Genetic Programming with A New Representation to Automatically Learn Features and Evolve Ensembles for Image Classification

Ying Bi, *Student Member, IEEE*, Bing Xue, *Member, IEEE*, and Mengjie Zhang, *Fellow, IEEE,*

***Abstract*—Image classification is a popular task in machine learning and computer vision, but it is very challenging due to high variations crossing images. Using ensemble methods for solving image classification can achieve higher classification performance than using a single classification algorithm. However, to obtain a good ensemble, the component (base) classifiers in an ensemble should be accurate and diverse. To solve image classification effectively, feature extraction is necessary to transform raw pixels into high-level informative features. However, this process often requires domain knowledge. This paper proposes an evolutionary approach based on genetic programming to automatically and simultaneously learn informative features and evolve effective ensembles for image classification. The new approach takes raw images as inputs and returns predictions of class labels based on the evolved classifiers. To achieve this, a new individual representation, a new function set and a new terminal set are developed to allow the new approach to effectively find the best solution. More importantly, the solutions of the new approach can extract informative features from raw images and can automatically address the diversity issue of the ensembles. In addition, the new approach can automatically select and optimise the parameters for the classification algorithms in the ensemble. The performance of the new approach is examined on 13 different image classification data sets of varying difficulty and compared with a large number of effective methods. The results show that the new approach achieves better classification accuracy on most data sets than the competitive methods. Further analysis demonstrates that the new approach can evolve solutions with high accuracy and diversity.**

***Index Terms*—Genetic Programming; Representation; Feature Learning; Ensemble Learning; Image Classification**

I. Introduction

IMAGE classification is a popular task in machine learning and computer vision in recent years [1]. The task of image classification is to automatically assign each image X_i in $\{X_i\}_{i=1}^N$ with a class label C_j from $\{C_j\}_{j=1}^M$ based on the content in X_i (N is the number of images and M is the number of classes). The human vision system can easily obtain or capture information from an image to decide which group the image is, while a computer can only see matrix or numbers from the image, which indicates that image classification is difficult. Despite the success achieved by many techniques [2], image classification remains a challenging task due to the high variations of images, such as in illumination, background, rotation, deformation, and occultation.

Ensemble methods have been widely used for solving classification problems [3, 4]. An ensemble consists of multiple base learners (classifiers) to solve a classification problem [5]. In an ensemble, each classifier is trained using the training data and a commonly used classification algorithm such as decision tree (DT) and support vector machines (SVMs) [3, 6]. Then a combination method such as voting and averaging is used to combine the outputs of classifiers to make a good prediction. An ensemble often achieves a better generalisation performance than a single classifier on unseen data [6]. The performance of an ensemble can be defined as $E = \overline{E} - \overline{A}$, where $\overline{E}$ denotes the average of the generalisation errors of the classifiers and $\overline{A}$ denotes the average ensemble ambiguity (diversity) [5, 7]. As indicated by this equation, to obtain a good ensemble, the classifiers should be accurate and diverse. In the ensemble building process, the diversity of classifiers is often considered as a key factor affecting the performance of ensembles [8]. The diversity indicates that the errors achieved by the classifiers in the ensemble are uncorrelated, which is not straightforward to measure. Many techniques have been developed to build a set of diverse classifiers, such as bagging and boosting, which use different data sets to train each classifier in the ensemble [5, 9]. Other effort can be seen in [10, 11]. However, the problem of diversity is still an open issue in ensemble learning [8].

Using ensemble methods to solve image classification, a key process is to extract a set of informative features from the images as the raw pixel values are often not effective. Then the extracted features are fed into a set of classification algorithms to train the classifiers of an ensemble [12]. Feature extraction transforms low-level raw pixel values of images into high-level informative features, which are discriminative for classification. However, due to high image variations and the high dimensionality of the image data, it is difficult to obtain discriminative and invariant features from images. Although many methods such as local binary patterns (LBP) [13], histogram of orientated gradients (HOG) [14] and scale-invariant feature transform (SIFT) [15] have been proposed, they are limited to particular image domains and need domain knowledge when applying them to new domains. Feature learning techniques can address these issues by automatically learning effective features from raw images with the goal of achieving a high classification performance. Typical work

This work was supported in part by the Marsden Fund of New Zealand Government under Contracts VUW1509, and VUW1615, and the SfTI Spearhead Trench 2 Grant E3603/2903. This work of Ying Bi was supported by China Scholarship Council (CSC)/Victoria University Scholarship.

The authours are with School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand (e-mail: ying.bi@ecs.vuw.ac.nz; bing.xue@ecs.vuw.ac.nz; mengjie.zhang@ecs.vuw.ac.nz).

can be seen in [16, 17]. However, since image classification involves a large range of domains, i.e., face images, texture images, scene images, and other object images, to learn informative features for dealing with different types of image classification tasks is still challenging.

As an evolutionary computation (EC) technique, genetic programming (GP) aims to automatically find solutions in the form of programs for solving particular problems [18]. GP has been used to construct ensembles for classification since each GP program can be a classifier for binary or multi-class classification [19], such as using bagging or boosting methods [9]. Other GP-based ensemble methods can be seen in [20, 21, 22]. However, most methods cannot be directly used for tackling with image classification tasks due to the necessity and difficulty of feature extraction.

On the other hand, many GP-based methods have been proposed to automatically learn features for image classification, such as in [16, 17, 23, 24]. The learnt features are often through multiple levels of transformation, i.e., linear and nonlinear, in a tree-based representation. Typically, one high-level feature is constructed/learnt by GP from a raw image and can be easily used for binary image classification, such as in [23, 24]. However, most image classification tasks are multi-class and one feature is not effective for solving them. GP has been applied to learn multiple features for image classification and shown a promise [16, 17]. But most methods are limited to particular image domains, such as texture classification [17]. In addition, these methods only employ a single classifier for classification. Therefore, their classification performance could be further improved by using ensemble classifiers.

With a flexible tree-based representation, it is possible to integrate the processes of the feature learning and the ensemble learning into a single GP tree to automatically evolve effective solutions for image classification from raw images. Our previous work [25] developed an automated ensemble framework using GP (EGP) for image classification and achieved promising results on different types of image classification tasks. The EGP method can learn features using filtering and pooling operators and use these features to build classifiers of an ensemble. The main advantage of EGP is the significant reduction of domain knowledge requirement in the whole process, where the selections of image-related operators, classification algorithms and combination methods are automatically conducted and optimised during the evolutionary learning process. However, EGP learns features through filtering and pooling, which may not be effective for complex image classification such as scene classification. Furthermore, EGP cannot automatically tune parameters for the classification algorithms, which is not flexible and efficient for different image classification tasks.

The overall goal of this paper is to develop a new effective GP-based approach by addressing the limitations of EGP to automatically learn effective features and evolve ensembles for image classification. The proposed approach is called improved EGP (IEGP). To achieve this goal, a new representation with an input layer, a filtering & pooling layer, a feature extraction layer, a concatenation layer, a classification layer, a combination layer, and an output layer will be developed in IEGP. Each functional layer, i.e., except for the input and output layers, will have a number of functions to allow IEGP to automatically evolve combinations of them to form the solutions. Each solution will produce the combined predictions of class labels for the input images. The proposed approach will be evaluated on 13 data sets of varying difficulty, including large data sets with noisy. The performance of IEGP will be compared with a large number of benchmark methods. A deep analysis will be conducted on IEGP to show its effectiveness.

The characteristics of the new IEGP approach can be summarised in the following five aspects.

1) A new multi-layer individual representation is developed in IEGP to allow it to automatically and simultaneously learn features and evolve ensembles for image classification. The new representation has seven layers with different functionalities, which make it different from the current multi-layer representations of GP, such as in [16, 23]. The new representation allows IEGP to produce solutions of ensembles from raw images without human intervention and domain expertise.
2) IEGP can learn high-level features through multiple transformations, i.e., filtering, pooling, complex feature extraction functions, and concatenation. The learnt features are invariant to certain variations such as rotation, which can improve the classification performance.
3) IEGP is able to automatically select and optimise the parameters for the classification algorithms in the evolved ensemble. The important parameters, such as the number of trees in random forest (RF) and the regularisation parameter in logistic regression (LR), are designed as terminals of IEGP with a predefined range to allow them to be optimised during the evolutionary learning process. This leads to the evolved ensembles containing effective and efficient classifiers for producing promising results.
4) IEGP can automatically address the diversity issue when building the ensembles, where the strategies belong to input feature manipulation and learning parameter manipulation [6].
5) The evolved solutions of IEGP can be easily visualised as trees, which provide high interpretability on the target problems, i.e., what types of features are learnt, what classification algorithms are used to build the ensemble, and why they produce good results.

II. Background and Related Work

A. GP and Strongly Typed GP

GP is an EC technique and is able to evolve computer programs for solving problems [18]. Similar to genetic algorithms (GAs), GP has an evolutionary learning process to search for the best individual/solution. Different from GAs, each individual in GP is commonly represented by a tree, having the functions as internal nodes and terminals as leaf nodes. This representation needs a tree generation method, i.e., *full*, *grow*, and *ramped half-and-half*, to build the initial population, and the *subtree crossover* and *subtree mutation* for crossover and mutation, respectively.

Classical tree-based GP is effective for dealing with one data type, such as the commonly used functions, $+, -, \times$, and protected $\%$ operate on floating-point numbers. The final output of a standard GP tree is a number as well. However, for some tasks, multiple data types are often involved, which requires a GP method having the ability to deal with multiple data types. Strongly typed GP (STGP) [26] has been proposed for this purpose. STGP defines an input type and an output type to each function and defines an output type to each terminal. When building a GP tree using the functions and terminals, each node of the tree must obey the type constraint that the types of arguments of a node must equal to the defined ones. To satisfy the type constraint and to integrate functions and terminals into trees, a program structure is often needed. On image data, STGP is more popular than GP since it can deal with multiple data types such as arrays.

B. Representation of GP on Image Data

Liu et al. [27] developed a GP-based method to automatically learning spatio-temporal representations from 3D sequences for action recognition. This method has a new tree representation with an input tier, a filtering tier, a max-pooling tier, and an output tier to learn informative features from 3D sequences. In [17], a novel GP method was proposed for texture description, where a new root node was developed to transform the outputs of the children nodes into binary codes. Then the binary codes are transformed into decimal numbers and the histogram of the decimal numbers were used as features. The features extracted in this way have achieved better classification performance than the other texture features. Bianco et al. [28] developed a GP-based method for simultaneous video change detection algorithm selection, combination and processing. This method uses change detection algorithms as functions in the trees. However, these representations are task-specific and cannot be directly used for other tasks.

