Using Feature Clustering for GP-Based Feature Construction on High-Dimensional Data

Binh Tran[⋆], Bing Xue[⋆], and Mengjie Zhang

School of Engineering and Computer Science, Victoria University of Wellington,
PO Box 600, Wellington 6140, New Zealand
{binh.tran,bing.xue,mengjie.zhang}@ecs.vuw.ac.nz

Abstract. Feature construction is a pre-processing technique to create new features with better discriminating ability from the original features. Genetic programming (GP) has been shown to be a prominent technique for this task. However, applying GP to high-dimensional data is still challenging due to the large search space. Feature clustering groups similar features into clusters, which can be used for dimensionality reduction by choosing representative features from each cluster to form the feature subset. Feature clustering has been shown promising in feature selection; but has not been investigated in feature construction for classification. This paper presents the first work of utilising feature clustering in this area. We propose a cluster-based GP feature construction method called CGPFC which uses feature clustering to improve the performance of GP for feature construction on high-dimensional data. Results on eight high-dimensional datasets with varying difficulties show that the CGPFC constructed features perform better than the original full feature set and features constructed by the standard GP constructor based on the whole feature set.

Keywords: Genetic programming, feature construction, feature clustering, classification, high-dimensional data

1 Introduction

In machine learning, there has been an immense increase in high-dimensional data such as microarray gene expression, proteomics, images, text, and web mining data [1]. These datasets usually have thousands to tens of thousands of features. This enormity leads to the curse of dimensionality that tends to limit the scalability and learning performance of many machine learning algorithms, including classification methods. Furthermore, they usually contain a significant number of irrelevant and redundant features. The existence of these features not only enlarges the search space but also obscures the effect of relevant features on showing the hidden patterns of the data. As a result, they may significantly degrade the performance of many learning algorithms [2].

[⋆] Corresponding authors.

Feature construction (FC) has been used as an effective pre-processing technique to enhance the discriminating ability of the feature set by creating new high-level features from the original features [3, 4]. In order to create new features with better discriminating power, a FC method needs to select informative features and appropriate operators to combine the selected features. Therefore, the search space of the FC problem is very large, which requires a powerful search technique for a FC method.

With a flexible representation and a global search technique, Genetic programming (GP) [5] has been widely used in FC methods [6, 3, 4] in general and in high-dimensional data [7–9] as well.

GP has been proposed as a FC method with filter, wrapper, or embedded approaches [10]. While wrapper methods evaluate constructed features based on the performance of a learning algorithm, filter methods rely on the intrinsic characteristics of the training data. As a result, wrapper methods usually achieve better classification accuracy than filters. On the other hand, the later are usually faster than the former. Since GP can also be used as a classification algorithm, GP embedded FC methods have also been proposed where the constructed features are evaluated by applying GP itself as a classifier to the training data.

Although GP has been shown promising in generating better discriminating features than original features [4], its application to high-dimensional data is still challenging. In a recent study [8], Tran et al. proposed a GP-based feature construction method (or GPFC) that produced a combination of one constructed feature and a set of features selected to construct it. Experiment results demonstrated that GPFC is a promising approach on high-dimensional data. However, since these datasets may have a large number of redundant features, there may exist a high chance that GP selects redundant features to construct a new feature from the whole feature set. Therefore, the performance of GP in constructing new features may be degraded accordingly. Such a problem can be addressed if GP can avoid choosing redundant features when constructing a new feature.

Feature clustering groups similar *features* into one cluster, which is different from the common data mining task of clustering that groups similar *instances* into clusters [11]. Based on the resulting clusters, one or several features from each group can be chosen as representatives of each group. Using this approach, feature clustering has been proposed and shown promising performance in many feature selection methods [12–15]. However, applying feature clustering to FC is still limited.

Goals

This study proposes a cluster-based feature construction method called CGPFC for high-dimensional data. CGPFC uses feature clustering to improve the performance of GP for FC on high-dimensional data. Its performance will be compared with the standard GP for feature construction (GPFC) proposed in [8]. Specifically, we ask the following questions:

– How to automatically group features into clusters,

– Whether the proposed method can construct features with better discriminating ability (better classification accuracy) than the original full feature set and the one constructed by GPFC,
– Whether CGPFC can select a smaller number of features than GPFC to construct a better new feature, and
– Whether the CGPFC combination sets perform better than those created by GPFC.

2 Background

2.1 Genetic Programming for Feature Construction

When applying GP to FC, a constructed feature is usually represented as a *tree* in which terminal nodes are features selected from the original feature set and internal nodes are operators [3, 4]. Although GP has a built-in capability to select good features based on the guide of fitness function, its performance is still affected when applied to high-dimensional data due to the large search space [8]. Therefore, it is critical to narrow this search space for GP performance improvement.

