A Gaussian Filter-based Feature Learning Approach Using Genetic Programming to Image Classification

Ying Bi, Bing Xue and Mengjie Zhang

School of Engineering and Computer Science,
Victoria University of Wellington, Wellington 6140, New Zealand
{Ying.Bi, Bing.Xue, Mengjie.Zhang}@ecs.vuw.ac.nz

Abstract. Image classification is an important task in computer vision. Image features are often needed for classification, but the majority of feature extraction methods require domain experts and human intervention. To learn image features automatically from the problems being tackled is more effective. However, it is very difficult due to image variations and the high dimensionality of image data. This paper proposes a new feature learning approach based on Gaussian filters and genetic programming (GauGP) for image classification. Genetic programming (GP) is a well-known evolutionary learning technique and has been applied to many visual tasks, showing good learning ability and interpretability. In the proposed GauGP method, a new program structure, a new function set and a new terminal set are developed, which allow it to detect small regions from the input image and to learn discriminative features using Gaussian filters for image classification. The performance of GauGP is examined on six different data sets of varying difficulty and compared with four GP-based methods, eight traditional approaches and convolutional neural networks. The experimental results show GauGP achieves significantly better or similar performance in most cases. Further analysis on example GP programs demonstrates the good interpretability of GauGP and the importance of Gaussian filters in GauGP in terms of feature learning and image classification.

Keywords: Feature Learning, Genetic Programming, Image Classification, Gaussian Filter, Evolutionary Computation, Feature Extraction

1 Introduction

Image classification as an important task in computer vision has received much attention in recent years [1]. The task is to assign class labels to images based on the content in images. It is a challenging task due to image variations, such as scale, illumination, deformation, and rotation variations. Generally, image features including shape, texture and edge features are employed to feed machine learning classification algorithms such as support vector machines (SVMs), k-nearest neighbour (KNN), decision tree (DT) and random forest (RF) to perform classification [2]. However, the majority of the existing approaches require domain experts to extract features, which is time-consuming and expensive [3]. In addition, the classification performance cannot be guaranteed by these handcrafted features when dealing with a new task.

Feature learning is to automatically learn informative features from the raw data without domain knowledge and human intervention for visual tasks [4]. Typically, the learnt features are able to achieve good performance on specific tasks as they are problem-dependent. However, to learn discriminative features from the raw pixel values for effective classification is difficult due to the variations and high dimensionality of the image data. The state-of-the-art convolutional neural networks (CNNs) have achieved significant success in feature learning and image classification [4]. However, deep CNNs require a large number of training instances and computing resources. Moreover, the learnt representation is not necessarily meaningful [5], and is often hard to understand.

In contrast to CNNs, genetic programming (GP), as an evolutionary computation (EC) technique, can evolve solutions with good interpretability and understandability [6]. GP aims at automatically evolving computer programs to solve problems without the assumption of solution structures through the evolutionary learning process [7]. Motivated by the principles of biological evolution, GP has good learning ability and has been widely applied to visual tasks, including edge detection [8], image segmentation [9] and feature learning [6]. Therefore, this paper designs a new feature learning approach based on GP.

The commonly used representation of GP is tree-based, which is very flexible and is able to integrate different functions and terminals into feasible solutions [7] [10]. Image operators/descriptors, such as histogram of gradient descent (HOG), histogram equalisation, Sobel edge detector, and Laplacian, are designed as GP functions to allow the GP system to learn advanced and discriminative features for classification [6] [11] [12]. However, there are a number of existing advanced image-related operators, which can be employed as GP functions to facilitate feature learning.

The Gaussian filter is well-known filter and widely used in image processing and computer vision [8]. For example, the edge and blob detection operator Laplace of Gaussian (LoG) applies Laplace filter to the image after Gaussian filtering/blurring. The Difference of Gaussian (DoG), where Gaussian filters with different standard deviation values are used, is designed for blob or keypoints detection, such as in the well-known scale-invariant feature transform (SIFT) algorithm. In addition, the derivatives of the Gaussian filter are important for salient feature detection/description, such as in the Canny edge detector and the Hessian matrix. Thus, this work integrates Gaussian filter and its derivatives in GP to achieve informative feature learning for image classification.