Multi-tier or multi-layer representation of GP has been found in the literature for classifying images from raw pixels. A general image classification process often contains the processes of feature extraction, feature selection and classification. These processes can be integrated into a GP tree using a multi-tier or multi-layer representation to evolve solutions to address them simultaneously. Typically, each functional layer has a number of functions available for GP to search for them to build the trees. Atkins et al. [23] developed a multi-tier GP (named as 3TGP in [24]) for binary image classification. The 3TGP method has an image filtering tier, an aggregation tier and a classification tier, which performs image filtering, region detection, feature extraction, feature construction, and classification in a single tree. Bi et al. [29] proposed a multi-layer GP method to achieve region detection, feature extraction, feature construction layer, and classification simultaneously. However, these representations are only effective for solving binary image classification tasks because the output of each GP tree is a floating-point number, which is naturally suitable for binary classification. Therefore, very few work on GP with a multi-layer representation has been proposed for multi-class image classification. Shao et al. [16] developed a multi-objective GP method with a multi-layer representation to learn features that are transformed by multiple filtering and pooling functions. This method has shown promising results on four different image classification tasks, including difficult scene classification. Existing work shows that using GP to effectively solve image classification, the representation is essential. A good representations of GP can achieve promising performance, such as in [16, 17]. However, most existing representations are only effective for particular tasks. When a new task, i.e., ensemble learning, is integrated in GP, it is necessary to develop a new representation.

C. Image Feature Extraction and Learning

Many effective feature extraction methods have been developed to extract informative features from images. Among the existing methods, the most popular ones include LBP [13], SIFT [15] and HOG [14]. LBP [13] is a simple but effective method for texture feature extraction. The standard LBP method labels each pixel in an image using a decimal number calculated by quantizing the relationship between the pixel and its neighbours. The histogram of the LBP labels is often extracted as features for dealing with particular tasks such as texture categorising [13, 17]. To deal with certain variations, LBP variants have been developed, e.g., uniform LBP (uLBP), completed LBP (CLBP) and local derivative pattern (LDP) [1]. The SIFT method [15] is a local feature description method by detecting keypoints from an image and then extracting features from the detected keypoints. To reduce the computational cost, a dense SIFT method [30] has been developed to extract histogram features of gradient magnitude and direction for the whole image instead of each detected keypoint. The HOG method [14] can extract histogram features of gradient orientation in a different way compared with SIFT. It extracts locally normalised histogram features of gradient orientation in a dense overlapping grid. The features extracted by HOG are known as shape and appearance features.

Compared with feature extraction methods, feature learning methods can automatically and dynamically learn more informative and meaningful features for solving particular tasks. Convolutional neural networks (CNNs) are typical examples following in this category, where image features are learnt via multiple layers of non-linear transformation. Li and Gong [31] proposed a self-paced CNN (SPCN) by assigning weights to the training samples during the learning process in order to enhance the learning robustness of CNNs. Several variants of CNNs have been developed, such as scattering convolution networks (ScatNet) [32] and principal component analysis network (PCANet) [33], where the convolution filters are generated using different ways. In [34], a feedforward convolutional conceptor neural network (FCCNN) was developed for image classification. The contractive auto-encoder (CAE) was employed by Rifai et al. [35] for digits recognition.

EC-based methods have been used in this process to obtain effective features for solving particular tasks. Fu et al. [36] developed a GP-based method to evolve solutions that can detect edge features from the image. Albukhanajer et al. [37] applied an evolutionary multiobjective algorithm to optimise the functionals in the trace transform to extract robust and invariant image features for object classification. Wei and Tang

[38] developed a GA-based method to learn conceptual shape representation for object recognition. The contour of the shape is preprocessed and the important information is extracted and encoded in the genes, which allows GA to search for the optimal ones. Mistry et al. [1] proposed a micro-GA embedded particle swarm optimisation (PSO) to optimise the extracted horizontal and vertical neighborhood pixel comparison LBP (hvnLBP) features for facial expression classification.

D. Ensemble Methods for Classification

Ensemble methods have been widely applied to classification tasks in machine learning. Nag and Pal [39] proposed a multiobjective GP algorithm to simultaneous feature selection and ensemble evolving. In this algorithm, the multi-class classification problem is decomposed to multiple binary classification tasks and an ensemble of GP programs is created for each task. A detailed review on ensemble methods for the class imbalance problem can be found in [3], where the main methods include cost-sensitive boosting, booting-based, bagging-based, and hybrid methods. Yu et al. [4] developed a new feature selection-based semi-supervised framework for classification. In the ensemble, each classifier is trained on a subspace selected by a feature selection method.

To employ an ensemble method for image classification often needs a different design since before the general classifier training process the feature extraction process is needed to extract meaningful features from raw images. Dittimi and Suen [12] used CNNs to extract image features and used principal component analysis (PCA) to reduce the dimension of the features. Then different base learners such as RF, DT and SVMs are selected and trained on the features. Finally, these classifiers are combined to obtain the predictions.

EC-based ensemble methods have seldom been proposed to solve image classification. Majid et al. [40] proposed a GP-based method to find optimal combinations (ensembles) of SVM classifiers for gender classification from frontal face images. This method achieved better results than using single SVM classifier with different kernel functions. Other GP or EC-based methods can be found to solve classification but not image classification, such as [20, 39]. To the best of our knowledge, [25] firstly introduced a GP-based automated ensemble learning framework for image classification. This method can automatically and simultaneously learn features and build ensembles from images, which is different from the methods in [39] requiring manually extracted image features. However, the method in [25] has shown an inferior performance on large image classification data sets, which may due to the learnt features are not effective. In addition, the method cannot automatically select parameters for the classification algorithms, which is not flexible and efficient for different image classification tasks. Therefore, this paper significantly improves the EGP method in [25] by developing a new individual representation, a new function set and a new terminal set to automatically learn meaningful features and select suitable parameters for classification algorithms to build ensembles for image classification.

III. The New Approach

The general process of traditional ensemble methods for image classification is shown in Fig. 1. The overall process includes feature extraction, base learners/classifiers selection, training and combination [12]. To improve the connection of each process, IEGP is proposed in this paper to automatically and simultaneously learn informative features and evolve effective ensembles for image classification. As shown in Fig. 1, the inputs of an IEGP solution are images and the outputs are class labels. All the processes such as feature extraction and base learner selection are within a single IEGP solution/program. To achieve this, a novel individual representation, a new function set and a new terminal set are developed in IEGP. This section will describe these three important components of IEGP. Then it outlines the overall algorithm of IEGP for image classification.

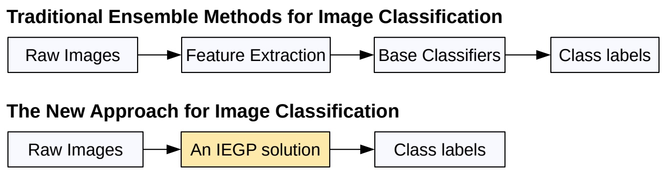

Traditional Ensemble Methods for Image Classification
Raw Images
Feature Extraction
Base Classifiers
Class labels
The New Approach for Image Classification
Raw Images
An IEGP solution
Class labels

Fig. 1. The outlines of the traditional ensemble methods [12] and the new approach (IEGP) for image classification.

A. Novel Individual Representation

GP has a tree-based representation, which is known for variable lengths of evolved solutions. The individual representation of IEGP is based on STGP [26], where each function has input and output types, and each terminal has an output type. To define the type constraints, a new program structure is developed, as shown in Fig. 2. The new program structure has the input, filtering & pooling, feature extraction, concatenation, classification, combination, and output layers. Each layer except for input and output has different functions for different purposes. The input layer represents the input of the IEGP system, such as the terminals. The filtering & pooling layer has filtering and pooling functions, which operate on images. The feature extraction layer extracts informative features from the images using existing feature extraction methods. The concatenation layer concatenates features produced by its children nodes into a feature vector. The filtering & pooling, feature extraction and concatenation layers belong to the process of feature learning, where informative features are learnt from raw images. The learnt features can be directly fed into any classification algorithm for classification. Therefore, a classification layer is connected with the concatenation layer. The classification layer has several classification functions that can be used to train the classifiers using the learnt features. The combination layer has several combination functions to combine the outputs of the classification functions. The classification and combination layers belong to the process of ensemble learning, where the classification functions are selected and trained, and the outputs of the classifiers are combined. Finally, the output layer performs the plurality voting on the outputs produced by the combination layer to obtain the combined predictions of the class labels.

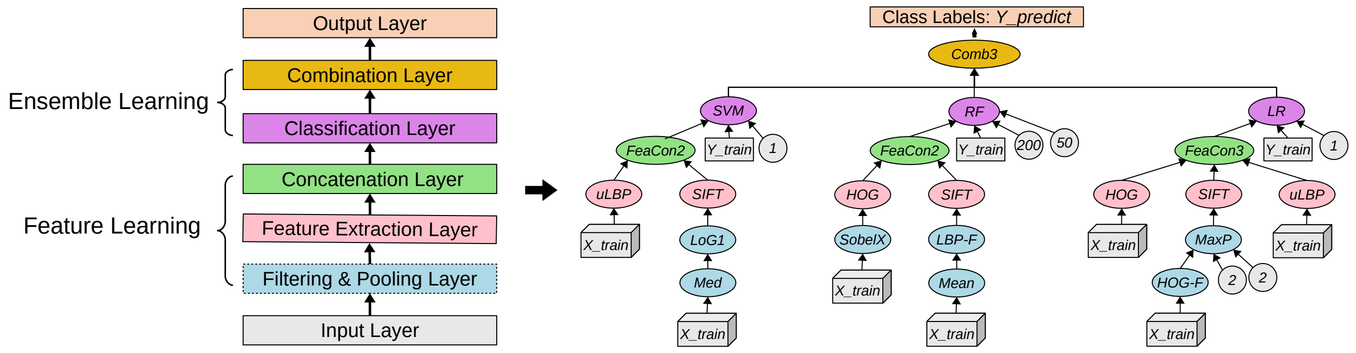


Fig. 2. The new program structure of IEGP and an example solution/program that can be evolved by IEGP. The example program is $Comb3(SVM(FeaCon2(uLBP(X\_train),\ SIFT(LoG1(Med(X\_train)))), Y\_train, 1),\ RF(FeaCon2(HOG(SobelX(X\_train)),\ SIFT(LBP{-}F(Mean(X\_train)))), Y\_train, 200, 50)),\ LR(FeaCon3(HOG(X\_train),\ SIFT(MaxP(HOG(X\_train), 2, 2)),\ uLBP(X\_train)), Y\_train, 1)$

Compared with the program structure of EGP in [25], the new program structure of IEGP has one more layer, i.e., the feature extraction layer. The features learnt by EGP are from the filtering and pooling operations, which may not be robust to variations such as scale and rotation. To improve the effectiveness of the learnt features, a feature extraction layer is inserted between the filtering & pooling layer and the concatenation layer. The feature extraction layer utilises three well-developed methods, i.e., LBP, SIFT, and HOG, to extract invariant and informative features from images, which will be introduced in Section III-B.