Attempt to reduce the number of features in the GP terminal set has been proposed for different applications. In [16], GP is proposed as a feature selection method for high-dimensional data. The GP terminal set is a combination of 50 top ranked features selected by Information Gain and ReliefF methods. Results show that GP achieves better feature subsets than baseline methods. However, it is necessary to choose a good number of top ranked features. Furthermore, many redundant features may still exist when combining selected features from the two feature selection methods. Readers are referred to [17] for more examples. In [18], GP is used to build c non-dominated sets of classifiers for a c-class problem. In this method, GP uses a relevance measure combined with some threshold as the probability of choosing features to form the trees. During the evolutionary process, GP also eliminates features that do not appear in the population. Results show that the proposed method achieves better classification accuracy than the compared ones.

Results from the above studies have shown that helping GP select appropriate features is critical for enhancing its performance. However, this approach has not been investigated in GP for FC. Therefore, in this study, we propose to use clustering to eliminate redundant features in order to improve GP performance for FC.

2.2 Feature Clustering

Clustering or cluster analysis is one of the main tasks in exploratory data mining. It aims to group similar objects into the same group or cluster. Literature has proposed different clustering algorithms using different measures to evaluate the similarity between objects as well as different ways of grouping objects [19].

Clustering has been used for decades as an unsupervised task where instances are grouped into clusters based on the similarity between them [20]. In machine learning and data mining, the "clustering" terminology is usually meant *instance clustering* or instance grouping. Recently, clustering techniques have been proposed to group similar features, thus called *feature clustering*, to achieve feature selection [11]. Similar features are grouped into the same cluster. Then one or more representative features from each cluster were used to form final subsets. Clustering techniques and different strategies of using the resulting feature clusters have been proposed in feature selection methods [21–23].

Correlation coefficient (CC) is a popular measure to evaluate feature dependency or redundancy. To group redundant features, CC is used to replace the proximity measure in the k-means clustering algorithm [24]. The final feature subset is then formed by gathering the most relevant feature from each cluster. A feature is considered as the most relevant of the cluster if its CC value with the class label is the highest. Experiments on two datasets with hundreds of features show a better result than a compared method and worse than the other. However, when k-means is used, the performance of the proposed method depends on other methods in estimating the number of clusters. Feature clustering is used as an approach to dimensionality reduction not only in classification but also in regression. In [25], CC is used to calculate the input coherence I_c and output incoherence O_c of each feature f to each feature cluster. Feature f is put into the cluster with the highest I_c if its I_c and O_c values satisfy two predefined thresholds. Then k new features are constructed from k clusters based on a weight matrix. Results on four regression problems with 13 to 363 features showed that the proposed method has better performance than the compared methods.

While CC measures the level of correlation between features, mutual information indicates how much knowledge one can get for a variable by knowing the value of another variable. Symmetric uncertainty (SU) [26], which is a normalised version of information gain (IG) [27], is also used to identify irrelevant and redundant features. A feature is considered irrelevant if it does not give any information about the target concept or its SU with the class label is very small. Two features are redundant if their mutual information or their SU is high [28]. For example, in [23], SU is combined with minimum spanning tree (MST) to group features. Firstly, features are considered as irrelevant and removed if their SU with the class label is lower than a predefined threshold. Then a MST is built on the remaining features where SU between two features are used as the cost of each edge. The MST is then partitioned into disconnected trees, each of which contains features that have SU between them higher than their SU to the class. The most relevant feature in each tree is chosen to form the final subset. Results on 35 high-dimensional data including microarray, text and images showed that the proposed method achieve better performance on micro-array data than the state of the art feature selection methods.

Different from previous approaches, statistical clustering [29] takes into account interactions between features in evaluating feature redundancy. It is used for the purpose of dimensionality reduction in [21, 22]. Particle swarm optimisation is used to select features from each cluster to form the final subset. Performances on UCI datasets of the proposed methods show promising results. However, these statistical clustering algorithms are computationally expensive, thus not suitable for high-dimensional data.

In summary, by reducing the number of redundant features in feature set, feature clustering is a promising approach to feature selection. However, it has not been investigated in GP for FC especially on high-dimensional data. In this study, we propose to use feature clustering to automatically group features into clusters and the best feature of each cluster will be chosen for FC.

3 The Proposed Approach

3.1 The Redundancy Based Feature Clustering Method: RFC

K-means is a well-known clustering algorithm. It is a simple and effective method. However, it is essential to predefine a suitable number of clusters, which is not easy especially in high-dimensional data with thousands to tens of thousands of features. An inappropriate value may lead to clusters with uncorrelated or non-redundant features. In addition, the number of clusters in feature clustering is not as meaningful as the number of clusters in instance clustering, which represents the number of different types of objects/instances. Therefore, instead of grouping features based on a predefined number of clusters, in this study, we propose a new algorithm to group features based on the redundancy levels between them (called RFC). Different from the number of clusters, the redundancy or correlation level of two features is a value in the range of 0 and 1, representing no and full correlation between them, respectively. RFC uses a simple approach to ensure that all features in the same cluster are redundant features with their correlation level higher than a predefined threshold.

The main principle of RFC is to group features that have their redundancy level higher than a given threshold. If two features X and Y have their CC(X,Y) larger than this threshold, they will be grouped into the same cluster. In this way, the number of clusters will be automatically determined. If a dataset has a large number of redundant features, the number of clusters will be much smaller than the number of features, and vice versa. Furthermore, using this strategy enables us to ensure that the generated clusters include only features having their correlation levels higher than or equal to the predefined redundancy threshold.

