# Multi-Objective Genetic Programming for Feature Learning in Face Recognition

Ying Bi, Bing Xue, and Mengjie Zhang

*School of Engineering and Computer Science, Victoria University of Wellington, P.O. Box 600, Wellington, New Zealand*

## Abstract

Face recognition is a challenging task due to high variations of pose, expression, ageing, and illumination. As an effective approach to face recognition, feature learning can be formulated as a multi-objective optimisation task of maximising classification accuracy and minimising the number of learned features. However, most of the existing algorithms focus on improving classification accuracy without considering the number of learned features. In this paper, we propose new multi-objective genetic programming (GP) algorithms for feature learning in face recognition. To achieve effective face feature learning, a new individual representation is developed to allow GP to select informative regions from the input image, extract features using various descriptors, and combine the extracted features for classification. Then two new multi-objective genetic programming (GP) algorithms, one with the idea of non-dominated sorting (NSGPFL) and the other with the idea of Strength Pareto (SPGPFL) are proposed to simultaneously optimise these two objectives. NSGPFL and SPGPFL are compared with the single-objective GP for feature learning (GPFL), a single-objective GP for weighting two objectives (GPFLW), and a large number of baseline methods. The experimental results show the effectiveness of the NSGPFL and SPGPFL algorithms by achieving better or comparable classification performance and learning a small number of features.

*Keywords:* Multi-Objective Optimisation; Evolutionary Computation; Feature Learning; Genetic Programming; Face Recognition.

*Email address:* `{Ying.Bi; Bing.Xue; Mengjie.Zhang}@ecs.vuw.ac.nz` (Ying Bi, Bing Xue, and Mengjie Zhang)

## 1. Introduction

Face recognition is an active research area in computer vision [1]. The task of face recognition is to identify the face of a person from a number of face images. Face recognition has a wide range of applications in entertainment, smart cards, information security, law enforcement and surveillance [2]. However, due to the wide variations of pose, illumination, ageing, expression, resolution, and occlusion, face recognition remains a challenging task.

A face recognition system typically has two main stages: a face representation stage and a face matching stage [3]. Face representation aims to extract a set of discriminative features so that the face images can be easily distinguished. Face matching aims to develop effective classifiers to classify the face images into different groups using the extracted features. In general, compared with face matching, face representation has more significant effects on the recognition/classification performance and is more challenging [4]. Many methods have been developed to obtain an effective face representation, such as scale-invariant feature transform (SIFT) [5], local binary patterns (LBP) [6] and Gabor wavelets [7]. The process that uses one of these methods to extract features often needs human intervention and domain knowledge. Different from feature extraction, feature learning aims to automatically learn/extract features from images without human intervention and domain knowledge. Many feature learning methods have been proposed in recent years to automatically learn representations of faces and have achieved better performance than the methods using manually extracted features [4]. Most feature learning methods are based on neural networks (NNs), which learn representations using many non-linear layers from raw data. However, NN-based methods have their limitations, such as require a large number of training instances, have poor interpretability of the learned representation, have a fixed model complexity, and require rich expertise to design an effective architecture. Except for NNs, genetic programming (GP) has also been applied to automatically learn features for image classification [8, 9]. GP is an evolutionary algorithm, aiming to automatically evolve computer programs to solve a task without any predefined solution structure [10]. Compared with NNs, GP has a flexible representation to evolve variable-length solutions for solving a task. In addition, GP can evolve tree-based solutions with high interpretability from a small number of training instances [11]. Many face recognition tasks often have a small number of instances in each class, which may be difficult to train an effective NN without any data argumentation. In contrast, the solutions of GP often have fewer parameters and it is effective to use GP to learn features from the such a small number of training instances [11]. Therefore, this study develops a new GP-based approach to learning features for face recognition.

The dimension of features in face representation is important and a small number of image features is typically preferred for fast applications. In many traditional face recognition systems, dimensionality reduction methods, such as using principal component analysis (PCA) and linear discriminant analysis (LDA), are employed to reduce the dimension of the features [12, 13]. A small number of features can not only shorten the training time of a classification algorithm but also have potentially higher interpretability. However, the majority of the existing feature learning methods, such as convolutional neural networks (CNNs) and auto-encoders (AEs), focus on improving classification accuracy and ignore the number of learned/extracted features [2, 14]. To address this, it is possible to simultaneously maximise the classification performance and minimise the number of learned/extracted features. Typically, these two objectives are potentially conflicting because a small number of features represent limited information of the data and the between-class similarity may be reduced. A simple and straightforward way to deal with these two objectives is to combine them as a single-objective using a weighted sum approach [15]. However, it is difficult to set the weights of these two objectives because the optimal number of features is unknown for solving a task. Instead, this problem can be formulated as a multi-objective optimisation problem and directly solved using an existing multi-objective algorithm. Although many algorithms have been developed for feature learning [8, 16], very few works focused on multi-objective feature learning with simultaneous maximising the classification performance on the training set and minimising the number of learned/extracted features. To this end, this study aims to fill this gap by developing multi-objective GP-based feature learning algorithms for face recognition.

In recent years, evolutionary multi-objective optimisation algorithms have been widely applied to solve many real-world tasks, such as network planning [17], bound-constrained real-world problems [18], and spread spectrum radar polyphase code design problem [19]. As evolutionary algorithms, multi-objective GP algorithms have been proposed for symbolic regression and modelling [20], and morphological filters optimisation [21]. It can also be found that multi-objective GP has been applied for feature extraction and construction [22, 23, 24]. However, no GP-based algorithms have been developed for multi-objective feature learning. Existing multi-objective GP methods focus on maximising the performance and minimising the tree size rather than the number of features, such as in [22, 23]. In many GP-based feature learning algorithms, the number of features is dynamically and automatically changed during the evolutionary process [16, 25, 26]. These methods may learn a large number of features if the objective has no constraint on the feature number. The multi-objective GP methods have seldom been used to simultaneously maximise the classification performance and minimise the number of learned/extracted feature. It is noted that feature selection also has these two objectives and a number of evolutionary multi-objective feature selection algorithms have been proposed, such as in [27, 28]. However, the maximum number of features in feature selection is known but it is unknown in feature learning. This difference makes these two tasks and the behaviours/landscapes of these two objectives very different. This also makes the multi-objective feature learning task is more difficult. Therefore, it is necessary to investigate multi-objective feature learning and develop a new multi-objective feature learning algorithm to solve it.

### *1.1. Goals*

The overall goal of this study is to develop new multi-objective feature learning algorithms for face recognition using GP with the objectives of maximising the classification performance and minimising the number of learned features. To effectively learn features from face images, a new representation, a new function set and a new terminal set will be developed to allow GP to automatically detect regions from the input images, use descriptors to extract features and combine the extracted features for classification. Then we develop two single-objective feature learning algorithms and two multi-objective feature learning algorithms based on GP with the new representation:

- single-objective GP for feature learning (GPFL)
- single-objective GP for weighting two objectives (GPFLW)
- multi-objective GP for feature learning using the idea of non-dominated sorting (NSGPFL)
- multi-objective GP for feature learning using the idea of strength Pareto (SPGPFL)

These four algorithms will be examined on four face recognition datasets of different image sizes, numbers of instances and difficulty. Specifically, we will investigate

1. whether NSGPFL and SPGPFL can achieve better classification performance and learn a smaller number of features than GPFL;
2. whether NSGPFL and SPGPFL can achieve better classification performance and learn a smaller number of features than GPFLW with different weighting factors;
3. Which method of NSGPFL and SPGPFL is better than the other in improving the classification performance and the number of learned features;
4. Whether NSGPFL and SPGPFL with the new individual representation can achieve better classification performance than 34 non-GP-based baseline methods, including CNN-based methods and the methods using well-known face features;

### *1.2. Organisation*

The rest of the paper is organised as follows. Section 2 provides background of this study and reviews typical related work. Section 3 proposes the new representation of GP, new single-objective GP-based feature learning algorithms and new multi-objective GP-based feature learning algorithms. The experimental design is presented in Section 4. Section 5 discusses and analyses the experimental results. The final section presents conclusions and future work.

## 2. Background and Related Work

This section provides background about multi-objective optimisation and GP. It also reviews typical work on face recognition and GP for image feature learning.

### *2.1. Multi-Objective Optimization*

Multi-objective optimisation aims to simultaneously optimise two or more conflicting objective functions. A multi-objective optimisation problem of maximising multiple objective functions can be mathematically formatted as follows

$$maximise\ F(x) = [f_1(x),\ f_2(x),\ \ldots,\ f_k(x)], \tag{1}$$

subject to:

$$g_i(x) \leq 0, i = 1,\ 2,\ \ldots,\ m, \tag{2}$$

$$h_i(x) = 0, i = 1,\ 2,\ \ldots,\ l. \tag{3}$$

where $x$ represents a vector of decision variables and $k$ ($k > 1$) represents the number of objective functions. $f_i(x)$ is the $i$th objective function to be maximised. $g_i(x)$ and $h_i(x)$ are the constraints functions of the problem.

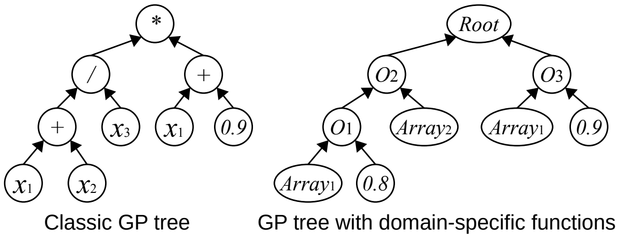
A multi-objective optimisation problem is to find a Pareto front of many non-dominated solutions. Let $y$ and $x$ be two solutions of the above problem. The relation of two solutions $y$ and $z$ can be defined as $y$ dominates $z$ ($y$ is better than $z$ or $z$ is dominated by $y$) if the following condition is satisfied :

$$\forall i \in k : f_i(y) \geq f_i(z) \ \ and \ \ \exists i \in k : f_i(y) > f_i(z). \tag{4}$$

Feature learning can be formulated as a two-objective problem of minimising the number of learned features and minimising the classification error rate. It is noticeable that feature learning for images is different from feature extraction or feature construction. Feature extraction or construction often extracts/constructs a fixed number of features from raw images or pre-defined features [22, 23]. Therefore, feature extraction or construction does not need to consider the objective of the number of features. These two objectives are the same as that for feature selection [27]. The maximum number of features in feature selection is known, while it is unknown in feature learning. This makes the two tasks different and the behaviours/landscapes of the two objectives are different. Therefore, it is necessary to investigate the task of feature learning by formulating it as a multi-objective problem. In this study, we formulate it as a two-objective problem of maximising the classification performance and minimising the number of learned features.

### *2.2. Genetic Programming (GP)*

GP is an evolutionary algorithm of automatically evolving computer programs to solve a problem [10]. Compared with other evolutionary algorithms, GP has a flexible representation, i.e., tree-based representation. In a GP tree, the internal nodes are functions and the terminal nodes are features/variables. A classic example tree of GP is shown in the left part of Figure 1. For handling other types of data, such as image data, strongly typed GP (STGP) [29] is often used since it can deal with multiple data types. With STGP, many domain-specific functions/operators can be employed in GP. An example program is shown in the right part of Figure 1, where $O_1, O_2$ and $O_3$ are the functions/operators that can deal with arrays and/or floating-point numbers. STGP has been widely used on image data by employing many domain-specific operators as functions (internal nodes) of GP trees. Based on STGP, this study develops a new GP approach with a new representation, a new function set and a new terminal set to feature learning for face recognition.