An example solution of IEGP is shown in Fig. 2 to further demonstrate the new representation. Fig. 2 shows that different layers have different functions, which will be introduced in the following subsections. In the feature learning layers, i.e., the filtering & pooling layer and the feature extraction layer, the image-related operators, such as $LoG1$, HOG and $SIFT$, are employed. The classification layer has functions such as SVM, RF and LR, which have new additional leaf nodes compared with that in EGP. These leaf nodes are important parameters of the classification functions. In the combination layer, new combination functions, e.g., $Comb3$, are used to combine the predicted class labels in a flexible way. Therefore, the output of the example program in Fig. 2 is the combined output of the SVM, RF and LR classifiers.

1) Diversity of Ensembles: There are four commonly used strategies to enhance the diversity of an ensemble [6], (1) data sample manipulation, (2) input feature manipulation, (3) learning parameter manipulation, and (4) output representation manipulation. The ensembles evolved by IEGP can automatically address the diversity issue, where the strategies are the input feature manipulation and learning parameter manipulation. The new program structure allows that in an evolved program different classification functions have different tree branches, as the example program shown in Fig. 2. The different tree branches with functions can produce various features to form the inputs of the classification functions, which is the input feature manipulation strategy. The parameters of the classification functions are developed as terminals of IEGP, which allows IEGP to automatically fine-tune the parameters during the evolutionary learning process. This is the learning parameter manipulation. In addition, the new program structure allows IEGP to evolve ensembles of the same or different classifiers, which is more flexible than the other ensemble methods with fixed classifiers, such as in [6].

2) Flexibility of Representation: The new program structure enables IEGP to evolve programs with various tree sizes (nodes) and depths. In the new representation, the input layer and the output layer have a fixed tree depth of one. The feature extraction layer transforms images into features, which is irreversible data transformation, so that the depth of this layer is one. Similarly, the depth of the classification layer is one, where the inputs are transformed into class labels. The remaining layers, i.e., the filtering & pooling layer, the concatenation layer, and the combination layer, have a flexible tree depth, which indicates that the tree depth can be automatically adjusted according to the problems. This maintains the flexibility of the evolved solutions by IEGP. Another point is that the filtering & pooling layer may not be necessary for a particular task. Therefore, the filtering & pooling layer is developed as a flexible layer, which indicates that it may be or not be in the evolved IEGP programs. As shown in the example program in Fig. 2, the inputs X_train can be directly fed into the feature extraction functions $uLBP$ and HOG without any filtering & pooling operations.

B. New Function Set

The new function set has a number of different functions for different purposes. The functions for each layer are summarised in Table I. The introduction of these functions is from the bottom layer to the top layer.

TABLE I
New Function Set of IEGP

Layer	Function
Filtering & Pooling	Gau, $GauD$, $Gabor$, Lap, $LoG1$, $LoG2$, $Sobel$, $SobelX$, $SobelY$, $LBP{-}F$, $HOG{-}F$, Med, $Mean$, Min, Max, $Sqrt$, $W{-}Add$, $W{-}Sub$, $ReLU$, $MaxP$
Feature Extraction	$SIFT$, HOG, $uLBP$
Concatenation	$FeaCon2$, $FeaCon3$, $FeaCon4$
Classification	LR, SVM, RF, ERF
Combination	$Comb3$, $Comb5$, $Comb7$

1) Filtering & Pooling Functions: The filtering & pooling functions operate on arrays. They take a number of images as inputs and conduct corresponding operation on each image. The filtering functions keep the size of the output images consistent with the input size. The pooling function reduces the size of images by subsampling maximum values from a small window of the image. In IEGP, the filtering and pooling functions are the same as that in EGP [25]. Gau is the Gaussian filter generated by a 2D Gaussian function $Gau(x,\ y) = \frac{1}{2\pi\sigma^2}\exp[-\frac{x^2+y^2}{2\sigma^2}]$, where the standard deviation σ is a terminal of IEGP. $GauD$ represents the derivative of the Gaussian function with the standard deviation σ in the orders $(o_1,\ o_2)$ along the X axis and the Y axis, respectively. The three parameters are the terminals of IEGP. The kernel of the $Gabor$ function is generated according to $Gabor(x,\ y) = Gau(x,y) * sin[(\frac{2\pi(cos\theta x+sin\theta y)}{\lambda} + \psi)]$. The frequency f $(\frac{1}{\lambda})$ and the orientation θ are developed as terminals of IEGP. Lap is the Laplacian filter, which is commonly used to detect the flat area. In the case that the results produced by Laplacian are noisy, the Laplacian of Gaussian filter is used to convolve the image produced by Laplacian using the Gaussian filter. $LoG1$ and $LoG2$ are the Laplacian of Gaussian filters, where the σ for the Gaussian function is 1 and 2, respectively. $SobelX$, $SobelY$ and $Sobel$ conduct edge detection on the images. $HOG-F$ and $LBP-F$ produce the HOG and LBP images with informative features. Med, $Mean$, Min, and Max convolve images by returning the median, mean, minimum, and maximum values of each 3×3 sliding window, respectively. $Sqrt$ returns the sqrt root of each pixel value and is protected by returning 1 if the pixel value is negative. $W-Add$ and $W-Sub$ take two images and two weights as inputs and calculate the weighted sum or subtraction of the images. In the case that the input images have different sizes, the two functions overlap the two images at the left top point and cut them to have the same sizes for sum or subtraction. $ReLU$ returns 0 if the pixel value is negative, otherwise it returns the pixel value.

2) Feature Extraction Functions: Three commonly used feature extraction methods, uLBP [13], HOG [14] and SIFT [15], are developed as functions for IEGP. These functions transform the input image into a set of effective features, which are invariant to certain variations. The $uLBP$ function extracts 59 histogram features of the LBP image. In $uLBP$, the radium is set to 1.5 and the number of neighbours is set to 8 [13]. The HOG function extracts the mean value of each 4×4 grid from the HOG image to form the feature vector. The parameters for the HOG method are the same as that in [14]. The $SIFT$ function produces 128 features for each image by taking the image as a keypoint [30]. It is noticeable that the three functions produce various numbers of features.

3) Concatenation Functions: The concatenation functions in IEGP are different from that in the EGP method [25]. The $FeaCon2$, $FeaCon3$ and $FeaCon4$ functions convert two, three and four vectors as a vector by concatenating them, respectively. The concatenation functions can take the feature extraction functions or the concatenation functions as their children nodes, which indicates that a combination of features is produced by each concatenation function.

4) Classification Functions: Any commonly used classification algorithm can be developed as functions in the classification layer for IEGP. To narrow the search space, the previous EGP method [25] employs six classification algorithms, LR, k-nearest neighbour (KNN), SVM, RF, extremely randomised trees (ERF), and $AdaBoost$, according to [41]. However, since KNN is an instance-based method and is computationally expensive when the training data is large, the KNN function is not employed in IEGP. The $AdaBoost$ method achieves an inferior performance on image classification [25] so that it is removed from the function set. Therefore, only four classification functions are employed in IEGP. They are LR, SVM, RF, and ERF, including linear classification methods and tree-based classification methods. Note that RF and ERF are ensemble methods and the IEGP approach can build ensembles of ensembles.

The previous EGP method fixes the parameters for the classification functions, which is not effective and efficient for solving different tasks. Therefore, the key parameters for LR, SVM, RF, and ERF are developed as terminals of IEGP to allow their values to be automatically generated or optimised during the evolutionary process. These key parameters and their ranges are introduced in Section III-C.

5) Combination Functions: The previous EGP method [25] conducts voting multiple times, which is computationally expensive. To address this, three new functions are developed in IEGP to combine the classifiers in an effective way. The functions are $Comb3$, $Comb5$ and $Comb7$, which take 3, 5 and 7 class labels as inputs and concatenate these labels to form the output, respectively. These functions can be root nodes or internal nodes of a GP tree, which represents different ways of combining the classifiers, as shown in Fig. 3. In the left example (Ensemble 1) of Fig. 3, the $Comb3$ function is used as the root node. The inputs for $Comb3$ are 1 from SVM, 1 from RF, and 0 from LR. The $Comb3$ function combines these inputs and returns 110. In the right example (Ensemble 2) of Fig. 3, $Comb3$ is an internal node and $Comb5$ is the root node. The outputs of $Comb3$ are 012 and the outputs of $Comb5$ are 31001221 from the nodes of SVM, RF, $Comb3$, ERF, and LR. After obtaining the outputs from the root node, the plurality voting is conducted to produce the predicted class label for an instance/image. As shown in the Fig. 3, the final output (class label) of Ensemble 1 is 1 (from 110) and the final output (class label) of Ensemble 2 is 1 (from 3101221).

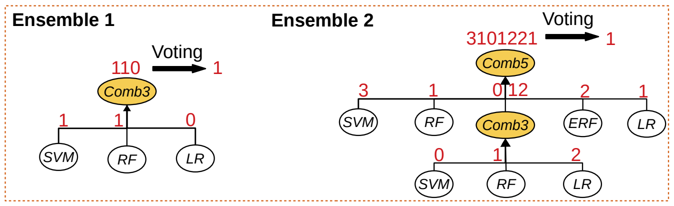


Fig. 3. Example ensembles with the new $Comb3$ and $Comb5$ functions as root nodes and internal nodes.

C. New Terminal Set

The terminal set represents the inputs of the IEGP system. Table II lists the terminals and their ranges of IEGP. The $X\_train = \{X_i\}_{i=1}^N$ represents N input training images and the $Y\_train = \{Y_i\}_{i=1}^N$ represents the class labels of the N images. In X_train, X_i represents an image of size $M \times L$, where the pixel values in the image are in the range of $[0,\ 1]$. In Y_train, Y_i is an integer representing the class label. The

σ, o_1, o_2, θ, n_1, n_2, k_1, and k_2 terminals are the parameters for particular functions in the filtering & pooling layer and their values are in the commonly used ranges, respectively [25].