Algorithm 1 shows the pseudo code of RFC for a given training set and a redundancy threshold θ. First of all, we analyse features to remove irrelevant ones (lines 4-9). In this study, a feature is considered irrelevant if it does not give any information about the class label. Since the class label is a discrete variable, SU is a suitable measure for feature relevancy. Therefore, in this step, all features whose SU with the class label are equal to zero will be removed.

SU between a feature X and the class C is calculated based on Eq.(1) which gives a value between 0 and 1 representing no to full correlation, respectively. As SU is an entropy-based measure, it can only be applied on category or nominal data. Therefore, we discretise data before calculating SU using MDL [30], which is a popular discretisation method.

Algorithm 1: The pseudo code of RFC

```
Input  : Training data, redundancy_threshold $\theta$
Output: Clusters of features
begin
    $F \leftarrow \varnothing$ ;
    $clusters \leftarrow \varnothing$ ;
    for each feature $f_i$ in Training data do
        $su \leftarrow SU(f_i, class)$ (based on Eq.(1)) ;
        if $su > 0$ then
            $F \leftarrow F \cup \{f_i\}$;
        end
    end
    while $(F \neq \varnothing)$ do
        $f_i \leftarrow$ next feature in $F$;
        $F \leftarrow F \setminus \{f_i\}$;
        $new\_cluster \leftarrow \{f_i\}$;
        while $(F \neq \varnothing)$ do
            $f_j \leftarrow$ next feature in $F$;
            $F \leftarrow F \setminus \{f_j\}$;
            $cc \leftarrow CC(f_i, f_j)$ (based on Eq.(3)) ;
            if $(cc > \theta)$ then
                $new\_cluster \leftarrow new\_cluster \cup \{f_j\}$;
            end
            $clusters \leftarrow clusters \cup new\_cluster$;
        end
    end
    Return $clusters$;
end
```

$$SU(X,C) = 2\left[\frac{IG(X|C)}{H(X)+H(C)}\right] \tag{1}$$

where

$$IG(X|C) = H(X) - H(X|C) \tag{2}$$

and $H(X)$ is the entropy of X and $H(X|C)$ is the conditional entropy of X given C.

All remaining features are then grouped into exclusive or non-overlapped clusters (lines 10-23). In this step, CC is used to measure redundancy level between features because it can be directly applied to numerical data. Although CC can only measure the linear relationship between variables, it has been shown effective in many feature selection methods [24, 25]. The CC measure gives a value between -1 and 1 whose absolute value represents the correlation level between two features. Given that n is the number of instances in the dataset, CC between feature X and Y is calculated based on Eq.(3) which gives a value in [0,1].

$$CC(X,Y) = \left| \frac{\sum_{i=1}^{n} X_i Y_i - n\bar{X}\bar{Y}}{\sqrt{\sum_{i=1}^{n} X_i^2 - n\bar{X}^2}\sqrt{\sum_{i=1}^{n} Y_i^2 - n\bar{Y}^2}} \right| \tag{3}$$

3.2 The Proposed Method: CGPFC

After grouping features into clusters, collection of the best features of each cluster is used to construct one new feature. This section introduces the proposed cluster-based feature construction method, called CGPFC. Note that while the feature clustering algorithm (RFC) uses a filter measure to group features, the feature construction algorithm follows the wrapper approach.

Representation. CGPFC aims at constructing a single feature using a tree-based representation. Each GP individual has one tree which represents a constructed feature. We follow [8] to create a combination of the constructed and selected features from the GP tree. Fig. 1 provides an example of creating this combination from the best GP tree.

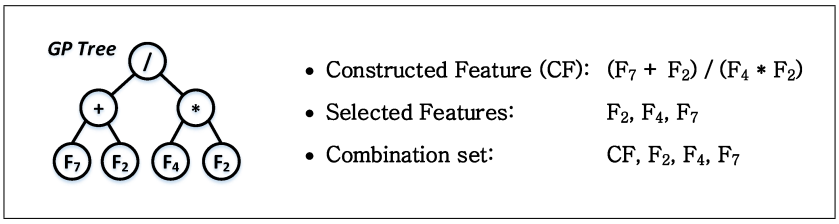
GP Tree
/
+
*
F_7 F_2 F_4 F_2
• Constructed Feature (CF): $(F_7 + F_2) / (F_4 * F_2)$
• Selected Features: F_2, F_4, F_7
• Combination set: CF, F_2, F_4, F_7

Fig. 1: Constructed Feature and Selected Features

Fitness Function. The cluster-based feature construction method developed in this work follows the wrapper approach. All classification algorithms can be used to evaluate the performance of the constructed feature. To evaluate a GP individual, the training set is transformed based on the feature constructed by the GP tree. Then the transformed training set classification performance is tested using L-fold cross validation (CV). The average accuracy is used as the fitness of the individual.

