighbor and P denotes the total number of neighbors being considered. In this project, the circular LBP is used. Fig. 2.3 shows an example of a circular neighbourhood with 8 sampling points on a circle of radius 2.

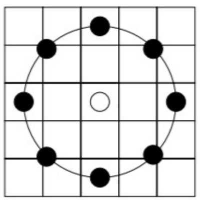

Figure 2.3: A circular neighborhood with 8 sampling points on a circle of radius 2. The pixel values are bilinearly interpolated whenever the sampling point is not in the center of a pixel.

A useful extension to the original operator known as the uniform LBP, which can be used to reduce the length of the feature vector [52]. Instead of using all values of the pixel, only 59 LBP values are considered. Using uniform patterns, the length of the feature vector for a single cell reduces from 256 to 59. The 58 uniform binary patterns correspond to the integer values 0, 1, 2, 3, 4, 6, 7, 8, 12, 14, 15, 16, 24, 28, 30, 31, 32, 48, 56, 60, 62, 63, 64, 96, 112, 120, 124, 126, 127, 128, 129, 131, 135, 143, 159, 191, 192, 193, 195, 199, 207, 223, 224, 225, 227, 231, 239, 240, 241, 243, 247, 248, 249, 251, 252, 253, 254 and 255. The 59^{th} bin contains all non-uniform patterns. Using uniform LBP greatly reduces the size of the feature vector when it comes to higher number of sampling points.

Local Quinary Patterns

LQP, which is also known as elongated quinary patterns, has been widely used in many applications such as mammogram analysis [5], facial expression recognition [54] and biomedical indexing [15]. It is known to be more robust with noise and large illumination variations [51].

LQP is based on a five level scale for encoding the local gray-scale difference proposed by Loris Nanni et al.[37], who suggested to use a five-value encoding in order to obtain a more robust descriptor. As opposed to the standard LBP which uses the difference between the intensity of the central pixel and that of the neighboring pixels to create a binary pattern, LQP uses the difference between the intensity of the central pixel v_c and that of the neighboring pixels v_p, and two fixed thresholds (t_1 and t_2) to create four binary patterns. The LQP variant assumes five values shown in Equation (2.3) instead of two values in standard LBP shown in Equation (2.2). The equations to calculate the LQP value is shown below.

$$f(v_c, v_p, t_1, t_2) = \begin{cases} 2, & v_p \geq v_c + t_2 \\ 1, & v_c + t_1 \leq v_p < v_c + t_2 \\ 0, & v_c - t_1 \leq v_p < v_c + t_1 \\ -1, & v_c - t_2 \leq v_p < v_c - t_1 \\ -2, & \text{otherwise} \end{cases} \tag{2.3}$$

The quinary pattern is split into four binary patterns according to the following binary function.

$$b_c(x) = \begin{cases} 1, & x = c \\ 0, & \text{otherwise} \end{cases}, c \in \{2, 1, 0, -1, -2\} \tag{2.4}$$

Using Equation (2.4), the four binary patterns are respectively obtained considering $c = 2$, $c = 1$, $c = -1$ and $c = -2$.

1	-2	1
-1		2
1	-1	0

Figure 2.4: Quinary pattern

(a)

0	0	0
0		1
0	0	0

(b)

1	0	1
0		0
1	0	0

(c)

0	0	0
1		0
0	1	0

(d)

0	1	0
0		0
0	0	0

Figure 2.5: Binary pattern for (a) $c = 2$, (b) $c = 1$, (c) $c = -1$, (d)$c = -2$

Fig. 2.4 displays a quinary pattern extracted from a square neighborhood in the size of 3×3 according to Equation (2.3) using two predefined thresholds (t_1 and t_2) and the intensity of the neighboring pixels and central pixel. The quinary pattern is then divided into four binary patterns shown in Fig. 2.5 using Equation (2.4). Finally, the binary patterns is multiplied by the weights given to the corresponding pixels as shown in Fig. 2.2(c), similar to the LBP operator, and finally summed in order to obtain four LQP values. For example, the LQP value for Fig. 2.5(a) will be 8.

After the four LQP values are obtained for all pixels of the image, four histograms are computed based on the four LQP values. The histograms are concatenated and the resulting histogram is then used as features for image classification.

2.4 Related Work

2.4.1 Image Classification using Canonical Machine Learning Methods

Many popular supervised machine learning techniques such as Random Forest and Support Vector Machines have been used in many image classification tasks with promising results. The process often involves manually choosing, extracting and processing features before feeding them into the classifier.

In 1999, Chapelle et al. [11] proposed a method that chooses support vector machine (SVM) as its classifier because of its high generalization performance, even with high dimensional input space. The inputs are color histograms in the hue-saturation-value space extracted from images. The model produced a good classification performance with error rates as low as 11%. Szummer et al. [50] introduced a method that uses the k-nearest neighbour method as the classifier using a range of different features as inputs. Each of the features represents the image's color, texture and frequency information. The best classification results were generally obtained by combining color features with texture features. However, it was difficult to determine the suitable combination of features or the usage of any single features with k-nearest neighbour to produce consistent results.

Bosch et al. [10] proposed a method that uses automatic selection of the regions of interest in training, extract features that represents shape and appearance of the image and use random forests to perform classification. The result showed comparable performance over other image classification method using SVM. The use of automatic region detection provided a significant benefit to the performance.

Nayak et al. [38] presented presents an automated system for brain magnetic resonance (MR) image classification that combines both AdaBoost and Random Forests. The dimension of features were reduced from 1024 normalized features to 13 significant features using probabilistic principal component analysis. The method yielded superior performance compared to all the existing methods such as k-nearest neighbor, neural network and SVM.

2.4.2 Image Classification using Genetic programming

Over the years, GP has been used in various image classification problems such as pulmonary nodule detection [12] and remote sensing classification [16]. Kobashigawa et al. [29] showed that GP achieved better results compared to neural networks, even with the increase in problem difficulty levels.

One approach of image classification is using GP to perform feature extraction due to its ability to manipulate features without any domain knowledge. Experimental results have shown that the performance was able to outperform a human-designed extraction system. Shelton et al. [46] proposed a GP method to extract features using uniform LBP that evolves the number, position and size of regions for feature extraction. The result showed that using 36% of the image for feature extraction gained a higher accuracy than of that of using a standard LBP approach that covered the entire image.

Some earlier works involve using previously extracted image features to evolve a program that mathematically combine them to classify the images. In 1999, Zhang and Ciesielski [61] proposed a GP method that used a domain independent image feature extraction method (simplified as FeEx in this paper) and evolve a GP program based on these features for object detection. The performance of this approach was able to outperform a neural network-based method. Oechsle and Clark [39] proposed a GP method that evolves both a feature extraction system and a classification system in two separate evolution. However, these techniques often require domain specific knowledge and human intervention.

To address the limitation of requiring human intervention, GP-based techniques that perform feature extraction and classification in a single program tree have been proposed by several researchers. Atkins et al. [8] proposed a three-tier GP (3TGP) method to connect the feature extraction and classification steps into one single program tree. This allows the program to operate directly on raw pixels rather than pre-defined features. The three tiers consist of image filtering tier, aggregation tier and classification tier. However, 3TGP has some limitations in the image filtering tier which were addressed by Al-Sahaf et al. in [3].

Al-Sahaf et al. introduced a two-tier GP (2TGP) method which simplified the 3TGP method to only consist of two-tiers: aggregation tier and classification tier. The features discovered have a higher predictive power than manually extracted features. The extension of 2TGP proposed in [4] was also able to outperform 3TGP by extracting features from image regions instead of the entire image.

Since 2TGP and 3TGP were only able to extract low level features, Lensen et al. [33] developed another GP approach which extracted high-level features using functions based on HOG (GP-HOG). HOG descriptor was used as it has a high ability to describe specific image features including shape and appearance. The new function was able to produce more advanced features and increase the classification performance.

Bi et al. [9] proposed a multi-layer GP method (MLGP) to also perform high-level feature extraction and classification using 11 image-related operators. The operators helped to reduce noise, increase contrast or detect edges of an image. The result showed that MLGP was not only able to achieve significantly better results than other GP methods, but also more stable performance. MLGP dealt with complex image classification tasks such as texture image or scene image classification with good results without any prior domain knowledge.

Al-Sahaf et al. [2] proposed a GP method that manually combine the detected keypoints using an operator inspired by LBP that is also rotation-invariant. While the experiment considers evolving the two parameters of radius and neighboring pixels, there are many LBP variants that have outperformed the standard LBP approach. Experimental results in [37] showed that the LQP operator has a high ability to classify textural images in comparison with other LBP variants, there are some limitations in the existing LQP operator.

2.5 Summary of Limitations of Existing Work

The methods of image classification using canonical machine learning techniques typically involve two processes - feature extraction and classification. This causes the performance to be highly dependent on the feature extraction method and the classification method for dealing with different image classification tasks. Many of these methods do not generalize well with other data sets as the most effective feature extraction methods varies with different tasks.

Most of the existing methods of image classification using GP involve the use of image operators using predetermined parameters. These image operators are not tuned with various parameter values in order to adapt its operation to work with a wide range of problems. For example, MLGP uses the LBP operator with a fixed radius value for all tasks, which may not be effective for certain classification tasks. GP-HoG also uses the HoG image operator with a fixed number of bins. These existing image operators may not use the appropriate choice parameters in the final program, which could potentially affect the performance. Manually tuning the parameters require human intervention that in many cases can be a very difficult task. Evolving the parameters used in these image operators can aid in choosing parameters that are tailored towards different data sets.

Furthermore, one important drawback of the LQP method is the use of the static thresholds. These thresholds need to be set by the user as input parameters and are usually determined via trial and error. The accuracy of the LQP analysis is therefore very sensitive to the threshold values. Each class of texture may require different threshold values in order to get a good performance. To determine threshold values for separate problems requires both time and domain knowledge which is not always available.

Therefore, the methods proposed in this project is designed to investigate the usage of LBP and LQP in GP while evolving the parameters involved for image classification.

Chapter 3

The LQP-GP Approach

3.1 Introduction

This chapter presents a novel approach of using GP with LQP to extract and construct high-level features for image classification called LQP-GP. This new method is tested on two datasets from different domains, and compared to existing GP methods. The new GP program structure, the new function set, the new terminal set and the fitness function are described here.

3.2 Chapter Goals

This chapter aims to investigate how LQP can be used with GP for image classification. This will be achieved with the following objectives:

1. Design a new function set and a terminal set for a GP program to determine suitable parameters for LQP to extract features. GP has shown promising performance using both low-level and high-level image features. However, literature using GP to construct high-level image features are limited.
2. Design a new program structure to select regions for feature extraction with the LQP algorithm. The standard approach of LQP extracts features by considering the entirety of an image within one run of the algorithm. This approach may not work well in images that do not have useful features throughout the whole image.
3. Evaluate LQP-GP by comparing its performance against other GP methods on a range of datasets.
4. Analyse the results and provide discussion regarding the performance of the LQP-GP.

3.3 Program Structure

A strongly-typed GP [36] is used to create the new LQP-GP program tree structure in order to introduce restrictions on the inputs and outputs of the different nodes. The tree can be virtually divided into five layers, i.e Input, Region Detection, Feature Extraction, Feature Construction, and Classification, inspired by the program structure designed in [9]. This program structure is presented in Fig. 3.1, where each layer is shown in a different colour.