Classic GP tree

GP tree with domain-specific functions

Figure 1: Two different example trees of GP.

### *2.3. Related Work*

#### *2.3.1. Face Recognition Methods*

Existing face recognition methods can be broadly classified into three groups: holistic methods, feature-based structural matching methods, and hybrid methods [2]. Holistic methods extract global features from the whole images and use these features as the inputs to a face recognition system. Popular methods are PCA and LDA, where the extracted features are known as eigenfaces and Fisherfaces [13]. The eigenfaces and Fisherfaces have been widely used in many face recognition tasks [2]. Feature-based structural matching methods extract local features of the partial face, such as eye, mouth and nose, and use these features for recognition. The well-known features are Gabor wavelets [6] and LBP features[7]. Hybrid methods use both holistic and local features for face recognition. In the past decades, many algorithms have been developed and most of them follow one of these three groups [2, 1]. In recent years, it is popular and effective to automatically learn features directly from face images for classification, such as using GP or deep NNs. More works related to deep NNs for face recognition can refer to [30]. It is known that NN-based methods often learn a high-dimensional feature vector and require a large number of training instances. Therefore, it is necessary to develop non-NN-based methods to overcome these limitations. In this study, we propose a new GP-based approach to automatically extracting local features for face recognition.

### *2.3.2. GP for Face Recognition and Feature Learning*

Early work on GP for face recognition can be found in [31, 32]. Bozorgtabar and Rad [31] extract eigenfaces as features and employ GP to construct classifiers for face recognition. Ibrahem *et al.* [32] apply GP to find a model/expression of face feature description. However, this method has only been examined on one dataset.

Liang *et al.* [33] apply GP to construct high-level features and classifiers for figure-ground segmentation using seven different types of image features. The experimental results showed that these methods using different types of features can achieve better performance than four widely-used segmentation techniques. Choi and Choi [34] propose a GP-based system for pulmonary nodule detection on computed tomography images. In the final step of the system, GP is employed to evolve classifiers for categorising nodules and non-nodules using 2D geometric features, 3D geometric features, 2D intensity-based statistic features, and 3D intensity-based statistic features. Zhang *et al.* [35] develop a GP method for multi-class object detection using domain-independent features. The results show that this method achieves promising results on multi-class object detection. Lee and Muhammad [36] apply GP to classify non-classified feature points into the inlier and outlier classes from the features computed using various distances and angle information in the image registration system. Peng *et al.* [37] develop a new GP approach to automatically extracting and constructing high-level features from raw signals for fault type classification. This method achieves better performance than many traditional methods for fault type classification.

Many GP-based feature learning methods have been proposed for image classification. A multi-tier GP approach with an image filtering tier, an aggregation tier and a classification tier is proposed in [38] to learning high-level features for image classification. Al-Sahaf *et al.* [39] simplify this method in the proposed two-tier (2TGP) methods, which only have the aggregation and classification tiers to construct high-level features for image classification. The 2TGP methods can detect rectangle, square, line, and circle regions from the input image and extract pixel statistics from the detected regions. Lensen *et al.* [40] introduce HOG in GP to learn HOG histogram and distance features based on the framework of 2TGP for image classification. In [41], a set of image operators, e.g., Gaussian filter, LoG and Sobel, are cooperated in a multi-layer GP method to facilitate feature learning for image classification. However, these methods have only been examined on binary image classification. Shao *et al.* [22] develop a multi-objective GP algorithm with many filtering and pooling operators to feature learning for image classification with simultaneously maximising the classification performance and minimising the tree size. This method has achieved better performance than several commonly used image classification algorithms [22]. However, it generates a high-dimensional feature vector from an image so that PCA is employed for dimensionality reduction. Al-Sahaf *et al.* [11] propose a GP-based algorithm to automatically extract a flexible number of features from images for texture classification. This method has achieved better performance than the well-known texture descriptors, LBP and its variants. However, this method is only for texture classification [11]. Bi *et al.* [25] develop a GP-based feature learning algorithm with convolution operators for image classification. This method employs convolution and pooling operators as internal nodes of GP trees to extract informative features from images. However, this method could produce a large number of features if few pooling functions are used as internal nodes in the GP trees or the input image is large [25]. Bi *et al.* [42] develop an ensemble method that uses GP to simultaneously learn features and evolve ensembles for image classification. As an ensemble method, it has achieved promising results in many image classification tasks. But the evolved solutions have several complex classifiers, which are difficult to explain.

Although many methods (e.g., [38, 39, 40, 22, 11, 25, 42]) have been developed for feature learning, they have some limitations. In addition, no multi-objective GP algorithms have been applied for multi-objective feature learning with simultaneously maximising the classification performance and minimising the number of learned features. Therefore, this study addresses this by developing new multi-objective GP-based feature learning algorithms for face recognition.

## 3. The Proposed Approaches

This section describes the proposed single-objective and multi-objective GP-based feature learning approaches. First, it introduces the new representation, the function set and the terminal set. Second, it presents the four GP-based feature learning algorithms, i.e., single-objective GP for feature learning (GPFL), single-objective GP for feature learning with a weighted objective (GPFLW), multi-objective GP for feature learning using the idea of non-dominated sorting (NSGPFL), and multi-objective GP for feature learning using the idea of strength Pareto (SPGPFL).

### *3.1. Representation*

The new representation is based on STGP [29], which can integrate multiple different data types into a single tree. To extract informative features from the face, the detection of the partial face such as nose, eye or mouth, is important. The new representation can integrate multiple processes, i.e., region selection, feature extraction and feature combination, into a single tree. In this representation, region selection aims to select a small important region from the input image. For example, it may select the nose, eye or mouth area of the face, which contains informative face features. Feature extraction is to extract features from the selected regions using one of the predefined image descriptors. Feature combination is to combine the features extracted by various descriptors to produce a feature vector. Feature combination allows the new approach to produce a combination of various features, which are potentially more effective for classification than a single type of features.

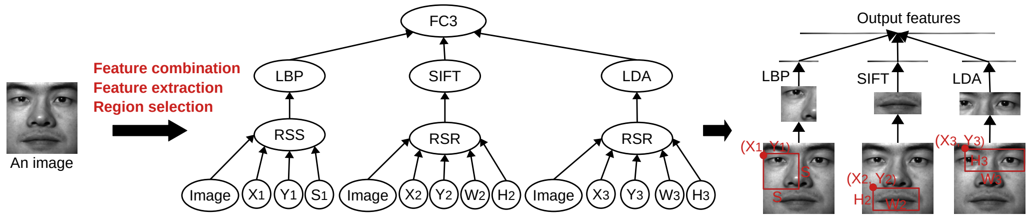


Figure 2: An example program/tree and the process of feature generation using this example tree.

Figure 2 shows an example program/tree and the process of feature generation from an example image using this example program. This example program selects three regions from a face image, uses $LBP$, $SIFT$ and $LDA$ to extract features from the three regions, respectively, and combines these features to form the final feature vector. From the example image, it can be found that informative regions of the face may be selected by the example program. From these regions, effective features can be extracted using the well-known descriptors. The trees of the new approach can be more wide or deep to produce more features for classification during the evolutionary process.

### *3.1.1. Terminal Set*

The new terminal set has the $Image$, $X$, $Y$, $S$, $W$, and $H$ terminals. $Image$ represents the input image and is a matrix/array. The values of the image are normalised into the range of [0, 1] by diving 255, which follows the commonly used method for normalisation. $X$ and $Y$ represent the coordinates of the top-left point of the selected region in the image. $S$ represents the size of a square region selected by the region selection function $RSS$. The $W$ and $H$ terminals represent the width and height of a region selected by the region selection function $RSR$. In the proposed approach, the $X$, $Y$, $S$, $W$, and $H$ terminals are ephemeral random constants. The ranges of $S$, $W$ and $H$ are set to [20, 50]. These allow the new approach to select a region with a size ranging from $20 \times 20$ to $50 \times 50$. The ranges of $X$ and $Y$ are set to $[0, Image\_width - 20]$ and $[0, Image\_height - 20]$, respectively. $Image\_width$ and $Image\_height$ represent the width and height of the image.

### *3.1.2. Function Set*

The operators/functions in the function set are region selection functions, feature extraction functions and feature combination functions.

**Region Selection Functions:** Two region selection functions are employed in the proposed approach. The first function is $RSS$, which can select a square region from the input image. The $RSS$ function has four arguments, i.e., $Image$, $X$, $Y$, and $S$. The region selected by this function is $Image[X : min(Image\_width, X + S), Y : min(Image\_height, Y + S)]$. The second function is $RSR$, which can select a rectangle region from the input image. The $RSR$ function has five arguments, i.e., $Image$, $X$, $Y$, $W$, and $H$. The region selected by this function is $Image[X : min(Image\_width, X + W), Y : min(Image\_height, Y + H)]$.

**Feature Extraction Functions:** To extract effective features from the selected region, four commonly used descriptors are developed as feature extraction functions in the proposed approach. The four descriptors are $LDA$ [13], $SIFT$ [5], $LBP$ [6], and $Conca$. The $LDA$ function extracts the features that can maximise the between-class distance and minimise the within-class distance. The extracted features are known as Fisherfaces, which have been widely used in face recognition [13]. The number of features extracted by $LDA$ is $C - 1$, where $C$ is the number of classes of the dataset. The $SIFT$ function extracts histogram features of gradient magnitudes and directions. In the proposed approach, the dense SIFT method [43] is employed to extract features from the whole image without keypoint detection. From each image/region, the $SIFT$ method produces 128 features. The $LBP$ method extracts invariant texture features from an image/region. The LBP features have been applied for face recognition [6]. In the $LBP$ function, the number of neighbours is set to 8 and the size of the radius is set to 1.5. The uniform version of LBP is employed as the extracted features are invariant to rotation. From each image/region, the $LBP$ function extracts 59 features. The $Conca$ function concatenates all the rows in the image/region into a vector. This function returns raw pixels of the region without any transformation since some regions may contain sufficient information for classification. Based on the region detection functions, the number of features produced by $Conca$ is between 400 to 2500. With these feature extraction functions, various numbers of features can be produced by different GP trees during the evolutionary process.

**Feature Combination Functions:** Feature combination functions aim to concatenate features extracted from different regions into a feature vector. The feature combination functions are $FC2$ and $FC3$. These two functions combine features extracted from 2 and 3 regions, respectively. These two functions can be the root node of the GP trees and can also be the children nodes of each other. This allows the GP trees to have a flexible depth/length with the capability to produce more features.

### *3.2. Single-Objective GP for Feature Learning (GPFL)*

With the new representation, a single-objective GP-based feature learning (GPFL) algorithm can be applied for face recognition. The goal of GPFL is to automatically learn a number of features to maximise the classification performance. The framework of GPFL is shown in Algorithm 1. The fitness/objective function is the classification accuracy defined by

$$F_{acc} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{5}$$

where TP, TN, FP, and FN indicate the number of true positives, the number of true negatives, the number of false positives, and the number of false negatives, respectively. The value range of $F_{acc}$ is in $[0, 1]$.

