Class Dependent Multiple Feature Construction Using Genetic Programming for 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. Genetic Programming (GP) has shown promise in feature construction where high-level features are formed by combining original features using predefined functions or operators. Multiple feature construction methods have been proposed for high-dimensional data with thousands of features. Results of these methods show that several constructed features can maintain or even improve the discriminating ability of the original feature set. However, some particular features may have better ability than other features to distinguish instances of one class from other classes. Therefore, it may be more difficult to construct a better discriminating feature when combing features that are relevant to different classes. In this study, we propose a new GP-based feature construction method called CDFC that constructs multiple features, each of which focuses on distinguishing one class from other classes. We propose a new representation for class-dependent feature construction and a new fitness function to better evaluate the constructed feature set. Results on eight datasets with varying difficulties showed that the features constructed by CDFC can improve the discriminating ability of thousands of original features in most cases. Results also showed that CFDC is more effective and efficient than the hybrid MGPFC method which was shown to have better performance than standard GP to feature construction.

Keywords: Genetic programming · Class-dependent · Feature construction · Feature selection · Classification · High-dimensional data

1 Introduction

More and more high-dimensional data appears in machine learning especially in classification thanks to the advances in data collection technologies [15]. With thousands of features, these datasets bring challenges to learning algorithms not only because of the curse of dimensionality but also the existence of many irrelevant and redundant features. Therefore, dimensionality reduction is an essential step in preprocessing these datasets. By eliminating irrelevant and redundant features, feature selection is a popular technique for this purpose. In addition to

© Springer International Publishing AG 2017
W. Peng et al. (Eds.): AI 2017, LNAI 10400, pp. 182–194, 2017.
DOI: 10.1007/978-3-319-63004-5_15

feature selection, feature construction is also promising in reducing the number of features by creating a smaller number of high-level features from the original ones while maintaining or even improving the discriminating ability of the data.

Feature construction methods can be categorised into wrapper, filter, or embedded based on how a learning algorithm is involved in the feature construction process [6]. Wrapper methods use a learning algorithm to evaluate the goodness of the constructed features, while a measure based on data characteristics is used in filter methods. Therefore, filters are usually faster than wrappers while wrappers usually acquire higher classification accuracy than filters. In order to synthesise strengths of both approaches, combination of filter and wrapper (or hybrid) has been proposed in [11] as well. Different from wrappers and filters, embedded methods construct features during the process of learning a model.

Compared to feature selection, feature construction is more challenging due to its large search space. Feature construction has to choose informative features from 2^N possible combinations of N features and appropriate operators to combine them. Because of the high complexity, feature construction methods need a global and powerful search technique to generate better high-level features.

Genetic programming (GP) [7] is an evolutionary computation technique which can automatically evolve good solutions or models from a population of potential ones. Using a predefined fitness function, individuals are evaluated and selected to evolve new solutions using crossover or mutation. The process of evaluation-selection-evolution will be continued until a stopping criterion is met. Then GP returns the fittest individual found so far as the best solution. GP has shown promise for feature construction in general [10,12] as well as in high-dimensional data [1,12]. The most popular representation of constructed features in GP is tree with internal nodes being operators (e.g. $+, -$, etc.) and terminal nodes being original features.

GP has been proposed for multiple feature construction for high-dimensional data using either single-tree [1] or multiple-tree [11] representation. Results of these methods have shown that GP is promising in feature construction. However, some particular features may have better ability than other features to distinguish instances of one class from other classes [13]. For example, a feature may be good at distinguishing samples of class A from those of class B, C and D, but may not be good at differentiating samples of class B from those of C and D. Therefore, it may be more difficult to construct a better discriminating feature when combing features that are relevant to different classes. In this study, we propose a new multiple feature construction method that takes into account class-dependency in selecting features for feature construction.