1. **Input**
   Images and randomly generated constants are fed from this layer to the GP method.

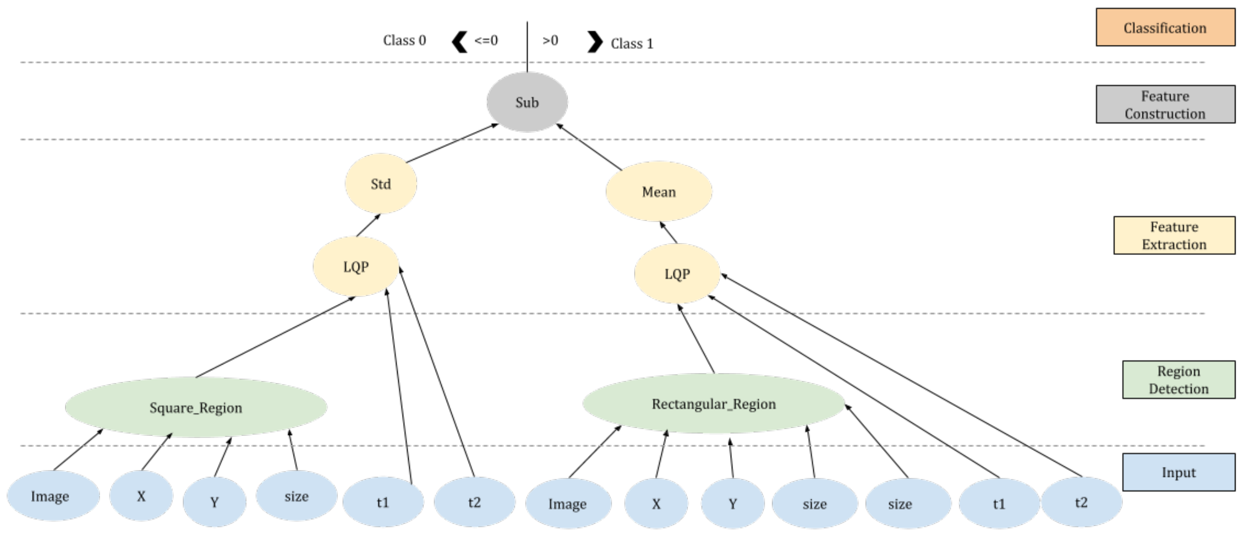

Figure 3.1: Example LQP-GP program structure

2. **Region Detection**
   Detects square or rectangular region in an image located by the program that effectively describes the relevance of the image.

3. **Feature Extraction**
   The layer in which the LQP operator is applied or pixel statistics are calculated to extract image features from detected region. The thresholds for LQP are randomly generated constants.

4. **Feature Construction**
   The extracted features by LQP are further constructed to form a new high-level feature.

5. **Classification**
   The top layer of the program, which assigns a class label to the input image according the value of the constructed feature and a threshold.

The program structure of the proposed LQP-GP method is comprised of the five layers in a bottom-up manner.

3.4 Functions and Terminals

3.4.1 Terminals

The terminal set shown in Table 4.1 includes parameters for functions in region detection layer and feature extraction layer. These parameters include Image, X, Y and size. In addition to the input image parameters, there are also two additional terminals that provide the thresholds for the LQP operator in the feature extraction layer. They are t1 and t2. The values of size, X, Y, t1 and t2 are randomly generated and evolve during evolutionary process according to their corresponding range stated in the table.

Table 3.1: Terminal set

Name	Output	Details
Image	Image	The image being classified
size	size	Random integer value in [3, min(min_{width}, min_{height})]
X	X	Random integer value in [0, $Image_{width}$] that provides the horizontal location in the image
Y	Y	Random integer value in [0, $Image_{height}$] that provides the vertical location in the image
t1	t1	Random integer value in [2, 8] that provides the parameter of the lower threshold in the LQP operator
t2	t2	Random integer value in [9, 15] that provides the parameter of the upper threshold in the LQP operator

3.4.2 Functions

Functions of region detection layer

Table 3.2 shows the functions used in the region detection layer. These two functions detects a square or rectangular region with a suitable size and position by taking arguments from the terminal set. If the area in the detected region is beyond the boundaries of the image, only the area inside the input image is used as the detected region.

Table 3.2: Functions of region detection layer

Name	Input	Output	Details
Square_Region	Image, X, Y, size	region	Detects a square region at a selected position in an image with a suitable size
Rectangular_Region	Image, X, Y, size, size	region	Detects a rectangular region at a selected position in an image with a suitable size

Functions of feature extraction layer

Table 3.3 shows the functions used in the feature extraction layer. The LQP operator is a new function, inspired by the LQP algorithm. The standard approach of LQP computes a histogram of 256 bins for each quinary pattern ($c = 2$, $c = 1$, $c = -1$ and $c = -2$) and concatenate the resulting four histograms into one histogram of $256 * 4 = 1024$ bins. The LQP operator used in LQP-GP differs from the standard LQP approach in which a histogram of 32 bins is calculated for each quinary pattern instead. Each bin contains of a range of 8 gray values of the image. The first bin contains the frequency of the gray values from 0 to 7 and the second bin contains of that from 8 to 15 and so on. Consequently, the final histogram have a total of $32 * 4 = 128$ bins. This helps to reduce the number of features in order to reduce computational cost.

Another important change from the standard LQP approach is that the final histogram is normalized. This reduces the search space to ensure that images with high gray values is able to output a negative number in the classification layer.

Aside from the LQP operator, statistical information are extracted from the resulting histogram. They are standard deviation, mean, minimum and maximum.

The next function is distance, which finds the Euclidean distance between two histograms outputted from two LQP operators. This produces a float value which gives a measure of how dissimilar the two histograms are.

Table 3.3: Functions of feature extraction layer

Name	Input	Output	Details
LQP	region, t1, t2	histogram	Performs the LQP operator on a region
Std	histogram	float	Calculate the standard deviation of a histogram
Mean	histogram	float	Calculate the mean of a histogram
Min	histogram	float	Calculate the minimum value of a histogram
Max	histogram	float	Calculate the maximum value of a histogram
Distance	histogram, histogram	float	Calculate the distance between two histograms

Functions of feature construction layer

There are 3 arithmetic function used for this layer. They are Sub($-$), Add($+$) and Mul($\times$) which subtracts, adds and multiplies respectively, two floating-point numbers and returns a floating-point number.

Table 3.4: Functions of classification layer

Name	Input	Output
Sub ($-$)	float, float	float
Add ($+$)	float, float	float
Mul ($\times$)	float, float	float

Functions for classification layer

The function for this layer is that if the output from the feature construction layer is positive, the class label for the input image is 1 (class 1), otherwise the class label is 0 (class 0).

3.5 Fitness Function

Fitness function is a crucial element of GP. In binary image classification, the commonly used fitness function is the classification accuracy. The equation to calculate the classification accuracy is in Eq. 3.1.

$$\frac{TP + TN}{TOTAL} \times 100\% \tag{3.1}$$

where TP denotes the total number of True Positives, which are positive instances classified as positives, TN denotes the total number of True Negatives, which are negative instances classified as negative, and TOTAL denotes the total number of instances in the dataset.

A GP program with a higher accuracy would have a better fitness value and is more likely to participate in the next generation. The proposed method works by maximizing the fitness function / classification accuracy.

3.6 Overall Algorithm

The overall algorithm process is shown in Figure 3.2. Each image undergoes region detection to find a local region that best describes the relevance of the image. Then, the local region undergoes feature extraction using the LQP operator to output a histogram of 128 bins. The statistical information such as standard deviation, mean, minimum or maximum value from the resulting histogram or Euclidean distance between histograms are computed. The results then undergoes arithmetic operation such as subtract, add or multiply to construct higher-level features. The constructed features are then classified with class label 1 when the output of the float is positive or class label 0 otherwise.

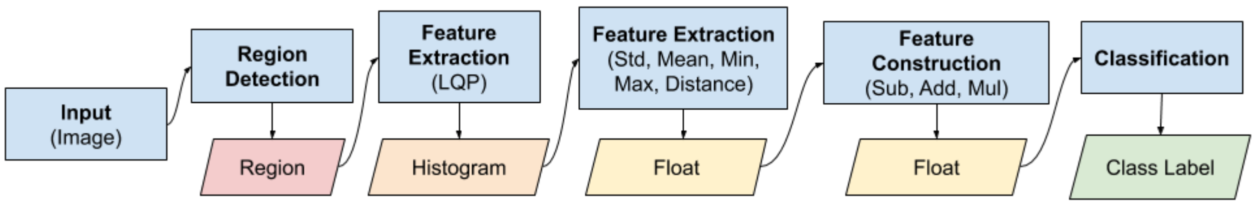

Figure 3.2: Flowchart of LQP-GP approach

3.7 Experiment Design

3.7.1 Datasets

Two different datasets are used to evaluate the performance of the proposed method. They are **JAFFE** [34] and **BIRDS** [55]. Each data set is split into a training set, a validation set and a test set, having 50%, 25%, 25% images respectively.

The training set is used to evolve programs to search for an optimal individual among the space of all individual using GP. The validation set is used to measure its generalization capability and select the best individual for testing. The test set is used to measure the performance of the best individual.

3.7.2 Baseline methods

In order to evaluate the performance of the proposed method, five GP-based methods are evaluated with the same data set. The five GP-based methods comprise of MLGP [9], 2TGP [3], FeEx+GP [61], Hist+GP and uLBP+GP. The Hist+GP method extracts 64 histogram features and and the uLBP+GP method extracts 59 uniform LBP histogram features. Both methods use the extracted features as input for GP. Fig. 3.3 and 3.4 shows example images from each data set.

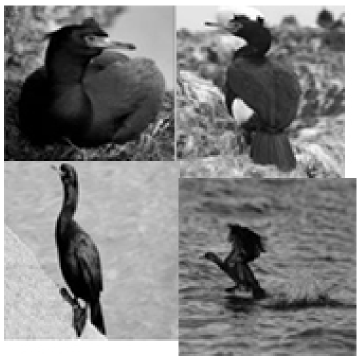

Figure 3.3: BIRDS dataset: Pelagic cormorant vs Red-faced cormorant

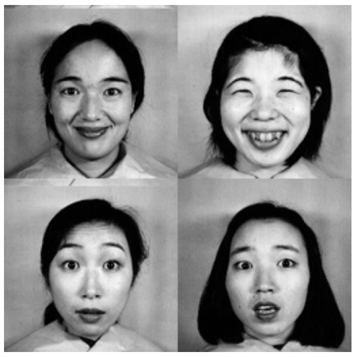

Figure 3.4: JAFFE dataset: Happy vs Surprised

3.7.3 Parameter Settings

The proposed method is implemented in Python based on the DEAP (Distributed Evolutionary Algorithm in Python) [20] package. Each experiment is run independently 30 times using different seeds for each method for each data set.