TABLE II
TERMINAL SET

Terminal	Type	Description
X_train	Array	N training images with a size of $M \times L$
Y_train	Array	The class labels of N training images
σ	Integer	The standard deviation of the Gaussian filter in the Gau and $GauD$ functions. Its range is $\{1,\ 2,\ 3\}$
o_1, o_2	Integer	The orders of the Gaussian derivatives. They are in range $\{0,\ 1,\ 2\}$
θ	Float	The orientation of the $Gabor$ function. It is in the range of $[0,\ 7\pi/8]$ with a step of $\pi/8$ [42]
f	Float	The frequency of the $Gabor$ function. It equals to $(\pi/2)/(\sqrt{2}^v)$, where v is an integer in the range of $[0,\ 4]$ [42]
n_1, n_2	Float	The parameters for the *W-Add* and *W-Sub* functions. They are randomly generated from the range of $[0,\ 1)$
k_1, k_2	Integer	The kernel size for $MaxP$. They are in range $\{2,\ 4\}$
C	Integer	The penalty term/parameter in LR and SVM is 10^C, where C is in the range of $[-2,\ 5]$ according to [43]
NT	Integer	The number of trees in RF and ERF. It is in the range of $[50,\ 500]$ with a step of 10
MD	Integer	The maximum tree depth of the decision tree in RF and ERF. It is in the range of $[10,\ 100]$ with a step of 10

The important parameters for the classification functions are designed as terminals of IEGP. The parameters are C, NT and MD. The C terminal is an integer and is related to the penalty term/parameter for the classification functions LR and SVM. The range for C is set to $[-2,\ 5]$ [43], resulting the penalty term/parameter (10^C) in the range of $\{10^{-2},\ 10^{-1},\ \ldots,\ 10^4,\ 10^5\}$. The number of decision trees and the maximum tree depth are two important parameters for RF and ERF according to [6]. Therefore, they are developed as terminals, NT and MD. The values for NT is in the range of $[50,\ 500]$ with a step of 10 and the values for MD is in the range of $[10,\ 100]$ with a step of 10. The maximum values for NT and MD are set according to that in [6]. To reduce the computational cost, a smaller NT and MD are desired to be found for RF and ERF. In addition, a step of 10 is used to avoid a large search space of IEGP.

D. Overall Algorithm

With the new program structure, the new function set and the new terminal set, IEGP can automatically learn features and ensembles for image classification. The overall algorithm of IEGP is described in Algorithm 1. The flowchart of the overall algorithm (include the training and test processes) is shown in Fig. 4.

The IEGP approach starts with randomly initialising the population P_0 using the *ramped half-and-half* method. Each individual in P_0 is evaluated by a fitness function. During the evolutionary learning process (at generation g), the elitism, subtree crossover and subtree mutation operators are used to create a new population P_g. The new P_g is then evaluated. When g equals to the maximum number of generations, the evolutionary learning process stops and the best individual (ensemble) is returned as the output.

Algorithm 1: Framework of IEGP

```
Input  : $X\_train$: $N$ training images; $Y\_train$: the class labels
         of $N$ training images.
Output : $Best\_Individual$: the best program tree.
1  $P_0 \leftarrow$ Initialise a population based on the new representation, the
     new function set and the new terminal set;
2  $g \leftarrow 0$;
3  $Cache\_Table \leftarrow \emptyset$;
4  for each individual $p$ in $P_0$ do
5      $f_p \leftarrow$ Evaluate $p$ using the fitness function
6  end
7  $Cache\_Table \leftarrow P_0$;
8  while $g < G$ do
9      $I \leftarrow$ Best individuals of $P_g$ using elitism operator;
10     $S \leftarrow$ Selected individuals from $P_g$ by tournamanet selection;
11     $O_{g+1} \leftarrow$ Offspring generated from $S$ using subtree crossover
         and subtree mutation;
12     for each individual $o$ in $O_{g+1}$ do
13         if $o$ in $Cache\_Table$ then
14             $f_o \leftarrow$ the fitness value of $o$ in $Cache\_Table$;
15         else
16             $f_o \leftarrow$ Evaluate $o$ using the fitness function;
17         end
18     end
19     $P_{g+1} \leftarrow O_{g+1} \cup I$;
20     Update $Best\_Individual$ and $Cache\_Table$;
21     $g \leftarrow g + 1$;
22 end
23 Return $Best\_Individual$.
```

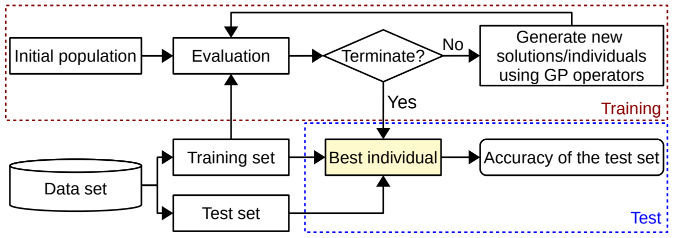


Fig. 4. The flowchart of the overall IEGP algorithm. It includes the flowchart of the training and test processes.

During the evolutionary learning process, a subtree caching method is used to reduce the evaluation time since GP is known of computationally expensive on image data [44]. A $Cache\_Tabel$ is used to store individuals and their fitness values. At generation 0, the $Cache\_Table$ stores the P_0 with fitness values. At generation g, the $Cache\_Table$ stores the best individuals of the past generations and all the individuals in generation $g-1$ [44]. To evaluate individual o at g ($g > 1$), a search is conducted in $Cache\_Table$ to check whether o is evaluated before. If o is in the $Cache\_Table$, the fitness value is directly assigned to o, otherwise, o is evaluated using the fitness function. Generally, any individuals can be stored in $Cache\_Table$ but a tradeoff between the searching time and the evaluation time for an individual should be considered. Therefore, in IEGP, the size of $Cache\_Table$ is set to $5 \times N_p$ (N_p is the population size), which is the same as that in [25].

1) Training Process and Fitness Function: In the training process of IEGP, the training data X_train and Y_train are fed into the IEGP system. Since the classification functions in each IEGP tree/individual have a training process. We employ stratified k-fold cross-validation on the training set (X_train and Y_train) to build and evaluate the classifier in the training process of IEGP. For each classification function in the evolved IEGP tree, each time $k-1$ folds are used to train the classifier and the remaining one fold is used to test the classifier. The predicted class labels for the one fold are recorded. This process repeats k times to obtain the predicted class label for each instance in X_train. The value of k is set to 3 according to [6]. The predicted class labels from different nodes (classification functions) are then combined by the combination functions and voted by the plurality voting to form the output $Y\_predict$.

The fitness function for IEGP is the classification accuracy, which is the percentage of the number of correctly classified images out of the total number of images in the training set. In IEGP, the classification accuracy is calculated according to $Y\_predict$ and Y_train.

2) Test Process: The test process is to evaluate the best IEGP tree on an unseen data set. In this process, the classification algorithms in the best IEGP tree are trained using the transformed X_train and Y_train without k-fold cross-validation (X_train were transformed into features by certain nodes in the IEGP tree). Then the IEGP tree, which is an ensemble of trained classifiers, is applied to the unseen data X_test to obtain the class labels. Based on the class labels, the accuracy of the test set is calculated and reported.

IV. Experiment Design

A number of experiments have been conducted to show the effectiveness of the new approach. The detailed design of experiments is described in this section.

A. Data Sets

To show the performance of the proposed approach, 13 well-known benchmark data sets of varying difficulty are employed in the experiments. Nine data sets are the same as that employed in [25]. Besides the nine data sets, four new data sets with a large size of training and test data are employed to conduct experiments. Therefore, in the experiments, 13 data sets are used, which are FEI_1 [45], FEI_2 [45], JAFFE [46], ORL [47], KTH (KTH-TIPS2) [48], FS (13 natural scene categories) [49], MB (mnist-basic) [50], MRD (mnist-rot) [50], MBR (mnist-back-rand) [50], MBI (mnist-back-image) [50], Rectangle [50], RI (rectangle-image) [50], and Convex (convex sets) [50]. These data sets include a broad variety of image classification tasks, i.e., facial expression classification: FEI_1, FEI_2 and JAFFE; face recognition: ORL; texture classification: KTH; scene classification: FS; and object classification: MB, MRD, MBR, MBI, Rectangle, RI, and Convex. In the object classification tasks, different image variations are included, such as the random background in MBR and RI, rotations in MRD, and additional image background in MBI. The variety of image classification tasks and the image variations are two main considerations of selecting these benchmark data sets, which can be employed to comprehensively investigate the performance of the proposed approach on different types of image classification tasks.

Table III lists the detailed information of the 13 data sets. To simplify, all the data sets are numbered. The images of the data sets 1-6 are resized or converted to gray-scale images in order to reduce the computational cost. Figures 5 - 7 show several example images of the 13 data sets. The data sets 1-6 are split using a proportion to form the training set and the test set. The proportion is set according to the size of the data set to obtain a balanced training set. The number of training images per class of the data sets 1-6 is shown in the bracket of Table III. In contrast, data sets 7-13 are public data sets and have the separated training and test sets as listed in Table III, which can be directly used in experiments [1].

TABLE III
BENCHMARK DATA SETS

No.	Data Set	Image Size	Training Set Size	Test Set Size	#Class
1	FEI_1	60×40	150 (75)	50	2
2	FEI_2	60×40	150 (75)	50	2
3	JAFFE	55×55	140 (20)	73	7
4	ORL	50×55	240 (6)	160	40
5	KTH	50×50	480 (48)	330	10
6	FS	55×55	1300 (100)	2559	13
7	MB	28×28	12000	50000	10
8	MRD	28×28	12000	50000	10
9	MBR	28×28	12000	50000	10
10	MBI	28×28	12000	50000	10
11	Rectangle	28×28	1200	50000	2
12	RI	28×28	12000	50000	2
13	Convex	28×28	8000	50000	2

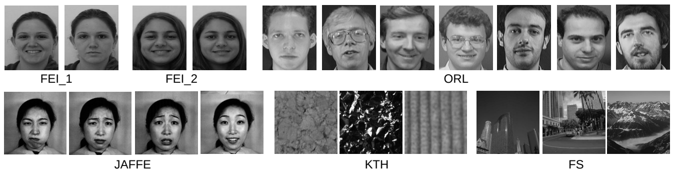


Fig. 5. Example images from the FEI_1, FEI_2, JAFFE, ORL, KTH, and FS data sets.

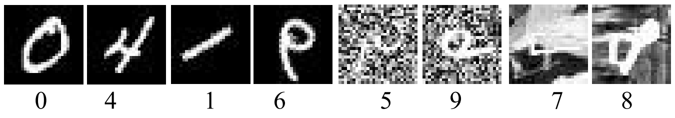


Fig. 6. Two example images from the MB, MRD, MBR, and MBI data sets, respectively. The class label of each image is listed below the image.