Goals: The overall goal of this paper is to develop a new GP-based approach, which is able to benefit from the Gaussian filter and its derivatives, to learning discriminative features for image classification. To achieve this goal, a new program structure, a new function set and a new terminal set is designed and employed in the new GP approach to allow it to detect small regions from the input image, and learn discriminative features from the detected regions using the Gaussian-based filters for classification. The performance of the new GP approach will be evaluated on six different data sets of varying difficulty and compared with 13 state-of-the-art methods. Specifically, this goal can be divided into the following three objectives.

- Develop a new GP approach with a new program structure, a new function set and a new terminal set to detect small regions and learn informative features by using Gaussian filter and its derivatives from the input image for classification,
- Investigate whether the learnt Gaussian filter-based features are powerful for image classification and can obtain better classification performance than four GP-based methods, eight effective traditional feature extraction approaches and CNN, and
- Investigate whether the learnt features by the new GP approach can be easily interpreted and whether the Gaussian filters are important in the new method in terms of feature learning.

2 Genetic Programming

GP automatically evolves computer programs to solve problems through the evolutionary learning process [7]. It is a population-based technique, where a population represents a set of individuals/solutions. A typical representation of GP is tree-based. Specifically, each individual in GP is represented by a program tree in Lisp S-expression. To start a GP method, it is necessary to design or select a function set and a terminal set. In a GP tree, the root node and the internal nodes are constructed by selecting functions from the function set, and the leaf nodes are formed by selecting terminals from the terminal set. During the evolutionary process, the GP trees (population) are updated and evolved in order to find the optimal solution.

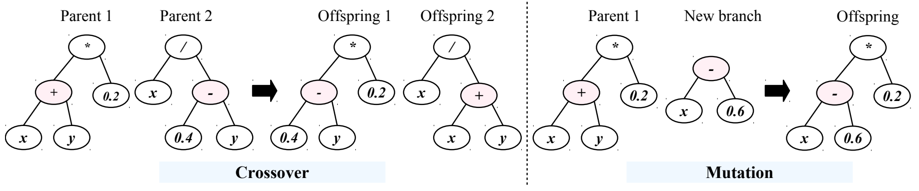

Fig. 1. Crossover and mutation operations.

The evolutionary learning process of GP starts with randomly initialising a population. Each individual in the population is evaluated by the predefined fitness function and assigned a fitness value. At each generation, the population is updated using a selection operator and genetic operators, i.e. elitism, crossover and mutation. In *selection*, the individual with better fitness value has a larger chance to be selected. The *elitism* operator reproduces a small number of the best individuals to the new population. The crossover and mutation operations are shown in Fig. 1. In *crossover*, two selected individuals (parents) swap their branches according to the selected nodes to generate two new individuals (offspring). In *mutation*, a selected individual (parent) randomly replace one of its old branches with a new branch to generate a new individual (offspring). By these operators, a new population is generated. The population updating and fitness evaluation process proceeds until the termination criterion is satisfied. The termination criterion might be reaching the maximum number of generations or obtaining the desired fitness value. Finally, the best individual/solution is returned.

3 Proposed Method

This section presents the new GauGP approach in detail, including the program structure, the function set, the terminal set, and the overall process.

3.1 Program Structure

The GauGP approach uses a tree-based representation, and is based on strongly typed GP (STGP) [10]. An example program of GauGP is shown in Fig. 2, where an input image goes through region detection, filtering, max-pooling, and feature concatenation process. In the region detection process, region detection functions, i.e., *Region_R* and *Region_S*, are employed to find the discriminative rectangle and square regions from the input image. Then the detected regions are processed by the filtering functions, i.e., *GauD* and *Gau*, and max-pooling functions, i.e. *MaxP*. Finally, a feature vector is generated using feature concatenation functions as the final output of the GP system.