Table 3.5: GP runtime 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

The parameter settings in all the GP methods are the same as listed in Table 3.5. During the evolutionary process, each individual is evaluated at each generation on the training set. The best individual on the training set is evaluated on the validation set in order to avoid overfitting. The validation set selects a configuration of the runtime that maximizes the generalization performance.

3.8 Results and Discussions

Tables 3.6 and 3.7 shows the test results in terms of maximum, mean and standard deviation of classification accuracy obtained by LQP-GP and the other five GP methods on two data sets in 30 runs.

On the JAFFE data set, LQP-GP obtains a better performance than FeEx+GP, Hist+GP and uLBP+GP. However, it is outperformed by MLGP and 2TGP. On BIRDS data set, LQP-GP obtains the lowest maximum accuracy out of all GP methods, while achieving a slightly higher average accuracy than 2TGP and Hist+GP. This shows the LQP-GP may not be suitable for object recognition (BIRDS), but performs relatively well for facial recognition (JAFFE).

Table 3.6: Classification accuracy (%) of all the GP methods on JAFFE

JAFFE			
	Max	Average	St. dev
LQP-GP	95.00	75.00	12.89
MLGP	100.00	86.50	7.89
2TGP	100.00	79.00	12.55
FeEx+GP	95.00	69.00	11.99
Hist+GP	50.00	50.00	0.00
uLBP + GP	75.00	51.67	9.94

Table 3.7: Classification accuracy (%) of all the GP methods on the BIRDS

Birds			
	Max	Average	St. dev
LQP-GP	60.71	53.90	4.06
MLGP	71.43	61.67	6.45
2TGP	67.86	51.79	7.70
FeEx+GP	64.29	54.64	5.77
Hist+GP	78.57	51.67	9.53
uLBP + GP	71.43	60.36	7.57

Based on the experimental results, the proposed LQP-GP approach is not able to compete with some of the baseline GP methods. The performance of LQP-GP on the JAFFE data set is rather unstable and fluctuates a lot, which can be seen in the standard deviation value. While it is able to exceed FeEx+GP, Hist+GP and uLBP+GP, it is not able to surpass the more recent GP approaches for image classification such as 2TGP and MLGP. This indicates that 2TGP and MLGP are able to construct key discriminatory features with their function sets that LQP-GP is unable to replicate. Besides, JAFFE is a relatively small dataset, which shows that LQP-GP is unable to sufficiently learn from its limited training instances, as compared to 2TGP and MLGP.

This shows that LQP-GP may need to be further improved. However, due to inadequate computational power to handle the complexity of LQP operator and time constraints, this may be done in the future.

Fig. 3.6 displays four binary images created by each quinary pattern using Equation 2.3 and 2.4 on one of the instance in the BIRDS data set shown in Fig. 3.5. The binary images for quinary pattern $c = 2$ and $c = -2$ are relatively similar looking in which they extracted the edges of the bird in the image. On the other hand, The binary images for quinary pattern $c = 1$ and $c = -1$ also look relatively similar in which they extracted the textural information of the image.

The low performance of LQP-GP may be due to the presence of a large number features presented by LQP-GP of 128 bins from each LQP operator over a small number of training instances. This phenomenon may have contributed to the occurrence of overfitting. Not all of the features extracted by the LQP operators are relevant. As shown in the similarities between the binary images in Fig. 3.6, there is a considerable amount of redundant and duplicate features extracted by LQP. This may have caused the resulting feature from feature construction to be less useful for classification.

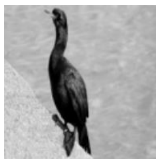

Figure 3.5: Raw image before being processed by the LQP operator

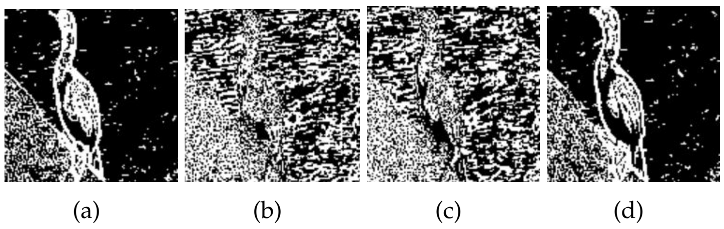

Figure 3.6: Binary images returned by the LQP operator for each quinary pattern (a) $c = 2$, (b) $c = 1$, (c) $c = -1$, (d)$c = -2$

Apart from that, the average training time of LQP-GP is approximately 12 hours on JAFFE and 18 hours on BIRDS, likely due to the large amount of computation required by the LQP operator. The computation of the quinary patterns is the most computational heavy part of the algorithm as it considers all the neighborhood pixels for each central pixel in the image.

3.9 Chapter Summary

This chapter introduced a new approach called LQP-GP which aimed to improve the performance of image classification by combining the LQP algorithm with GP techniques. The approach is shown to have limited success. While LQP-GP has produced decent results, its average training time is extremely high due to the complex nature of algorithm. Furthermore, using LQP as the only feature extraction technique may not be sufficient to derive useful features for subsequent learning. Steps to use functions with lesser computation power such as LBP as well as other image operators are considered in the next chapter to address these issues.

Chapter 4

The uLBP-GP approach

4.1 Introduction

This chapter presents a novel approach of using GP to extract and construct high-level features using uniform LBP (uLBP) called uLBP-GP. A new GP program design with uLBP is proposed in order to improve the performance of image classification. The main components of the newly proposed methods, and results on various datasets are presented.

While the LQP-GP approach proposed in Chapter 3 produces good general features, fine tuning the features further would allow more accurate classification. Furthermore, the LQP operator is computationally expensive and extracts a large number of redundant features.

The uLBP operator is an exceptional visual descriptor that has been widely used in various application with promising results. uLBP is highly discriminative and its invariance to monotonic gray level changes and computational efficiency make it suitable for demanding image classification tasks such as texture classification and facial recognition. By using the concept of uniform pattern, it can also filter out noises [56].

4.2 Chapter Goals

This chapter aims to develop a new approach which uses LQP's predecessor, uLBP. The new approach also adds another layer to the GP program that uses various image operators to process the input image before performing feature extraction using the uLBP operator or calculating pixel statistics. As before, GP is still used for region selection, feature extraction and classification. The specific objectives of this chapter are as follows:

1. Propose a new program structure to investigate the effectiveness of adding an image processing layer to a GP tree to remove image discrepancies such as noises or enhance image by improving contrast before using uLBP for feature extraction.

2. Combine the new layer with the region detection approach (as in Chapter 3) to perform region detection, feature extraction and classification in one tree.

3. Evaluate this new approach against general classifiers and GP classifiers on six datasets.

4. Analyse the programs of some high-performing individuals to understand how they are able to achieve high classification accuracy.

4.3 Program Structure

In order to combine various stages of image classification into a single tree, a strongly-typed GP [36] was employed to enforce a tiered structure. The tree can be virtually divided into six layers, i.e Input, Region Detection, Image Processing, Feature Extraction, Feature Construction and Classification. The program structure is presented in Fig. 4.1 where each layer is shown in a different colour.

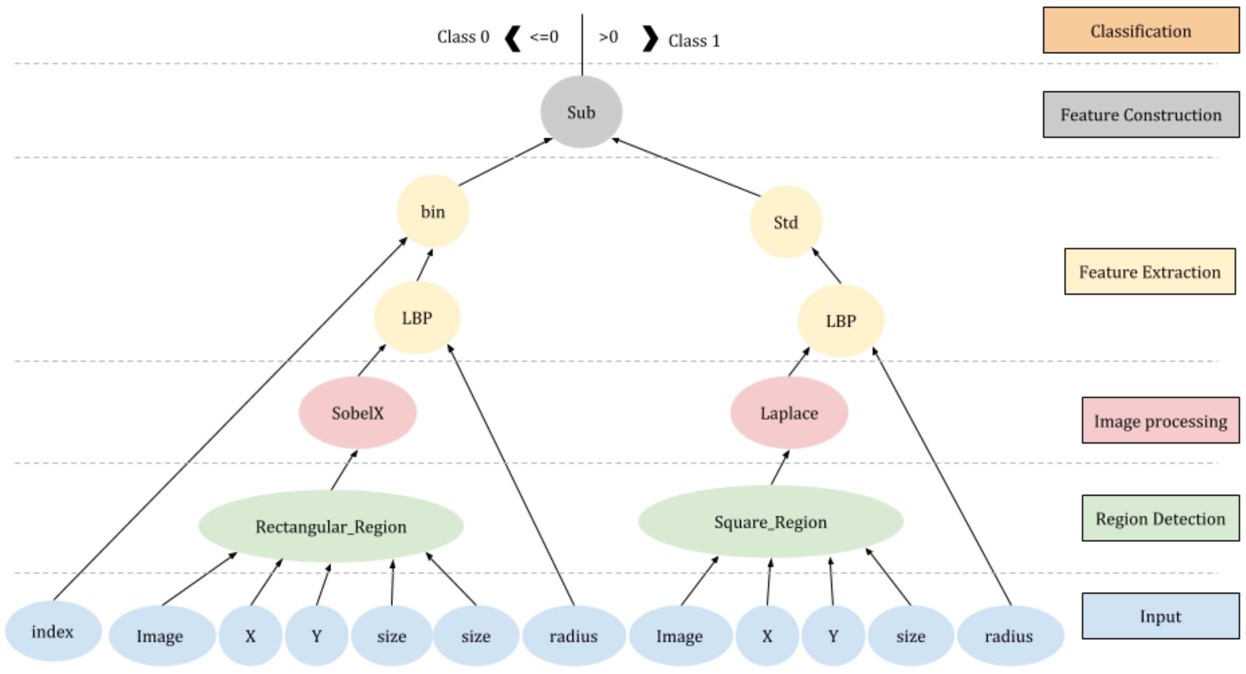

Figure 4.1: Example uLBP-GP program structure

1. **Input**
   Images and randomly generated constants are fed from this layer to the GP method. Note that some of the constant parameters are used by successive layers other than the region detection layer, i.e *radius* is a parameter used by LBP.

2. **Region Detection**
   Detects square or rectangular region in an image that effectively describes the relevance of the image.

3. **Image processing**
   The layer in which the various image operators are applied to further enhance the input image by removing noises or increase contrast to aid the feature extraction process.

4. **Feature Extraction**
   The layer in which the uLBP operator is applied or pixel statistics are calculated to extract image features from detected regions.

5. **Feature Construction**
   The extracted features are further constructed into a new high-level feature.

6. **Classification**
   The top layer of the program, which assigns a class label to the input image according the value of the constructed feature and the predetermined threshold.

   Note: While the term image processing and feature extraction are often used interchangeably, in the uLBP-GP approach, they are primarily different in terms of the type of outputs returned. The functions in the image processing layer returns an image of the same size, while feature extraction layer returns either a histogram or *float*.

The program structure of the proposed uLBP-GP method is constructed according to the six layers in a bottom up manner. The tree-based representation has operators which consists of internal nodes and terminals consists of leaf nodes.

4.4 Functions and Terminals

4.4.1 Terminals

There are six types of terminals for this layer, which represents the input image and the constant parameters used in the proposed uLBP-GP method. The parameters are used for functions in the region detection and feature extraction layers. The values of size, X, Y, index and radius are randomly generated and evolve during the evolutionary process according to their corresponding range detailed in Table 4.1.