Goals

This study proposes a class-dependent feature construction (CDFC) method using GP for high-dimensional data. CDFC constructs a small number of new high-level features, each of which focuses on distinguishing one class from other classes. The small set of constructed features is expected to improve the classification performance of common learning algorithms including k-Nearest Neighbour

(KNN), Naive Bayes (NB) and Decision Tree (DT). Performance of the CDFC constructed features will be tested and compared with the original feature set and those constructed by the hybrid MGPFC method [11]. Since [11] has shown a better performance achieved by MGPFC over standard GP, we do not compare CDFC with standard GP to save space in this paper. Specifically, we will investigate the following research objectives:

- How to construct class-dependent features;
- Whether the class-dependent constructed features achieve better classification accuracy than the original full feature set and those constructed by MGPFC;
- Whether the features selected by CDFC have better discriminating features than those selected by MGPFC; and
- Whether CDFC is faster than MGPFC.

2 Related Work

GP has been used to solve different tasks such as classification [6], feature selection [9], and feature construction [10]. Here we only review the prior work related to GP for feature construction. Readers are referred to [6,14] for broader and more comprehensive reviews.

GP has been proposed to construct multiple features using different strategies. In [8], each individual comprised of $2m$ trees representing m new features and m hidden features where m is the desired number of constructed features. Hidden features are good features which are kept out of the evolutionary process to avoid loosing them. The results showed that constructed features improved the DT performance on five out of six datasets. Cooperative coevolutionary GP was also proposed to construct m new features in [3] using m populations. Another approach is to run a single-tree GP method multiple times [10] to construct multiple features, each for one class. Experiments on datasets with small numbers of features showed promising results. However, this approach required to run GP as many times as the number of constructed features, which may be inefficient especially for high-dimensional data.

For high-dimensional data, Ahmed et al. [1] proposed to use a single-tree GP representation to construct multiple features from all possible subtrees of the best individual. Two fitness functions were proposed to create two feature construction methods; one uses classification accuracy of random forest (RF), and the other uses entropy gain of RF and the p-value of an ANOVA test on the selected features. Results showed that the latter had better generalisation ability with a smaller number of features than the former. However, the features constructed in this way may be redundant since one subtree can be a multiplication of a constant with its subtree, generating two correlated features from these subtrees. This may explain why their performance is inferior to the combination of the constructed feature from the root and the selected features in terminal nodes as demonstrated in [12]. Multiple-tree GP was proposed in [11] to construct as many features as required based on a user-defined ratio. Using a hybrid approach of filter and wrapper in evaluating the constructed features, the proposed method obtained a better results than [12].

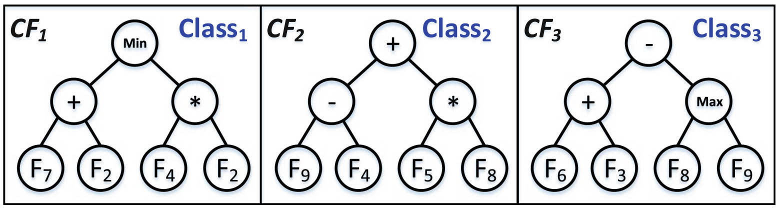
In summary, with a very flexible representation and a global search technique, GP has shown its high potential in feature construction. However, constructing features from class-dependent features which could potentially further improve the performance has not been investigated.

3 The Proposed Method CDFC

3.1 Representation

The aim of this study is to construct multiple features, each of which focuses on discriminating one class from the others. Furthermore, constructing only one feature per class may not be enough to represent complex problems. Therefore, CDFC allows to construct multiple features for one class depending on a user given ratio r which can be 1, 2, 3, etc. The number of constructed features m is equal to r multiplied by the number of classes. For example, with a given ratio $r = 2$, CDFC will construct $m = 4$ features for a binary-class problem, and $m = 6$ features for a problem with 3 classes. Each GP individual has m trees, each of which corresponds to one constructed feature of a class. Figure 1 shows an example of a GP individual with $r = 1$ for a three-class problem.