Since many of the high-dimensional datasets are unbalanced data, the balanced classification accuracy [31] is used in this fitness function. Given c as the number of classes, TP_i as the number of correctly identified instances in class i, and $|S_i|$ as the total number of instances in class i, the balanced accuracy is calculated based on Eq. (4). Since no bias is given to any specific class, the weight here is set equally to $1/c$.

$$fitness = \frac{1}{c} \sum_{i=1}^{c} \frac{TP_i}{|S_i|} \tag{4}$$

Overall Algorithm. Algorithm 2 describes the pseudo code of CGPFC. Given a training set and a redundancy threshold, the algorithm will return the combination of one constructed feature and the selected features constructing it.

First of all, the feature clustering procedure RFC is called to create a set of clusters from the training data. The most relevant feature (based on SU measure)

Algorithm 2: The pseudo code of CGPFC

```
   Input  : Training_data, redundancy_threshold $\theta$
   Output: Constructed feature and selected features
1  begin
2      $clusters \leftarrow RFC(Training\_data, \theta)$;
3      Initialise population using the best feature in each cluster of $clusters$;
4      while Maximum iterations or the best solution is not found do
5          for $i = 1$ to Population Size do
6              $transf\_train \leftarrow$ Calculate constructed feature of individual $i$ on Training data
                 ($transf\_train$ has only one new feature) ;
7              Evaluate $transf\_train$ using the learning algorithm with L-fold CV;
8              $fitness \leftarrow$ average test accuracy based on Eq.(4);
9          end
10         Select parent individuals using tournament;
11         Create offspring individuals by applying crossover or mutation on the selected
             parents;
12         Place new individuals into the population of the next generation;
13     end
14     Return the constructed feature and selected features in the best individual;
15 end
```

in each cluster is employed to initialise GP individuals. Lines 5-9 are used to evaluate GP individuals. The loop of evaluation-selection-evolution (lines 4-13) is executed until the stopping criteria are met.

4 Experiment Design

Datasets. Eight binary-class gene expression datasets with thousands of features are used to examine the performance of the proposed method on high-dimensional data. These datasets are publicly available at *http://www.gems-system.org*, and *http://csse.szu.edu.cn/staff/zhuzx/Datasets.html.*

Table 1: Datasets

Dataset	#Features	#Instances	Class Distribution
Colon	2,000	62	35% - 65%
DLBCL	5,469	77	25% - 75%
Leukemia	7,129	72	35% - 65%
CNS	7,129	60	35% - 65%
Prostate	10,509	102	50% - 50%
Ovarian	15,154	253	36% - 64%
Alizadeh	1,095	42	50% - 50%
Yeoh	2,526	248	83% - 17%

Table 1 describes these datasets in detail. It can be seen that these datasets have a small number of instances compared to the number of features. These datasets are challenging tasks in machine learning due to the curse of dimensionality issue. On top of this challenge, these are also unbalanced data with very different percentage of instances in each class as shown in the last column of the table.

Furthermore, gene expression data usually contains substantial noise generated during the data collection in laboratories. Therefore, discretisation is applied to reduce noise as suggested in [32]. Each feature is first standardised to have zero mean and unit variance. Then its values are discretised into -1, 0 and

1 representing three states which are the under-expression, the baseline and the over-expression of gene. Values that fall in the interval $[\mu - \sigma/2, \mu + \sigma/2]$, where μ and σ are mean and standard deviation of the feature values, are transformed to state 0. Values that are in the left or in the right of this interval will be transformed to state -1 or 1, respectively.

Parameter Settings. Table 2 shows the parameter settings for GP. The function set includes four basic arithmetic operators and three functions that used to construct a numeric feature from the selected features and random constants. Since the numbers of features in these datasets are quite different, ranging from about two thousand to fifteen thousand, the search spaces of these problems are very different. Therefore, we set the population size proportional to the terminal set size or the number of clusters (#clusters $\cdot\ \alpha$). α is set to 20 in this experiment. The mutation rate is set to 0.2, but after generation 10 it gradually increases with a step of 0.02 in every generation to avoid stagnation in local optima. The crossover rate is also updated accordingly to ensure the sum of these two rates always equal to 1. The redundancy threshold is empirically set to 0.9. The stopping criterion is either GP reaches the maximum generation or the best solution is found.

Table 2: GP Parameter Settings

Function set	$+$, $-$, $\times$, %, min, max, if
Terminal set	Features and random constant values
Population size	#clusters $\cdot\ \alpha$
Maximum generations	50
Initial population	Ramped Half-and Half
Initial maximum tree depth	2
Maximum tree depth	17
Selection method	Tournament Method
Tournament size	7
Crossover rate	0.8
Mutation rate	0.2
Elitism	True
Redundancy threshold	0.9

Experiment Configuration. To test the performance of CGPFC, we compared the discriminating ability of the constructed feature versus the original features and the one constructed by the standard GP [8] based on the classification accuracy of K-nearest neighbour (KNN), Naive Bayes (NB) and Decision Tree (DT). This comparison is also conducted on the combination feature set of the constructed and the selected features.