Table 4.1: Terminal set

Name	Output	Details
Image	Image	The image being classified
size	size	Random integer value in [20, 70]
X	X	Random integer value in [0, $Image_{width}$] that provides the horizontal location in the image
Y	Y	Random integer value in [0, $Image_{height}$] that provides the vertical location in the image
radius	radius	Random integer value in [1, 5] that provides the parameters of the radius of the neighbourhood in the LBP operator
index	index	Random integer value in [0, 58] to specify the bin of a histogram to be used

4.4.2 Functions

Functions of region detection layer

The same function set for the region detection layer is used in the LQP-GP method. The details of the function set is defined in Table 3.2.

Functions of image processing layer

Table 4.2 shows the functions used in the image processing layer. These are 5 image-related operators, which are similar to the operators used in [9] in the feature extraction layer. The **Hist_Eq** operator is designed to increase contrast and equalize the histogram of an image.

The **Gau1** operator reduces noise. **SobelX** and **SobelY** perform edge detection by calculating the gradient of image intensity along the X and Y axis respectively. Finally, the **Laplace** operator is used to identify and highlight fine edges. The type returned from these functions is *processed* which denotes the processed image using one of the operators listed in Table 4.2.

Table 4.2: Functions of image processing layer

Name	Input	Output	Details
Hist_Eq	region	processed	Histogram equalization
Gau1	region	processed	Gaussian smooth filter with σ=1
SobelX	region	processed	Sobel filter along X axis
SobelY	region	processed	Sobel filter along Y axis
Laplace	region	processed	Laplacian filter

Functions of feature extraction layer

LBP is used in the function set for describing the shape and texture information of an image. In the LBP operator, the radius is set as a terminal and the number of neighbours is set to the value of radius$\times$8. After the LBP operator computes a histogram of 59 bins representing the 59 uniform binary patterns, the resulting histogram is normalized. This reduces the search space in order to ensure that images with high gray values are able to return a negative number in the classification layer.

Another operator for this layer is **Distance** which computes the Euclidean distance between two histograms returned by two LBP operators. The function produces a float value which gives a measure of how dissimilar the two histograms are.

The **bin** function extracts a value at a specific bin in the LBP histogram (denoted by the index) to return a float value.

Finally, statistical information such as standard deviation, mean, minimum and maximum are extracted from the resulting LBP histogram from LBP operator. All functions except LBP and bin are similar to the function set in the feature extraction layer in the LQP-GP method.

Table 4.3: Functions of feature extraction layer

Name	Input	Output	Details
LBP	processed, radius	histogram	Performs the uniform LBP operator on a processed image region and calculate the normalised LBP histogram.
Distance	histogram, histogram	float	Calculate the Euclidean distance between two histograms
bin	index, histogram	float	Returns the float at histogram[*index*]
Std	histogram	float	Calculate the standard deviation of a histogram
Mean	histogram	float	Calculate the mean of a histogram
Min	histogram	float	Calculate the minimum value of a histogram
Max	histogram	float	Calculate the maximum value of a histogram

Functions of feature construction layer

There are four arithmetic function used for this layer. They are **Sub**($-$), **Add**($+$), **Mul**($\times$) and **Protected_Div**($\div$), which subtracts, adds, multiplies and divides respectively, two floating-point numbers and returns a floating-point number. Protected_Div returns a value of zero if either of the float values are equal to zero to prevent any floating-point number overflow errors.

In addition to that, there are also trigonometric functions such as **Cos** and **Sin** which relates the lengths of the sides of a triangle to the cosine or sine of one of its angles respectively. Finally, the **Neg** function simply negates a floating-point number to return a negated floating-point number.

Table 4.4: Functions of feature construction layer

Name	Input	Output
Sub ($-$)	float, float	float
Add ($+$)	float, float	float
Mul ($\times$)	float, float	float
Protected_Div ($\div$)	float, float	float
Neg	float	float
Cos	float	float
Sin	float	float

Functions of classification layer

The same function is used in the LQP-GP approach. This functions of the classification is defined in Section 3.4.2.

4.5 Fitness Function

The same fitness function is used in the LQP-GP approach. This fitness function is defined in Section 3.5.

4.6 Overall Algorithm

The overall algorithm process is shown in Figure 4.2. Each image is undergoes region detection to find a local region that best describes the relevance of the image. Then, the local region undergoes image processing using various image operators to output a processed region. Then, the processed region undergoes feature extraction using the uLBP operator to output a histogram of 58 bins. The statistical information such as standard deviation, mean, minimum or maximum value from the resulting histogram or Euclidean distance between two histograms or a selected bin value from this histogram is calculated.

The result then undergoes mathematical operations such as subtract, add, multiply, protected division, negation, cosine or sine to construct higher-level features. The constructed features are then classified with class label 1 when the output of the float is positive or class label 0 otherwise.

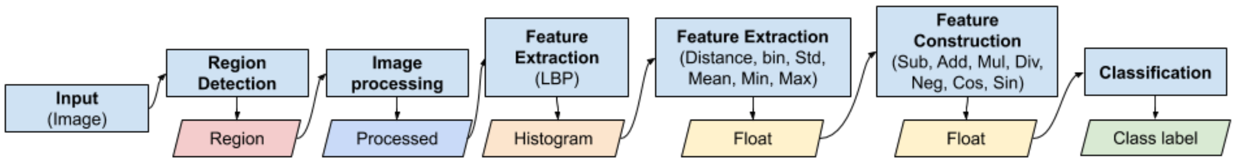

Figure 4.2: Flowchart of uLBP-GP approach

4.7 Experiment Design

4.7.1 Datasets

Six different data sets of varying difficulties are employed in the experiments. The proposed method is evaluated using three texture image datasets (OutexTC, KyWiRo, KTH-TIPS2) to test its performance on texture classification. Two face datasets (FEI1, JAFFE) are employed to evaluate its performance in binary classification of facial expression. One scene dataset (Scene) classification is used to evaluate its performance in scene classification.

The image datasets are gray-scale images, in which each pixel is ranging between 0 (black) and 255 (white). All the images from each data set have been preprocessed accordingly. Each data set is split into the training set, the validation set and the test set, having 50%, 25%, 25% images respectively. Fig. 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8 shows example images from each data set. Details of the data sets are listed in Table 4.5.

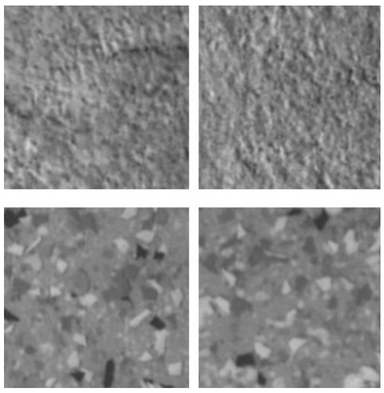

Figure 4.3: OutexTC dataset: Carpet vs Tile

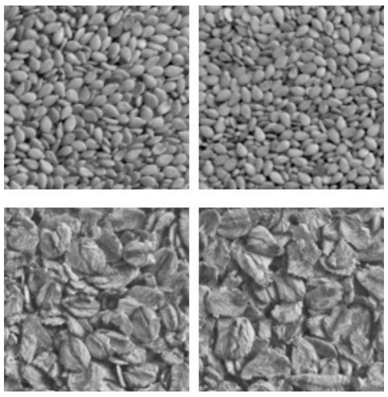

Figure 4.4: KyWiRo dataset: Sesame seeds vs Oats

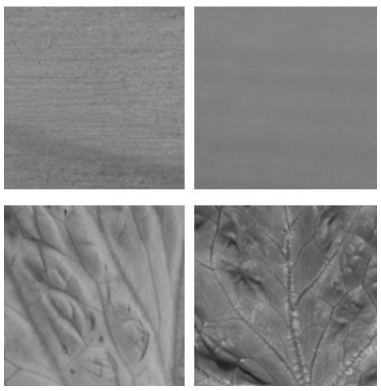

Figure 4.5: KTH-TIPS2 dataset: Wood vs Lettuce

OutexTC

This dataset was drawn from the Outex Texture Classification (OutexTC) [40] dataset for texture classification. The dataset was formed by extracting two classes of ”tile” and ”carpet” from the content of *Outex_TC_*00010 as the focus is on binary classification. Each class contains a total of 180 images, resulting in 360 images in total. The original images are all of the size 128 $\times$ 128 pixels and gray-scale, so there was no prior preprocessing required.

Kylberg With Rotation (KyWiRo)

KyWiRo dataset was collected from the Kylberg texture dataset [32], formed by extracting two classes, ”sesame seeds” and ”oats” and sampled to 128 $\times$ 128 pixels. The instances of

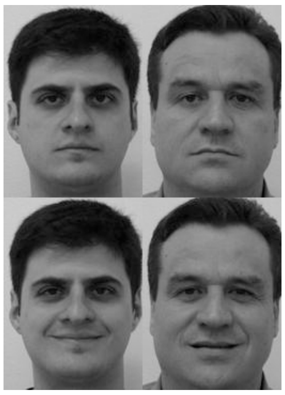

Figure 4.6: FEI1 dataset: Neutral vs Smiling

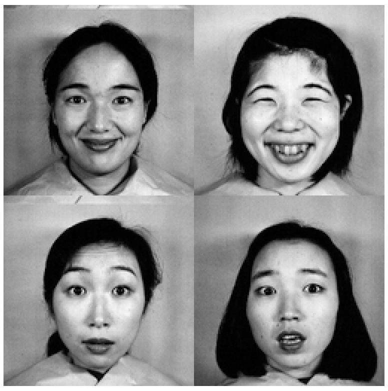

Figure 4.7: JAFFE dataset: Happy vs Surprised

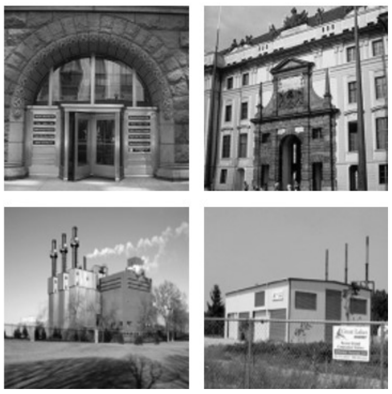

Figure 4.8: Scene dataset: Inside City vs Industrial

Table 4.5: Data set properties

Name	Size	Classes	Training set	Validation set	Test set
OutexTC	128 ×128	Tile	90	45	45
		Carpet	90	45	45
KyWiRo	128 ×128	Sesame seeds	960	480	480
		Oats	960	480	480
KTH-TIPS2	128 ×128	Wood	216	108	108
		Lettuce	216	108	108
FEI1	180 ×130	Neutral	50	25	25
		Smiling	50	25	25
JAFFE	128 ×128	Happy	10	10	10
		Surprised	10	10	10
Scene	128 ×128	Inside city	308	76	76
		Industrial	311	76	76

the with-rotation category are extended from the without-rotation instances by rotating each image in 12 angles around the center between 0°and 330°with a step size of 30°. This results in 160 × 12 = 1920 images for each class and 3840 images in total.