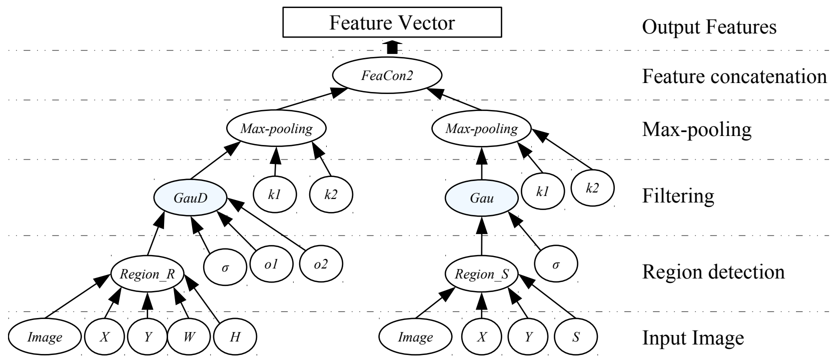

Fig. 2. An example program of GauGP.

It is noticeable that the tree depth of GauGP at the filtering, max-pooling and feature concatenation process as the corresponding functions can be used/appeared in a GP tree many times. This allows GP to evolve a simple tree for easy tasks or a complex tree with many functions for difficult tasks.

This program structure is inspired by the work in [11] [6]. Different from the approaches in [11], GauGP employs max-pooling functions and is able to extract a set of features rather than one high-level feature for classification. Compared with the approach in [6], GauGP can extract lower dimensional features as region detection and down-sampling contribute to dimension reduction. Moreover, GauGP only uses Gaussian-based filters in the filtering process rather than a set of different operators, which reduces the search space.

3.2 Function Set

The function set contains region detection functions, Gaussian-based filters, arithmetic functions, max-pooling functions and feature concatenation functions. Table 1 lists all the functions employed in the GauGP method.

Table 1. Function Set

Functions	Input	Output	Function Description
Root	2 vectors	1 vector	Concatenate two vectors to a vector
FeaCon2	2 images	1 vector	Concatenate two images into a vector
FeaCon3	3 images	1 vector	Concatenate three images into a vector
Max-pooling	1 image, k_1, k_2	1 image	Conduct max-pooling to the input image
Gau	1 image, σ	1 image	Gaussian filter with standard deviation σ
GauD	1 image, σ, o_1, o_2	1 image	The derivatives of Gaussian filter based on Eq. (2)
Mix_Add	2 images	1 image	Add two images with different sizes.
Mix_Sub	2 images	1 image	Subtract two images with different sizes
Mix_Mul	2 images	1 image	Multiply two images with different sizes
Mix_Div	2 images	1 image	Protected division on two images with different sizes
Region_S	1 images, X, Y, S	1 image	Detect a square region from the input image
Region_R	1 images, X, Y, W, H	1 image	Detect a rectangle region from the input image

The region detection functions *Region_S* and *Region_R* are to detect a square and rectangle region from the input image respectively, which are the same as that in [11]. The Gaussian-based filters include the *Gau* and *GauD* functions. *Gau* represents the Gaussian smooth filter with the standard deviation value of σ, which is generated according to the Gaussian function in Eq. (1).

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2}\, e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

GauD is the derivatives of the Gaussian function with a standard deviation value and two orders. It takes an image and three parameters, i.e., σ, o_1 and o_2 as input and returns an image. o_1 and o_2 represent the orders of derivative along X axis and Y axis. The *GauD* filter is defined as Eq.(2) according to the Gaussian function in Eq. (1). Fig. 3 shows an example image and the generated images after employing the corresponding *Gau* and *GauD* filters. It is obvious that salient information can be detected by the *GauD* function.

$$GD(x, y, \sigma, o_1, o_2) = \frac{\mathrm{d}^{o_1} G(x, y, \sigma)}{\mathrm{d}^{o_1} x} \cdot \frac{\mathrm{d}^{o_2} G(x, y, \sigma)}{\mathrm{d}^{o_2} y} \tag{2}$$

In the filtering step, except for the *Gau* and *GauD* filters, four arithmetic functions, i.e., *Mix_Add, Mix_Sub, Mix_Mul*, and *Mix_Div*, are employed to deal