GP Individual:



⇨ **1. Constructed feature set:** **CF_1, CF_2, CF_3**

- CF_1 = Min (($F_7 + F_2$), ($F_4 * F_2$))
- CF_2 = ($F_9 - F_4$) + ($F_5 * F_8$)
- CF_3 = ($F_6 + F_3$) - Max (F_8 , F_9))

⇨ **2. Selected feature set:** **F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9**

Fig. 1. Representation of a GP individual and the generated feature sets.

3.2 Class-Dependent Terminal Sets

Since one constructed feature aims at discriminating instances of a class (c) from instances of other classes, it should be constructed based on features that are relevant to class c. In other words, constructed features of different classes should select different sets of features. Therefore, different trees in an individual will have different terminal sets, each of which comprises of features that are relevant to the focused class only. Since one feature may be important to different classes, there may be overlap between different terminal sets.

A feature f is relevant to class c if its values appeared in class c are significantly different from its values in other classes. In CDFC, t-Test is used to measure how relevant a feature f is to class c. First of all, values of f will be divided into two groups, one comprises values belonging to class c and one from other classes. Then, the relevant measure $Rel_{f,c}$ is calculated based on Eq. (1). $Rel_{f,c}$ is set to 0 if two groups are not significantly different (i.e. p-value ≥ 0.05). Otherwise, it is equal to the absolute of t-value divided by p-value. Therefore, the larger the value of $Rel_{f,c}$, the more relevant the feature f to class c.

$$Rel_{f,c} = \begin{cases} 0, & \text{if p-value} \geq 0.05 \\ \frac{|\textbf{t-value}(f_{class=c}, f_{class \neq c})|}{\textbf{p-value}}, & \text{otherwise} \end{cases} \tag{1}$$

For each class c, features are ranked by its $Rel_{f,c}$ values. Then half of the top-ranked features will be used to form the terminal set of class c. By doing so, we not only eliminate irrelevant features but also narrow the search space so that the searching process will be more efficient.

3.3 A New Fitness Function

To evaluate an individual, its m constructed features are used to transform the training set into a new training set with m features. The discriminating ability of the transformed dataset indicates how good the constructed features are. While a wrapper measure based on a learning algorithm can improve its classification accuracy, the constructed features may not be general for other learning algorithms. Furthermore, the computational cost of wrappers is usually high. Therefore, in this study, we propose a new fitness function using two filter measures that are simple and fast to calculate, and can provide a good indicator for data discriminating ability. Equation (2) shows the fitness function that maximises two measures combined using a weight α. Individual size is used as a pressure for small-tree preference when two individuals have the same measure values.

$$Fitness = \alpha \cdot AvgIG + (1 - \alpha) \cdot Distance - 10^{-7} \cdot indSize \tag{2}$$

The first measure is the average information gain (IG) of the constructed features which is calculated based on Eq. (3) where f_{max} is the best feature with the highest IG among m constructed features. The f_{max}'s IG is added to bias toward those candidates that have f_{max} with higher IG. IG of feature f is calculated based on unconditional and conditional entropy H as in Eq. (4). Finally, $AvgIG$ is divided by the best IG to scale it to the range of [0,1].

$$AvgIG = \frac{\sum_{i=1}^{m} IG(f_i, class) + IG(f_{max}, class)}{(m+1)(\log_2 nbr\_classes)} \tag{3}$$

$$IG(f, class) = H(class) - H(class|f) \tag{4}$$

Although IG is a good measure for feature relevancy, it can only evaluate features individually. Therefore, it can not show how good the whole set of constructed features in discriminating instances of different classes. Therefore, we combine IG with distance measure to overcome this limitation. In CDFC, we use the *Distance* measure [2] as shown in Eq. (5) to maximise the distance of instances *between* class (D_b) and minimise the distance of instances *within* the same class (D_w). Therefore, the larger the *Distance* value, the better the feature set. As a result, the proposed fitness function is a maximisation function where α defines the weight of the IG measure.