KTH-TIPS2

The KTH-TIPS2 dataset was collected from the KTH-TIPS database [21] for texture classification. The KTH-TIPS2 dataset extends the KTH-TIPS dataset by having an additional illumination conditions taken in varying poses. The dataset was formed by extracting two classes of ″wood″ and ″lettuce″. There are a total of 864 images, with an even distributon of wood and lettuce images. All images were converted from color to grayscale and sampled from 200 × 200 pixels to 128 × 128 pixels.

FEI1

The FEI1 was drawn from the FEI Face Database [53], comprised of frontal images of neutral and smiling facial expressions for face classification. All the images were resized from

360 × 260 pixels to 180 × 130 pixels. There are 200 full frontal face images, with an even distribution of neural and smiling facial expressions for each of the dataset.

JAFFE

The JAFFE dataset was drawn from the JAFFE database [34]. Out of 7 facial expression, two classes of ”happy” and ”surprised” were used to form the dataset. All images were resized from 256 × 256 pixels to 128 × 128 pixels.

Scene

The Scene dataset was drawn from the 15-Scene Image Dataset [6] for scene classification. Out of 15 classes, two classes of ”inside city” and ”industrial” were used to form the dataset. All images were sampled to 128 × 128 pixels.

4.7.2 Baseline methods

GP classifiers

In order to evaluate the performance of the proposed method, five GP-based methods are also evaluated with the same data set. The five GP-based methods comprise of MLGP [9], 2TGP [3], FeEx+GP [61], Hist+GP and uLBP+GP.

General Classifiers

Image classification can be treated as a typical classification problem by representing each pixel as a feature. For instance, the image in Figure 4.9 shows an image of 28 × 24 pixels, which equates to a total of 952 pixels (red lines dividing the pixels). These pixels can be represented as 952 features in which each feature corresponds to the intensity of the pixel.

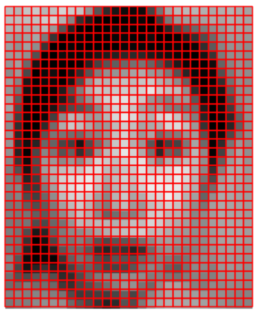

Figure 4.9: Sample features

These features are then used as input to train and test a classifier. The classifiers are deployed using the *scikit-learn* [43] package, a popular Python machine learning package. Classifiers were chosen from a range of paradigms to ideally give a broad range of results. The general classifiers used were: AdaBoost, Decision Trees, Nearest Neighbour, Naive Bayes and Random Forest. The five methods take the raw pixel values of each image as inputs and train classifiers for classification.

The goal of comparisons is to show whether uLBP-GP can achieve better performance than state-of-the-art methods.

4.7.3 Parameter Settings

The proposed method is implemented in Python based on the DEAP (Distributed Evolutionary Algorithm in Python) [20] package. Each training is run independently 30 times using different seed for each method for each data set.

The parameter settings in all the GP methods are similar to the parameters used in LQP-GP experiment as listed in Table 3.5.

4.8 Results and Discussions

This section shows the tabulated results and compares the performance in terms of testing accuracy and training time of uLBP-GP with the baseline methods.

The proposed method (uLBP-GP) was tested on the six datasets with a range of difficulties. Datasets from different domains were used to ensure that the proposed solution is domain-independent. The Wilcoxon rank-sum test with a 5% significance level is used to compare uLBP-GP with a baseline method on each dataset to test if the results are significantly different.

The results of each dataset is tabulated in a table. The table consists of the summary statistics for all the results from the classifiers (average, standard deviation and maximum for accuracy (%) and computational time on both training and testing). The symbols "$-$", "$+$" or "$=$" is used to denote if uLBP-GP is significantly worse, better or similar to the compared baseline method respectively in terms of testing accuracy. The first row of table shows the result of the proposed method, followed by the results of the GP classifiers in the next 5 rows and results of the general classifier in the last five rows.

Table 4.6: Summary statistics of all baseline methods on **OutexTC**

OutexTC								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	100.00	99.73±0.38	100.00	95.34±3.74	28763.71	17911.61±3808.71	0.25	0.17±0.05
MLGP	100.00	99.96±0.15	100.00	98.74±2.12−	321345.20	45962.09±70525.02	4.05	0.52±0.97
2TGP	97.70	86.16±7.38	86.60	72.18±10.24+	1488.20	755.42±411.90	0.02	0.01±0.01
FeEx+GP	92.70	87.38±2.98	87.70	79.62±6.83+	87.98	55.14±11.25	0.00	0.00±0.00
Hist+GP	100.00	99.61±0.88	100.00	97.77±2.46−	162.76	105.61±23.28	0.00	0.00±0.00
uLBP+GP	100.00	99.96±0.22	100.00	99.68±0.54−	251.35	153.05±42.28	0.00	0.00±0.00
AdaBoost	100.00	100.00±0.00	83.33	83.33±0.00+	26.13	24.16±1.01	0.30	0.16±0.03
Decision Trees	100.00	100.00±0.00	67.78	64.11±2.26+	1.64	0.96±0.16	0.01	0.01±0.00
Nearest Neighbour	53.33	53.33±0.00	53.33	53.33±0.00+	0.87	0.03±0.16	0.09	0.07±0.01
Naives Bayes	98.33	98.33±0.00	84.44	84.44±0.00+	0.76	0.10±0.13	0.09	0.06±0.01
Random Forest	100.00	99.59±0.36	94.44	87.30±2.67+	1.21	0.43±0.16	0.03	0.01±0.00

4.8.1 Compared with GP Classifiers

For the **OutexTC** dataset, the proposed method is able to significantly outperform 2TGP and FeEx+GP. This may be due to the images having different illumination settings and nine different rotation angles, causing the low-level features produced by these methods to be insufficient in classifying the images. Both 2TGP and FeEx+GP use only arithmetic operations without any high-level feature extraction techniques in their respective function

Table 4.7: Summary statistics of all baseline methods on **KyWiRo**

KyWiRo								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	99.70	96.12±3.74	99.00	94.74±4.68	328806.89	235362.58±55763.56	4.37	2.51±0.96
MLGP	100.00	97.84±1.54	100.00	96.95±1.90 =	937169.62	372319.03±340052.88	11.45	3.52±4.12
2TGP	99.80	93.28±3.07	98.60	91.66±3.26+	17586.99	5127.15±3148.09	0.27	0.06±0.05
FeEx+GP	90.20	87.97±1.22	89.70	87.72±1.18+	710.72	451.70±92.02	0.01	0.00±0.00
Hist+GP	100.00	99.60±0.22	100.00	99.52±0.23−	1448.11	1006.01±216.65	0.02	0.01±0.00
uLBP+GP	100.00	99.26±0.87	100.00	98.92±1.02−	2507.47	1467.81±394.85	0.03	0.02±0.01
AdaBoost	99.11	99.11±0.00	72.29	72.29±0.00+	126.33	125.85±0.09	1.20	1.19±0.00
Decision Tree	100.00	100.00±0.00	62.60	60.72±0.79+	21.12	20.86±0.16	0.01	0.01±0.00
Nearest Neighbour	72.92	72.92±0.00	67.50	67.50±0.00+	1.40	1.39±0.00	35.28	35.27±0.00
Naive Bayes	97.66	97.66±0.00	70.00	70.00±0.00+	0.17	0.17±0.00	0.09	0.09±0.00
Random Forest	95.16	93.85±0.79	78.44	76.07±1.42+	1.35	1.34±0.00	0.02	0.02±0.00

Table 4.8: Summary statistics of all baseline methods on **KTH-TIPS2**

KTH-TIPS2								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	100.00	99.88±0.29	100.00	98.90±1.14	87700.42	51554.76±13151.88	1.01	0.56±0.17
MLGP	100.00	99.44±0.50	99.50	97.47±1.14+	237517.97	78649.56±68223.91	3.44	0.76±0.89
2TGP	100.00	96.36±7.00	97.60	93.03±6.87+	4553.51	1891.99±1024.52	0.05	0.02±0.01
FeEx+GP	100.00	99.84±0.15	100.00	98.88±0.44 =	167.79	120.50±20.76	0.00	0.00±0.00
Hist+GP	99.00	98.56±0.44	98.60	95.80±1.60+	444.50	279.71±57.44	0.01	0.00±0.00
uLBP+GP	100.00	99.94±0.12	100.00	99.60±0.38 =	589.87	307.03±73.84	0.01	0.00±0.00
AdaBoost	100.00	100.00±0.00	83.80	83.80±0.00+	28.72	27.96±0.14	0.23	0.23±0.00
Decision Tree	100.00	100.00±0.00	80.56	76.65±2.02+	1.37	1.36±0.00	0.00	0.00±0.00
Nearest Neighbour	50.46	50.46±0.00	50.00	50.00±0.00+	0.15	0.15±0.00	1.73	1.72±0.00
Naive Bayes	92.13	92.13±0.00	91.20	91.20±0.00+	0.04	0.04±0.00	0.04	0.04±0.00
Random Forest	95.37	94.00±0.69	89.35	85.80±1.68+	0.34	0.34±0.00	0.01	0.01±0.00

Table 4.9: Summary statistics of all baseline methods on **FEI1**

FEI1								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	99.00	94.00±2.95	98.00	87.27±4.53	14837.16	7751.01±2066.07	0.15	0.07±0.02
MLGP	98.00	93.33±2.80	96.00	85.73±6.05+	45842.41	8816.65±11161.15	1.48	0.10±0.28
2TGP	99.00	96.30±2.07	96.00	90.00±2.92−	640.33	307.89±150.20	0.01	0.00±0.00
FeEx+GP	81.00	75.13±3.09	68.00	54.00±5.43+	73.82	46.87±10.05	0.00	0.00±0.00
Hist+GP	50.00	50.00±0.00	50.00	50.00±0.00+	47.92	38.96±5.60	0.00	0.00±0.00
uLBP+GP	88.00	80.33±3.31	68.00	52.87±7.50+	149.31	105.19±20.76	0.00	0.00±0.00
AdaBoost	100.00	100.00±0.00	94.00	93.13±1.01−	10.88	10.35±0.10	0.10	0.10±0.00
Decision Tree	100.00	100.00±0.00	92.00	86.67±3.21=	0.30	0.30±0.00	0.00	0.00±0.00
Nearest Neighbour	62.00	62.00±0.00	48.00	48.00±0.00+	0.01	0.01±0.00	0.14	0.13±0.00
Naive Bayes	90.00	90.00±0.00	80.00	80.00±0.00+	0.02	0.02±0.00	0.01	0.01±0.00
Random Forest	99.00	97.07±1.08	98.00	97.47±1.17−	0.13	0.13±0.00	0.01	0.01±0.00