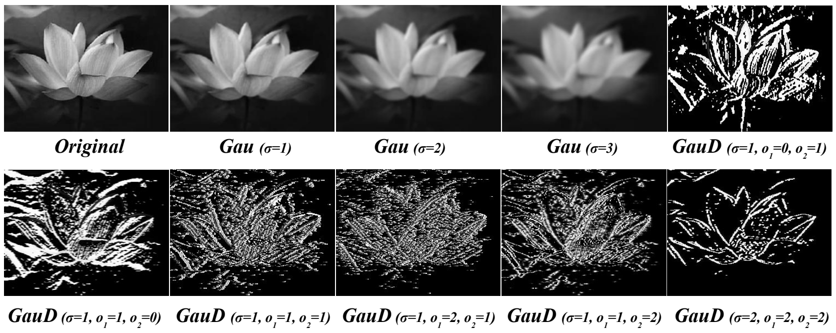

Fig. 3. An example image and the generated images after the corresponding filtering.

with two images. These four functions take two images with different sizes as input and return an image by performing the corresponding arithmetic operation to the images after cutting them to keep the size consistent. The aims of these functions are to combine two detected regions together to generate new features.

Max-pooling function takes an image and kernel size i.e., k_1 and k_2 as input and returns a smaller image as it is a down-sampling method in computer vision. The kernel size of max-pooling function is automatically generated and optimised during the evolutionary learning process. In max-pooling, the stride step is the same as the kernel size.

Feature concatenate functions contain the *FeaCon2, FeaCon3* and *Root* functions, which can be used to form the root node in the GauGP approach. These functions are to concatenate two or three images or vectors to a feature vector. Specifically, the *Root* function takes itself, *FeaCon2* and *FeaCon3* functions as children nodes. The *FeaCon2* and *FeaCon3* functions take max-pooling function as children nodes.

3.3 Terminal Set

Table 2. Terminal Set

Terminals	Type	Description
Image	Image	The input grey-scale image after normalisation
X, Y	Integer	The coordinates of the top left point of a detected region. They are in range $[0,\ Image_{width}-20]$ or $[0,\ Image_{height}-20]$
S, W, H	Integer	The size or width/height of a square/rectangle region in *Region_S*/ *Region_R* functions. They are in range [20, 50]
k_1, k_2	Integer	The kernel size of the *Max-pooling* function. They are in range [2, 10] with a step of 2
σ	Integer	The standard deviation of the Gaussian filter in the *Gau* and *GauD* functions. It randomly initialized from range [1, 3]
o_1, o_2	Integer	The order of Gaussian derivatives. They are randomly initialised from range [0, 2].

The type and description of terminals used in the GauGP method are listed in Table 2, including *Image, X, Y, S, W, H,* k_1, k_2, σ, o_1, and o_2. The *Image* represents the input normalised image, which is a 2D array with values in range

[0, 1]. The *X, Y, S, W,* and *H* terminals are the leaf nodes of the region detection functions *Region_S* and *Region_R*. The *X* and *Y* terminals represent the coordinates of the top left point of the detected region by the two functions. *S, W* and *H* are the size or width and height of the detected region. The k_1 and k_2 are the kernel size of the max-pooling function. The σ terminal is the standard deviation of a Gaussian filter, and the o_1 and o_2 terminals represent the order of derivatives in the *GauD* function. Except for the *Image* terminal, the values of the other terminals are randomly generated in the initialisation stage and are evolved/optimised during the evolutionary learning process. The range of these terminals are listed in Table 2.

3.4 Overall Process

The overall learning and testing process of GauGP on feature learning and classification is shown in Fig. 4. A training set is employed for GauGP to learn a set of discriminative features for image classification. In GauGP, each program can be considered as a feature extraction approach. During the GP learning process, the programs are updated using a selection operator and genetic operators, and evaluated using linear SVM with 5-fold cross-validation on the training set. The mean accuracy of the 5-fold are employed as the fitness function for GauGP. At the final stage of learning, the best program with the best mean accuracy on the training set is returned and tested on the test set, as shown in Fig. 4.

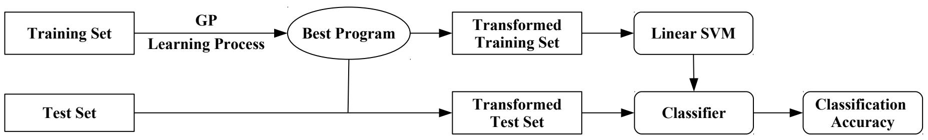

Fig. 4. The overall process of the proposed GauGP for image classification.