Due to the small number of instances in each dataset, two loops of CV are used to avoid feature construction bias as described in [8]. The outer loop uses a stratified 10-fold CV on the whole dataset. One fold is kept as the test set to evaluate the performance of each method, and the remaining 9 folds are used to form the training set for feature construction and classifier training. In the fitness function, an inner loop of 3-fold CV within the training set is run to evaluate the evolved feature (see Section 3.2). The learning algorithm used for fitness evaluation is Logistic Regression (LR) - a statistical learning method. We choose LR because it can help in scaling the constructed feature to determine the probabilities of each class through the use of a logistic function. Therefore, it can effectively test the ability of GP-constructed feature in discriminating classes.

Since GP is a stochastic algorithm, we run CGPFC on each dataset 30 times independently with different random seeds and report the average results to avoid statistical variations.

Therefore, a totally 300 runs (30 runs combined with 10-fold CV) are executed on each dataset. Experiments were runs on PC with Intel Core i7-4770 CPU @ 3.4GHz, running Ubuntu 4.6 and Java 1.7 with a total memory of 8GB. The results of 30 runs from each method were compared using Wilcoxon statistical significance test [33], with the significance level of 0.05.

5 Results and Discussions

5.1 Performance of The Constructed Feature

Table 3 shows the average test results of 30 independent runs of the proposed method (CGPFC) compared with "Full" (i.e. using the original feature set) and the GPFC [8]. The number of instances in each dataset is also displayed in parentheses under its name. Column "#F" shows the average size of each feature set. The following columns display the best (B), mean and standard deviation of the accuracy (M±Std) obtained by KNN, NB and DT on the corresponding feature set. The highest average accuracy of each learning algorithm for each dataset is bold. Columns S_1, S_2, and S_3 display the Wilcoxon significance test results of the corresponding method over CGPFC with significance level of 0.05. "+" or "–" means that the result is significantly better or worse than CGPFC and "=" means that their results are similar. In other words, the more "–", the better the proposed method.

CGPFC Versus Full. It can be seen from Table 3 that using only a single constructed feature by CGPFC, KNN obtains significantly better results than Full on 6 out of the 8 datasets. The highest improvement in the average accuracy is 8% on Ovarian and Yeoh, and 14% in the best result of Colon dataset. On Alizadeh, it achieves a similar performance on average and 10% higher than Full in the best case. Only on CNS, does CGPFC obtain slightly worse results than Full, however, still 7% better accuracy in the best case. With 7129 features and only 60 instances, this dataset can be considered as the most challenging dataset among the eight. With a small number of training instances, it is hard for GP to construct a feature that is generalised well to the unseen data.

Similar to KNN, the constructed feature by CGPFC also helps NB achieve 4% to 27% higher accuracy than Full on 5 datasets. Using only 1 constructed feature on Prostate, the best accuracy NB can achieve is 32% higher than Full. On Leukemia and CNS, although NB obtains 1% and 2% average result lower than Full, its best accuracy is still 6% and 5% higher, respectively. On Alizadeh, the accuracy of CGPFC constructed feature is significantly lower than Full. We also note that the accuracy of NB on the Full feature set of this dataset is much higher than KNN and DT.

Table 3: Test Accuracy of The Constructed Feature

Dataset	**Method**	**#F**	**B-KNN**	**M±Std-KNN**	S_1	**B-NB**	**M±Std-NB**	S_2	**B-DT**	**M±Std-DT**	S_3
Colon	Full	2000	74.29		–	72.62		–	74.29		–
(62)	GPFC	1	79.28	71.40 ±4.46	–	78.81	69.64 ±4.17	–	79.28	72.25 ±4.07	–
	CGPFC	1	88.81	**77.56 ±4.47**		88.81	**77.96 ±4.16**		88.81	**78.08 ±4.05**	
DLBCL	Full	5469	84.46		–	81.96		–	80.89		–
(77)	GPFC	1	96.07	86.65 ±3.76	–	92.32	86.27 ±4.28	–	94.64	86.51 ±4.08	–
	CGPFC	1	94.64	**88.62 ±2.92**		94.64	**88.74 ±2.90**		94.64	**88.62 ±2.92**	
Leukemia	Full	7129	88.57		–	91.96		+	91.61		=
(72)	GPFC	1	94.46	89.03 ±2.71	–	93.21	87.26 ±4.44	–	95.89	88.97 ±2.96	=
	CGPFC	1	95.89	**90.65 ±3.21**		97.32	90.73 ±3.16		95.89	90.65 ±3.21	
CNS	Full	7129	56.67		+	58.33		+	50.00		–
(60)	GPFC	1	70.00	**57.56 ±5.87**	=	70.00	**58.44 ±5.94**	=	70.00	**57.78 ±6.05**	=
	CGPFC	1	63.33	55.06 ±3.85		63.33	56.00 ±2.89		63.33	56.00 ±3.02	
Prostate	Full	10509	81.55		–	60.55		–	86.18		–
(102)	GPFC	1	90.18	83.72 ±3.18	–	90.18	83.18 ±3.68	–	90.18	83.82 ±2.85	–
	CGPFC	1	92.27	**87.40 ±3.62**		92.27	**87.31 ±3.51**		92.27	**87.40 ±3.62**	
Ovarian	Full	15154	91.28		–	90.05		–	98.42		–
(253)	GPFC	1	99.62	97.86 ±1.22	–	99.62	97.22 ±1.48	–	99.62	97.89 ±1.18	–
	CGPFC	1	100.00	**99.37 ±0.48**		100.00	**99.37 ±0.48**		100.00	**99.37 ±0.48**	
Alizadeh	Full	1095	77.00		=	92.50		+	78.50		=
(42)	GPFC	1	86.00	**77.88 ±5.53**	=	88.50	76.52 ±5.85	=	86.00	77.20 ±5.84	=
	CGPFC	1	87.50	77.12 ±5.85		87.50	76.88 ±6.06		87.50	77.20 ±6.11	
Yeoh	Full	2526	89.97		–	93.57		–	97.57		=
(248)	GPFC	1	99.17	97.04 ±1.01	=	97.57	95.11 ±2.72	–	99.17	97.05 ±0.99	–
	CGPFC	1	98.77	**97.32 ±1.59**		98.77	**97.38 ±0.84**		98.77	**97.71 ±0.67**	