In the fitness evaluation process, five-fold cross-validation on the training set $\mathcal{D}_{train}$ is employed to obtain the classification accuracy [16]. Because the features extracted by different feature extraction functions are in different scales, the learned features are normalised using the min-max normalisation method. Then the normalised data are split using five-fold cross-validation

**Algorithm 1:** Framework of GPFL and GPFLW

```
begin
    Split an image dataset into a training set $\mathcal{D}_{train}$ and a test set $\mathcal{D}_{test}$;
    $P_0 \leftarrow$ Initialise a population of $N$ trees based on the new representation;
    Evaluate $P_0$ using the objective function on $\mathcal{D}_{train}$;
    $g \leftarrow 1$;
    Update $Best\_Ind$ according to $P_0$;
    while $g \leq G$ do
        $Selected \leftarrow$ Select trees from $P_{g-1}$ using torunament selection;
        $P_g \leftarrow$ Generate a new population from $Selected$ using crossover and
          mutation operators;
        Evaluate $P_g$ using the objective function on $\mathcal{D}_{train}$ using five-fold cross
          validation;
        Update $Best\_Ind$ according to $P_g$;
        $g \leftarrow g + 1$;
    end
    Calculate the number of learned features by $Best\_Ind$;
    Calculate the test accuracy of $Best\_Ind$ on $\mathcal{D}_{test}$;
    Return $Best\_Ind$, the number of learned featues, and the test accuracy.
end
```

to feed into a linear support vector machine (SVM) for training and testing. The linear support vector machine is employed because it is commonly used in image classification [22] and it has fewer parameters than SVM with other kernels. The mean test accuracy of the five folds is set as the objective value for the evaluated individual.

### *3.3. GPFL for Weighting Two Objectives (GPFLW)*

Single-objective GPFL may learn a high-dimensional feature vector from images because the fitness function (as listed in Equation 5) only focuses on maximising classification accuracy and does not make any constrain on the number of features. To address this, a new fitness function is proposed with the goals of maximising the classification accuracy and minimising the number of learned features. Instead of using the number of learned features as an objective function directly, we use the inverse ratio of the number of features and change this objective to be a maximisation problem. With a ratio $\alpha$ ($\alpha \in [0,\ 1]$), the new fitness/objective function is defined by

$$F_{ratio} = \frac{Min(C-1, 59)}{NF} \tag{6}$$

$$F_{comb} = \alpha * F_{acc} + (1 - \alpha) * F_{ratio}. \tag{7}$$

where $\alpha$ ($\alpha \in [0, 1]$) and $(1-\alpha)$ indicates the importance of the two objectives $F_{acc}$ (Equation 5) and $F_{ratio}$ (Equation 6). $C$ indicates the number of classes and $NF$ indicates the number of learned features. $F_{ratio}$ indicates the inverse ratio of the number of learned features, which is in the range of $[0, 1]$. In this objective, the values of $F_{ratio}$ is set to 0 if $NF < Min(C-1, 59)$. $Min(C-1, 59)$ indicates the minimal number of learned features. Based on the algorithm design, it can be found that the minimal number of learned features is the minimal of $C-1$ and 59. $C-1$ is the number of features extracted by the $LDA$ function in the function set and 59 is the number of features extracted by the $LBP$ function. The integrated/weighted fitness function is $F_{comb}$ defined by Equation 7. The value of $F_{comb}$ is in the range of $[0, 1]$. The fitness evaluation process of GPFLW is the same as that of GPFL except for the calculation of the fitness values. The algorithm framework of GPFLW is outlined in Algorithm 1.

### *3.4. Multi-Objective GPFL with Non-Dominated Sorting (NSGPFL)*

The fitness function in Equation 7 jointly optimises the two objectives, i.e., the classification accuracy and the number of learned features. However, the weight $\alpha$ is difficult to set because the importance of the two objectives are unknown. To address this, it is necessary to optimise these two objectives using a multi-objective optimisation algorithm. A multi-objective GP-based feature learning algorithm using the idea of non-dominated sorting (NSGPFL) is developed optimise the two objectives ($F_{acc}$ and $F_{ratio}$) described by Equations 5 and 6. The idea of non-dominated sorting is from non-dominated sorting genetic algorithm II (NSGA-II) [44], which is one of the most popular evolutionary multi-objective optimisation algorithms. The overall algorithm of NSGPFL is described in Algorithm 2. The main procedure of NSGPFL is the same as that of NSGA-II [44]. In the fitness evaluation of NSGPFL, the calculation of the two objective functions are based on Equation 6. After the evolutionary process, the non-dominated solutions are tested on the test set and the numbers of learned features are calculated.

### *3.5. Multi-Objective GPFL with Strength Pareto (SPGPFL)*

In order to further investigate the use of evolutionary multi-objective algorithms for feature learning in image classification, another multi-objective

**Algorithm 2:** Framework of NSGPFL

```
begin
    Split an image dataset into a training set $\mathcal{D}_{train}$ and a test set $\mathcal{D}_{test}$;
    $P_0 \leftarrow$ Initialise a population of $N$ trees based on the new representation;
    Evaluate $P_0$ using the two objective functions on $\mathcal{D}_{train}$;
    $O_0 \leftarrow$ Generate a new population using crossover and mutation operators;
    $g \leftarrow 0$;
    while $g \leq G$ do
        Evaluate $O_g$ using the two objective functions on $\mathcal{D}_{train}$;
        $R_g \leftarrow\ P_g \cup O_g$;
        $front = (front_1, front_2, \dots) \leftarrow$ Identify different levels of
          non-dominated fronts in $R_g$ using fast non-dominated sorting;
        $P_{g+1} \leftarrow \emptyset$ and $i = 1$;
        while $|P_{g+1}| < N$ do
            if $|P_{g+1}| + front_i < N$ then
                $P_{g+1} \leftarrow\ P_{g+1} \cup front_i$;
                $i = i + 1$;
            else
                Calculate crowding distance of each individual in the front
                  $front_i$;
                $P_{g+1} \leftarrow$ Add $(N - |P_{g+1}|)$ least crowded indiviudals of $front_i$;
            end
        end
        $O_{g+1} \leftarrow$ Generate a new population using crossover and mutation
          operators;
        $g \leftarrow g + 1$;
    end
    Calculate the number of learned features by each non-dominated solution in
      $front_1$;
    Calculate the test accuracy of each non-dominated solution in $front_1$ on
      $\mathcal{D}_{test}$;
    Return each individual in $front_1$, the number of learned features, and the
      test accuracy.
end
```

evolutionary algorithm using the idea of strength Pareto [45] is proposed. The same as NSGPFL, the SPGPFL algorithm aims to maximise the classification accuracy and minimise the number of learned features. Different from NAGAII, the strength Pareto evolutionary algorithm (SPEA2) [45] is an evolutionary multi-objective algorithm that performs environmental selection based on fitness and density. SPEA2 is a widely used evolutionary multi-objective algorithm and has been employed in GP for other tasks, such as figure-ground image segmentation [23]. The major steps of the proposed SPGPFL algorithm are the same as that of SPEA2. The algorithm framework of SPGPFL is described in Algorithm 3. In SPGPFL, the *Archive* is employed to store non-dominated solutions. More details about SPEA2 can be found in [45]. After the evolutionary process, the non-dominated solutions from *Archive* are identified and tested on the test set.

## 4. Experiment Design

### *4.1. Datasets*

To examine the performance of the proposed approaches, four well-known face recognition datasets of varying difficulty are used for conducting the experiments. The four datasets are ORL [46], Extended Yale B [47], Aberdeen [48], Faces95 [49]. The details of the four datasets are listed in Table 1. It is found that these datasets have different image sizes, numbers of classes, numbers of training and test instances. It is noted that these four datasets are not extremely large datasets but they cover typical face image variations, i.e., the variations of illumination, occlusion, pose, expression, background, and facial details.

The **ORL** dataset [46] is a small dataset of 400 images, i.e., ten images per class. The face images have variations of pose, illumination condition and expression. In the experiments, five images per class are randomly selected for training and the remaining images are used for testing. The **Extended Yale B** dataset [47] has 2424 face images of 38 different subjects. The images were sampled under various light conditions. Each class has 63/64 images. In the experiments, 20 images per class are randomly selected for training and the remaining images are used for testing. The original 168×192 images are resized to 90×100 to reduce the computational cost. The **Aberdeen** dataset [47] has 690 images in 116 classes. Some classes have a small number of images, i.e., smaller than eight. Therefore, only 30 classes with a larger number of images, i.e., over ten images, are selected for the experiments. The

**Algorithm 3:** Framework of SPGPFL

```
begin
    Split an image dataset into a training set $\mathcal{D}_{train}$ and a test set $\mathcal{D}_{test}$;
    $P_0 \leftarrow$ Initialise a population of $N$ trees based on the new representation;
    $g \leftarrow 0$;
    $Archive \leftarrow \emptyset$;
    while $g \leq G$ do
        Evaluate $P_g$ using the two objective functions on $\mathcal{D}_{train}$;
        $Union \leftarrow P_g \cup Archive$;
        Calculate the raw fitness of each individual in $Union$;
        Calculate the density of each individual in $Union$;
        Calculate the fitness of each individual based on the raw fitness and the
          density value;
        Identify non-dominated solutions in $Union$ and add then to $Archive$;
        if $|Archive| < Maximum\ Archive\ Size$ then
            Add the $Maximum\ Archive\ Size - |Archive|$ non-dominated
              solutions in the current population to $Archive$;
            $i = i + 1$;
        else
            Remove the most similar individuals from $Archive$;
        end
        $Selected \leftarrow$ Select individuals as parents from $Archive$;
        $P_{g+1} \leftarrow$ Generate a new population using crossover and mutation
          operators from $Selected$;
        $g \leftarrow g + 1$
    end
    Obtain non-domained soltuions from $Archive$;
    Calculate the number of learned features by each non-domined solution;
    Calculate the test accuracy of each non-domained solution;
    Return each individual in $Archive$, the number of learned featues, and the
      test accuracy.
end
```

sizes of the images vary from 336×480 to 624×544. The images are resized to 100×100 and the colour images are converted to gray-scale images to reduce the computational cost. In the experiments, eight images per classes are randomly selected for training and the remaining images are used for testing. The final dataset is **Faces95** [49], having 1440 images in 72 classes. Each class has 20 images with the size of 180×200. The images have variations of background, expression and illumination. In the experiments, ten images per class are randomly selected for training and the remaining images are used for testing. The original colour images are converted to gray-scale images and are resized to 90×100 for reducing the computational cost. Example images of these datasets are shown in Figure 3.

Table 1: Summary of the datasets

| Datasets | #Class | Image Size | Train Set (Per Class) | Test Set |
|---|---|---|---|---|
| ORL | 40 | 112×92 | 200 (5 images) | 200 |
| Extended Yale B | 38 | 90×100 | 760 (20 images) | 1664 |
| Aberdeen | 30 | 100×100 | 240 (8 images) | 269 |
| Faces95 | 72 | 90×100 | 720 (10 images) | 720 |

### *4.2. Baseline Methods*

A large number of baseline methods are used for comparisons, i.e., two NN-based methods and 32 traditional methods using different classification algorithms and face features. The two NN-based methods are LeNet [50] and a five-layer CNN [51]. The other 32 methods use eight types of well-known features and four commonly used classification algorithms. The eight types of features are Fisherfaces, Eigenfaces1, Eigenfaces2, SIFT, LBP, Original, Downsample4, and Downsample8. Fisherfaces are $C-1$ features extracted by LDA [13]. The Eigenfaces1 and Eigenfaces2 features are generated using PCA [12]. The values of explained variance in PCA are set to 0.8 and 0.9 to generate various numbers of features to form the Eigenfaces1 and Eigenfaces2 features, respectively. The SIFT [5] and LBP [6] features are extracted from the images using the feature extraction functions in the proposed approach. The Original features indicate the raw pixels. Downsample4 and Downsample8 features are down-sampled from the images using 4×4 and 8×8 windows [52], respectively. The four commonly used classification algorithms are k-nearest neighbour (KNN) [11], SVM [22], sparse representation-based classification (SRC) [52], and random forest (RF) [53].