In the testing process, the best program is used to extract features from the training set and the test set. Then transformed training set is employed to learn a SVM classifier and the learnt classifier is tested on the transformed test set. The classification accuracy on the test set is reported.

4 Experiment Design

4.1 Data Sets

To examine the performance of the proposed method, six different data sets of varying difficulty are employed for conducting experiments. They are JAFFE [13], YALE [14], FEI_1 [15], FEI_1 [15], SCENE [16], and TEXTURE [17]. The JAFFE, YALE, FEI_1 and FEI_2 are facial expression classification data sets, where the images are sampled from different people with different expressions, such as happy, sad, surprised, and natural. The SCENE data set contains natural scene images and the TEXTURE data set contains texture images involving scale and illumination variations. Without losing generalisation ability, all the data sets are binary classification tasks and only have gray-scale images. More details of these data sets, including the size of images, the class labels, and the number of images in the training set and the test set, are listed in Table 3.

Table 3. Data Set Properties

Name	Size	Class labels	Training set	Test set
JAFFE	128×128	*Happy*	20	10
		Surprised	20	10
YALE	128×128	*Happy*	20	10
		Sad	20	10
FEI_1	130×180	*Smile*	75	25
		Natural	75	25
FEI_2	130×180	*Smile*	75	25
		Smile	75	25
SCENE	128×128	*Highway*	130	65
		Streets	146	73
TEXTURE	100×100	*Cork*	324	108
		Brown bread	324	108

4.2 Baseline Methods

To show the performance of the proposed method, 13 advanced approaches are implemented for comparisons, including four GP-based approaches, eight traditional approaches and a CNN approach. The four GP-based approaches are 2TGP [18], DIF+GP [19], Histogram+GP, and uLBP+GP [20]. The eight traditional approaches extract image features using domain independent feature (DIF), Histogram, grey-level co-occurrence matrix (GLCM), Gabor bank features (Gabor), SIFT, HOG, local binary patterns (LBP), and uniform LBP (uLBP) methods, respectively, and employ an SVM with linear kernel for classification. The CNN method is the famous LeNet [21] with ReLU as activation function and softmax for classification.

4.3 Parameter settings

The GauGP method and the GP-based baseline methods are implemented in Python based on the *DEAP* (*Distributed Evolutionary Algorithm Package*) package [22]. The number of generations is set as 50 and the population size is 500. The crossover rate is 0.8, the mutation rate is 0.19 and the elitism rate is 0.01. The selection method is Tournament selection and the size is 7. In the initialisation step, *Ramped half-and-half* is used for population generation and the tree depth is $2-6$.

In the four GP-based baseline methods, the classification accuracy is employed as the fitness function. The experiments of the GP-based approaches and the eight traditional approaches on each data set run 30 times independently. The experiments of LeNet run 10 times due to the high computation cost. In LeNet, the number of epochs is 500 and the batch size is 128. The training set is employed for training and the classification accuracy on the test set is reported as the final results.

5 Results and Discussions

This section compares and discusses the results obtained by the proposed GauGP method and the 13 baseline methods. Table 4 lists the maximum classification accuracy, mean accuracy and standard deviation on the test set of the six data sets by the total 14 methods. In the table, each block represents all the results obtained by these methods on a data set. The Wilcoxon signed-rank test with a 5% significance level is used to compare the GauGP method with a baseline method to show the significance of performance improvement. The symbols "+", "=" and "–" in Table 4 denote the GauGP method is significantly better, similar or significantly worse than the competitor.