Table 4.10: Summary statistics of all baseline methods on **JAFFE**

JAFFE								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	100.00	94.17±3.73	90.00	66.67±14.87	2982.11	1674.91±506.25	0.07	0.03±0.01
MLGP	100.00	94.33±5.04	100.00	86.50±7.89−	7664.00	1665.63±1688.61	0.16	0.03±0.04
2TGP	100.00	91.50±6.04	100.00	79.00±12.55−	192.93	55.39±41.81	0.00	0.00±0.00
FeEx+GP	100.00	94.67±4.72	95.00	69.00±11.99 =	26.81	19.08±3.16	0.00	0.00±0.00
Hist+GP	50.00	50.00±0.00	50.00	50.00±0.00+	18.32	13.63±1.65	0.00	0.00±0.00
uLBP+GP	100.00	99.17±1.90	75.00	51.67±9.94+	47.26	30.77±6.73	0.00	0.00±0.00
AdaBoost	100.00	100.00±0.00	80.00	77.17±2.52−	0.98	0.07±0.17	0.01	0.00±0.00
Decision Tree	100.00	100.00±0.00	80.00	76.67±2.40−	0.05	0.04±0.01	0.00	0.00±0.00
Nearest Neighbour	60.00	60.00±0.00	55.00	55.00±0.00+	0.01	0.00±0.00	0.01	0.01±0.00
Naive Bayes	100.00	100.00±0.00	85.00	85.00±0.00−	0.02	0.02±0.00	0.03	0.02±0.00
Random Forest	100.00	100.00±0.00	100.00	98.50±2.67−	0.15	0.12±0.01	0.02	0.01±0.00

Table 4.11: Summary statistics of all baseline methods on **Scene**

Scene								
	Training Accuracy		**Testing Accuracy**		**Training Time**		**Testing Time**	
	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.	Max	Average±Std.
uLBP-GP	80.90	78.78±1.94	80.90	72.88±3.32	47032.25	26770.43±7944.19	0.51	0.22±0.09
MLGP	85.30	79.48±3.09	80.20	73.00±3.71 =	142972.55	35368.13±42239.58	1.59	0.30±0.46
2TGP	80.30	75.05±2.72	74.30	68.04±3.62+	2264.81	976.28±584.93	0.02	0.01±0.00
FeEx+GP	76.50	72.83±1.85	67.70	63.39±3.39+	168.25	107.52±29.57	0.00	0.00±0.00
Hist+GP	83.40	80.66±1.87	76.30	71.54±2.88+	280.42	168.02±46.07	0.00	0.00±0.00
uLBP+GP	91.40	88.94±1.47	90.70	87.26±2.39−	397.26	230.36±66.20	0.00	0.00±0.00
AdaBoost	100.00	100.00±0.00	68.42	68.42±0.00+	27.64	27.18±0.09	0.16	0.16±0.00
Decision Tree	100.00	100.00±0.00	70.39	65.68±1.97+	1.96	1.95±0.01	0.00	0.00±0.00
Nearest Neighbour	72.70	72.70±0.00	67.11	67.11±0.00+	0.09	0.09±0.00	0.97	0.96±0.00
Naive Bayes	69.21	69.21±0.00	71.71	71.71±0.00=	0.03	0.03±0.00	0.02	0.02±0.00
Random Forest	83.49	80.92±1.19	75.66	71.43±1.52=	0.32	0.31±0.00	0.01	0.01±0.00

sets. Consequently, they are not robust against harsh illumination settings and various rotation angles. On the other hand, the proposed method achieved a slightly lower accuracy (approximately 3-4% decrease) compared to MLGP, Hist+GP and uLBP+GP. These methods uses a higher level feature extraction techniques. However, it was still able to obtain the maximum accuracy of 100%.

The **KyWiRo** dataset is the largest one out of all datasets. Again, the proposed method achieves a better performance than 2TGP and FeEx+GP which may be caused by the low-level features constructed by these methods. Similar to OutexTC, the proposed method obtains a lower accuracy compared to Hist+GP and uLBP+GP by approximately 5% and 4% in the average testing accuracy respectively. The high and stable performance from Hist+GP and uLBP+GP may be due to the nature of constructing high-level features from the entire image instead of local image regions.

In the **KTH-TIPS2** dataset, the uLBP-GP approach outperforms MLGP, 2TGP and Hist+GP while achieving similar performances as FeEx+GP and uLBP+GP. This dataset is easy, so all the GP methods are able to obtain high maximum accuracies. The high performance from 2TGP and FeEx+GP proves that low level features are sufficient to distinguish the images of different classes.

As for the **FEI1** dataset, the highest performing GP classifier is 2TGP in terms of average accuracy. However, the proposed method, MLGP and 2TGP are able to obtain the same maximum accuracy of 96%. FeEx+GP, Hist+GP and uLBP+GP perform significantly lower at about 50% accuracy, which shows that they are unable to find optimal solutions using the constructed features during the evolution process. While both 2TGP and FeEx+GP constructs low-level features, 2TGP uses GP to select local regions while FeEx+GP selects 6 pre-defined local regions.

The proposed method significantly outperforms FeEx+GP, Hist+GP and uLBP+GP. These methods uses static local regions or the entire image as the input. This indicates that GP selected local regions far are more effective than manually chosen local regions for face classification tasks. The significantly lower performance from uLBP+GP and Hist+GP concludes local features are able to obtain better performance than global features on FEI1 dataset. Global features may be effective for texture classification but not for facial expression classification.

The **JAFFE** dataset is a relatively small dataset. The proposed method seems to have encountered the issue of overfitting with an average training accuracy of 94% and testing accuracy of 66%. The new approach is unable to perform well in small datasets as compared to other GP classifiers such as MLGP and 2TGP.

As for the **Scene** dataset, the highest performing GP classifier is uLBP+GP. Unlike facial expression images, scene images do not have a specific local image regions that can effectively differentiate all the images. Using global features is more effective than local features for this task, which is why uLBP-GP, MLGP, 2TGP and FeEx+GP are not able to shine on this data set. Constructing high-level features using texture descriptors extracted by uLBP on the entire image discovers the highest performing solutions. These findings indicate that the Scene images contain important textural information and global features are very important for good classification performance.

4.8.2 Compared with General Classifiers

The proposed method was able to outperform all general classifiers for all texture datasets (**OutexTC, KyWiRo** and **KTH-TIPS2**). This proves general classifiers are not able to effectively learn from pixels in the original image for texture classification tasks and GP classifiers are able to produce better learned classifiers.

As for the **FEI1** dataset, Random Forest and AdaBoost perform well on this dataset. Both classifiers are ensemble learning algorithms that use a number of classifiers to construct a strong classifier. This is unsurprising as ensemble learning algorithms like AdaBoost have shown good results in facial expression recognition tasks in [58], [60] and [42].

The proposed method is outperformed by all general classifiers in the **JAFFE** dataset except for Nearest Neighbour. It is likely that there are key discriminatory pixels in the original image which were detected by the AdaBoost, Decision Tree, Naive Bayes and Random Forest, and high-level feature extraction techniques like uLBP were not a requirement for achieving high testing accuracy.

As for the **Scene** dataset, the uLBP-GP approach performs significantly better than AdaBoost, Decision Tree and Neighbour and achieves a similar performance as Naive Bayes and Random Forest.

The general classifiers that achieved significantly better results than the proposed method in some cases are mainly AdaBoost and Random Forest. These methods are boosting and ensemble classifiers, while the proposed method only uses a single evolved program. Besides, the datasets that general classifiers was able to achieve state-of-art accuracies were mainly facial expression classification tasks. This indicates that raw pixels extracted from the whole image are more effective, as opposed to single constructed feature from smaller regions for this classification task.

It requires additional effort to extract features prior to the training process when using general classifiers. It can be difficult to determine suitable feature extraction techniques for each dataset as they must be carefully selected and suited when dealing with image classification task. However, the proposed uLBP-GP method is able to achieve comparable results without such considerations.

4.8.3 Training time

The proposed uLBP-GP method obtains a comparable performance compared to MLGP with a decrease average training time. However, it has the highest training time after MLGP, which is likely due to the large amount of computation required by the image processing and uLBP operators. The gap between the average training times of MLGP and uLBP-GP increases further with large datasets such as KyWiRo.

All other baseline methods have lower average training times. While the increase in training time is a downside compared to the other methods, the time required (testing test) to apply the trained solution to new, unseen images is still relatively quick. Long training times are common when GP is used as long as the final programs are not overly complex, they are often fast to use after the training process.

4.9 Further Analysis

The uLBP-GP produces programs which can be interpreted and understood easily. This section analyses a few high-performing evolved programs to understand how they can perform classification with high accuracy.

4.9.1 Example Program on the KTH-TIPS2 data set

As the KTH-TIPS data set is relatively easy, majority of the programs evolved by uLBP-GP achieved greater than 98% in classification accuracy on training, validation and test sets. To display the good interpretability and understandability, a simple program is selected for analysis, as shown in Fig. 4.10. This program achieves 99.7% classification accuracy on the training set, 99.07% accuracy on the validation set and 98.1% accuracy on the test set. Figure 4.10 shows the example program, the example image from the two different classes and the outputs of each nodes of the example program. In the figure, the red colour represents the outputs of the *lettuce* class, and the green colour represents that of the *wood* class.

This program detects one rectangular region and one square region at different positions in the input image. Both regions overlaps each other slightly. The left detected region with a size of 36×30 is smaller than the right region with a size of 63×63. Both regions captures the differences of the wood and lettuce classes by capturing the an area on the top right side of the image. In this region, the *lettuce* class shows the vein of the lettuce leaves travelling upwards diagonally, while the *wood* class displays a much smoother surface. The fluctuations in the gray values in the *lettuce* class changes more drastically than of that in the *wood* class. The square region captures a larger portion of the top right area.

In the left region, the *Laplace* operators are evolved to highlight regions of rapid intensity change. In the right region, the *SobelY* operator is evolved to detect edges along the vertical direction. By these evolved operators, the textural information in the image is enhanced further and extracted by the uLBP operator. The regions are then converted into normalized histograms. The mean of histogram in the left region is computed to give an idea of average value of LBP code that occurred in the histogram. Unsurprisingly, the *lettuce* class has a higher mean value than of that of the *wood* class as it has more areas with significantly darker regions. On the other hand, the normalized histogram value at bin 48 was selected which indicates that the value of LBP code occurred in the *lettuce* class only (The *wood* class has a normalized frequency of zero). The two values from the mean of the histogram and value of the specific bin from the histogram are constructed by the *Sub* operator further for classification.

