VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

Particle Swarm Optimisation For Feature Construction

Yan Dai

Supervisors: Mengjie Zhang, Bing Xue

18 October, 2013

Submitted in partial fulfilment of the requirements for
Bachelor of Computer Science and Engineering with
Honours in Software Engineering.

Abstract

Feature construciton aims to construct new high-level features from the original dataset. The constructed feature is expected to improve the representation quality of the original dataset. Particle swarm optimisation (PSO) is an effective search technique and has been widely used in feature selection tasks. However, PSO has seldom been used for feature construction. In this project, we propose an algortihm using PSO for feature construction in binary classification. Firstly, we propose two different representations for particles in PSO. The experimental results show that the single constructed new feature can improve the classification performance. Secondly, we developed a PSO based approach to construct one single high-level feature for multi-class classification problem. The experimental results show that the constructed feature is not informative enough for multi-class classification problem. Thirdly, we developed a PSO based approach to construct multiple high-level features for classification problem. The experimental results show that the developed approach can improve the classification performance for some datasets.

Contents

Chapter 1

Introduction

Classification is one of the most important tasks in machine learning and data mining. The large number of features is a difficult obstacle when people are trying to solve real-world problems. The difficulty of classification will be increased by including irrelevant and redundant features. The consequence is that most classification algorithms cannot achieve good classification performance. Therefore, one concern is reducing the number of original features and improving the representation quality.

Feature transformation is one of the most popular approaches used today in which feature selection and feature construction are included. Feature selection aims to choose a subset of existing features from original features. The selected subset of features should be able to fully represent and describe the concept shown by the original features. Compared to feature selection, the objective of feature construction is to construct new high-levl feature(s) based on some functional expression which uses the original features as input [1]. A constructed feature usually acts as a function of the original low-level features [13]. The constructed feature(s) should be able to discover the hidden relationship amongst the original features. Feature construction can improve the quality of the representation, reduce its complexity and increase the classification performance [18].

Feature construction needs to produce a function or a set of functions which uses a subset of the original features as input to construct new features. Two major factors will significantly influence the performance of the constructed feature. The first is how the subset of original features can be selected. Secondly, what operators will be used in the feature construction function. Exhaustively searching all the possible solutions will be expensive or even impractical when the total number of original features is large. Therefore, an effective and efficient global search technique is essential to a good feature construction method.

Evolutionary computation techniques are global heuristic search techniques, which have been widely used in many areas [5]. Feature construction using Genetic Programming (GP) [14, 8, 15] and Genetic Algorithms (GAs) [1] has been successfully developed. GP provides a terminal set which can reserve all original features and a function set where the potential operators can be reserved. The terminal and function sets provided by GP are advantages for solving the feature construction problem. Mayfield et al. [11] propose an approach for feature construction using GP to a language processing task. The presence or absence of features is represented as boolean statements in the proposed approach. The potential operators are AND and XOR. The results show that the unigram feature space can be improved by introducing newly constructed features. Alfred [1] proposes a feature construction method called Dynamic Aggregation of Relational Attributes (DARA) to represent records stored in the non-target tables. GAs are also used to construct a relevant set of features for DARA. Each chromosome is initialised with the format of $< X, A, B >$, where X represents a list of the features from the original data set, A represents the number of features combined, and B

represents the point of crossover. GAs is used as a heuristic search strategy to avoid the local optima and find the global optimal solutions. The results show that the data summarisation results can be improved by the constructed features.

Particle swarm optimisation (PSO) is a relatively recent evolutionary computation technique [7]. PSO is inspired by social behaviour of birds flocking and fish schooling. PSO is a less computational cost and quicker converged technique compared with GP and GAs. In PSO, fewer parameters are introduced than GP and GA, which makes PSO an easier controlled algorithm. There are several successful implementations using PSO solving real-world problems, such as feature selection [3, 19], process planning and scheduling [6]. PSO has been successfully used for feature selection, but has seldom been used for feature construction. This is one of the main motivations for using PSO for feature construction.

To construct effective feature(s), the functional expression is essential for constructing feature(s). One of the most important steps for making the functional expression is the input features selection from the original dataset. Therefore, a good technique for selecting a subset of useful features from the original features is one of the most important factors in feature construction. PSO can be a good technique for selecting features from the original dataset as inputs for the feature construction function. However, PSO has seldom been used for feature construction, which is one of the motivations to propose this project. One of the reasons making PSO difficult for feature construction is that the current particle encoding scheme and the velocity updating mechanisms in PSO are not designed to allow the function operators to be included in the evolutionary process.