D_b is approximated by the average distance between an instance and its nearest miss which is the nearest instance of other classes. D_w is approximated by the average distance between an instance and its farthest hit which is of the same class. Let S be the training set, they are computed based on Eqs. (6 and 7).

$$Distance = \frac{1}{1 + e^{-5(D_b - D_w)}} \tag{5}$$

$$D_b = \frac{1}{|S|} \sum_{i=1}^{|S|} \min_{\{j|j \neq i, class(V_i) \neq class(V_j)\}} Dis(V_i, V_j) \tag{6}$$

$$D_w = \frac{1}{|S|} \sum_{i=1}^{|S|} \max_{\{j|j \neq i, class(V_i) = class(V_j)\}} Dis(V_i, V_j) \tag{7}$$

where $Dis(V_i, V_j)$ is an approximate distance between two vectors V_i and V_j. In this study, we use Czekanowski distance [4] which is calculated based on the shared portion between two vectors as shown in Eq. (8). Its value is bounded in the interval [0,1] where 1 is the best case showing two dissimilar vectors.

$$Czekanowski(V_i, V_j) = 1 - \frac{2 \sum_{d=1}^{m} \min(V_{id}, V_{jd})}{\sum_{d=1}^{m} (V_{id} + V_{jd})} \tag{8}$$

3.4 The Overall System

Figure 2 shows the overall system of CDFC. Based on the training set, CDFC forms different terminal sets for different classes using the relevance measure in Eq. (1). These terminal sets are then input to GP for class-dependent feature construction. The best individual is used to generate the constructed and selected feature sets as described in Fig. 1. The training and test sets will be transformed based on these feature sets and used to evaluate the performance of CDFC.

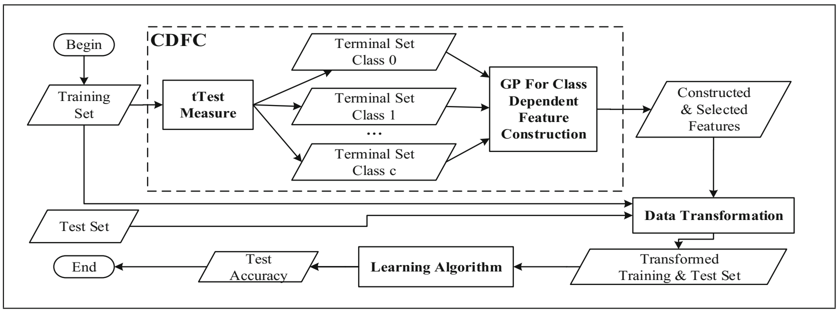


Fig. 2. CDFC overall system.

4 Experiment Design

Datasets. To test the performance of CDFC, eight gene datasets[1] with thousands to tens of thousands of features are used in the experiments. As shown in Table 1, the number of instances of these datasets are much smaller than the numbers of features, which makes these problems challenging due to the curse of dimensionality. On top of that, they are more challenging due to the unbalanced class distribution as shown in the last column of the table.

Table 1. Datasets

Dataset	#Features	#Ins.	#Class	Class 1	Class 2	Class 3	Class 4
Colon	2,000	62	2	35%	65%	-	-
DLBCL	5,469	77	2	25%	75%	-	-
Leukemia	7,129	72	2	35%	65%	-	-
CNS	7,129	60	2	35%	65%	-	-
Prostate	10,509	102	2	50%	50%	-	-
Ovarian	15,154	253	2	36%	64%	-	-
Leukemia1	5,327	72	3	13%	35%	53%	-
SRBCT	2,308	83	4	13%	22%	30%	34%

Another challenge when working with gene expression data is the existence of noise generated during microarray experiments involving reverse transcription, hybridisation of the samples, etc. Therefore, in this study, data is pre-processed before entering the feature construction process using a simple discretisation method suggested in [5] and used in [11,12]. Each feature is first standardised to have zero mean and unit variance. Then its values are discretised into -1, 0

[1] These datasets are publicly available at http://www.gems-system.org, and http://csse.szu.edu.cn/staff/zhuzx/Datasets.html.

and 1 representing three states which are the under-expression, the baseline and the over-expression of gene. Values that are in $[-0.5, 0.5]$ will belong to state 0. Values that are smaller than -0.5 or larger than 0.5 will belong to state -1 or 1, respectively.