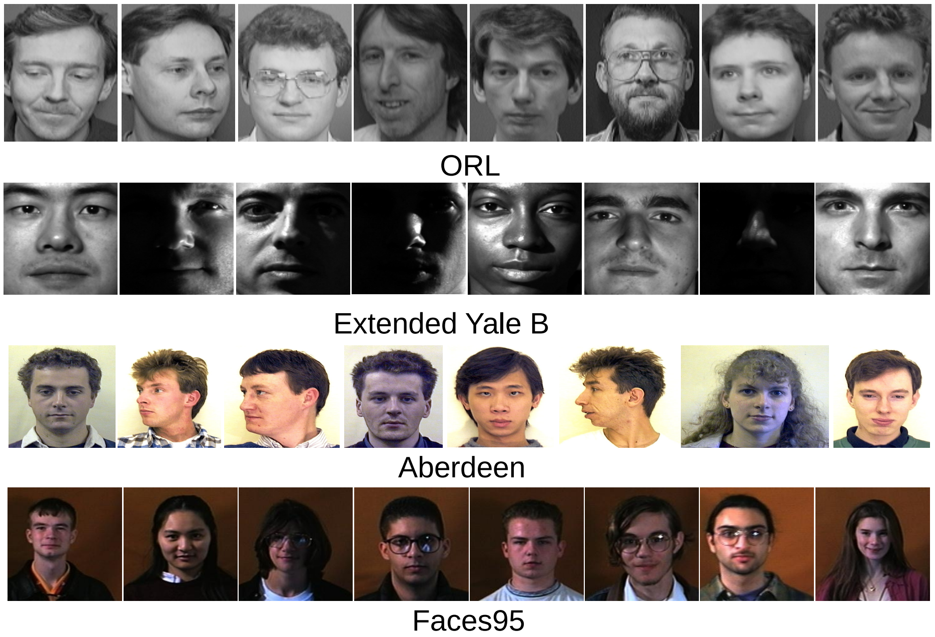


Figure 3: Example images of the **ORL, Extended Yale B, Aberdeen**, and **Faces95** datasets.

It is noticeable that there are several evolutionary multi-objective algorithms that can optimise the objective of classification accuracy and the objective of the number of features, such as in [27, 28]. However, these algorithms are developed for feature selection, which is very different from image feature learning. Feature selection is based on the dataset with well-established features, while image feature learning is based on raw images. In feature selection, the number of features is known, while the number of features is unknown in image feature learning. Feature selection only selects and produces a small set of existing features, while feature learning generates new features. Therefore, the current evolutionary multiobjective algorithms cannot be employed for comparisons.

### *4.3. Parameter Settings*

In the four GP-based feature learning algorithms, the maximum number of generations is set to 50 and the population size is set to 100 [51, 42]. The crossover and mutation rates are set to 0.8 and 0.2, respectively. Elitism is not employed in these GP algorithms as the best individual or the non-dominated solutions are recorded/updated at each generation. Tournament selection with size 7 is employed to select individuals for genetic operations. The minimum tree depth is set to 2 and the maximum tree depth is set to 8. The *ramped-half-and-half* method is used for generating the initial population. The stopping criteria is reaching the maximum number of generations. For GPFLW, $\alpha$ is a weighting factor and is subjective to the relative importance of the two objectives, i.e., the objective of classification accuracy and the objective of the number of learned features. The best value of $\alpha$ may vary with the dataset and tuning is needed to find it. In this paper, we investigated the performance of GPFLW with two representative values of $\alpha$, i.e., i.e., $\alpha = 0.5$ and $\alpha = 0.8$, and keep them the same for all the datasets for simplification. For $\alpha$=0.5, we would like to investigate the performance of GPFLW when these two objectives are equally important. For $\alpha$=0.8, we would like to investigate the performance of GPFLW when the first objective is more important than the second one.

The parameters for the classification algorithms are the commonly used settings. In KNN, the number of neighbours is 1 [11]. The linear kernel is employed in SVM as it has fewer parameters than the other kernels [22]. In RF, the number of trees is 500 and the minimum tree depth is 100 [53]. In LeNet and CNN, the batch size is set to 20 and the number of epochs is set to 100, referring to [53].

The GP-based feature learning algorithms are implemented using the *DEAP* (Distributed Evolutionary Algorithm in Python) package [54]. The classification algorithms, i.e., KNN, SVM and RF, are based on the *scikit-learn* package [55]. The implementations of LeNet and CNN are based on the *Keras* package [56]. For fair comparisons, each algorithm runs independent 30 times on each dataset. The test results from the 30 runs are reported in the following section.

To compare the performance of NSGPFL and SPGPFL, one commonly used indicator, hypervolume [57], is used. Hypervolume only needs one reference point, which is earlier to set in contrast to a set of reference points required in other indicators, such as the inverted generational distance (IGD) indicator [58] . In each run, the NSGPFL or SPGPFL algorithm obtains two Pareto front approximations, which are from the training set and the test set, respectively. The two sets of metric values (the values of $F_{acc}$ and $F_{ratio}$) are used to calculate the hypervolume. In the calculation, the values of $1 - F_{acc}$ and $1 - F_{ratio}$ are used and the reference point is set to (1, 1), indicating the worst values of $1 - F_{acc}$ and $1 - F_{ratio}$. A larger hypervolume value indicates a better algorithm.

## 5. Results and Discussions

This section discusses and analyses the performance of GPFL, GPFLWs, NSGPFL, SPGPFL, and the baseline methods on the four different datasets. The classification results and the number of learned features of the GPFL, GPFLWs, NSGPFL, and SPGPFL on the four datasets are shown in Table 2, and Figures 4 and 5. In Figures 4 and 5, "NSGPFL-Best" and "SPGPFL-Best" indicate the approximated Pareto fronts of the 30 runs. The comparisons with NSGPFL, SPGPFL and a large number of non-GP-based baseline methods are demonstrated in Tables 4 and 5. A statistical test: Wilcoxon rank-sum test with a 5% significance level [59], is employed to compare the performance between GPFL, GPFLWs, NSGPFL, SPGPFL, and the benchmark algorithms.

### *5.1. Comparisons between GPFL, GPFLWs, NSGPFL, and SPGPFL*

The test accuracy and the number of learned features of GPFL, GPFLWs (GPFLW with a weighting factor $\alpha$=0.5 and GPFLW with a weighting factor $\alpha$=0.8), NSGPFL, and SPGPFL are listed in Table 2. The results are the maximum test accuracy, the mean accuracy and standard deviation, the maximum number of learned features, the average number of learned features and standard deviation. To compare with these methods, the results of NSGPFL and SPGPFL are obtained using the individuals with the best classification performance of the training sets. In Table 2, the symbols "+", "=" and "–" ("↑", "→" and "↓" ) indicate that NSGPFL (SPGPFL) achieves significantly better, similar and worse results than/to the compared algorithm, respectively.

### *5.1.1. NSGPFL and SPGPFL versus GPFL*

From Table 2, it can be found that NSGPFL achieves similar classification performance than GPFL on the ORL, Aberdeen and Faces95 datasets and significantly worse performance on the Extended Yale B dataset. Compared with GPFL, SPGPFL achieves similar results on the four datasets. Although the mean accuracy obtained by GPFL is slightly higher than that by SPGPFL, there are no significant differences. Table 2 also shows that GPFL learns a larger number of features than NSGPFL or SPGPFL on the four datasets. More importantly, the average number of features learned by GPFL is 721.63 on ORL, which is much larger (almost double the number) than that of NSGPFL (385.83) or SPGPFL (440). On Aberdeen, NSGPFL

Table 2: Test accuracy (%) and the number of learned features on the four datasets. The symbols "+", "=" and "–" ("↑", "→" and "↓" ) indicate that NSGPFL (SPGPFL) achieves significantly better, similar and worse results than/to the compared algorithm

| | **Test Accuracy** | | **Number of Learned Features** | |
|---|---|---|---|---|
| | **Max** | **Mean±St.dev** | **Max** | **Mean±St.dev** |
| **Dataset** | **ORL** | | | |
| GPFL | 99.50 | 97.90±1.11 =→ | 1348 | 721.63±273.78 |
| GPFLW ($\alpha$=0.5) | 90.50 | 86.13±2.43 + ↑ | 59 | 59.00±0.00 |
| GPFLW ($\alpha$=0.8) | 99.50 | 95.18±3.16 + ↑ | 600 | 216.37±145.00 |
| NSGPFL | 99.50 | 97.48±1.66 → | 934 | 385.83±174.75 |
| SPGPFL | 99.50 | 97.40±1.19 | 649 | 440.00±115.18 |
| **Dataset** | **Extended Yale B** | | | |
| GPFL | 99.10 | 97.49±1.46 − → | 2496 | 1167.00±469.16 |
| GPFLW ($\alpha$=0.5) | 67.13 | 19.61±27.73 + ↑ | 74 | 69.50±6.87 |
| GPFLW ($\alpha$=0.8) | 98.62 | 79.78±34.41 = ↑ | 1944 | 500.53±405.61 |
| NSGPFL | 98.32 | 95.32±2.83 ↑ | 1560 | 753.60±308.46 |
| SPGPFL | 98.92 | 96.60±2.15 | 2719 | 1120.53±560.74 |
| **Dataset** | **Aberdeen** | | | |
| GPFL | 99.26 | 97.37±1.51 =→ | 3460 | 830.83±592.38 |
| GPFLW ($\alpha$=0.5) | 85.50 | 75.68±17.44 + ↑ | 59 | 58.87±0.34 |
| GPFLW ($\alpha$=0.8) | 97.40 | 95.32±2.65 + ↑ | 472 | 154.20±83.62 |
| NSGPFL | 98.88 | 97.06±1.34 → | 659 | 417.60±142.91 |
| SPGPFL | 99.63 | 97.30±1.08 | 1238 | 562.13±238.62 |
| **Dataset** | **Faces95** | | | |
| GPFL | 99.44 | 97.95±0.88 =→ | 708 | 232.07±153.80 |
| GPFLW ($\alpha$=0.5) | 97.64 | 54.02±35.16 + ↑ | 118 | 89.80±23.03 |
| GPFLW ($\alpha$=0.8) | 98.61 | 96.58±2.25 + ↑ | 177 | 134.07±25.95 |
| NSGPFL | 99.17 | 97.82±0.85 → | 354 | 180.93±71.34 |
| SPGPFL | 99.17 | 97.62±0.78 | 708 | 184.87±121.62 |

learns a maximum number of 659 features and SPGPFL learns a maximum number of 1238 features, which are much smaller than that by GPFL, i.e., 3460 features. The results show that both NSGPFL and SPGPFL achieve comparable performance to GPFL but learn a smaller number of features. The reason is that GPFL does not have a limit on the number of learned features when maximising the classification performance. In contrast to GPFL, NSGPFL and SPGPFL find the solutions that achieve similar performance but extract a smaller number of features via optimising these two objectives (the classification accuracy and the number of features) and identifying non-dominated solutions.