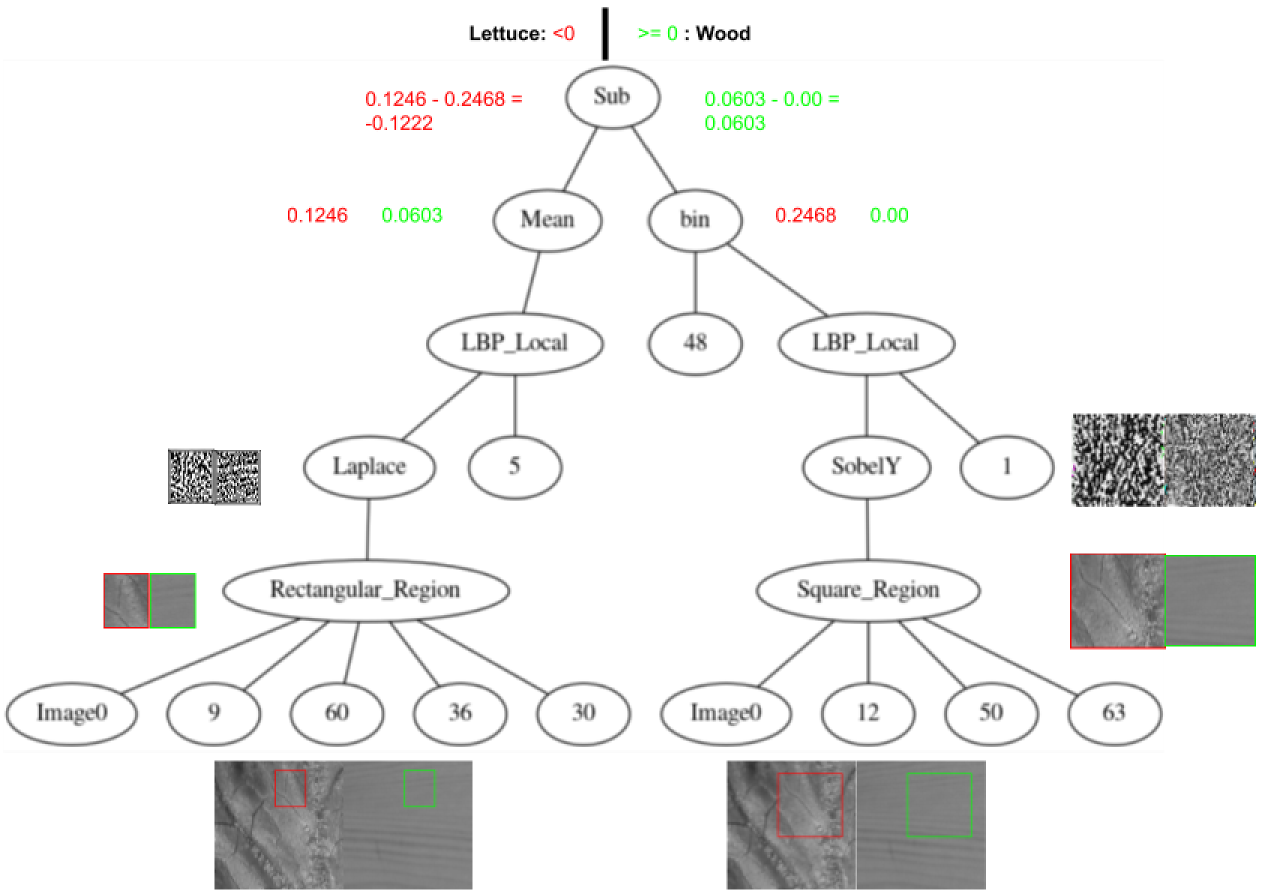

Figure 4.10: An example program evolved by the uLBP-GP method on the KTH-TIPS2 data set.

4.9.2 Example Program on the FEI1 data set

Fig. 4.11 demonstrates an example program evolved by the uLBP-GP method on the FEI1 data set. This program achieves 97% classification accuracy on the training set, 96% accuracy on the validation set and 94% accuracy on the test set. This program detects two rectangular regions of an image. Both detected regions do not show any significant differences in their sizes or positions. Both regions capture a rectangular region spanned from the left eye to the corner of the lip line. However, the distinctive difference between the *Smiling* and *Neutral* can be discovered with the presence of the upper creases of the mouth when the subject smiles which causes more changes in the gray values in the regions.

In the both regions, the *Gau1* operators are evolved to reduce image noise. The regions are then converted into normalized histograms after extracting the LBP code by the uLBP operator. Two different neighbourhood radius size are used to compute the LBP code, 2 and

1 for the left and right regions, respectively.

The standard deviation of histogram in the left region is computed to give an overall picture of the fluctuations of the LBP code values. On the other hand, the mean of histogram in the right region is computed to give an idea of average value of LBP code that occurred in the histogram. However, the results from both classes garnered the same value, which indicates that the program may have been slightly more complex than it should have been. The two value are constructed by the *Sub* operator further for classification.

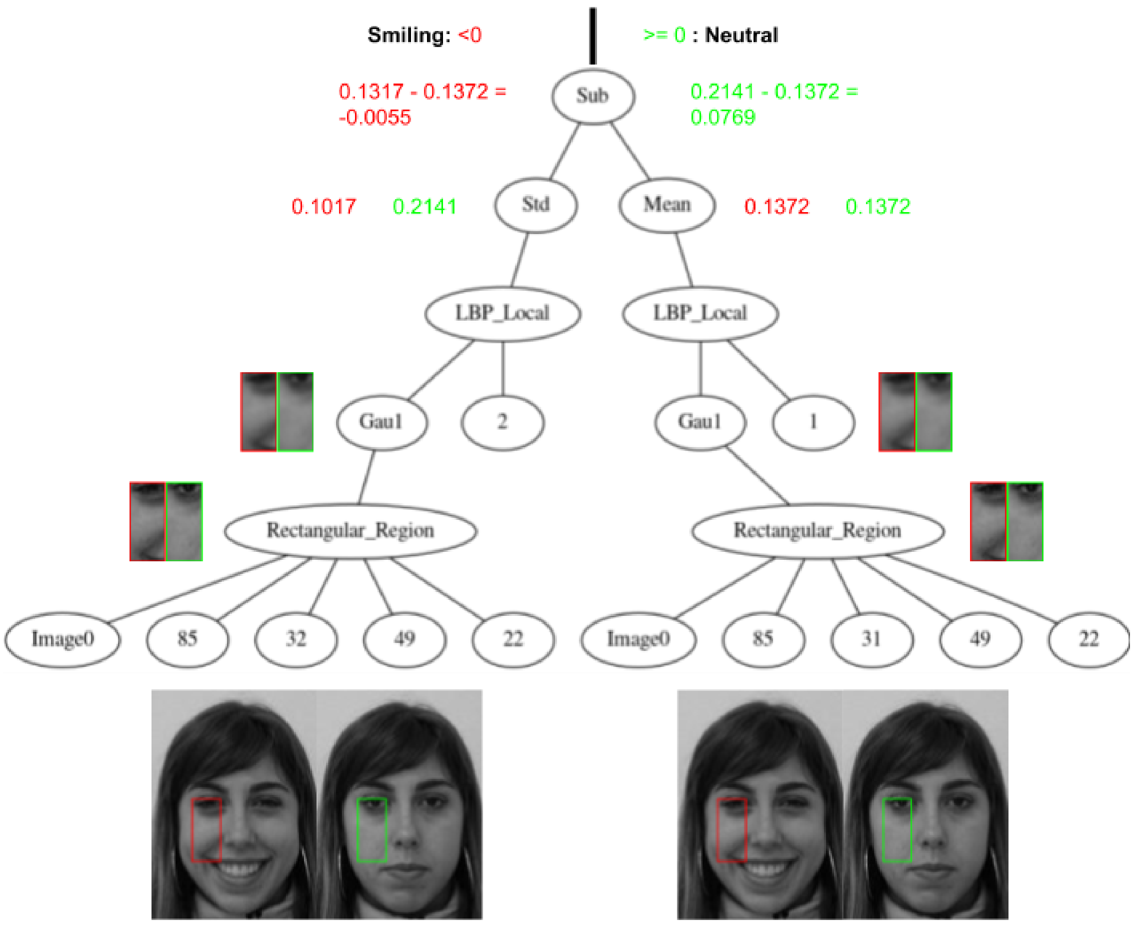

Figure 4.11: An example program evolved by the uLBP-GP method on the FEI1 data set.

4.10 Chapter Summary

This chapter introduced a new approach called uLBP-GP which aims to improve the performance of image classification using uniform LBP algorithm. In terms of maximum testing accuracy achieved, the proposed method achieved 100% or 99% for all texture datasets. Furthermore, it outperformed all general classifiers in terms of both maximum and average testing accuracy. The proposed solution achieved faster training time than MLGP while achieving a comparable performance. However, it struggled to provide state-of-the-art performance on facial expression and scene datasets which are datasets with less prominent textural information.

A clear advantage of the proposed method is the interpretability of the model. This chapter showed that uLBP-GP was able to automatically extract and construct useful, high-level features. Performance evaluation showed promising results using the proposed method. The analysis of high-performing solutions showed that uLBP-GP could perform well using simple programs. The adaptation of the uniform LBP algorithm as a function set in a GP tree was shown to be an effective method of performing high-level feature extraction.

In a nutshell, the proposed uLBP-GP method struggles to provide consistent and state-of-art results on facial expression and scene classification tasks. However, it excels in texture classification tasks due to its ability to effectively extract textural information.

Chapter 5

Conclusions and Future Work

5.1 Major Conclusion

The overall goal of the project was to develop a new domain-independent image classification by combining GP with existing feature extraction algorithms. A number of approaches were developed which delivered promising results when compared to general classifiers. The two main approaches used LQP and uLBP respectively with GP techniques with a slight change in the program structure. The uLBP-GP approach adds another layer to the GP program that uses various image operators to process the input image before using the uLBP operator to remove any prior image discrepancies. The LQP approach was shown to produce unpromising results with a tremendously high training time. The second approach, uLBP-GP approach has more potential and was shown to be more effective with a reduced amount of training time. The step to use the operators with various parameters values to adapt their operations to cater to a wide range of problems did not show a drastic improvement in performance, but did display state-of-art performances in all texture datasets in terms of maximum accuracy.

Several major conclusions can be drawn as a result of this work:

1. New GP methods using widely used feature extraction techniques such as LBP and LQP can be designed to give promising performance results. Using GP to select regions of feature extractions to classify these features was effective in the proposed uLBP-GP method.

2. uLBP-GP was able outperform non-GP methods on all texture datasets and some GP and non-GP methods on non-texture datasets. The proposed method was also able to perform comparably when compared to other high performing GP methods with a lesser training time (i.e MLGP).

3. It was shown that the features extracted and constructed by good programs using the uLBP-GP approach were relatively easily understood by humans, giving an insight into how region detection and parameter tuning can be used to improve classification performance. This concluded that the methods proposed in this work were effective as they produced solutions which could be analysed and understood.

5.2 Future Work

There are some limitations with the current approaches, which could be addressed in the future with further development on the following points.

The uLBP-GP approach displayed a drop in performance when tested using the JAFFE dataset, which is a dataset with a small number of instances. This issue can be addressed by designing an adequate fitness function to allow GP to still learn effectively from a small number of instances.

Compared to other state-of-art general classifiers, a large limitation is the training time taken to learn the model. Further work can be done to improve the computational efficiency of the work.

Furthermore, only one high-level feature is constructed by uLBP-GP for image classification which may be less effective. Features extracted by the feature construction layer of uLBP-GP can be further investigated to determine whether the extracted/constructed features automatically learned can be useful for other classification algorithms, such as Nearest Neighbour or Decision Trees.

Bibliography

[1] AHONEN, T., HADID, A., AND PIETIKÄINEN, M. Face recognition with local binary patterns. In *European conference on computer vision* (2004), Springer, pp. 469–481.

[2] AL-SAHAF, H., AL-SAHAF, A., XUE, B., JOHNSTON, M., AND ZHANG, M. Automatically evolving rotation-invariant texture image descriptors by genetic programming. *IEEE Transactions on Evolutionary Computation 21*, 1 (Feb 2017), 83–101.

[3] AL-SAHAF, H., SONG, A., NESHATIAN, K., AND ZHANG, M. Extracting image features for classification by two-tier genetic programming. In *2012 IEEE Congress on Evolutionary Computation* (June 2012), pp. 1–8.