Experiment Configuration and Parameter Settings. Due to the small number of instances in each dataset, 10-fold cross validation (10F-CV) is used to generate training and test set for evaluating CDFC performance. As GP is a stochastic algorithm, 30 independent runs of CDFC with 30 different random seeds are applied on each training set. Average of 300 results (30×10) are used in comparisons to eliminate statistical variations. Performance of the constructed and selected feature sets generated by CDFC is compared with the original feature set and those created by MGPFC using the average test accuracy of KNN, NB and DT.

Table 2. GP and parameter settings

Function set	$+, -, \times, max, if$	Generations	50
Population size	#features x β	Crossover rate	0.8
Initial population	Ramped half-and half	Mutation rate	0.2
Maximum tree depth	8	Elitism size	1
Selection method	Tournament method	Tournament size	7
CF_Ratio r	2	Fitness weighting α	0.8

Table 2 describes the parameter settings of GP. CDFC uses the same settings as MGPFC. The function set comprises of 3 arithmetic operators ($+, -, \times$), max function which returns the maximum values from the two inputs and if function which returns the second argument if the first argument is positive and returns the third argument otherwise. No constant values are used in terminal set for simplicity. The population size is set proportional to the dimensionality of the problem using a coefficient β which is set to 3 for Colon dataset and to 2 for others due to memory limit.

5 Results and Analysis

5.1 Performance of the Constructed Features

Table 3 shows the average test accuracy of KNN, NB and DT using constructed features obtained from 30 independent runs of CDFC compared with "Full" (i.e. using the original feature set) and MGPFC [11]. For each learning algorithm, the best (B), the mean and standard deviation (M $\pm$ Std) results are displayed. The best result among the three compared methods on each dataset is highlighted. In

Table 3. Results of the constructed features

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 (62)	Full	2000	74.28		–	72.62		–	74.29		–
	MGPFC	4	85.47	72.48 ± 5.95	–	85.48	71.10 ± 6.32	–	85.48	71.42 ± 5.82	–
	CDFC	4	88.81	**82.09 ± 3.16**		90.47	**83.89 ± 2.75**		87.38	**77.68 ± 4.35**	
DLBCL (77)	Full	5469	84.46		–	81.96		–	80.89		–
	MGPFC	4	97.32	89.50 ± 3.23	–	97.32	89.01 ± 3.55	–	94.82	87.47 ± 4.34	–
	CDFC	4	98.75	**95.76 ± 2.10**		98.75	**95.28 ± 2.24**		95.00	**90.31 ± 3.27**	
Leuk (72)	Full	7129	88.57		–	91.96		–	91.61		+
	MGPFC	4	95.89	92.71 ± 1.89	–	95.89	92.45 ± 1.88	–	95.89	90.99 ± 2.76	=
	CDFC	4	98.57	**94.80 ± 1.73**		97.32	**93.92 ± 1.49**		94.46	90.19 ± 2.46	
CNS (60)	Full	7129	56.67		–	58.33		–	50.00		–
	MGPFC	4	71.67	58.00 ± 8.27	–	71.67	58.67 ± 7.88	–	78.33	58.00 ± 8.93	=
	CDFC	4	71.67	**65.39 ± 4.03**		71.67	**65.94 ± 3.44**		68.34	**61.00 ± 5.39**	
Prostate (102)	Full	10509	81.55		–	60.55		–	86.18		–
	MGPFC	4	92.18	86.44 ± 3.08	–	92.18	85.89 ± 2.73	–	91.27	85.29 ± 3.04	–
	CDFC	4	95.18	**92.43 ± 1.75**		96.09	**92.41 ± 1.52**		94.09	**88.30 ± 2.52**	
Ovarian (253)	Full	15154	91.28		–	90.05		–	98.41		–
	MGPFC	4	100.00	99.23 ± 0.40	–	100.00	99.23 ± 0.52	–	99.60	98.82 ± 0.50	=
	CDFC	4	100.00	**99.79 ± 0.31**		100.00	**99.59 ± 0.32**		99.60	**98.86 ± 0.56**	
Leuk1 (72)	Full	5327	88.57		–	88.75		–	94.46		+
	MGPFC	6	95.89	89.53 ± 3.25	–	95.89	91.55 ± 2.70	=	95.89	91.53 ± 2.07	+
	CDFC	6	95.89	**92.87 ± 1.86**		95.89	**92.49 ± 2.03**		94.46	88.79 ± 3.41	
SRBCT (83)	Full	2308	80.83		–	97.50		+	72.36		–
	MGPFC	8	95.14	87.80 ± 3.74	–	94.03	87.69 ± 3.87	–	91.39	83.10 ± 4.45	–
	CDFC	8	100.00	**95.91 ± 1.74**		100.00	94.87 ± 2.51		93.89	**87.84 ± 3.01**	