DT also gets benefit from the constructed feature, shown as significantly improvement in its performance on 5 datasets and obtaining a similar result on the remaining datasets. The highest improvement is on DLBCL with 8% increase on average and 14% in the best case. Although the average accuracy is slightly worse on two datasets, namely Leukemia and Alizadeh, the best accuracy DT obtained for each dataset is always higher than Full.

In general, over 24 comparisons between CGPFC and Full on 8 datasets and 3 learning algorithms, the constructed feature by CGPFC wins 16, draws 4 and loses 4. The results indicate that the CGPFC constructed feature has much higher discriminating ability than the original feature set with thousands of features.

CGPFC Versus GPFC. Compared with GPFC, CGPFC helps KNN further improve its results to achieve the best results on 6 out of the 8 datasets. The result is significantly better than GPFC on 5 datasets and similar on the other three. Similarly, using the CGPFC constructed feature, NB obtains significantly better results than using GPFC constructed feature on 6 datasets with the highest improvement of 8% on Colon. Applying the CGPFC constructed feature on DT also gives similar results as KNN with significantly improvement on 5 datasets and equivalent on the remaining ones.

In summary, the CGPFC constructed feature wins 16, draws 8, loses 0 out of the 24 pairs of comparisons. The results show that by reducing the irrelevant and the redundant features in the GP terminal set, the constructed feature has a better discriminating ability than the one constructed from the full feature set.

5.2 Performance of The Constructed and Selected Features

In GP-based method, there is a built-in feature selection process which selects informative features from the original set to construct the new feature. Results in [8] has shown that the combination of these selected features and the constructed feature from the GP tree has better performance than other combinations of the constructed and original features. This finding is also supported in this study with an even better results.

The average size of this combination created by GPFC and CGPFC over the 30 runs is shown in Fig. 2. The average accuracy of the three learning algorithms, namely KNN, NB and DT, on this combination are shown in Fig. 3, 4, and 5, respectively. In these figures, each group of bars shows the results of GPFC and CGPFC on each dataset. On the CGPFC bars, results of the significance test comparing CGPFC against GPFC are displayed. "+" and "–" mean that CGPFC is significantly better or worse than GPFC. "=" means that they are similar.

First of all, let us examine the size of this combination of features. Since both methods construct only one feature, the difference between their sizes comes from the different numbers of distinct features they select. It can be seen from Fig. 2 that CGPFC always select a much smaller number of features than GPFC. On four datasets, namely DLBCL, Leukemia, Ovarian and Alizadeh, CGPFC selects less than half the number of features selected by GPFC. With a smaller number of selected features, if the CGPFC combination sets have better classification performance than those created by GPFC, it can be inferred that the selected features by CGPFC have better discriminating power than those selected by GPFC.

Results in Fig. 3 show that the CGPFC combination sets obtain a higher KNN accuracy than those created by GPFC on 7 out of the 8 datasets. This result is significantly better on 5 datasets and similar on the other three. Similar patterns are seen in Fig. 4 and 5 for NB and DT with significantly better results on 4 and 5 datasets, respectively. In general, CGPFC either improves or maintains the performance of GPFC on all the 8 datasets.

As can be seen in Fig. 2-5, the error bars of CGPFC are always smaller than the corresponding error bar of GPFC in all datasets. This indicates that CGPFC produces more robust results than GPFC.

Results from the combination of constructed and selected features on the 8 datasets show that CGPFC uses a smaller number of features to construct a new feature with better discriminating ability than GPFC. This indicates that by reducing the number of redundant features in the terminal set, feature clustering helps GP to improve its performance.

5.3 Cluster Analysis

To validate the structure of the clusters generated by our algorithm, in this section, we investigate the cohesion or compactness within each cluster as well as the separation or isolation between different clusters.

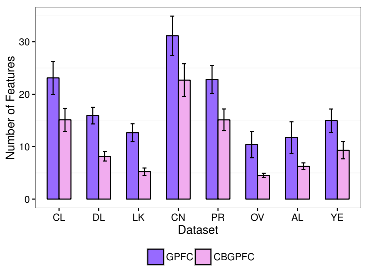

Fig. 2: Size of CFTer.