On the training sets, as shown in Figure 4, NSGPFL or SPGPFL finds a set of non-dominated solutions that extract various numbers of features and achieve different training performance. Compared with the solutions of GPFL, the solutions of NSGPFL or SPGPFL are more diverse in the classification performance and the number of features. On the test sets, as shown in Figure 5, most of the solutions of GPFL achieve better test accuracy but use a larger number of features than that of NSGPFL or SPGPFL. From these two figures, however, it can be found that when the number of features increases, the classification performance of the training or test set does not always increase significantly. This confirms the necessity of optimising both the objectives of the number of features and the classification performance.

To sum up, the results suggest that NSGPFL and SPGPFL learn a smaller number of features and achieve comparable classification performance than the single-objective GPFL by simultaneously optimising these two objectives: the number of learned features and the classification performance.

### *5.1.2. NSGPFL and SPGPFL versus GPFLWs*

The results in Table 2 show that NSGPFL achieves significantly better or similar results than GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) on the four datasets. The results show that SPGPFL significantly outperforms GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) on the four datasets. From Table 2, the results show that GPFLW ($\alpha$=0.5) achieves the lowest mean accuracy of 19.61% and GPFLW ($\alpha$=0.8) achieves a mean accuracy of 79.7% on the Extended Yale B dataset. GPFLW ($\alpha$=0.5) achieves the lowest mean accuracy of 75.68% on the Aberdeen dataset and of 54.02% on the Faces95 dataset. On these three datasets, both NSGPFL and SPGPFL achieve a mean accuracy of over 95%. The results show that NSGPFL and SPGPFL achieve better classification performance than GPFLW with the two different weighting factors. Table 2 shows that GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) learn a smaller number of features than NSGPFL or SPGPFL.

The training and test results of the 30 runs are shown in Figures 4 and 5. With a constraint on the number of the features in the objective, GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) learn a small number of features and achieve low classification performance of the training or test set. On the training sets, as shown in Figures 4, GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) achieve worse classification performance than NSGPFL or SPGPFL when the number of learned features is similar or the same. Compared with GPFLW ($\alpha$=0.8), GPFLW ($\alpha$=0.5) extracts smaller numbers of features in most cases but

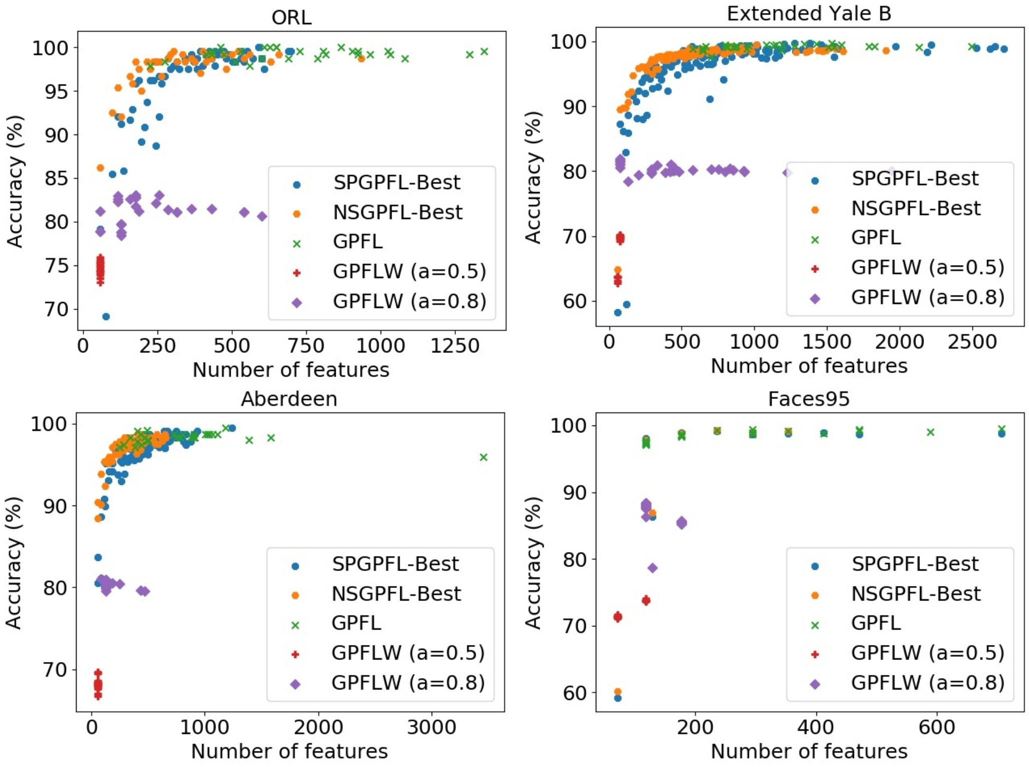


Figure 4: Training results (i.e.. the number of learned features and accuracy) of the 30 runs obtained by GPFL, GPFLW ($\alpha$=0.5), GPFLW ($\alpha$=0.8), NSGPFL, and SPGPFL on the four datasets.

achieves worse classification performance on the training set. It is noted that some of the solutions of GPFLW ($\alpha$=0.8) still extract a large number of features, such as on the ORL and Extended Yale B datasets. The results show that the classification performance is difficult to be optimised in the weighted objective/fitness function of GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8). In contrast, the objective of the classification performance can be optimised in the multi-objective algorithms, i.e., NSGPFL and SPGPFL.

On the test sets, most solutions of GPFLW ($\alpha$=0.8) can achieve good classification performance, but not as good as some solutions of NSGPFL and SPGPFL. Figure 4 shows that the solutions of GPFLW ($\alpha$=0.5) achieve worse classification accuracy than most solutions of NSGPFL and SPGPFL when the number of features is similar or the same.

To sum up, compared with GPFLWs, NSGPFL and SPGPFL are more

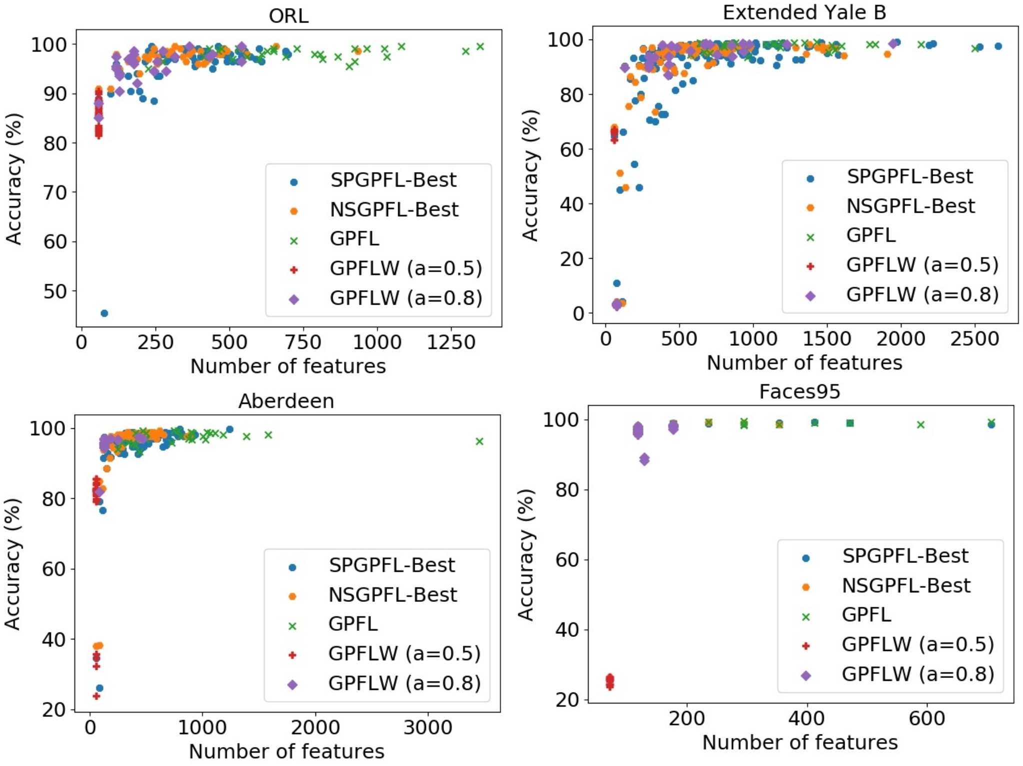


Figure 5: Test results (i.e.. the number of learned features and accuracy) of the 30 runs obtained by GPFL, GPFLW ($\alpha$=0.5), GPFLW ($\alpha$=0.8), NSGPFL, and SPGPFL on the four datasets.

effective for optimising these two objectives. Because the trade-off between the two objectives is unknown, it is difficult to integrate them into a single-objective to be jointly optimised as that in GPFLWs. The results show that GPFLWs with different $\alpha$ values achieve different performances on the four datasets, indicating that $\alpha$ is an important factor in GPFLW but is difficult to set. Compared with GPFLWs, NSGPFL and SPGPFL are more effective by simultaneously optimising the objectives of the classification performance and the number of features.

### *5.1.3. NSGPFL versus SPGPFL*

Table 2 shows that SPGPFL achieves significantly better performance than NSGPFL on the Extended Yale B dataset and similar performance on the remaining three datasets. On the ORL and Faces95 datasets, NSGPFL achieves better mean accuracy than SPGPFL. However, SPGPFL achieves better maximum accuracy and mean accuracy than NSGPFL on Extended Yale B and Aberdeen. Table 2 shows that NSGPFL learns a smaller number of features than SPGPFL on the four datasets. The results in Table 2 show that SPGPFL achieves better classification performance than NSGPFL but learns a larger number of features. Because the results in Table 2 are obtained by the solutions with the best classification performance of the training set, these results cannot comprehensively show the performance of NSGPFL and SPGPFL in multi-objective feature learning. Therefore, more detailed comparisons are conduced as follows.

On the training sets, as shown in Figure 4, both NSGPFL and SPGPFL obtain a set of diverse solutions with different training accuracies and numbers of features. The approximated Pareto fronts obtained by NSGPFL and SPGPFL confirm that the classification performance of the training set increases with the number of learned features. From Figure 4, it can be found that the solutions of NSGPFL achieve better classification performance of the training sets than that of SPGPFL when the number of learned feature is similar or the same on the four datasets. Compared with NSGPFL, SPGPFL finds more solutions with a larger numbers of features, especially on the Extended Yale B and Aberdeen datasets. It is also noted that some solutions of NSGPFL and SPGPFL obtain a small number of features with poor classification performance of the training sets.

On the test sets, as shown in Figure 5, NSGPFL and SPGPFL obtain different approximated Pareto fronts from that on the training sets. The results show that NSGPFL obtains better classification accuracy than SPGPFL when the number of features is similar or the same. It can also be found that some solutions of NSGPFL and SPGPFL achieve poor classification performance when the numbers of features become too small, such as on the ORL and Extended Yale B datasets. It is noted that the solutions with poor classification performance of the training sets and small numbers of features achieve very low classification accuracy of the test sets. This indicates that when the number of learned features is small, the generalisation performance is poor.