addition, the Wilcoxon significance test is applied on the results with 5% significance level. Its results for KNN, NB and DT are displayed in column S_1, S_2, and S_3, respectively. "+" or "–" indicates that the corresponding method is significantly better or worse than the proposed method CDFC. "=" means they have similar performance. In other worse, the more "–", the better the proposed method.

CDFC Versus Full. All the "–"s appeared in column S_1 of Table 3 show that the constructed features help KNN achieve significantly higher accuracy than using full feature sets on all the eight datasets. The highest improvement is on SRBCT dataset with 15% on average and 20% in the best case, reaching 100% accuracy. The modest improvement is still 4% on average and 10% in the best case on Leukemia. The results show that the discriminating ability of the constructed features is much higher than the original all features although the number of constructed features is negligible to the original dimensionality.

For NB, the features constructed by CDFC also obtain better performance than Full on almost all datasets. For example, using the 4 constructed features on Prostate dataset, NB achieves 32% higher accuracy than using the whole 10,509 features. Similarly on Colon and CNS, the improvement is 11% and 14% on average with 18% and 17% in the best case, respectively. Only on SRBCT, CDFC has about 2.6% lower accuracy than Full. However, the best accuracy achieved by CDFC is 100% which is 2.5% higher than the Full accuracy.

Compared to using Full, DT using features constructed by CDFC also has significantly better performance on six datasets. An improvement of at least 10% on average accuracy is achieved on three datasets, namely SRBCT, CNS and DLBCL, with the best accuracy improved from 15% to 21%. CDFC obtains about 1% and 6% lower average accuracy than Full on Leukemia and Leukemia1. However, their best results are higher and equal to Full, respectively.

In general, over 24 comparisons with Full using the three learning algorithms on 8 datasets, CDFC wins 21 and loses 3 in terms of average accuracy. However, in term of the best accuracy, CDFC outperforms Full in all 24 cases except for the NB result on SRBCT. Results showed that CDFC can construct a very small number of features with high discrimination and generalised well to the three learning algorithms in most cases.

CDFC Versus MGPFC. As shown in Table 3, although both methods construct the same number of features, KNN using features constructed by CDFC achieves significantly better performance than using those of MGPFC on all datasets. The highest improvement of 10% on average is found on Colon which MGPFC failed to maintain its Full accuracy. The results show that using terminal sets comprising of features that are relevant to a specific class, CDFC achieves much better results than MGPFC, allowing it to obtain the best KNN accuracy on all datasets.

Similarly, NB using features constructed by CDFC achieves better accuracy on 7 datasets than using those constructed by MGPFC. Among these datasets, CDFC further improves the performance of MGPFC from 6% to 12% on 5 datasets. Only on Leukemia1, CDFC obtains a similar accuracy as MGPFC.

For DT, features constructed by CDFC obtain significantly better performance than those of MGPFC on four datasets, namely Colon, DLBCL, Prostate and SRBCT with a further improvement of 3% to 6%. Among the other five datasets, CDFC has similar performance as MGPFC on four and worse on one.