1.1 The Problem

To consturct a high-level feature(s), a good set of functions are necessary. The purpose of using PSO is to produce such function(s). Two of the most important steps for feature construction using PSO are the feature selection from the original data set and operators selection from the potential operators set. The advantage of using PSO for feature construcion is that PSO can easily handle feature selection from the original data set. However, having the operators seleciton included in the PSO evolutionary process is difficult. Normally, the potential operators set will be defined before making the feature construciton function. During the evolution, the operators will be selected from the potential operators set. In PSO, the numeric representation is used for all candidate solutions. However, the potential operators are not easy to be converted to a numeric representation. A normal way for replacing the character-based operators is to give each potential operator a numeric index. Therefore, during the evolution, the operators' index is going to be included rather than the operator themselves. The problem is that how an effective PSO encoding scheme and updating machanism can be implemented.

In PSO, each particle will represent a candidate solution. The evaluation of each candidate solution is shown by particle's position [21]. At the end of the evolution of PSO, the particle with the best position will be selected. In feature construction, each particle can be seen as one candidate for one constructed feature. At the end, the best particle with the one constructed feature will be chosen. However, in real-world problems, a single constructed feature may not be enough for complex problems. Therefore, how to construct multiple features could be another challenging problem.

1.2 Project Goals

The overall goal is to develop a PSO based approach to feature construction for classification problems. Feature construction aims to summarise data from the original data set and construct high-level feature(s). By using the constructed feature(s), the representation quality of the data set should be improved compared with the original data set. This should include the classification accuracy improving and the computation time decreasing. To accomplish the overall goal, three sequential sub-goals are raised, which are shown in Figure 1.1.

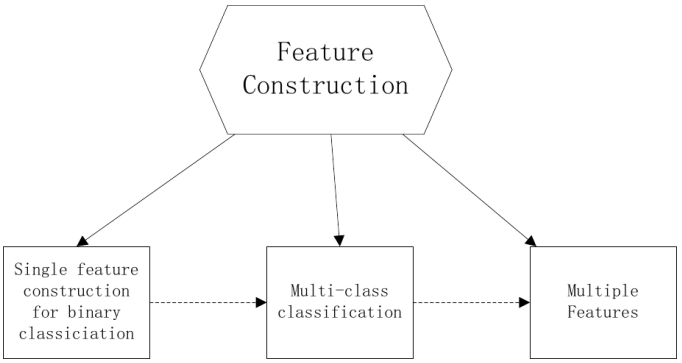

Figure 1.1: THE GOALS OF THE PROJECT

One of the most difficult obstacles in PSO for feature construction is the representation problem. In PSO, the candidate solutions are encoded as particles. To do feature construction, the output of PSO algorithm should be a function or a set of functions using the original features as input and produce new output as a constructed new feature. However, the encoding scheme and the updating mechanisms in PSO only allow using numeric representation for the candidate solutions. For feature construction, the operators for the feature construction function are not easy to be used in the numeric representation in PSO. Therefore, the first goal of the project is to figure out a representation design and encoding machanism for feature construction using PSO. At this stage, a single constructed new feature for binary classification is expected.

Secondly, the first sub-goal of the project only targets on the binary classification problem. However, in reality, the classification problem may include multiple classes. Using a single feature for multi-class classification is going to be a challenging task. This is also a good way to test the effectiveness of the algorithm designed for the first sub-goal because during the design and implemention of the single feature in sub-goal 1, only binary classificaiton task is used for testing. Therefore, in the second sub-goal, the multi-class classification can help with tuning the proposed representation from sub-goal 1.

With the working on the representation for the feature construction using PSO discussed above, the sub-goal 3 is to produce multiple constructed features. The sub-goal 1 and sub-goal 2 only produce a single constructed feature, which may not be enough for large, complex data sets. Therefore, constructing multiple features for multi-class classification problems is going to be the third goal of the project.