Table 3 lists the maximum values, mean values and standard deviations of the hypervolume of the NSGPFL and SPGPFL algorithm on the training and test sets. The "↑" and "↓" symbols indicate that SPGPFL is significantly better or worse than NSGPFL. It can be found that SPGPFL is significantly worse than NSGPFL in seven comparisons and significantly better than NSGPFL in one comparison out of the total eight comparisons. Compared with

Table 3: Hypervolume of the training and test sets. The “↑” and “↓” symbols indicate that SPGPFL is significantly better and worse than NSGPFL, respectively

| | **Training** | | **Test** | |
|---|---|---|---|---|
| | **Max** | **Mean±St.dev** | **Max** | **Mean±St.dev** |
| **Dataset** | **ORL** | | | |
| NSGPFL | 0.6083 | 0.5948±0.0067 ↓ | 0.6210 | 0.6041±0.0079 ↓ |
| SPGPFL | 0.5758 | 0.5506±0.0141 | 0.6122 | 0.5856±0.0174 |
| **Dataset** | **Extended Yale B** | | | |
| NSGPFL | 0.8704 | 0.8633±0.0034 ↓ | 0.7957 | 0.7674±0.0214 ↓ |
| SPGPFL | 0.8305 | 0.7580±0.0353 | 0.7725 | 0.7233±0.0285 |
| **Dataset** | **Aberdeen** | | | |
| NSGPFL | 0.4678 | 0.4625±0.0045 ↓ | 0.4430 | 0.3662±0.0308 ↑ |
| SPGPFL | 0.4435 | 0.4274±0.0092 | 0.4374 | 0.4178±0.0269 |
| **Dataset** | **Faces95** | | | |
| NSGPFL | 0.4332 | 0.4292±0.0045 ↓ | 0.3618 | 0.3566±0.0060 ↓ |
| SPGPFL | 0.4289 | 0.4098±0.0112 | 0.3580 | 0.3399±0.0132 |

SPGPFL, NSGPFL obtains higher maximal and mean hypervolume values in most cases. The results show that the non-dominated solutions of NSGPFL cover larger space than that of SPGPFL to the reference point. As a result, NSGPFL with the idea of non-dominated sorting is better than SPGPFL for multi-objective feature learning in face recognition.

The reason that NSGPFL is better than SPGPFL can be found in Figures 6 and 7, which show the median fronts of NSGPFL and SPGPFL on the training sets and the test sets. The median front of each datasets is from a single run that obtains the median hypervolume value on the training or test set [59]. Figures 6 and 7 show that SPGPFL is able to find more boundary solutions than NSGPFL but the solutions of NSGPFL achieve better performance than SPGPFL. The results show that SPGPFL finds solutions with better diversity than NSGPFL, but NSGPFL can find a set of non-dominated solutions with better performance than SPGPFL.

The results suggest that both NSGPFL and SPGPFL automatically find a set of non-dominated solutions that reduce the number of learned features and improve the classification performance. Compared with SPGPFL, NSGPFL finds a set of solutions with higher classification performance of the training and test sets when the numbers of features are similar or the same. The results of hypervolume indicator show that NSGPFL is more effective than SPGPFL for multi-objective feature learning in face recognition.

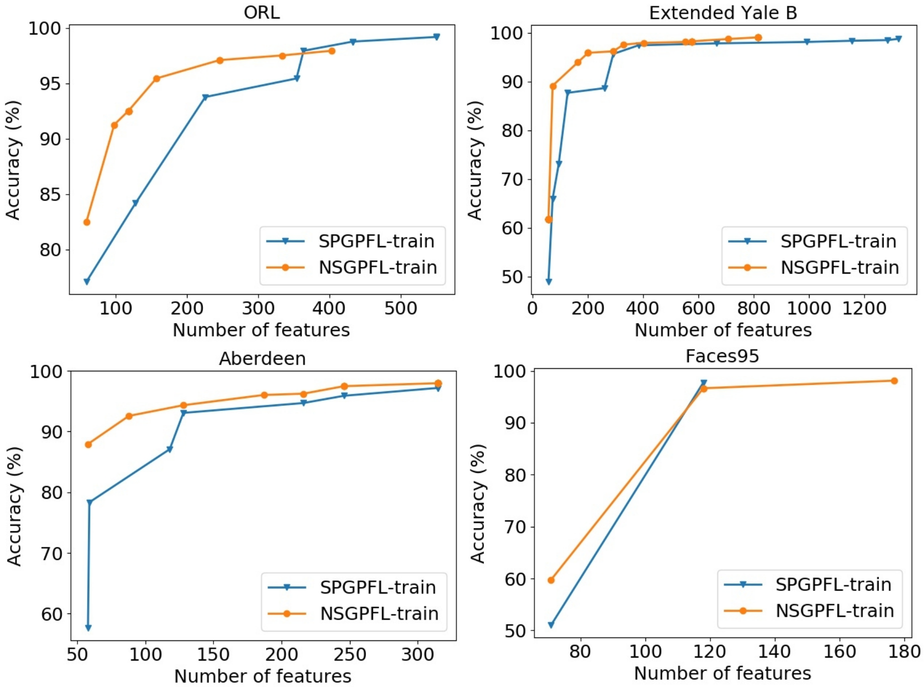


Figure 6: Median fronts of NSGPFL and SPGPFL on the training sets of the four datasets.

*5.1.4. GPFL versus GPFLWs:*

The results in Table 2 show that GPFL achieves better classification performance than GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) on the four datasets. GPFLWs have a weighted fitness function (Equation 7), aiming to maximise the classification accuracy and minimise the number of learned features, while GPFL does not have a limit on the number of learned features so that it can learn a larger number of features to achieve better classification performance than GPFLWs.

The number of features in Table 2 and the results from the 30 runs in Figures 4 and 5 confirm that GPFL learns a larger number of features to achieve a better classification performance than GPFLWs. GPFL learns over 400 features on the Faces95 dataset and over 1000 features on the remaining datasets, while GPFLWs learn less than 200 features on Faces95 and less than 1000 features on the remaining three datasets. The results suggest that both GPFLW ($\alpha$=0.5) and GPFLW ($\alpha$=0.8) significantly reduce the number of learned features but have a limited ability to improve the classification accuracy of the four datasets.

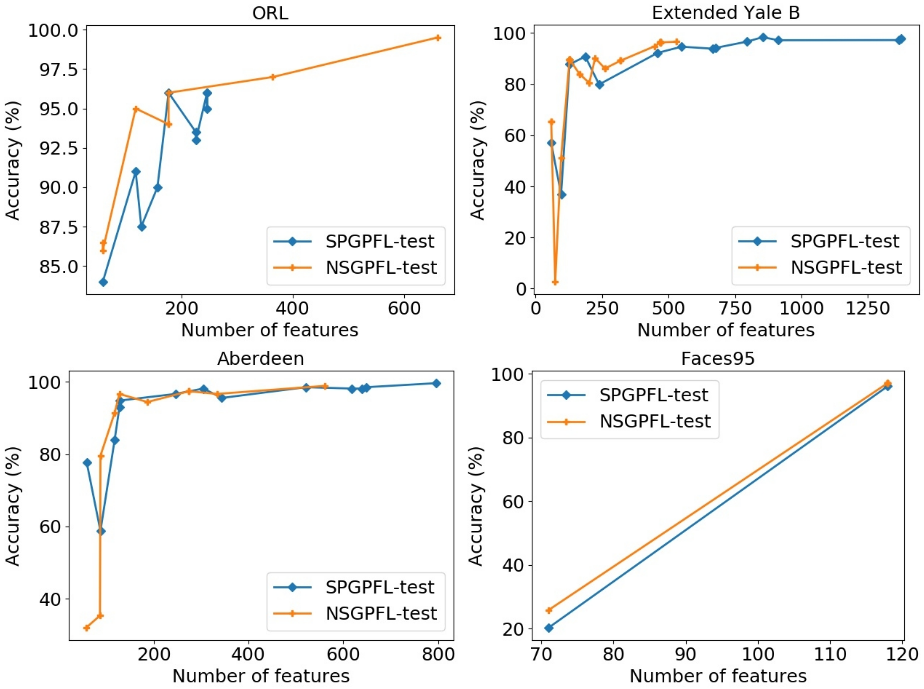


Figure 7: Median fronts of NSGPFL and SPGPFL on the test sets of the four datasets.

### *5.2. Comparisons with Baseline Methods*

To show the effectiveness of the proposed GP-based feature learning algorithms, we compare the results obtained by NSGPFL and SPGPFL with the 34 non-GP-based baseline methods. The results of these methods on the four datasets are listed in Tables 4 and 5. In Tables 4 and 5, the symbols "+", "=" and "–" (or '↑", "→" and "↓") indicate that NSGPFL (SPGPFL) achieves significantly better, similar and worse results than/to the compared algorithm, respectively. The final row of each block in Tables 4 and 5 summarises the overall results of the significance tests.

#### *5.2.1. Comparisons with CNNs*

It can be found from Table 4 that both NSGPFL and SPGPFL significantly outperform the two CNN methods on the four face recognition datasets. Importantly, the mean accuracy obtained by NSGPFL and SPGPFL is at least 10% higher than that by LeNet and CNN on the Faces95 dataset and at least 5% higher on the Extended Yale B dataset. The reason is that LeNet and CNN require a large number of training instances to obtain a

Table 4: Test accuracy (%) on the four datasets. The symbols “+”, “=” and “–” (“↑”, “→” and “↓” ) indicate that NSGPFL (SPGPFL) achieves significantly better, similar and worse results than/to the compared algorithm

| | **Max** | **Mean±St.dev** | **Max** | **Mean±St.dev** |
|---|---|---|---|---|
| **Dataset** | **ORL** | | **Extended Yale B** | |
| LeNet | 93.50 | 88.33±2.91+↑ | 93.33 | 88.42±2.63+↑ |
| CNN | 96.00 | 93.72±1.30+↑ | 92.55 | 89.98±1.24+↑ |
| NSGPFL | 99.50 | 97.48±1.66 | 98.32 | 95.32±2.83 |
| SPGPFL | **99.50** | 97.40±1.19 | 98.92 | 96.60±2.15 |
| **Overall** | **2 + / 2 ↑** | | **2 +/ 2 ↑** | |
| **Dataset** | **Aberdeen** | | **Faces95** | |
| LeNet | 94.05 | 90.14±2.01+↑ | 89.72 | 85.83±1.55+↑ |
| CNN | 96.28 | 94.67±0.97+↑ | 89.58 | 87.21±1.16+↑ |
| NSGPFL | 98.88 | 97.06±1.34 | 99.17 | 97.82±0.85 |
| SPGPFL | 99.63 | 97.30±1.08 | 99.17 | 97.62±0.78 |
| **Overall** | **2+ / 2 ↑** | | **2 +/ 2 ↑** | |

higher generalisation performance. The number of instances in per class of the four datasets is small so that only several instances can be used for training. The current training data may not be sufficient to train LeNet and CNN although they only have a few layers. Therefore, the performance of LeNet and CNN is limited to the small number of training instances. It is noted that using more training data can improve the performance of LeNet and CNN, but it is beyond the scope of this paper. The comparisons show that both NSGPFL and SPGPFL achieve significantly better performance than the two CNN methods on the four face recognition datasets.

### *5.2.2. Comparisons with Other 32 Baseline Methods*

From Table 5, it is noticeable the both NSGPFL and SPGPFL achieve significantly better or similar performance in all the comparisons. Specifically, NSGPFL achieves significantly better performance in 124 comparisons, similar performance in two comparisons, and significantly worse performance in two comparisons out of the total 128 comparisons. The NSGPFL algorithm achieves significantly better performance in 124 comparisons and similar performance in four comparisons out of the 128 comparisons. These results indicate that both NSGPFL and SPGPFL significantly outperform a large number of traditional face recognition algorithms.