Table 4. Classification accuracy(%) of the proposed GauGP method and the baseline methods on the six data sets

Algorithms	Max	Mean±St.D.	Max	Mean±St.D.	Max	Mean±St.D.
Data Sets	**JAFFE**		**YALE**		**FEI_1**	
2TGP	95.00	68.83±13.64+	95.00	74.67±13.66+	96.00	88.13±6.22+
DIF+GP	90.00	75.83±7.20+	75.00	60.33±9.74+	80.00	56.67±6.88+
Histogram+GP	80.00	53.33±11.13+	80.00	54.50±11.57+	70.00	48.93±7.22+
uLBP+GP	75.00	50.33±9.99+	65.00	49.17±9.84+	66.00	50.87±7.48+
DIF+SVM	90.00	85.17±5.24+	85.00	74.50±7.89+	74.00	61.13±4.89+
Histogram+SVM	60.00	51.17±2.79+	55.00	50.00±2.24+	54.00	48.13±3.38+
GLCM+SVM	70.00	54.50±6.50+	55.00	50.33±1.25+	50.00	49.67±0.75+
Gabor+SVM	**100.0**	96.17±5.87+	75.00	60.50±6.50+	82.00	71.60±7.87+
SIFT+SVM	80.00	80.00±0.00+	75.00	75.00±0.00+	82.00	82.00±0.00+
HOG+SVM	90.00	90.00±0.00+	85.00	85.00±0.00+	94.00	94.00±0.00+
LBP+SVM	75.00	74.33±1.70+	80.00	78.00±3.32+	68.00	62.47±3.49+
uLBP+SVM	80.00	73.17±5.08+	85.00	76.00±5.54+	64.00	56.87±5.18+
LeNet	**100.0**	**100.0±0.00**–	90.00	85.50±2.69+	**98.00**	94.40±1.96=
GauGP	**100.0**	99.17±1.86	**100.0**	**92.17±4.60**	**98.00**	**94.67±2.09**
Data Sets	**FEI_2**		**SCENE**		**TEXTURE**	
2TGP	**94.00**	85.47±5.98+	93.48	87.85±2.20+	86.11	79.40±3.42+
DIF+GP	72.00	60.33±8.38+	89.13	85.22±2.24+	88.43	84.46±2.54+
Histogram+GP	60.00	48.80±6.14+	84.06	79.98±1.83+	92.13	87.36±2.15+
uLBP+GP	72.00	48.73±7.87+	95.65	91.79±2.98=	96.76	93.89±2.01=
DIF+SVM	72.00	62.80±6.10+	87.68	81.09±6.87+	86.11	80.93±6.74+
Histogram+SVM	54.00	50.13±2.53+	59.42	56.74±3.09+	52.31	52.31±0.00+
GLCM+SVM	54.00	50.13±0.72+	93.48	90.56±6.73=	88.89	73.60±11.13+
Gabor+SVM	74.00	65.67±5.14+	82.61	75.14±7.98+	50.93	50.17±0.30+
SIFT+SVM	78.00	78.00±0.00+	97.10	**97.10±0.00**–	85.19	85.19±0.00+
HOG+SVM	88.00	88.00±0.00+	91.30	90.17±0.40+	75.46	72.71±1.26+
LBP+SVM	66.00	57.60±3.56+	95.65	94.49±1.05–	**99.54**	**99.09±0.08**–
uLBP+SVM	56.00	51.93±2.34+	**97.83**	94.15±2.79=	**99.54**	98.72±3.29–
LeNet	**94.00**	**90.80±1.83**=	94.93	92.90±1.77=	96.76	81.67±20.77=
GauGP	**94.00**	90.27±2.41	95.65	92.37±1.92	97.69	94.66±1.53

From Table 4, it is obvious that the GauGP method obtains good performance on the six different data sets, especially on the face data sets, i.e. JAFFE, YALE, FEI_1, and FEI_2. Compared with the four GP-based approaches, the GauGP method achieves significantly better results in 22 cases and similar results in 2 cases out of the total 24 cases. Compared with the eight commonly used feature extraction methods, the GauGP method obtains significantly better or similar performance in 44 cases and significantly worse results in 4 cases out of the total 48 cases. Compared with LeNet, the GauGP method achieves significantly better results in 1 case, similar results in 4 cases and significantly better results in 1 case on the six data sets.

In the total 78 ($13 \times 6 = 78$) comparisons, GauGP achieves significantly better results in 65 cases, similar results in 8 cases and significantly worse results in 5 cases. Specifically, GauGP obtains significantly better or similar results than all the other baseline methods on the YALE, FEI_1 and FEI_2 data sets. On the SCENE and TEXTURE data sets, the GauGP method achieves worse results than the traditional methods, i.e., SIFT+SVM, LBP+SVM and uLBP+SVM, but achieves better or similar results than the remaining approaches. It is noticeable that the LBP descriptors are very powerful for texture description.