1.3 Major Contributions

We started our first work on PSO based feature construction in [21]. In [21], PSO is used for selecting features from the dataset and an independent operator selection function is used to determine the operators for the constructed feature. A single high-level feature is

constructed for binary classification problem and the experimental results show that the constructed feature can improve the classification performance in some cases.

In this project, the major contributions are shown below:

- We proposed an array and pair representations for PSO to evolve and select the features and operators. In the area of feature construction, PSO has never been used to evolve and select operators for the constructed feature. Compared with [21] which has the operator selection as independent function, this project shows how operator selection can be done as part of feature construction using PSO for constructing high-level feature. This is the first work using PSO for doing operator selection. The experimental results shows that the proposed approach can effectively select a combination of features and operators to construct a relatively good single high-level feature. The performance of binary classification can benefit from using the constructed feature.
- We developed an approach to construct a single high-level feature using PSO for multi-class classification. The project shows how to construct a single high-level feature using PSO for multi-class classification problems. PSO has never been used to construct single feature for multi-class classification problems. The experimental results show that the proposed approach can construct a high-level feature for multi-class classification dataset. The single newly constructed high-level feature with the original features together can improve the classification performance. We also compared the operator selection performance of using the array and pair representations. And the experimental results shows that the probability of a operator can be chosen by array representation is more evenly distributed than using pair representation for each candidate operator.
- We developed an approach to construct multiple new high-level features using PSO for classification problems. The project shows how to construct multiple high-level features using PSO for classification problem, which has never been done by using PSO. The proposed approach is based on [15] who is using GP to construct multiple features and calculate the class interval purity to evaluate the constructed feature during the evolutionary process of PSO. Different from [15], we separated the dataset for finding class interval and calculate the class interval purity. The experimental results show that the proposed method can construct a set of highly informative high-level features for some datasets.

1.4 Organisation

The report is organised in the following way. The second chapter will show the background information of the research. The third chpater will propose two differnet representation for PSO to construct one single feature for binary classificaiton problem. In the fourth section, we will introduce a way to construct one single high-level feature for multi-class classification problem. In the fifth section, we will propose an approach to construct multiple high-level features using PSO for classification problem. In the last section, we will conclude the project.

Chapter 2

Background

2.1 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) [7, 17] is an evolutionary computation technique. PSO is one of the techniques based on swarm intelligence. PSO was firstly introduced by Eberhart [7] in 1995. The intention of PSO is to stimulate social behaviours of birds flocking and fish schooling. In PSO, there are one basic element called particle. Each candidate solution is encoded as a particle. At the beginning of a PSO algorithm, a population of particles is initialised with random positions and the population is called the swarm in PSO. During the evolution of PSO, all the particles move in the search space to find the optimal solutions. For any particle i, a vector $x_i = (x_{i1}, x_{i2}, ..., x_{iD},)$ is to represent the its position and a vector $v_i = (v_{i1}, v_{i2}, ..., v_{iD},)$ represents its velocity, where D is the dimensionality of the search space. During the evolutionary process, each particle can remember the best position visited so far as the personal best position, which is called *pbest*. The best position for the whole swarm is remembered as the global best, *gbest*. The *gbest* is determined based on how a particle shares information with its neighbours. Each particle updates its position in each iteration based on its velocity and previous position. The velocity is also updated in PSO for each particle. The updating equations used in PSO are presented below:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{2.1}$$

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_{1i} * (p_{id} - x_{id}^{t}) + c_2 * r_{2i} * (p_{gd} - x_{id}^{t}) \tag{2.2}$$

In the two equations, t shows the tth iteration. $d \in D$ shows the dth dimension. w is the inertia weight used to balance the local search and global search abilities of PSO. c_1 and c_2 are acceleration constants. r_{1i} and r_{2i} are random constants uniformly distributed in $[0, 1]$. p_{id} and p_{gd} denote the values of *pbest* and *gbest* in the dth dimension. A predefined maximum velocity v_{max} is used to limit the v_{id}^{t+1}.

A Binary Particle Swarm Optimisation (BPSO) is developed by Kenny and Eberhart [6]. in BPSO, x_{id}, p_{id} and p_{gd} are either 1 or 0. The velocity in BPSO represents the probability of the corresponding position taking value of 1. To transfer the velocity value between 0 and 1, we uses a sigmoid function to achieve this. The following equation is used to update the position of each particle:

$$x_{id} = \{^{1, if rand() < s(v_{id})}_{0, otherwise} \tag{2.3}$$

where

$$s(v_{id}) = \frac{1}{1 + e^{-v_{id}}} \tag{2.4}$$

The $rand()$ is a random number selected from a uniform distribution in (0, 1).