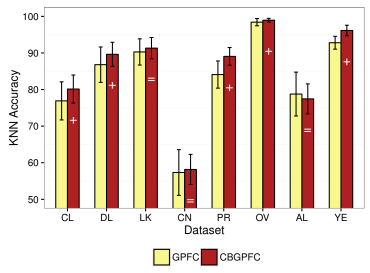

Fig. 3: KNN Accuracy of CFTer.

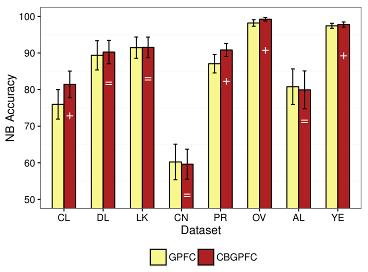

Fig. 4: NB Accuracy of CFTer.

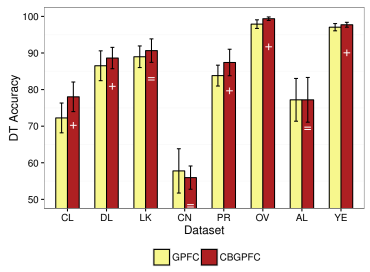

Fig. 5: DT Accuracy of CFTer.

Silhouette analysis [34] is a popular method to study both the cohesion and the separation of clusters. Eq. (5) displays the calculation of the silhouette coefficient of a feature i in which a_i is the average distance of feature i to all other features in its cluster, and b_i is the minimum average distance of feature i to other clusters. Given that c_i is the cluster that includes feature i, and c_k is other clusters, $CC(f_i, f_j)$ is the CC between features i and j, a_i and b_i are calculated based on Eq. (6) and (7). Since the CC (see (Eq. 3)) measures the correlation level or similarity between 2 features and has a value between 0 and 1, we use (1 - CC) as a distance or dissimilarity measure between them.

$$s_i = \frac{(b_i - a_i)}{max(a_i, b_i)} \tag{5}$$

where

$$a_i = \frac{1}{size(c_i)} \sum_{j=1}^{size(c_i)} (1 - CC(f_i, f_j)), i \neq j \tag{6}$$

$$b_i = \min_{\forall c_k \neq c_i} (\frac{1}{size(c_k)} \sum_{j=1}^{size(c_k)} (1 - CC(f_i, f_j))) \tag{7}$$

The value of the silhouette coefficient ranges from -1 to 1, where -1 is the worst and 1 is the best case. Average silhouette coefficient (ASC) of all features is an overall measure indicating the goodness of a clustering. Since the experiments were conducted based on a 10-fold CV framework on each dataset, we calculate the ASC for each fold and report the average of 10 ASCs. Table 4 shows the original number of features, the average number of clusters generated with redundancy level of 0.9, the percentage of dimensionality reduction, and the average of ASC of clustering on each dataset.

Table 4: Cluster Analysis

Dataset	**#Features**	**#Clusters**	**%Dimensionality reduction**	**Silhouette coefficient**
Colon	2000	104.10	0.95	0.80
DLBCL	5469	819.20	0.85	0.96
Leukemia	7129	901.30	0.87	0.98
CNS	7129	79.30	0.99	1.00
Prostate	10509	1634.80	0.84	0.85
Ovarian	15154	601.20	0.96	0.31
Alizadeh	1095	93.60	0.91	0.94
Yeoh	2526	97.60	0.96	1.00

As can be seen from the fourth column of Table 4, all datasets obtain at least 84% of dimensionality reduction after the proposed feature clustering algorithm is applied. The number of input features into GP is significantly reduced with the largest reduction of 99% on CNS and 96% on Ovarian and Yeoh. The third column of Table 4 also shows differences in the number of clusters generated on different datasets regardless of its original number of features. For example, CNS has much smaller number of clusters than Colon although its feature set size is more than three times larger than Colon. The silhouette coefficient of each dataset is quite good except for Ovarian, where features in different clusters are still correlated but with a smaller level than 0.9. Even though its silhouette coefficient is not good enough, the results of this dataset shown in Table 3 and Fig. 3-5 reveal that feature clustering method enables the constructed feature perform significantly better than the feature constructed from the whole feature set.

6 Conclusions and Future Work

This study is the first work that aims to apply feature clustering to GP for FC in classification in order to improve its performance on high-dimensional data. The goal has been achieved by proposing a new feature clustering algorithm to cluster redundant features in one group based on a correlation or redundancy level. Then the best feature from each cluster is fed into GP to construct a single new high-level feature. Performance of the constructed and/or selected features is tested on three different classification algorithms. Results on eight gene expression datasets have shown that feature clustering helps GP construct features with better discriminating ability than those generated from the whole feature set.

The clustering technique proposed in this study has an advantage of automatically determining the number of clusters. It guarantees that features in one cluster have their correlated level higher than the given redundancy level. Although determining a redundancy level is easier than the number of clusters as in the case of K-means clustering technique, the proposed method still has some limitations, such as features in different clusters may also correlated to each other with a lower level than the given threshold. Furthermore, the proposed feature clustering method is threshold sensitive. These limitations can be solved by integrating feature clustering into feature construction process so that the performance of GP could be used to automatically adjust the feature clusters.