These results and comparisons illustrate that the proposed GauGP method is able to learn discriminative features from the input image with good classification accuracy. Especially, the learnt features by GauGP are very powerful for facial expression classification. The experiments confirm the difficulty of feature extraction by the traditional approaches as they perform differently on different data sets. For example, the HOG method performs well on the face image data sets, the SIFT method performs well on the scene data set, and the LBP and uLBP methods perform well on the texture data set. This also reveals that feature learning approaches are more powerful and adaptive than these existing feature extraction approaches.

6 Further Analysis

This section conducts further analysis on an example program evolved by the proposed GauGP method to fully understand why it can achieve good performance and to investigate whether the Gaussian-base filters are important for feature learning in GauGP.

An example program is selected from the YALE data set as the GauGP method has obtained the best performance on this data set compared with all the competitors. The selected example program is shown in the left part of Fig. 5. The right part of Fig. 5 shows the procedure of feature extraction on two example images from the *Happy* and *Sad* classes, respectively. The features extracted by this example program with a linear SVM has achieved 100% classification accuracy on both the training set and the test set.

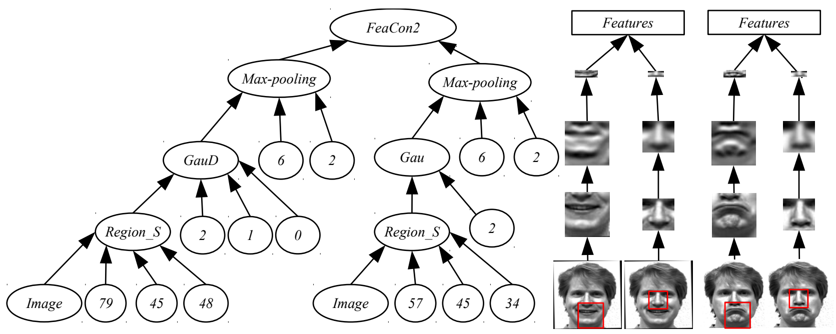

Fig. 5. An example program (left) evolved by the proposed GauGP method on the YALE data set and two example images (right) to show feature extraction by this program. The regions with red rectangle represent the detected regions by this program.

YALE is a facial expression data set, where the face images are sampled from 15 individuals with or without glasses and wink under three different illumination conditions. By this example program, two square regions with the sizes of 48×48 and 34×34 are detected, as shown in the right part of the Fig. 5. These two regions capture the mount and nose areas in the face respectively, which are salient under the two different expressions. The *GauD* function with the standard deviation of 2, the order along the X axis of 1 and the order along Y axis of 0 is evolved to deal with the bigger (left) region. The regions after this filter become more salient in two different classes. The *Gau* function with the standard deviation of 2 is to smooth the smaller (right) region. Overall, this example program extracts 253 features from the input 128×128 image.

To illustrate the importance of the *Gau* and *GauD* functions, we remove these two functions and use the rest of this program as a feature extraction approach to extract features. The extracted features are fed to the linear SVM for classification using the same training and test sets. However, it only obtains 80% classification accuracy on the test set. This confirms the importance of the *Gau* and *GauD* functions in feature learning.

7 Conclusions

The goal of this paper was to design a feature learning approach based on the Gaussian-based filters and GP to image classification. This goal has been successfully achieved by proposing a GauGP method with a new program structure, a new function set and a new terminal set, and examining it on six different data sets. The GauGP method was able to detect small regions from the input image, evolve Gaussian-based filters and max-pooling functions for feature learning, and produce a set of discriminative features for classification. The performance of the proposed method was examined on six different data sets and compared with 13 state-of-the-art approaches, including four GP-base methods, eight commonly used feature extraction methods and a CNN method. The experimental results demonstrated that GauGP was able to achieve significantly better or similar results in the majority cases than the 13 state-of-the-art competitors. The further analysis of the example program revealed the good interpretability of GauGP and the importance of Gaussian-based filters in feature learning.

At the current stage, GauGP has only been examined on binary image classification tasks. In the future, it will be further improved for feature learning to multi-class classification tasks.