[4] AL-SAHAF, H., SONG, A., NESHATIAN, K., AND ZHANG, M. Two-tier genetic programming: towards raw pixel-based image classification. *Expert Systems with Applications 39*, 16 (2012), 12291 – 12301.

[5] AL-SUMAIDAEE, S. A., ABDULLAH, M. A., AL-NIMA, R. R. O., DLAY, S. S., AND CHAMBERS, J. A. Multi-gradient features and elongated quinary pattern encoding for image-based facial expression recognition. *Pattern Recognition 71* (2017), 249–263.

[6] ALI, N., AND ZAFAR, B. 15-scene image dataset, Aug 2018.

[7] ALPAYDIN, E. *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[8] ATKINS, D., NESHATIAN, K., AND ZHANG, M. A domain independent genetic programming approach to automatic feature extraction for image classification. In *2011 IEEE Congress of Evolutionary Computation (CEC)* (June 2011), pp. 238–245.

[9] BI, Y., XUE, B., AND ZHANG, M. An automatic feature extraction approach to image classification using genetic programming. In *Applications of Evolutionary Computation* (Cham, 2018), K. Sim and P. Kaufmann, Eds., Springer International Publishing, pp. 421–438.

[10] BOSCH, A., ZISSERMAN, A., AND MUNOZ, X. Image classification using random forests and ferns. In *2007 IEEE 11th international conference on computer vision* (2007), Ieee, pp. 1–8.

[11] CHAPELLE, O., HAFFNER, P., AND VAPNIK, V. N. Support vector machines for histogram-based image classification. *IEEE transactions on Neural Networks 10*, 5 (1999), 1055–1064.

[12] CHOI, W.-J., AND CHOI, T.-S. Genetic programming-based feature transform and classification for the automatic detection of pulmonary nodules on computed tomography images. *Inf. Sci. 212* (Dec. 2012), 57–78.

[13] CYGANEK, B. *Object detection and recognition in digital images: theory and practice*. John Wiley & Sons, 2013.

[14] DALAL, N., AND TRIGGS, B. Histograms of oriented gradients for human detection. In *international Conference on computer vision & Pattern Recognition (CVPR'05)* (2005), vol. 1, IEEE Computer Society, pp. 886–893.

[15] DEEP, G., KAUR, L., AND GUPTA, S. Local quantized extrema quinary pattern: a new descriptor for biomedical image indexing and retrieval. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization 6*, 6 (2018), 687–703.

[16] DOS SANTOS, J. A., FERREIRA, C. D., DA SILVA TORRES, R., GONALVES, M. A., AND LAMPARELLI, R. A. C. A relevance feedback method based on genetic programming for classification of remote sensing images. *Inf. Sci. 181* (2011), 2671–2684.

[17] EBERHART, R., AND KENNEDY, J. Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks* (1995), vol. 4, Citeseer, pp. 1942–1948.

[18] EIBEN, A. E., AND SCHOENAUER, M. Evolutionary computing. *Information Processing Letters 82*, 1 (2002), 1–6.

[19] EIBEN, A. E., SMITH, J. E., ET AL. *Introduction to evolutionary computing*, vol. 53. Springer, 2003.

[20] FORTIN, F.-A., RAINVILLE, F.-M. D., GARDNER, M.-A., PARIZEAU, M., AND GAGNÉ, C. Deap: Evolutionary algorithms made easy. *Journal of Machine Learning Research 13*, Jul (2012), 2171–2175.

[21] FRITZ, M., HAYMAN, E., CAPUTO, B., AND OLOF EKLUNDH, J. The kth-tips database, 2004.

[22] FU, W., JOHNSTON, M., AND ZHANG, M. Genetic programming for edge detection: a global approach. In *2011 IEEE Congress of Evolutionary Computation (CEC)* (2011), IEEE, pp. 254–261.

[23] HARALICK, R. M., SHANMUGAM, K., ET AL. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, 6 (1973), 610–621.

[24] HOLLAND, J. H., ET AL. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press, 1992.

[25] JUNGES, R., AND KLÜGL, F. Evolution for modeling: a genetic programming framework for sesam. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation* (2011), ACM, pp. 551–558.

[26] KAELBLING, L. P., LITTMAN, M. L., AND MOORE, A. W. Reinforcement learning: A survey. *Journal of artificial intelligence research 4* (1996), 237–285.

[27] KAWAGUCHI, S., AND NISHII, R. Hyperspectral image classification by bootstrap adaboost with random decision stumps. *IEEE Transactions on Geoscience and Remote Sensing 45*, 11 (2007), 3845–3851.

[28] KENNEDY, J. Swarm intelligence. In *Handbook of nature-inspired and innovative computing*. Springer, 2006, pp. 187–219.

[29] KOBASHIGAWA, J. S., YOUN, H., ISKANDER, M. F., AND YUN, Z. Classification of buried targets using ground penetrating radar: Comparison between genetic programming and neural networks. *IEEE Antennas and Wireless Propagation Letters 10* (2011), 971–974.

[30] KOZA, J. R. Introduction to genetic programming. In *Proceedings of the 9th Annual Conference Companion on Genetic and Evolutionary Computation* (New York, NY, USA, 2007), GECCO '07, ACM, pp. 3323–3365.

[31] KWAK, J. T., XU, S., AND WOOD, B. J. Efficient data mining for local binary pattern in texture image analysis. *Expert Systems with Applications 42*, 9 (2015), 4529 – 4539.

[32] KYLBERG, G. The kylberg texture dataset v. 1.0. External report (Blue series) 35, Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, Uppsala, Sweden, September 2011.

[33] LENSEN, A., AL-SAHAF, H., ZHANG, M., AND XUE, B. Genetic programming for region detection, feature extraction, feature construction and classification in image data. In *Genetic Programming* (Cham, 2016), M. I. Heywood, J. McDermott, M. Castelli, E. Costa, and K. Sim, Eds., Springer International Publishing, pp. 51–67.

[34] LYONS, M. J., AKAMATSU, S., KAMACHI, M., GYOBA, J., AND BUDYNEK, J. The japanese female facial expression (jaffe) database. In *Proceedings of third international conference on automatic face and gesture recognition* (1998), pp. 14–16.

[35] MIKOLAJCZYK, K., AND SCHMID, C. A performance evaluation of local descriptors.

[36] MONTANA, D. J. Strongly typed genetic programming. *Evolutionary computation 3*, 2 (1995), 199–230.

[37] NANNI, L., LUMINI, A., AND BRAHNAM, S. Local binary patterns variants as texture descriptors for medical image analysis. *Artificial Intelligence in Medicine 49*, 2 (2010), 117 – 125.

[38] NAYAK, D. R., DASH, R., AND MAJHI, B. Brain mr image classification using two-dimensional discrete wavelet transform and adaboost with random forests. *Neurocomputing 177* (2016), 188–197.

[39] OECHSLE, O., AND CLARK, A. F. Feature extraction and classification by genetic programming. In *Computer Vision Systems* (Berlin, Heidelberg, 2008), A. Gasteratos, M. Vincze, and J. K. Tsotsos, Eds., Springer Berlin Heidelberg, pp. 131–140.

[40] OJALA, T., MAENPAA, T., PIETIKAINEN, M., VIERTOLA, J., KYLLONEN, J., AND HUOVINEN, S. Outex-new framework for empirical evaluation of texture analysis algorithms. In *Object recognition supported by user interaction for service robots* (2002), vol. 1, IEEE, pp. 701–706.

[41] OJALA, T., PIETIKÄINEN, M., AND HARWOOD, D. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition 29*, 1 (1996), 51–59.

[42] OWUSU, E., ZHAN, Y., AND MAO, Q. R. An svm-adaboost facial expression recognition system. *Applied Intelligence 40*, 3 (Apr 2014), 536–545.

[43] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., AND DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[44] POLI, R. Genetic programming for feature detection and image segmentation. In *AISB Workshop on Evolutionary Computing* (1996), Springer, pp. 110–125.

[45] RAMPUN, A., SCOTNEY, B., JOHN MORROW, P., WANG, H., AND WINDER, J. Breast density classification using local quinary patterns with various neighbourhood topologies. *Journal of Imaging 4* (01 2018), 14.

[46] SHELTON, J., V. DOZIER, G., BRYANT, K., ADAMS, J., POPPLEWELL, K., ABEGAZ, T., PURRINGTON, K., L. WOODARD, D., AND RICANEK, K. Genetic based lbp feature extraction and selection for facial recognition. pp. 197–200.

[47] SMART, W., AND ZHANG, M. Tracking object positions in real-time video using genetic programming. In *Proceeding of Image and Vision Computing International Conference* (2004), pp. 113–118.

[48] SUTTON, R. S., AND BARTO, A. G. *Reinforcement Learning: An Introduction*, second ed. The MIT Press, 2018.

[49] SZEGEDY, C., VANHOUCKE, V., IOFFE, S., SHLENS, J., AND WOJNA, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 2818–2826.

[50] SZUMMER, M., AND PICARD, R. W. Indoor-outdoor image classification. In *Proceedings 1998 IEEE International Workshop on Content-Based Access of Image and Video Database* (1998), IEEE, pp. 42–51.

[51] TAN, X., AND TRIGGS, W. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing 19*, 6 (2010), 1635–1650.

[52] TAPIA, J. E., PEREZ, C. A., AND BOWYER, K. W. Gender classification from iris images using fusion of uniform local binary patterns. In *European Conference on Computer Vision* (2014), Springer, pp. 751–763.

[53] THOMAZ, C. E. Fei face database. Tech. rep., Department of Electrical Engineering Centro Universitario da FEI, So Bernardo do Campo, So Paulo, Brazil, March 2006.

[54] VIPPARTHI, S. K., AND NAGAR, S. K. Color directional local quinary patterns for content based indexing and retrieval. *Human-centric Computing and Information Sciences 4*, 1 (2014), 6.

[55] WAH, C., BRANSON, S., WELINDER, P., PERONA, P., AND BELONGIE, S. The caltech-ucsd birds-200-2011 dataset.

[56] WANG, X., HAN, T. X., AND YAN, S. An hog-lbp human detector with partial occlusion handling. In *2009 IEEE 12th international conference on computer vision* (2009), IEEE, pp. 32–39.

[57] XUE, B., AND ZHANG, M. Evolutionary feature manipulation in data mining/big data. *ACM SIGEVOlution 10*, 1 (2017), 4–11.

[58] YING ZILU, AND FANG XIEYAN. Combining lbp and adaboost for facial expression recognition. In *2008 9th International Conference on Signal Processing* (Oct 2008), pp. 1461–1464.

[59] YUAN, J.-H., ZHU, H.-D., GAN, Y., AND SHANG, L. Enhanced local ternary pattern for texture classification. In *International Conference on Intelligent Computing* (2014), Springer, pp. 443–448.

[60] YUBO WANG, HAIZHOU AI, BO WU, AND CHANG HUANG. Real time facial expression recognition with adaboost. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* (Aug 2004), vol. 3, pp. 926–929.

[61] ZHANG, M., AND CIESIELSKI, V. Genetic programming for multiple class object detection. In *Australasian Joint Conference on Artificial Intelligence* (1999), Springer, pp. 180–192.