References

1. Zhang, J., Wang, S., Chen, L., Gallinari, P.: Multiple Bayesian discriminant functions for high-dimensional massive data classification. Data Mining and Knowledge Discovery (2016) 1–37
2. Liu, H., Motoda, H.: Feature Extraction, Construction and Selection: A Data Mining Perspective. Kluwer Academic Publishers, Norwell, MA, USA (1998)
3. Krawiec, K.: Evolutionary feature selection and construction. In: Encyclopedia of Machine Learning. (2010) 353–357
4. Neshatian, K., Zhang, M., Andreae, P.: A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming. IEEE Transactions on Evolutionary Computation **16** (2012) 645–661
5. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992)
6. Hiroyasu, T., Shiraishi, T., Yoshida, T., Yamamoto, U.: A feature transformation method using multiobjective genetic programming for two-class classification. In: IEEE Congress on Evolutionary Computation (CEC). (2015) 2989–2995
7. Ahmed, S., Zhang, M., Peng, L., Xue, B.: Multiple feature construction for effective biomarker identification and classification using genetic programming. In: Proceedings of Genetic and evolutionary computation conference, ACM (2014) 249–256
8. Tran, B., Xue, B., Zhang, M.: Genetic programming for feature construction and selection in classification on high-dimensional data. Memetic Computing **8** (2015) 3–15
9. Tran, B., Xue, B., Zhang, M.: Multiple feature construction in high-dimensional data using genetic programming. In: IEEE Symposium Series on Computational Intelligence (SSCI). (2016)
10. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182
11. Butterworth, R., Piatetsky-Shapiro, G., Simovici, D.A.: On feature selection through clustering. In: ICDM. Volume 5. (2005) 581–584
12. Gupta, A., Gupta, A., Sharma, K.: Clustering based feature selection methods from fmri data for classification of cognitive states of the human brain. In: 3rd International Conference on Computing for Sustainable Global Development (INDIACom), IEEE (2016) 3581–3584
13. Jaskowiak, P.A., Campello, R.J.: A cluster based hybrid feature selection approach. In: Brazilian Conference on Intelligent Systems (BRACIS), IEEE (2015) 43–48
14. Krier, C., François, D., Rossi, F., Verleysen, M., Chesnay Cedex, F.: Feature clustering and mutual information for the selection of variables in spectral data. In: European Symposium on Artificial Neural Networks (ESANN). (2007) 157–162
15. Rostami, M., Moradi, P.: A clustering based genetic algorithm for feature selection. In: Conference on Information and Knowledge Technology. (2014) 112–116
16. Ahmed, S., Zhang, M., Peng, L.: Feature Selection and Classification of High Dimensional Mass Spectrometry Data: A Genetic Programming Approach. In:

Proceedings of Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Springer Berlin Heidelberg (2013) 43–55
17. Xue, B., Zhang, M., Browne, W.N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Transactions on Evolutionary Computation **20** (2016) 606–626
18. Nag, K., Pal, N.: A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. IEEE Transactions on Cybernetics **46** (2016) 499–510
19. Xu, D., Tian, Y.: A comprehensive survey of clustering algorithms. Annals of Data Science **2** (2015) 165–193
20. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Transactions on neural networks **16** (2005) 645–678
21. Lane, M., Xue, B., Liu, I., Zhang, M.: Gaussian based particle swarm optimisation and statistical clustering for feature selection. In: Evolutionary Computation in Combinatorial Optimisation. Volume 8600 of Lecture Notes in Computer Science. (2014) 133–144
22. Nguyen, H.B., Xue, B., Liu, I., Zhang, M.: PSO and Statistical Clustering for Feature Selection: A New Representation. In: Simulated Evolution and Learning. Springer (2014) 569–581
23. Song, Q., Ni, J., Wang, G.: A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE Transactions on Knowledge and Data Engineering **25** (2013) 1–14
24. Hsu, H.H., Hsieh, C.W.: Feature selection via correlation coefficient clustering. Journal of Software **5** (2010) 1371–1377
25. Xu, R.F., Lee, S.J.: Dimensionality reduction by feature clustering for regression problems. Information Sciences **299** (2015) 42–57
26. Press, W.H., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical recipes in c. Cambridge University Press **1** (1988) 3
27. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, Inc. (1993)
28. Liu, H., Motoda, H.: Computational methods of feature selection. CRC Press (2007)
29. Pledger, S., Arnold, R.: Multivariate methods using mixtures: Correspondence analysis, scaling and pattern-detection. Computational Statistics & Data Analysis **71** (2014) 241–261
30. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Thirteenth International Joint Conference on Articial Intelligence. Volume 2., Morgan Kaufmann Publishers (1993) 1022–1027
31. Patterson, G., Zhang, M.: Fitness functions in genetic programming for classification with unbalanced data. In: Advances in Artificial Intelligence. Springer (2007) 769–775
32. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. Journal of bioinformatics and computational biology **3** (2005) 185–205
33. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics bulletin **1** (1945) 80–83
34. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics **20** (1987) 53–65