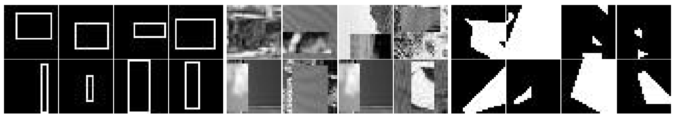


Fig. 7. Eight example images from the Rectangle, RI, and Convex data sets, respectively. Each row shows the images in one class.

B. Benchmark Methods

A large number of effective algorithms are used as benchmark methods for comparisons to show the effectiveness of IEGP. Since the data sets 7-13 have public training and test sets, the test accuracy reported in literature can be directly used for comparisons without reimplementation of the methods. We have collected these results from corresponding references on data sets 7-13. Therefore, there are 19 comparison methods on data sets 7-13, i.e., SVM+RBF [50], SVM+Poly [50], SAE-3 [35], DAE-b-3 [35], CAE-2 [35], SPAE [51], RBM-3 [35], ScatNet-2 [32, 33], RandNet-2 [33], PCANet-2 (softmax) [33], LDANet-2 [33], NNet [50], SAA-3 [50], DBN-3 [50], FCCNN

[1]The training and test sets of data sets 7-13 can be download from http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/PublicDatasets

[34], FCCNN (with BT) [34], SPCN [31], EGP [25], and EvoCNN [52]. Note that most of these methods are neural network-based methods and parameter tuning was conducted in some methods to obtain a good classification performance.

For the data sets 1-6, we use 13 different benchmark methods for comparisons. The 13 benchmark methods include traditional image classification methods, where the aim is to comprehensively investigate whether IEGP can learn informative features and evolve effective ensembles for image classification. These benchmark methods are six classification algorithms using raw pixels, i.e., SVM, KNN, LR, RF, AdaBoost, and ERF, four SVM methods using different features, i.e., uLBP+SVM, LBP+SVM, HOG+SVM, and SIFT+SVM, two CNNs, i.e., CNN-5 and CNN-8, and the previous EGP method [25]. The SVM, KNN, LR, RF, AdaBoost, and ERF methods take raw pixels as inputs and the classifiers are trained for classification. The uLBP+SVM, LBP+SVM, HOG+SVM, and SIFT+SVM methods use uLBP, LBP, HOG, and SIFT features as inputs, respectively, and train classifiers using these features. The two CNN methods have different architectures, i.e., 5 layers (CNN-5) and 8 layers (CNN-8). More details of these methods can be found in [25].

C. Parameter Settings

The parameter settings for IEGP are commonly used settings from the GP community, which are also the same as that for EGP [25, 53]. The population size is 100 and the maximum number of generations is 50. The elitism rate is 0.01, the crossover rate is 0.8 and the mutation rate is 0.19. The Tournament selection with a size of 7 is used to select individuals for mutation and crossover during the evolutionary learning process. The tree depth is between 2 and 8. In IEGP, the type constraint has high priority than the depth constraint so that the tree depth may over 8 in some cases. Note that we use the same parameter settings for IEGP on different data sets for generality, although tuning/optimising parameters could improve its performance.

The implementation of IEGP is in Python using the DEAP (*Distributed Evolutionary Algorithm in Python*) [54] package. The implementations of the classification algorithms in IEGP and the benchmark methods are based on the *scikit-learn* package [55] and the *Keras* package [56]. Note that the other parameters (except for the optimised ones) of these classification algorithms are the default settings in *scikit-learn* for simplification. The parameters settings for the benchmark methods on data sets 1-6 are described in [25]. The experiment of IEGP on each data set runs independent 30 times and the best tree of each run is tested on the test set.

V. Results and Discussions

This section discusses and analyses the classification performance of the proposed IEGP method and the benchmark methods including the EGP method on the 13 data sets.

A. Classification Accuracy on Data Sets 1-6

The classification results, i.e., the maximum accuracy (Max), the average accuracy and standard deviation (Mean±St.dev) of 30 runs by the IEGP method and the benchmark methods on the six data sets are listed in Table IV. The Wilcoxon rank-sum test with a 5% significance level is used to compare the IEGP method with a benchmark method to show the significance of the differences. The symbols "+", "–" and "=" in Table IV indicate that IEGP is significantly better, significantly worse or similar than/to the benchmark method. The final row of each table summaries the overall results of the significance test. The best accuracy and mean accuracy of each data set are highlighted in bold in Table IV.

Table IV shows that the proposed IEGP method performs significantly better than or similarly to any of the benchmark methods on FEI_1 and FEI_2, which are facial expression classification tasks. IEGP finds the best accuracy of 100% on these two data sets. Although RF achieves the best mean accuracy on FEI_1 and EGP achieves the best mean accuracy on FEI_2, there is no significant difference between IEGP and EGP or RF in the performance on the two data sets, which indicates that IEGP can achieve similar performance to the best methods on the two binary classification data sets.

The classification results on the JAFFE and ORL data sets in Table IV show that IEGP achieves significantly better results than eight methods on JAFFE and than any of these benchmark methods on ORL. JAFFE is a facial expression classification task with seven different expressions, where to learn informative features is difficult. IEGP can achieve comparable performance than most benchmark methods on JAFFE. On the ORL data set, IEGP is significantly better than any of the benchmark methods. ORL is a face recognition task of 40 classes and has a very small number of training images, which is challenging for some methods needing a large number of training instances such as CNN-5 and CNN-8. IEGP found the best accuracy of 100% and the best mean accuracy of 98.29% on ORL, which shows the effectiveness of IEGP on the data set with a small number of training instances.

The classification results on the KTH and FS data sets in Table IV indicates that IEGP achieves significantly better results than any of these benchmark methods, including the EGP method. The KTH data set has texture images and the FS data set has natural scene images. The SVM, KNN, LR, RF, and AdaBoost methods achieve very low accuracy on these two data sets, which indicates that using raw pixels to classify these two data sets is not effective. However, simple feature extraction cannot improve the accuracy as the uLBP+SVM, LBP+SVM, HOG+SVM, and SIFT+SVM methods achieve low accuracy as well. With automatically learning features and evolving ensemble for classification, IEGP achieves the best results on these two data sets, i.e., a maximum accuracy of 98.48% on KTH and a maximum accuracy of 92.45% on FS. Importantly, IEGP improves the mean accuracy by 13.72% on KTH and by 28.56% on FS. The results indicate that IEGP is very effective for texture classification and scene classification.

Table V summaries the overall results of the significance tests in the comparisons of different benchmark methods on the six data sets. IEGP achieves significantly better results than the six classification algorithms using raw pixels in 33 comparisons (in case A), which indicates that IEGP can learn informative features for effective image classification.

TABLE IV
CLASSIFICATION ACCURACY (%) OF IEGP AND 13 BASELINE METHODS ON DATA SETS 1-6

Method	**FEI_1**		**FEI_2**		**JAFFE**		**ORL**		**KTH**		**FS**	
	Max	Mean±St.dev	Max	Mean±St.dev	Max	Mean±St.dev	Max	Mean±St.dev	Max	Mean±St.dev	Max	Mean±St.dev
SVM	90.00	90.00±0.00+	88.00	88.00±0.00+	93.94	**91.06±0.73**–	94.38	94.38±0.00+	46.97	44.59±2.83+	20.63	20.30±0.15+
KNN	32.00	32.00±0.00+	8.00	8.00±0.00+	71.21	71.21±0.00+	94.38	94.38±0.00+	34.24	34.24±0.00+	24.35	24.35±0.00+
LR	92.00	92.00±0.00+	88.00	88.00±0.00+	89.39	89.39±0.00–	93.75	93.75±0.00+	48.79	48.79±0.00+	23.49	23.49±0.00+
RF	98.00	**97.07±1.01**=	90.00	89.20±1.13+	75.76	72.48±1.99+	93.12	92.33±0.63+	60.00	57.81±0.83+	37.36	36.53±0.49+
AdaBoost	80.00	78.67±1.32+	80.00	76.00±3.44+	53.03	47.93±2.68+	59.38	52.27±4.00+	37.88	33.44±1.37+	17.47	13.04±1.47+
ERF	94.00	93.27±0.98+	92.00	90.60±0.93+	77.27	73.89±1.72+	97.50	96.71±0.59+	61.52	59.83±0.86+	37.94	37.15±0.36+
uLBP+SVM	66.00	56.73±3.66+	68.00	62.53±3.52+	31.82	26.87±3.30+	87.50	87.42±0.21+	78.79	73.29±4.18+	49.79	33.27±8.90+
LBP+SVM	68.00	64.60±1.83+	74.00	69.80±0.00+	33.33	28.84±2.05+	88.12	87.52±0.20+	83.64	82.71±0.51+	53.50	50.45±1.80+
HOG+SVM	96.00	96.00±0.00=	82.00	82.00±0.00+	81.82	80.30±0.40+	91.25	91.25±0.00+	57.27	55.96±0.64+	12.11	7.91±2.47+
SIFT+SVM	56.00	56.00±0.00+	62.00	62.00±0.00+	33.33	33.33±0.00+	93.75	93.75±0.00+	65.76	65.76±0.00+	60.92	60.92±0.00+
CNN-5	98.00	95.40±1.30+	98.00	95.27±1.62+	**95.45**	90.96±2.68–	96.88	95.29±1.06+	85.76	82.56±1.87+	50.14	48.03±1.16+
CNN-8	98.00	95.33±1.32+	96.00	90.93±1.87+	90.91	84.54±4.33=	95.00	93.04±1.09+	76.36	71.63±3.18+	49.16	46.79±1.01+
EGP	**100.0**	96.20±2.06=	**100.0**	**98.07±1.70**=	92.42	84.24±4.28=	99.38	97.44±1.26+	87.88	77.53±5.17+	67.17	61.07±2.91+
IEGP	**100.0**	96.67±2.55	**100.0**	96.20±3.66	92.42	82.17±5.42	**100.0**	**98.29±0.97**	**98.48**	**96.43±1.26**	**92.54**	**89.63±1.47**
Overall		10+, 3=		12+, 1=		8+, 2=, 3–		13+		13+		13+

TABLE V
SUMMARY OF SIGNIFICANCE TEST ON DATA SETS 1-6

	A	B	C	D
Significantly better (+)	33	23	10	3
Similar (=)	1	1	1	3
Significantly worse (–)	2	0	1	0

A: compare IEGP with SVM, KNN, LR, RF, AdaBoost, and ERF
B: compare IEGP with uLBP+SVM, LBP+SVM, HOG+SVM, and SIFT+SVM
C. compare IEGP with CNN-5 and CNN-8
D. compare IEGP with EGP

Compared with the four methods (in case B), which using uLBP, LBP, SIFT, and HOG features as inputs for SVM, IEGP achieves significantly better or similar results than/to any of them. The results show that IEGP is more effective than the four methods by learning informative features and evolving an ensemble for classification. Compared with CNN-5 and CNN-8 (in case C), IEGP only achieves significantly worse results in one comparison, which shows that IEGP is more effective than simple CNN methods. The main advantage of IEGP over the CNN methods is the flexible length and depth of the evolved solutions. IEGP does not need to pre-define the model/solution structure (complexity) as it is able to find a suitable model during the evolutionary learning process. IEGP performs significantly better than or similarly to EGP (in case D). Compared with EGP, the proposed IEGP method has a new representation, a new function set and a new terminal set, which allow it to learn more effective features and to find more suitable classification algorithms with appropriate parameters to form the ensemble for classification.