In summary, features constructed by CDFC have significantly better performance than those constructed by MGPFC on 19 cases, similar on 4 and worse on 1. Note that results of CDFC on almost all datasets have smaller standard deviation than MGPFC despite of the learning algorithms being used. This indicates that by constraining the terminal sets to class relevant features, the performance of CDFC is better and more stable than MGPFC.

5.2 Performance of the Selected Features

To further investigate CDFC's performance, we also compared the performance of CDFC selected features with those selected by MGPFC. Figure 3 shows the differences in performance of KNN, NB and DT when using features selected by the two methods. The larger the difference, the better the performance of CDFC. If the difference is significantly different, a "*" is displayed above the corresponding bar.

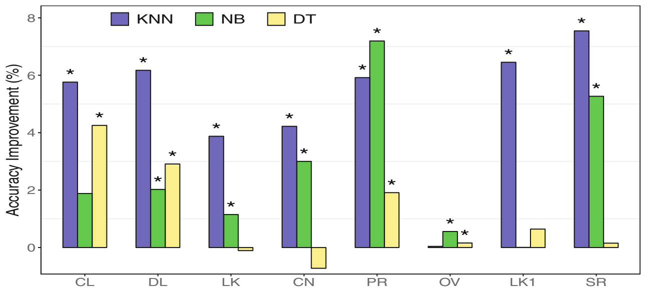


Fig. 3. Improvement of CDFC over MGPFC using the selected features (in colour).

As can be seen from Fig. 3, KNN combined with CDFC has significantly higher accuracy than with MGPFC on seven datasets. Among the three learning algorithms, KNN also has the largest improvements on six out of the eight datasets. NB has better performance on 6 datasets and similar on the remaining two. Similar pattern is seen in DT. Although the DT accuracy on CNS degrades 0.72%, the difference is not significant.

In general, over the 24 comparisons, CDFC selected features have significantly better performance than MGPFC on 17 cases and perform similar to MGPFC on the remaining 7 cases. The results indicate that by forming terminal sets with class-relevant features, thus narrowing the GP search space, CDFC can select more relevant features. These features are then used to construct features leading to further improvement in some datasets, e.g. from less than 2% to more than 12% for NB on the Colon dataset as shown in Table 3.

Another component contributing to CDFC superior performance is the new fitness function. Although both methods use the same distance measure described in Sect. 3.3, they also use an additional different measure in their fitness functions. While CDFC uses average IG, MGPFC uses average accuracy of DT. Although IG is the base measure of DT, the two fitness functions behave differently, leading to a significant improvement in the performance of CDFC. Using average IG, CDFC tries to maximise IG of all constructed features. On the other hand, using DT accuracy, MGPFC searches toward those set of constructed features that can produce better DT classifiers which may not use all constructed features. As a result, MGPFC fitness does not reveal the goodness of the whole set of constructed features.

In summary, performance of the three learning algorithms showed that the constructed and selected features by CDFC have better discriminating ability than MGPFC. The results also demonstrate the effectiveness of the proposed strategy in CDFC. In the next section, we will investigate how efficient CDFC is compared to MGPFC.

5.3 Computation Time

Figure 4 shows the average running time to complete a single run for MGPFC and CDFC. Compared with MGPFC running time, CDFC only takes less than one third in 4 out of 8 datasets, less than half in 3 datasets, and more than half in the largest dataset, Ovarian.

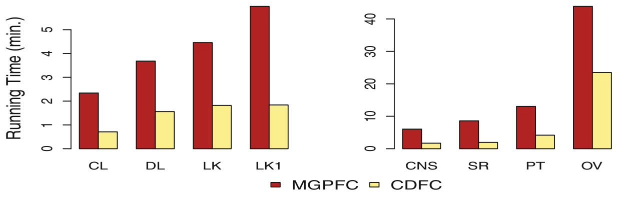


Fig. 4. Computation time of CDFC versus MGPFC (in colour).