References

1. Atkins, D., Neshatian, K., Zhang, M.: A domain independent genetic programming approach to automatic feature extraction for image classification. In: 2011 IEEE Congress on Evolutionary Computation. pp. 238–245. IEEE (2011)
2. Bi, Y., Zhang, M., Xue, B.: Genetic programming for automatic global and local feature extraction to image classification. In: 2018 IEEE Congress on Evolutionary Computation. pp. 1–6. IEEE (2018) to appear

3. Al-Sahaf, H., Zhang, M., Al-Sahaf, A., Johnston, M.: Keypoints detection and feature extraction: A dynamic genetic programming approach for evolving rotation-invariant texture image descriptors. IEEE Transactions on Evolutionary Computation pp. 825 – 844 (2017)
4. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature 521, 436–444 (2015)
5. Sun, Y., Yen, G.G., Yi, Z.: Evolving unsupervised deep neural networks for learning meaningful representations. IEEE Transactions on Evolutionary Computation (2018) DOI: 10.1109/TEVC.2018.2808689
6. Shao, L., Liu, L., Li, X.: Feature learning for image classification via multiobjective genetic programming. IEEE Transactions on Neural Networks and Learning Systems 25(7), 1359–1371 (2014)
7. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection. MIT press,Cambridge (1992)
8. Fu, W., Johnston, M., Zhang, M.: Genetic programming for edge detection: a gaussian-based approach. Soft Computing 20(3), 1231–1248 (2016)
9. Liang, Y., Zhang, M., Browne, W.N.: Genetic programming for evolving figure-ground segmentors from multiple features. Applied Soft Computing 51, 83–95 (2017)
10. Montana, D.J.: Strongly typed genetic programming. Evolutionary Computation 3(2), 199–230 (1995)
11. Bi, Y., Xue, B., Zhang, M.: An automatic feature extraction approach to image classification using genetic programming. In: International Conference on the Applications of Evolutionary Computation. pp. 421–438. Springer (2018)
12. Lensen, A., Al-Sahaf, H., Zhang, M., Xue, B.: Genetic programming for region detection, feature extraction, feature construction and classification in image data. In: European Conference on Genetic Programming. pp. 51–67. Springer (2016)
13. Lyons, M., Akamatsu, S., Kamachi, M., Gyoba, J.: Coding facial expressions with gabor wavelets. In: The Third IEEE International Conference on Automatic Face and Gesture Recognition. pp. 200–205. IEEE (1998)
14. Georghiades, A., Belhumeur, P., Kriegman, D.: Yale face database. Center for Computational Vision and Control at Yale University, http://cvc.yale.edu/projects/yalefaces/yalefa 2 (1997)
15. Thomaz, C.E.: Fei face database. online: http://fei.edu.br/˜cet/facedatabase.html (2012)
16. Fei-Fei, L., Perona, P.: A bayesian hierarchical model for learning natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 2, pp. 524–531. IEEE (2005)
17. Mallikarjuna, P., Targhi, A.T., Fritz, M., Hayman, E., Caputo, B., Eklundh, J.O.: The kth-tips2 database. Computational Vision and Active Perception Laboratory, Stockholm, Sweden (2006)
18. Al-Sahaf, H., Song, A., Neshatian, K., Zhang, M.: Extracting image features for classification by two-tier genetic programming. In: 2012 IEEE Congress on Evolutionary Computation. IEEE (2012, DOI: 101109/CEC20126256412)
19. Zhang, M., Ciesielski, V.B., Andreae, P.: A domain-independent window approach to multiclass object detection using genetic programming. EURASIP Journal on Advances in Signal Processing 2003(8), 841–859 (2003)
20. Ain, Q.U., Xue, B., Al-Sahaf, H., Zhang, M.: Genetic programming for skin cancer detection in dermoscopic images. In: 2017 IEEE Congress on Evolutionary Computation. pp. 2420–2427. IEEE (2017)
21. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

22. Fortin, F.A., De Rainville, F.M., Gardner, M.A., Parizeau, M., Gagné, C.: DEAP: Evolutionary algorithms made easy. Journal of Machine Learning Research 13, 2171–2175 (jul 2012)