B. Classification Accuracy on Data Sets 7-13

The classification accuracy (%) of the data sets 7-13 are listed in Table VI. On these data sets, 19 methods are employed for comparisons, where the results are collected from the corresponding references. Note that some of these 19 methods may not have results on some data sets such as RI, Rectangle and Convex. In Table VI, each column shows the results of all these methods on one data set. The results obtained by IEGP are listed at the bottom of the table. Because most of these benchmark methods have only reported the best results on these data sets, we compare IEGP with them using the best results. In Table VI, the symbol "+" denotes that IEGP achieves better accuracy than the corresponding benchmark method. The final two rows of Table VI summarise the ranking results of IEGP among all these benchmark methods.

Table VI shows that IEGP achieves better classification accuracy than any of the benchmark methods with reported results on two data sets, i.e., Rectangle and Convex. On the MB, MRD and RI data sets, the best accuracy of IEGP ranks second among all the methods, which indicates that only one method achieves better accuracy than IEGP on any of the three data sets. On the remaining two data sets, i.e., MBR and MBI, the best accuracy of IEGP ranks third among all the methods, which indicates that only two methods achieve better accuracy than IEGP on any of the two data sets. Importantly, IEGP improves the accuracy by 2.45% on Convex. IEGP achieves 100% accuracy on the Rectangle data set, although it is only 0.01% higher than the best results of the benchmark methods. Note that these benchmark methods have been explored extensively on these data sets, therefore, even 1% improvement in accuracy is very difficult to achieve.

On MB, the IEGP method achieves better (or the same) results than any of the 18 benchmark methods except for LDANet-2. IEGP achieves a maximum accuracy of 98.82%, which is slightly less than the accuracy of 98.95% achieved by LDANet-2. Although IEGP achieves worse results than LDANet-2 on MB, it achieves better results on the other six data sets. The three variants of MB, i.e., MRD, MBR, and MBI, are more difficult than MB by adding variational factors including rotation and background change. On MRD, IEGP achieves better (or the same) results than any of the 18 benchmark methods except for EvoCNN. On MBR and MBI, IEGP is better than any of the 18 benchmark methods except for SPCN and EvoCNN, which are CNN-based methods. SPCN is more effective than IEGP on these two data sets but less effective on the other four data sets. Especially, IEGP achieves 94.28% accuracy on MRD which is much higher than the accuracy (90.19%) achieved by SPCN. EvoCNN is a state-of-the-art CNN-based method by automatically evolving the architectures of CNNs. Compared with EvoCNN, IEGP achieves worse results on the difficult data sets, i.e., MBR and MBI, but IEGP achieves better (same) results on the MB, Rectangle and Convex data sets. This indicates that IEGP as a pure GP method is effective and promising for image classification. On the Rectangle data set, IEGP achieves 100% accuracy. RI as a variant of Rectangle is more difficult. IEGP achieves the best accuracy among all the methods on RI except for EvoCNN. But the difference of the best results achieved

TABLE VI
CLASSIFICATION ACCURACY (%) OF IEGP AND 19 EFFECTIVE METHODS ON DATA SETS 7-13

Method	MB	MRD	MBR	MBI	Rectangle	RI	Convex
SVM+RBF [50]	96.97(+)	88.89(+)	85.42(+)	77.39(+)	97.85(+)	75.96(+)	80.87(+)
SVM+Poly [50]	96.31(+)	84.58(+)	83.38(+)	75.99(+)	97.85(+)	75.95(+)	80.18(+)
SAE-3 [35]	96.54(+)	89.70(+)	88.72(+)	77.00(+)	97.86(+)	75.95(+)	–
DAE-b-3 [35]	97.16(+)	90.47(+)	89.70(+)	83.32(+)	98.01(+)	78.41(+)	–
CAE-2 [35]	97.52(+)	90.34(+)	89.10(+)	84.50(+)	98.79(+)	78.46(+)	–
SPAE [51]	96.68(+)	89.74(+)	90.99(+)	86.76(+)	–	–	–
RBM-3 [35]	96.89(+)	89.70(+)	93.27(+)	83.69(+)	97.40(+)	77.50(+)	–
ScatNet-2 [32, 33]	98.73(+)	92.52(+)	87.70(+)	81.60(+)	99.99(+)	91.98(+)	93.50(+)
RandNet-2 [33]	98.75(+)	91.53(+)	86.53(+)	88.35(+)	99.91(+)	83.00(+)	94.55(+)
PCANet-2 (softmax) [33]	98.60(+)	91.48(+)	93.15(+)	88.45(+)	99.51(+)	86.61(+)	95.81(+)
LDANet-2 [33]	**98.95**	92.48(+)	93.19(+)	87.58(+)	99.86(+)	83.80(+)	92.78(+)
NNet [50]	95.31(+)	81.89(+)	79.96(+)	72.59(+)	92.84(+)	66.80(+)	67.75(+)
SAA-3 [50]	96.54(+)	89.70(+)	88.72(+)	77.00(+)	97.59(+)	75.95(+)	81.59(+)
DBN-3 [50]	96.89(+)	89.70(+)	93.27(+)	83.69(+)	97.40(+)	77.50(+)	81.37(+)
FCCNN [34]	97.57(+)	91.09(+)	93.55(+)	86.77(+)	–	–	–
FCCNN (with BT) [34]	97.32(+)	90.41(+)	93.03(+)	89.20(+)	–	–	–
SPCN [31]	98.18(+)	90.19(+)	94.16	90.45	99.81(+)	89.40(+)	–
EvoCNN (best) [52]	98.82	**94.78**	**97.20**	**95.47**	99.99(+)	**94.97**	95.18(+)
EGP (best) [25]	97.19(+)	–	–	–	99.91(+)	–	93.97(+)
IEGP (best)	98.82	94.28	93.59	89.41	**100**	94.88	**98.26**
IEGP (mean)	98.69	93.78	92.65	88.42	99.94	89.02	97.76
IEGP (std)	0.08	0.24	0.35	0.64	0.05	2.1	0.26
Rank	2/20	2/19	3/19	3/19	**1/17**	2/16	**1/12**

by EvoCNN and IEGP on RI is very small, i.e., 94.97% (EvoCNN) vs. 94.88% (IEGP). On the Convex data set, IEGP obtains the best accuracy of 98.26% among all the methods, which is 3% higher than that achieved by EvoCNN.

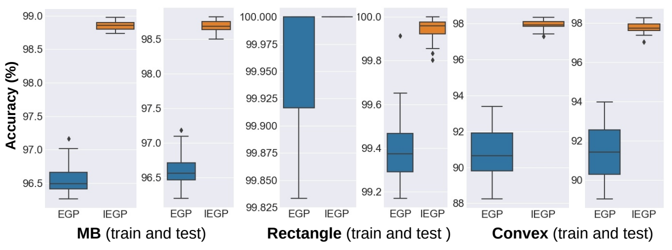


Fig. 8. Comparison of the distribution of the training and test results obtained by EGP and IEGP on the MB, Rectangle and Convex data sets.

Compared with EGP, IEGP is more effective for classifying large-scale data sets. On the three data sets, MB, Rectangle, and Convex, IEGP achieves better results than EGP. Importantly, IEGP achieves a maximum accuracy of 98.26% on Convex, which is 4% higher than that achieved by EGP. To further compare EGP with IEGP, Fig. 8 show the distributions of the training accuracy and test accuracy of EGP and IEGP on the MB, Rectangle, and Convex data sets. From this figure, it is clear that IEGP has much higher accuracy and median accuracy than EGP on the three data sets. This indicates that IEGP achieves better performance than EGP. In addition, the results obtained by IEGP are more clustered than that by EGP, which means that IEGP is more stable than EGP. The results demonstrate that IEGP significantly improves the EGP method by having a new representation and a new function set for image classification.

VI. FURTHER ANALYSIS

This section analyses the evolved trees/solutions by IEGP to further understand what features they extract and what classifiers in the evolved ensembles are built for image classification.

A. Visualisation of Example Solutions

1) An Example Solution on MRD: Since IEGP achieves the best results on the MRD data set, the best program/solution is selected from MRD for analysis and visualisation. The best solution is visualised in Fig. 9. This solution achieves 93.8% accuracy on the training set of MRD and 94.28% accuracy on the test set. Note that the example solution is used for testing so that the X_train node is replaced with the $Images$ node and the Y_train node is removed for simplification. It is clear from Fig. 9 that the example solution is an ensemble of three ERF classifiers with different parameters. The left and right classifiers have 450 decision trees with a maximum tree depth of 60, while the middle classifier has 450 decision trees with a maximum tree depth of 30. It is obvious that the example solution is an ensemble of ensembles.