From Table 5, it can be found that the performance of different features and different classification algorithms varies with the datasets. On the ORL

Table 5: Test accuracy (%) on the four datasets. The symbols "+", "=" and "–" ("↑", "→" and "↓" ) indicate that NSGPFL (SPGPFL) achieves significantly better, similar and worse results than/to the compared algorithm

| | **Mean±St.dev** | **Mean±St.dev** | **Mean±St.dev** | **Mean±St.dev** |
|---|---|---|---|---|
| | **KNN** | **SVM** | **SRC** | **RF** |
| **Dataset** | **ORL** | | | |
| Fisherfaces | 92.00±0.00+↑ | 92.47±0.13+↑ | 91.50±0.00+↑ | 95.85±0.49+↑ |
| Eigenfaces1 | 81.00±0.00+↑ | 96.00±0.32+↑ | 92.50±0.00+↑ | 91.58±0.87+↑ |
| Eigenfaces2 | 82.50±0.00+↑ | 94.50±0.00+↑ | 89.00±0.00+↑ | 90.57±0.84+↑ |
| SIFT | 87.50±0.00+↑ | 97.00±0.00+↑ | 98.00±0.00=→ | 97.78±0.49=→ |
| LBP | 67.50±0.00+↑ | 89.23±0.75+↑ | 66.50±0.00+↑ | 87.17±0.96+↑ |
| Original | 83.00±0.00+↑ | 95.50±0.00+↑ | 91.00±0.00+↑ | 94.13±1.00+↑ |
| Downsample4 | 82.50±0.00+↑ | 95.50±0.00+↑ | 90.50±0.00+↑ | 96.63±0.73+↑ |
| Downsample8 | 81.50±0.00+↑ | 94.50±0.00+↑ | 87.00±0.00+↑ | 96.53±0.59+↑ |
| **Overall** | **NSGPFL: 30 +, 2 =** | | **SPGPFL: 30 ↑, 2 →** | |
| **Dataset** | **Extended Yale B** | | | |
| Fisherfaces | 88.04±0.00+↑ | 77.93±2.08+↑ | 88.76±0.00+↑ | 82.68±0.28+↑ |
| Eigenfaces1 | 30.95±0.00+↑ | 76.02±0.54+↑ | 69.95±0.00+↑ | 76.47±0.43+↑ |
| Eigenfaces2 | 42.19±0.00+↑ | 86.87±0.37+↑ | 90.08±0.00+↑ | 84.38±0.32+↑ |
| SIFT | 34.38±0.00+↑ | 73.68±0.00+↑ | 71.75±0.00+↑ | 65.52±0.44+↑ |
| LBP | 18.15±0.00+↑ | 42.36±4.08+↑ | 20.01±0.00+↑ | 43.67±0.40+↑ |
| Original | 43.51±0.00+↑ | 90.75±0.03+↑ | 97.18±0.00–→ | 94.04±0.26+↑ |
| Downsample4 | 42.73±0.00+↑ | 90.32±0.00+↑ | 94.53±0.00+↑ | 91.66±0.23+↑ |
| Downsample8 | 40.81±0.00+↑ | 86.72±0.00+↑ | 87.02±0.00+↑ | 88.03±0.26+↑ |
| **Overall** | **NSGPFL: 31 +, 1–** | | **SPGPFL: 31 ↑, 1 →** | |
| **Dataset** | **Aberdeen** | | | |
| Fisherfaces | 91.08±0.00+↑ | 91.07±0.07+↑ | 87.36±0.00+↑ | 90.77±0.55+↑ |
| Eigenfaces1 | 65.43±0.00+↑ | 87.99±0.37+↑ | 85.13±0.00+↑ | 84.71±0.91+↑ |
| Eigenfaces2 | 69.52±0.00+↑ | 95.06±0.17+↑ | 94.80±0.00+↑ | 89.22±1.08+↑ |
| SIFT | 83.64±0.00+↑ | 91.07±0.00+↑ | 97.77±0.00–→ | 95.48±0.44+↑ |
| LBP | 39.78±0.00+↑ | 73.19±2.89+↑ | 62.83±0.00+↑ | 78.02±0.99+↑ |
| Original | 69.89±0.00+↑ | 94.81±0.07+↑ | 95.54±0.00+↑ | 90.66±0.80+↑ |
| Downsample4 | 66.91±0.00+↑ | 95.63±0.21+↑ | 94.80±0.00+↑ | 91.70±0.78+↑ |
| Downsample8 | 67.29±0.00+↑ | 92.95±0.07+↑ | 92.19±0.00+↑ | 89.64±0.80+↑ |
| **Overall** | **NSGPFL: 31 +, 1 –** | | **SPGPFL: 31 ↑, 1 →** | |
| **Dataset** | **Faces95** | | | |
| Fisherfaces | 85.69±0.00+↑ | 75.41±0.13+↑ | 76.53±0.00+↑ | 81.64±0.40+↑ |
| Eigenfaces1 | 73.33±0.00+↑ | 83.69±0.22+↑ | 73.06±0.00+↑ | 84.68±0.45+↑ |
| Eigenfaces2 | 71.11±0.00+↑ | 80.17±0.07+↑ | 79.17±0.00+↑ | 82.62±0.47+↑ |
| SIFT | 80.14±0.00+↑ | 94.31±0.00+↑ | 93.75±0.00+↑ | 92.57±0.28+↑ |
| LBP | 84.58±0.00+↑ | 94.24±1.27+↑ | 64.17±0.00+↑ | 94.61±0.38+↑ |
| Original | 70.83±0.00+↑ | 83.06±0.00+↑ | 74.58±0.00+↑ | 93.33±0.36+↑ |
| Downsample4 | 70.69±0.00+↑ | 83.47±0.00+↑ | 80.56±0.00+↑ | 93.60±0.37+↑ |
| Downsample8 | 68.89±0.00+↑ | 82.08±0.00+↑ | 78.06±0.00+↑ | 93.44±0.34+↑ |
| **Overall** | **NSGPFL: 32 +** | | **SPGPFL: 32 ↑** | |

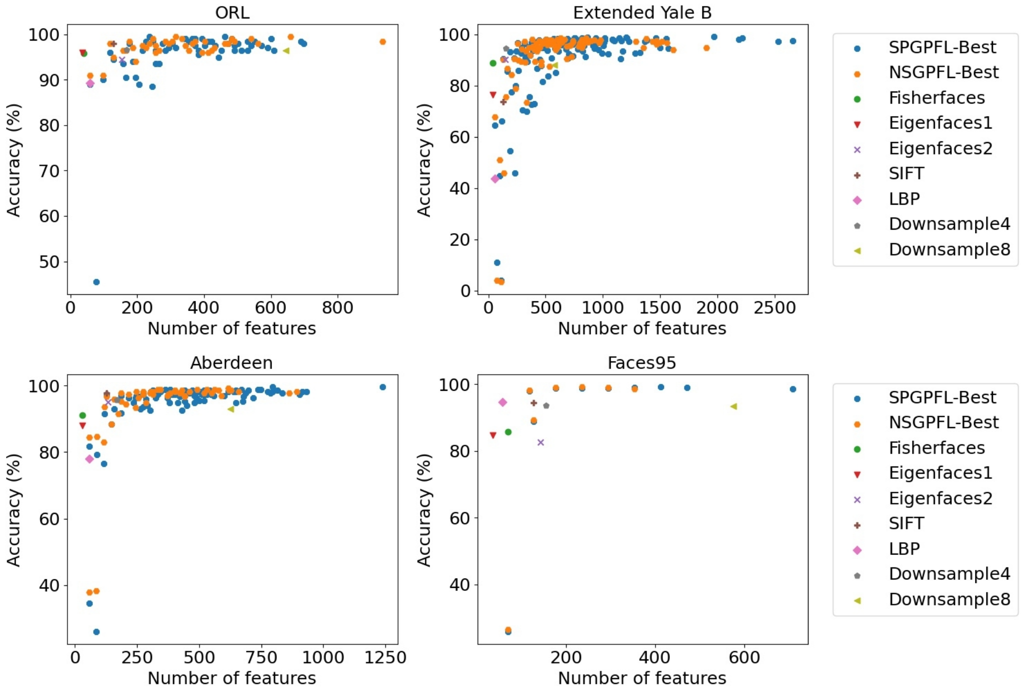


Figure 8: Comparisons between NSGPFL, SPGPFL and the baseline methods in terms of the classification accuracy and the number of learned features.

and Aberdeen datasets, SIFT+SRC (SRC using SIFT features) achieves better results than the other algorithms, while SIFT+SRC achieves worse results than SIFT+SVM on Extended Yale B and Faces95 and worse results than Original+SRC on Extended Yale B. These results indicate that it is typically difficult to manually extract features and choose a classification algorithm to build a face recognition system. Compared with these algorithms, NSGPFL and SPGPFL can automatically extract features to build a classification system for face recognition. The results show that NSGPFL and SPGPFL are more effective and adaptive than the algorithms using manually extracted features for various face recognition tasks.

Fig. 8 compares the classification performance and the number of learned features of NSGPFL, SPGPFL, and baseline methods. For the baseline methods, the best accuracy achieved by these features is presented in Fig. 8. For better comparisons, the Original features are not included because they are raw pixels with a high dimension e.g., 10,000 for a 100×100 image. From Fig. 8, it can be found that some solutions of NSGPFL and SPGPFL achieve better classification accuracy than the baseline methods when the number of features is similar. It can also be found that many solutions of NSGPFL and SPGPFL achieve better classification performance and use a larger number of features than these baseline methods. NSGPFL and SPGPFL are multi-objective algorithms and can find a set of solutions with the tradeoff between the classification performance and the number of learned features.

To sum up, the comparisons with NSGPFL, SPGPFL and the 34 baseline methods show that NSGPFL and SPGPFL achieve significantly better results in almost all comparisons on the four datasets. The results show that the features learned by NSGPFL and SPGPFL are more effective than many well-known manually extracted features and the features learned by LeNet and CNN for face recognition. Compared with the baseline methods using manually extracted features, some solutions of NSGPFL and SPGPFL achieve better performance when the number of features is similar. The results confirm that NSGPFL and SPGPFL are effective approaches to feature learning.

## 6. Conclusions

The goal of this paper was to develop multi-objective GP-based feature learning algorithms for face recognition by simultaneously optimising the objectives of the classification performance and the number of learned features. The goal has been successfully achieved by developing the NSGPFL and SPGPFL algorithms with new representation, a new function set and a new terminal set. With the new representation, the new GP algorithms can automatically select small regions from the input image, select descriptors to extract features from the regions and combine these features to form the final output features for classification. Based on the new representation, four GP-based algorithms were developed, which are single-objective GPFL with maximising the classification accuracy, single-objective GPFLW with jointly optimising the classification accuracy and the number of learned features, multi-objective NSGPFL and SPGPFL with simultaneously optimising the classification accuracy and the number of learned features.

The performances of these algorithms were examined on four face recognition datasets of varying difficulty and compared with 34 baseline methods. The results suggested that both NSGPFL and SPGPFL achieved comparable classification performance and learned a smaller number of features than GPFL. The results showed that NSGPFL and SPGPFL are more effective than GPFLWs with different weighting factors ($\alpha$=0.5 and $\alpha$=0.8) for optimising the two objectives. The comprehensive comparisons between NSGPFL and SPGPFL show that NSGPFL is more effective than SPGPFL for finding a set of non-dominated solutions for multi-objective feature learning in face recognition. The results show that NSGPFL can find a better approximated Pareto front but SPGPFL is able to find more boundary solutions with high diversity. Compared with SPGPFL, NSGPFL finds a set of solutions with higher classification performance of the training and test sets when the numbers of features are similar or the same. The results suggested that both NSGPFL and SPGPFL achieved significantly better or similar performance than 34 non-GP-based baseline methods, including CNN-based methods and the methods using well-known manually designed face features.