Note that both methods have the same population size and maximum generations. In other words, they have the same number of evaluations. However, CDFC running time is much shorter than MGPFC in all datasets. The main reason behind this reduction is the computation time of the fitness evaluations. While the CDFC fitness function comprises of two filter measures, distance and IG, MGPFC combines distance with a wrapper measure. This again confirms the efficiency of filter measures.

6 Conclusion

The goal of this work is to propose a class-dependent feature construction method (CDFC) that can produce a smaller number of high-level features which can improve the performance of common learning algorithms on high-dimensional data. The goal was achieved by creating different terminal sets for constructing different class-dependent features. t-Test is used as a relevance measure to rank features in the context of a given class. A new fitness function is also proposed to better evaluate a set of constructed features based on two filter measures.

Performances of the CDFC constructed and selected features are compared with those on the original set and those generated by MGPFC using KNN, NB and DT. Results on the eight high-dimensional datasets show that CDFC is not only more effective in almost all cases but also more efficient than MGPFC in all cases. The proposed strategies in CDFC demonstrate that by forming the GP terminal set with class-relevant features and a good fitness function, the proposed filter method can achieve better performance than a hybrid approach.

In CDFC, the fitness function combines two filter measures using a fix weight. Furthermore, the number of top-ranked features used to form terminal sets for each class is also predefined. A dynamic weight and an automatic adjustment of the terminal sets based on the performance of the constructed features can be investigated to further improve the CDFC performance. Finally, our future work also includes more analysis to provide more insights on the proposed method.

References

1. Ahmed, S., Zhang, M., Peng, L.: A new GP-based wrapper feature construction approach to classification and biomarker identification. In: IEEE Congress on Evolutionary Computation, pp. 2756–2763 (2014)
2. Al-Sahaf, H., Al-Sahaf, A., Xue, B., Johnston, M., Zhang, M.: Automatically evolving rotation-invariant texture image descriptors by genetic programming. IEEE Trans. Evol. Comput. **21**(1), 83–101 (2016)
3. Bhanu, B., Krawiec, K.: Coevolutionary construction of features for transformation of representation in machine learning. In: Proceedings of Genetic and Evolutionary Computation Conference, pp. 249–254. Press (2002)
4. Cha, S.H.: Comprehensive survey on distance/similarity measures between probability density functions. Int. J. Math. Models Methods Appl. Sci. **1**, 300 (2007)
5. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. J. Bioinform. Comput. Biol. **3**(02), 185–205 (2005)
6. Espejo, P., Ventura, S., Herrera, F.: A survey on the application of genetic programming to classification. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **40**(2), 121–144 (2010)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
8. Krawiec, K.: Genetic programming-based construction of features for machine learning and knowledge discovery tasks. Genet. Program. Evol. Mach. **3**, 329–343 (2002)
9. Nag, K., Pal, N.: A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. IEEE Trans. Cybern. **46**(2), 499–510 (2016)
10. Neshatian, K., Zhang, M., Andreae, P.: A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming. IEEE Trans. Evol. Comput. **16**(5), 645–661 (2012)
11. Tran, B., Zhang, M., Xue, B.: Multiple feature construction in classification on high-dimensional data using GP. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–8 (2016)
12. Tran, B., Xue, B., Zhang, M.: Genetic programming for feature construction and selection in classification on high-dimensional data. Memetic Comput. **8**(1), 3–15 (2015)
13. Wang, L., Zhou, N., Chu, F.: A general wrapper approach to selection of class-dependent features. IEEE Trans. Neural Netw. **19**(7), 1267–1278 (2008)
14. Xue, B., Zhang, M., Browne, W.N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Trans. Evol. Comput. **20**(4), 606–626 (2016)
15. Zhang, J., Wang, S., Chen, L., Gallinari, P.: Multiple Bayesian discriminant functions for high-dimensional massive data classification. Data Mining Knowl. Discovery **31**(2), 1–37 (2016)