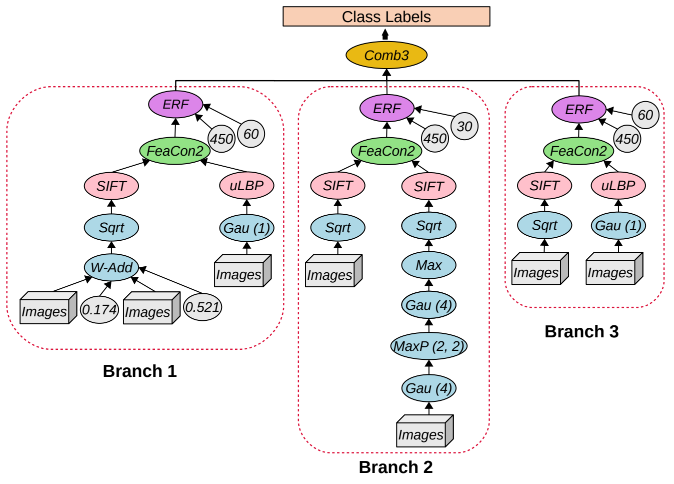


Fig. 9. An example solution found by IEGP on the MRD data set. The example program is $Comb3(ERF(FeaCon2(SIFT(Sqrt(W{-}Add(Images, 0.174, Images, 0.521))), uLBP(Gau(Images, 1))), 450, 60), ERF(FeaCon2(SIFT(Sqrt(Images)), SIFT(Sqrt(Max(Gau(MaxP(Gau(Images, 4), 2, 2), 4))))), 450, 30), ERF(FeaCon2(SIFT(Sqrt(Images)), uLBP(Gau(Images, 1))), 450, 60))$

As shown in this figure, it is obvious that the three classifiers are trained using different features. The first branch uses features that are the combination of 128 SIFT features and 59 uLBP features. Before extracting features using $SIFT$ and $uLBP$, each image is processed by the $Sqrt$ function or the Gau function (with standard deviation of 1). The second

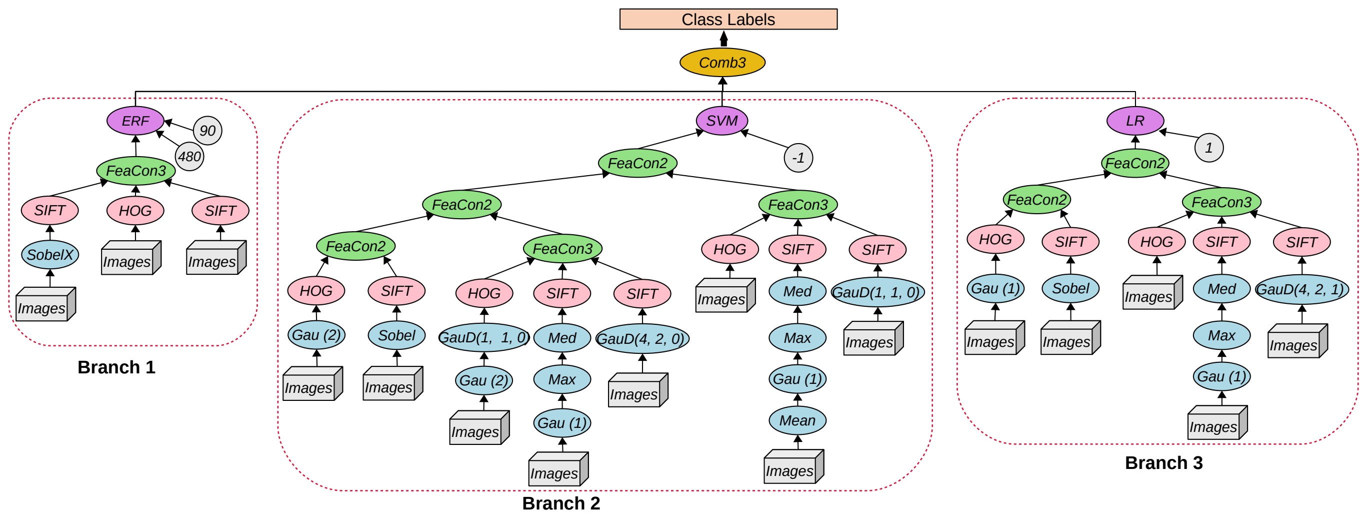


Fig. 10. An example solution found by IEGP on the MBR data set. The example program is $Comb3(ERF(FeaCon3(SIFT(SobelX(Images)), HOG(Images), SIFT(Images)), 480, 90), SVM(FeaCon2(FeaCon2(FeaCon2(HOG(Gau(Images, 2)), SIFT(Sobel(Images))), FeaCon3(HOG(GauD(Gau(Images, 2), 1, 1, 0)), SIFT(Med(Max(Gau(Images, 1)))), SIFT(GauD(Images, 4, 2, 0)))), FeaCon3(HOG(Images), SIFT(Med(Max(Gau(Mean(Images), 1)))), SIFT(GauD(Images, 1, 1, 0)))), -1), LR(FeaCon2(FeaCon2(HOG(Gau(Images, 1)), SIFT(Sobel(Images))), FeaCon3(HOG(Images), SIFT(Med(Max(Gau(Images, 1)))), SIFT(GauD(Images, 4, 2, 1)))), 1))$

branch uses 256 (128×2) SIFT features extracted from the images after corresponding transformations, such as $Sqrt$, Max, Gau, and $MaxP$. The third branch uses 128 SIFT features and 59 uLBP features for classification. The $Sqrt$ function and the Gau function are employed to rescale and smooth the images before feature extraction. The $SIFT$ and $uLBP$ functions transform the processed images into features and the features are fed into the classification functions. By this analysis, it is clear that each branch extracts various numbers and types of features from images after corresponding filtering or pooling operations and uses these features to build different classifiers. This indicates that the inputs for each classifier in the example solution are different, which enhance the diversity of the classifiers in the ensemble.

2) An Example Solution on MBR: An example solution on the MBR data set is visualised in Fig. 10. This solution achieves 93.38% accuracy on the training set and 93.21% accuracy on the test set. Different from the solution in Fig. 9, where the classifiers in the ensemble are trained from the same classification algorithm, this example solution is an ensemble of three classifiers trained from different classification algorithms. This shows that IEGP can evolve ensembles of the same or different classifiers, which is very flexible for solving different tasks.

In the ensemble in Fig. 10, the first classifier is ERF, having 480 decision trees with a maximum tree depth of 90. The second classifier is SVM, where the value of the penalty parameter is 10^{-1}. The third classifier is LR, where the value of the penalty parameter is 10. From Fig. 10, we can see that the three classifiers are trained using features extracted by different feature extraction functions in their children branches. The features extracted from this data set are the combinations of the SIFT features and the HOG features. Meanwhile, different filtering functions are employed to process the image before feature extraction. Accordingly, different inputs for the three classifiers further enhance the diversity of the constructed ensemble.

B. Further Analysis on Example Solutions

To further analyse the performance of the example ensembles in Fig. 9 and Fig. 10, we calculate the accuracy of each classifier in the example ensembles on the test set of MRD and MBR, respectively. Table VII lists the results obtained by the example ensembles (the second row) and the results obtained by each classifier in the branches circled in Fig. 9 and Fig. 10 (the third to the fifth rows). In addition, we use the raw pixels as inputs to train three classification algorithms that are used to build the example ensembles in Fig. 9 and Fig. 10, and build a new ensemble of the trained classifiers using the plurality voting. The results obtained by this ensemble are listed in the final two rows of Table VII.

TABLE VII
CLASSIFICATION ACCURACY ON THE TEST SETS OF MRD AND MBR

Method	**MRD**	**MBR**
Ensemble in Fig 9 or Fig. 10	**94.28%**	**93.21%**
Classifier in branch 1 in Fig 9 or Fig. 10	93.31%	88.37%
Classifier in branch 2 in Fig 9 or Fig. 10	93.53%	92.84%
Classifier in branch 3 in Fig 9 or Fig. 10	93.30%	91.29%
Ensemble of three ERFs using raw pixels	88.51%	
Ensemble of ERF, SVM and LR using raw pixels		66.04%

The ensemble in Fig. 9 achieves 94.28% accuracy on the test set of MRD and the ensemble in Fig. 10 achieves 93.21% accuracy on the test set of MBR, which are better than any of the three single classifiers or the ensemble using raw pixels, as listed in Table VII. Compared the example ensembles with the single classifiers, we can find that the combination of these three classifiers using the voting function enhances the classification accuracy. The reason may be that IEGP automatically selects classification functions to build an ensemble during the evolutionary process, which results in a good combination of classifiers. This indicates that IEGP can find a good ensemble to achieve better generalisation performance than a single classifier. Compared the example ensembles in Fig. 9 and Fig. 10 with the new ensembles listed in the final two rows of Table VII, it is obvious that feature extraction is necessary and important for improving the classification performance. The ensembles of classifiers built using raw pixels only achieve 88.51% on MRD and 66.04% on MBR, which are much lower than that of the ensembles found by IEGP. This indicates that the features learnt by IEGP are more discriminative than raw pixels and can further boost the performance of the ensembles. This shows that one of the objectives that uses the proposed IEGP approach to learn effective features has been successfully achieved.

To conclude, further analysis shows that the ensembles found by IEGP have high diversity by using different features to build classifiers to form ensembles. The analysis reveals that IEGP can find ensembles of the classifiers trained from the same or different classification algorithms. The analysis also shows that IEGP can find good ensembles of classifiers to achieve higher generalisation performance than a single classifier. In addition, the features learnt by IEGP are more discriminative features than raw pixels for classifying images.

VII. Conclusions

The goal of this paper was to develop a new GP-based method to automatically learn effective features and evolve ensembles for image classification. This goal has been successfully achieved by developing the IEGP approach with a new individual representation, a new function set, and a new terminal set. With the new multi-layer representation, the IEGP approach can learn informative features and evolve ensembles of diverse classification algorithms. The parameters of the classification algorithms in the evolved ensemble can be automatically optimised/tuned during the evolutionary process. The diversity issue of ensembles is automatically addressed by IEGP using the tree-based flexible representation. In addition, the evolved solutions of ensembles have a flexible length or depth, which is suitable for dealing with different types of image classification tasks.

The performance of IEGP has been examined on 13 image classification data sets of varying difficulty, including facial expression classification, face recognition, scene classification, texture classification, and object classification. The comparisons show that IEGP is more effective than traditional methods using pre-extracted features or raw pixels. On the large-scale data sets, i.e., data sets 7-13, IEGP achieves better results than all the benchmark methods on two data sets, is ranked the second on three data sets, and is ranked the third on the remaining two data sets. Compared with the previous EGP method, IEGP achieves significantly better or similar results on small-scale data sets and better results on large-scale data sets. The comparisons of EGP and IEGP in terms of the distribution of the training and test results show that IEGP achieves better and more stable results than EGP.

This study shows the potential of GP on learning features and evolving ensembles for image classification. However, there are many tasks in computer vision, such as video analysis and remote sensing image classification. It is possible to develop new GP-based methods to address these tasks.

References

[1] K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, and B. Fielding, "A micro-ga embedded pso feature selection approach to intelligent facial emotion recognition," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1496–1509, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*. Springer, Sep 2016, pp. 630–645.

[3] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Trans Syst. Man Cybern. C Appl. Rev.*, vol. 42, no. 4, pp. 463–484, 2012.

[4] Z. Yu, Y. Zhang, J. You, C. P. Chen, H.-S. Wong, G. Han, and J. Zhang, "Adaptive semi-supervised classifier ensemble for high dimensional data classification," *IEEE Trans. Cybern.*, no. 99, pp. 1–14, 2017.

[5] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2012.