The experimental results of GPFLWs confirm the difficulty of the simple combination of the two objectives as the exact range of the number of the learned features is unknown. In the future, we will further improve the performance of GPFLW by developing an adaptive weighting strategy to automatically adjust the weight when optimising the aggregation objective function. In this study, we investigated to use the ideas of non-dominated sorting and strength Pareto in GP-based feature learning algorithms. There are many other evolutionary multi-objective algorithms, such as decomposition-based algorithms [60]. In the future, we will develop new multi-objective GP-based feature learning algorithms to better search for the Pareto front of non-dominated solutions.

## Declaration of Competing Interest

The authors declare no conflict of interest.

## Acknowledgements

This work was supported in part by the Marsden Fund of New Zealand Government under Contracts VUW1509, VUW1615, VUW1913 and VUW1914, the Science for Technological Innovation Challenge (SfTI) fund under grant E3603/2903, the University Research Fund at Victoria University of Wellington grant number 223805/3986, MBIE Data Science SSIF Fund under the contract RTVU1914, and National Natural Science Foundation of China (NSFC) under Grant 61876169.

## References

[1] A. K. Jain, S. Z. Li, Handbook of Face Recognition, Springer, 2011.

[2] W. Zhao, R. Chellappa, P. J. Phillips, A. Rosenfeld, Face recognition: A literature survey, ACM Computing Surveys (CSUR) 35 (4) (2003) 399–458.

[3] C. Ding, J. Choi, D. Tao, L. S. Davis, Multi-directional multi-level dual-cross patterns for robust face recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (3) (2015) 518–531.

[4] J. Lu, V. E. Liong, X. Zhou, J. Zhou, Learning compact binary face descriptor for face recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 37 (10) (2015) 2041–2056.

[5] D. G. Lowe, Distinctive image features from scale-invariant keypoints, Proceedings of International Journal of Computer Vision 60 (2) (2004) 91–110.

[6] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: Application to face recognition, IEEE Transactions on Pattern Analysis & Machine Intelligence (12) (2006) 2037–2041.

[7] L. Shen, L. Bai, A review on gabor wavelets for face recognition, Pattern Analysis and Applications 9 (2-3) (2006) 273–292.

[8] H. Al-Sahaf, Y. Bi, Q. Chen, A. Lensen, Y. Mei, Y. Sun, B. Tran, B. Xue, M. Zhang, A survey on evolutionary machine learning, Journal of the Royal Society of New Zealand 49 (2) (2019) 205–228.

[9] Y. Bi, B. Xue, M. Zhang, A survey on genetic programming to image analysis, Journal of Zhengzhou University (Engineering Science) 39 (06) (2018) 3–13.

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

[11] H. Al-Sahaf, M. Zhang, A. Al-Sahaf, M. Johnston, Keypoints detection and feature extraction: A dynamic genetic programming approach for evolving rotation-invariant texture image descriptors, IEEE Transactions on Evolutionary Computation 21 (6) (2017) 825 – 844.

[12] M. Turk, A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neuroscience 3 (1) (1991) 71–86.

[13] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, IEEE Transactions on Pattern Analysis & Machine Intelligence (7) (1997) 711–720.

[14] D. Yi, Z. Lei, S. Liao, S. Z. Li, Learning face representation from scratch, arXiv preprint arXiv:1411.7923 (2014).

[15] L. M. Antonio, C. A. C. Coello, Coevolutionary multiobjective evolutionary algorithms: Survey of the state-of-the-art, IEEE Transactions on Evolutionary Computation 22 (6) (2017) 851–865.

[16] Y. Bi, B. Xue, M. Zhang, Genetic programming with image-related operators and a flexible program structure for feature learning to image classification, IEEE Trans. Evol. Comput. 25 (1) (2021) 87–101.

[17] J. Avilés, J. Mayo-Maldonado, O. Micheloud, A multi-objective evolutionary approach for planning and optimal condition restoration of secondary distribution networks, Applied Soft Computing 90 (2020) 106182.

[18] R. Tanabe, H. Ishibuchi, An easy-to-use real-world multi-objective optimization problem suite, Applied Soft Computing (2020) 106078.

[19] G. Li, Q. Lin, W. Gao, Multifactorial optimization via explicit multipopulation evolutionary framework, Information Sciences 512 (2020) 1555–1570.

[20] J. Liang, Y. Liu, Y. Xue, Preference-driven pareto front exploitation for bloat control in genetic programming, Applied Soft Computing (2020) 106254.

[21] A. M. B. Dourado, E. C. Pedrino, Multi-objective cartesian genetic programming optimization of morphological filters in navigation systems for visually impaired people, Applied Soft Computing 89 (2020) 106130.

[22] L. Shao, L. Liu, X. Li, Feature learning for image classification via multiobjective genetic programming, IEEE Transactions on Neural Networks and Learning Systems 25 (7) (2014) 1359–1371.

[23] Y. Liang, M. Zhang, W. N. Browne, Figure-ground image segmentation using feature-based multi-objective genetic programming techniques, Neural Computing and Applications 31 (7) (2019) 3075–3094.

[24] Y. Bi, B. Xue, M. Zhang, Automatically extracting features for face classification using multi-objective genetic programming, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020, pp. 117–118.

[25] Y. Bi, B. Xue, M. Zhang, An evolutionary deep learning approach using genetic programming with convolution operators for image classification, in: Proceedings of IEEE Congress on Evolutionary Computation, IEEE, 2019, pp. 3197–3204.

[26] Y. Bi, B. Xue, M. Zhang, An effective feature learning approach using genetic programming with image descriptors for image classification [research frontier], IEEE Computational Intelligence Magazine 15 (2) (2020) 65–77.

[27] B. Xue, M. Zhang, W. N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Transactions on Cybernetics 43 (6) (2012) 1656–1671.

[28] B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation 20 (4) (2015) 606–626.

[29] D. J. Montana, Strongly typed genetic programming, Evolutionary Computation 3 (2) (1995) 199–230.

[30] M. Wang, W. Deng, Deep face recognition: A survey, arXiv preprint arXiv:1804.06655 (2018).

[31] B. Bozorgtabar, G. A. R. Rad, A genetic programming-pca hybrid face recognition algorithm, Journal of Signal and Information Processing 2 (03) (2011) 170.

[32] H. Ibrahem, M. Nasef, M. Emam, Genetic programming based face recognition, International Journal of Computer Applications 69 (27) (2013) 1–6.

[33] Y. Liang, M. Zhang, W. N. Browne, Genetic programming for evolving figure-ground segmentors from multiple features, Applied Soft Computing 51 (2017) 83–95.

[34] W.-J. Choi, T.-S. Choi, Genetic programming-based feature transform and classification for the automatic detection of pulmonary nodules on computed tomography images, Information Sciences 212 (2012) 57–78.

[35] M. Zhang, V. B. Ciesielski, P. Andreae, A domain-independent window approach to multiclass object detection using genetic programming, EURASIP Journal on Advances in Signal Processing 2003 (8) (2003) 841–859.

[36] I. H. Lee, M. T. Mahmood, Adaptive outlier elimination in image registration using genetic programming, Information Sciences 421 (2017) 204–217.

[37] B. Peng, S. Wan, Y. Bi, B. Xue, M. Zhang, Automatic feature extraction and construction using genetic programming for rotating machine fault diagnosis, IEEE Transactions on Cybernetics, DOI: 10.1109/TCYB.2020.3032945 (2020).

[38] D. Atkins, K. Neshatian, M. Zhang, A domain independent genetic programming approach to automatic feature extraction for image classification, in: Proceedings of IEEE Congress on Evolutionary Computation, 2011, pp. 238–245.

[39] H. Al-Sahaf, A. Song, K. Neshatian, M. Zhang, Two-tier genetic programming: Towards raw pixel-based image classification, Expert Systems with Applications 39 (16) (2012) 12291–12301.

[40] A. Lensen, H. Al-Sahaf, M. Zhang, B. Xue, Genetic programming for region detection, feature extraction, feature construction and classification in image data, in: Proceedings of European Conference on Genetic Programming, Springer, Heidelberg, 2016, pp. 51–67.

[41] Y. Bi, B. Xue, M. Zhang, An automatic feature extraction approach to image classification using genetic programming, in: Proceedings of International Conference on the Applications of Evolutionary Computation, Springer, 2018, pp. 421–438.

[42] Y. Bi, B. Xue, M. Zhang, Genetic programming with a new representation to automatically learn features and evolve ensembles for image classification, IEEE Transactions on Cybernetics (2020) 1–15, DOI: 10.1109/TCYB.2020.2964566.

[43] A. Vedaldi, B. Fulkerson, Vlfeat: An open and portable library of computer vision algorithms, in: Proceedings of the 18th ACM international conference on Multimedia, ACM, 2010, pp. 1469–1472.

[44] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii, in: Proceedings of International conference on parallel problem solving from nature, Springer, 2000, pp. 849–858.

[45] E. Zitzler, M. Laumanns, L. Thiele, Spea2: Improving the strength pareto evolutionary algorithm, TIK-report 103 (2001).

[46] F. S. Samaria, A. C. Harter, Parameterisation of a stochastic model for human face identification, in: Proceedings of the Second IEEE Workshop on Applications of Computer Vision, 1994, pp. 138–142.

[47] K.-C. Lee, J. Ho, D. J. Kriegman, Acquiring linear subspaces for face recognition under variable lighting, IEEE Transactions on Pattern Analysis and Machine Intelligence (5) (2005) 684–698.

[48] M. in Research Methods in Psychology of Faces, Psychological image collection at stirling (pics), http://pics.psych.stir.ac.uk/ (2012).

[49] L. Spacek, Face recognition data, University of Essex. UK. Computer Vision Science Research Projects, http://cswww.essex.ac.uk/mv/allfaces/index.html (2012).

[50] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[51] Y. Bi, B. Xue, M. Zhang, An automated ensemble learning framework using genetic programming for image classification, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2019, pp. 365–373.

[52] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2008) 210–227.

[53] Z.-H. Zhou, J. Feng, Deep forest, National Science Review 6 (1) (2018) 74–86.

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

[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, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[56] F. Chollet, et al., Keras, `https://keras.io` (2015).

[57] A. Auger, J. Bader, D. Brockhoff, E. Zitzler, Theory of the hypervolume indicator: optimal distributions and the choice of the reference point, in: Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms, 2009, pp. 87–102.

[58] J. D. Knowles, L. Thiele, E. Zitzler, A tutorial on the performance assessment of stochastic multiobjective optimizers, TIK-Report 214 (2006).

[59] B. H. Nguyen, B. Xue, P. Andreae, H. Ishibuchi, M. Zhang, Multiple reference points based decomposition for multi-objective feature selection in classification: Static and dynamic mechanisms, IEEE Transactions on Evolutionary Computation 24 (1) (2019) 170–184.

[60] L. Li, Q. Lin, S. Liu, D. Gong, C. A. C. Coello, Z. Ming, A novel multi-objective immune algorithm with a decomposition-based clonal selection, Applied Soft Computing 81 (2019) 105490.