2.2 Related work on Feature Construction

The feature construction approaches can be categorized by whether a learning/classification algorithm is included in the feature construction process or not, which are wrapper approaches and filter approaches [10].

In a wrapper method, a learning/classification algorithm is included and classification performance is used to evaluate the quality of the constructed feature(s). Murthy et al. [12] introduces a feature construction system. The system can construct new features through linearly combining the original features in the process of learning a decision tree and the technique is recognised as an efficient technique for feature construction since GP can help with improving the accuracy by building good quality programs and expression in . Lim et al. [9] propose an explanation-based feature construction approach. In the explanation-based feature construction approach, an explanation-based interaction between training examples and prior domain knowledge is used to guide the automatical construction for task-relevant discriminative features. The presented results show that the feature construction approach can construct effective features for the most difficult and complex character pairs in the experiment.

In filter approaches, feature construction is a separate, independent preprocessing stage of any learning algorithm. The filter approaches improve the computational efficiency and generality so that the filter approach becomes a popular way for feature construction problems. Otero et al. [16] propose a GP method for feature construction and the information gain ratio is used in the fitness function. During the process, C4.5 classifier is used to evaluate the performance of the constructed feature. The result shows that the constructed feature can increase the classification performance combining with the original features. The result may benefit from either C4.5 classifier or GP with information gain ratio as fitness function. However, the effect of whether a different classifier can benefit the feature construction is still unknown.

Recently, GP has been widely used in feature construction task. Neshatian at al. [15] develop a GP based feature construction algorithm. The fitness function of the algorithm is based on the class dispersion and entropy. The experimental results show that the classification performance is improved. Then Neshatian and Zhang [13] develop a GP based feature construction method. This method uses a variable terminal pool constructed by the class-wise orthogonal transformations of the original features. The experimental results show that the proposed GP based algorithm can improve the principle component analysis method in terms of classification performance and dimensionality reduction. Neshatian at al. [14] develop another algorithm based on GP. The proposed GP algorithm includes an entropy-based fitness so that the the purity of class intervals can be maximised. A decomposable objective function is proposed so that the system is able to construct multiple high-level features. Experimental results show that the constructed featues are highly relevant to improve the classification performance.

2.3 PSO for Feature Construction

PSO is rarely used for feature construction. We start the first work on PSO for feature construction in [21]. The proposed algorithm uses PSO for feature construction and has a separated function operator selection algorithm. The constructed feature is used to perform a binary classification.

The process starts with randomly generated particles and predefined candidates operators. Each particle contains an array of binary numbers. The length of the array equals to the number of the features in the dataset. If the value equals to 1, the feature will be selected

and 0 otherwise. In each generation, an array of features from original dataset has been selected. Then we will run an algorithm to select operators based on the selected array. Four operators are used in the work and they are +, -, * and /. Exhaustively searching the best function operators sets for the selected features is time consuming. Therefore, we use local search to find a near-optimal combination of function operators and the selected features. The operator for the first selected feature are always set to "+" since we do not need form a function by a single feature. From the second selected feature, the function operators selection algorithm starts by searching a proper operator for the first two selected features, which construct a new feature (f_c) based on the first two selected feature. The operator with the best evaluation performance will be remembered and another operator selection process with the third selected features started. A new operator will be selected between the f_c and the third selected feature, and the f_c will be updated. The overall process will finished until the last selected feature has been used.

The experimental results show that the proposed feature construction method can construct useful feature for improving the classification performance. However, the benefit of the constructed feature can be caused by the local search for the operator selection which is computaionally expensive. The proposed algorithm made the feature selection and operator selection process separtately. Therefore, in this project, we want to use PSO for both feature selection and operator selection simultaneously.

Chapter 3

Two Representations For Feature Construction Using PSO For Binary Classification

3.1 Introduction

So far, we have only focus on the first sub-goal of using PSO for feature construction, which is the one of constructing single high-level feature for binary feature construction. In the rest of this chapter, we will describe the new methods and results.