[6] Z.-H. Zhou and J. Feng, "Deep forest," *Natl. Sci. Rev.*, vol. 6, no. 1, pp. 74–86, Oct 2018.

[7] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Proc. NeurIPS*, 1995, pp. 231–238.

[8] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Mach. Learn.*, vol. 51, no. 2, pp. 181–207, 2003.

[9] C. T. Tran, M. Zhang, P. Andreae, B. Xue, and L. T. Bui, "An effective and efficient approach to classification with incomplete data," *Knowl. Based Syst.*, vol. 154, pp. 1–16, 2018.

[10] Z. Yu, D. Wang, Z. Zhao, C. P. Chen, J. You, H.-S. Wong, and J. Zhang, "Hybrid incremental ensemble learning for noisy real-world data classification," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 403–416, 2017.

[11] R. Diao, F. Chao, T. Peng, N. Snooke, and Q. Shen, "Feature selection inspired classifier ensemble reduction," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1259–1268, 2013.

[12] T. V. Dittimi and C. Y. Suen, "Mobile phone based ensemble classification of deep learned feature for medical image analysis," *IETE Tech. Rev.*, pp. 1–12, 2019.

[13] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul 2002.

[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE CVPR*, vol. 1, 2005, pp. 886–893.

[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov 2004.

[16] L. Shao, L. Liu, and X. Li, "Feature learning for image classification via multiobjective genetic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 7, pp. 1359–1371, Jul 2014.

[17] H. Al-Sahaf, A. Al-Sahaf, B. Xue, M. Johnston, and M. Zhang, "Automatically evolving rotation-invariant texture image descriptors by genetic programming," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 83–101, Feb 2017.

[18] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT press, Cambridge, 1992.

[19] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *IEEE Trans Syst. Man Cybern. C Appl. Rev.*, vol. 40, no. 2, pp. 121–144, 2010.

[20] B. P. Evans, "Population-based ensemble learning with tree structures for classification," Master's thesis, Victoria University of Wellington, 2019.

[21] A. Khan, A. S. Qureshi, M. Hussain, M. Y. Hamza *et al.*, "A recent survey on the applications of genetic programming in image processing," *arXiv preprint arXiv:1901.07387*, 2019.

[22] A. Zameer, J. Arshad, A. Khan, and M. A. Z. Raja, "Intelligent and robust prediction of short term wind power using genetic programming based ensemble of neural networks," *Energy Convers. Manag.*, vol. 134, pp. 361–372, 2017.

[23] D. Atkins, K. Neshatian, and M. Zhang, "A domain independent genetic programming approach to automatic feature extraction for image classification," in *Proc. IEEE CEC*, 2011, pp. 238–245.

[24] H. Al-Sahaf, A. Song, K. Neshatian, and M. Zhang, "Two-tier genetic programming: Towards raw pixel-based image classification," *Expert Syst. Appl.*, vol. 39, no. 16, pp. 12 291–12 301, Nov 2012.

[25] Y. Bi, B. Xue, and M. Zhang, "An automated ensemble learning framework using genetic programming for image classification," in *Proc. GECCO*. ACM, 2019, pp. 365–373.

[26] D. J. Montana, "Strongly typed genetic programming," *Evol. Comput.*, vol. 3, no. 2, pp. 199–230, 1995.

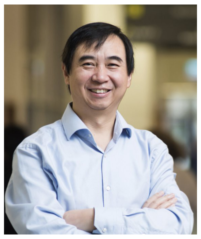
[27] L. Liu, L. Shao, X. Li, and K. Lu, "Learning spatio-temporal representations for action recognition: A genetic programming approach," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 158–170, 2016.

[28] S. Bianco, G. Ciocca, and R. Schettini, "Combination of video change detection algorithms by genetic programming," *IEEE Trans. Evol. Com-*

put., vol. 21, no. 6, pp. 914–928, Dec 2017.
[29] Y. Bi, B. Xue, and M. Zhang, "An automatic feature extraction approach to image classification using genetic programming," in *Proc. Int. Conf. Appl. Eov. Comput.*, 2018, pp. 421–438.
[30] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proc. 18th ACM Int. Conf. Multimedia*, 2010, pp. 1469–1472.
[31] H. Li and M. Gong, "Self-paced convolutional neural networks," in *Proc. IJCAI*, 2017, pp. 2110–2116.
[32] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug 2013.
[33] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: A simple deep learning baseline for image classification?" *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5017–5032, Dec 2015.
[34] G. Qian and L. Zhang, "A simple feedforward convolutional conceptor neural network for classification," *Appl. Soft Comput.*, vol. 70, pp. 1034–1041, Sep 2018.
[35] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proc. ICML.* Omnipress, 2011, pp. 833–840.
[36] W. Fu, M. Johnston, and M. Zhang, "Low-level feature extraction for edge detection using genetic programming," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1459–1472, 2014.
[37] W. A. Albukhanajer, J. A. Briffa, and Y. Jin, "Evolutionary multiobjective image feature extraction in the presence of noise," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1757–1768, 2015.
[38] H. Wei and X.-S. Tang, "A genetic-algorithm-based explicit description of object contour and its ability to facilitate recognition," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2558–2571, 2015.
[39] K. Nag and N. R. Pal, "A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 499–510, 2016.
[40] A. Majid, A. Khan, and A. M. Mirza, "Combination of support vector machines using genetic programming," *Inter. J. Hy. Intel. Syst.*, vol. 3, no. 2, pp. 109–125, 2006.
[41] S. Young, T. Abdou, and A. Bener, "Deep super learner: A deep ensemble for classification problems," in *Proc. 31st Can. Conf. Art. Intell.* Springer, 2018, pp. 84–95.
[42] C. Liu and H. Wechsler, "A gabor feature classifier for face recognition," in *Proc. ICCV*, vol. 2, 2001, pp. 270–275.
[43] B.-H. Asa and W. Jason, "A user's guide to support vector machine," *Data Mini. Tech. Life Sci.*, vol. 609, pp. 223–239, 2010.
[44] M. E. Roberts, "The effectiveness of cost based subtree caching mechanisms in typed genetic programming for image segmentation," in *Proc. Workshop Appl. Evol. Comput.* Springer, 2003, pp. 444–454.
[45] C. E. Thomaz, "Fei face database," *online: http://fei.edu.br/˜cet/facedatabase.html*, Mar 2012.
[46] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," in *Third IEEE Int. Conf. Auto. Face Gest. Recog.*, Apr 1998, pp. 200–205.
[47] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proc. Sec. IEEE Workshop Appl. Comput. Vis.*, 1994, pp. 138–142.
[48] P. Mallikarjuna, A. T. Targhi, M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh, "The kth-tips2 database," *Computational Vision and Active Perception Laboratory, Stockholm, Sweden*, pp. 1–10, Jun 2006.
[49] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *Proc. IEEE CVPR*, vol. 2, Jun 2005, pp. 524–531.
[50] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proc. ICML.* ACM, 2007, pp. 473–480.
[51] T. Yu, C. Guo, L. Wang, S. Xiang, and C. Pan, "Self-paced autoencoder," *IEEE Signal Proc. Let.*, vol. 25, no. 7, pp. 1054–1058, Jul 2018.
[52] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Evolving deep convolutional neural networks for image classification," *IEEE Trans. Evol. Comput.*, pp. 1–14, 2019.
[53] M. Iqbal, B. Xue, H. Al-Sahaf, and M. Zhang, "Cross-domain reuse of extracted knowledge in genetic programming for image classification," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 569–587, Aug 2017.
[54] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *J. Mach. Learn. Res.*, vol. 13, no. Jul, pp. 2171–2175, Jul 2012.
[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct 2011.
[56] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

Ying Bi (S'17) received the B.Sc. degree from Wuhan Polytechnic University, Hubei, China, in 2013 and M.Sc. degree from Shenzhen University, Shenzhen, China, in 2016. Currently, she is pursuing the Ph.D degree in computer science with the School of Engineering and Computer Science, Victoria University of Wellington (VUW), Wellington, New Zealand. She has over 20 journal and conference papers and her current research interests include evolutionary computation, computer vision and machine learning. She is a member of the IEEE Computational Intelligence Society and has been severing as reviewers for top international journals and conferences in this field.

Bing Xue (M'10) received the B.Sc. degree from the Henan University of Economics and Law, Zhengzhou, China, in 2007, the M.Sc. degree in management from Shenzhen University, Shenzhen, China, in 2010, and the PhD degree in computer science in 2014 at Victoria University of Wellington, New Zealand. She is currently an Associate Professor in School of Engineering and Computer Science at Victoria University of Wellington. She has over 200 papers published in fully refereed international journals and conferences and her research focuses mainly on evolutionary computation, feature selection, feature construction, image analysis, and transfer learning. Dr Xue is currently the Chair of IEEE Computational Intelligence Society (CIS) Data Mining and Big Data Analytics Technical Committee, Chair of the IEEE Task Force on Evolutionary Feature Selection and Construction, and Vice-Chair of IEEE CIS Task Force on Transfer Learning & Transfer Optimization, and Vice-Chair of IEEE CIS Task Force on Evolutionary Deep Learning and Applications. She is also an Associate Editor for many international journals including IEEE Transactions on Evolutionary Computation, IEEE Computational Intelligence Magazine, and ACM Transactions on Evolutionary Learning and Optimisation.



Mengjie Zhang (M'04-SM'10-F'19) received the B.E. and M.E. degrees from Artificial Intelligence Re- search Center, Agricultural University of Hebei, Hebei, China, and the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 1989, 1992, and 2000, respectively. He is currently Professor of Computer Science, Head of the Evolutionary Computation Research Group, and the Associate Dean (Research and Innovation) in the Faculty of Engineering. His current research interests include evolutionary computation, particularly genetic programming, particle swarm optimization, and learning classifier systems with application areas of image analysis, multi-objective optimization, feature selection and reduction, job shop scheduling, and transfer learning. He has published over 500 research papers in refereed international journals and conferences. Prof. Zhang is a Fellow of Royal Society of New Zealand and have been a Panel member of the Marsden Fund (New Zealand Government Funding), a Fellow of IEEE, and a member of ACM. He was the chair of the IEEE CIS Intelligent Systems and Applications Technical Committee, and chair for the IEEE CIS Emergent Technologies Technical Committee and the Evolutionary Computation Technical Committee, and a member of the IEEE CIS Award Committee. He is a vice-chair of the IEEE CIS Task Force on Evolutionary Feature Selection and Construction, a vice-chair of the Task Force on Evolutionary Computer Vision and Image Processing, and the founding chair of the IEEE Computational Intelligence Chapter in New Zealand. He is also a committee member of the IEEE NZ Central Section.