3.2 Chapter Goals

The first goal of this project is to develop a PSO based algorithm for feature construction in binary classification tasks. We expect that the newly constructed high-level features can benefit the classification performance either used solely or combined with the original features. We will propose two representations for the PSO based feature construction algorithm, an array representation and a pair representation.

Specifically, we will investigate:

- whether the new representations can do a good enough job for feature construction. We will compare the results of the new developed representations with [21]
- whether only use the newly constructed feature can improve the classification performance.
- whether combining the constructed feature with the original feature can improve the classification performance.

3.3 Proposed Approach

3.3.1 The Overall Design

In this section, we propose two PSO based feature construction and classification approaches. A single high-level feature will be constructed as the result of the algorithm. The different part for the two algorithms is the particle representation. The process starts with constructing a training set and a testing set from the original dataset. By using the training dataset, a high-level feature is constructed. The high-level feature is then used to construct a new

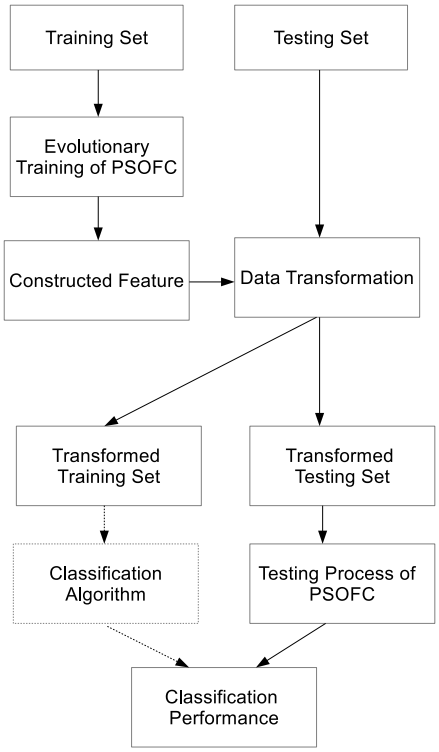

Figure 3.1: OVERALL OPERATOR SELECTION PROCESS

training set and a new testing set based on the original training and testing set. The new training and testing dataset only contain a single newly constructed feature and the corresponding class labels. The purpose of having these two new training and testing dataset is to test the classification performance of the newly constructed high-level feature. The Figure 3.1 shows the overall process of PSO for feature construction (PSOFC).

In order to construct a high-level feature, a set of function operators needs to be selected and be optimised with the selected features. This is the main challenge for using PSO for feature construction.

Two representations for particles in PSO are proposed. The process starts with initialising particles with random positions. The next step is to select features from the original dataset as input for a feature construction function. Meanwhile, according to the different representations, the corresponding function operators are also selected for the selected features. The details for selecting the operators for the two representations will be shown in the following two sections and the particle representation is the main difference for the two proposed representations. Then, in order to evaluate the solution and proceed the evolution, a binary classification process will be used to produce a fitness value based on the constructed feature. The evolution process will finally stop until the maximum number of iterations is reached.

3.3.2 Array Representation

In this proposed representation which we call it *array representation*, each particle is initialised with an array of float numbers. If the total number of features of a given dataset is n, the length of the array will be $2 * n - 1$. In each particle, each array is a combination of

$feature_1$	$operator_1$	$feature_2$	$operator_2$	...	$operator_{n-1}$	$feature_n$

Figure 3.2: ARRAY REPRESENTATION

$feature_1 \| operator_1$	$feature_2 \| operator_2$	...	$feature_n$

Figure 3.3: PAIR REPRESENTATION

features and function operators. The array can be seen in Figure 3.2:

The feature and operator selection shares the same value range. However, as the solution for feature selection is only selected or not selected, we will need to evenly split the value range to two parts. Since there can be more than two candidate operators, we will need to equally split the same value range according to the number of the candidate operators. During the evolutionary process, the algorithm will firstly decide a set of new features from the array of the particle based on a predefined threshold. Secondly, the algorithm will need to decide the operators for the features. If a feature is selected and with an index of m where the feature is not the last feature in the array, then the algorithm will get the number at position $m + 1$. And, base on the pre-defined range defined by human, we will get an operator. Algorithm 1 shows the process to select an array of operators (ops) for the selected feature. The process will not stop until the last dimension or element of the array in the particle is reached.

```
Data: position // the position should not be null
Data: operators // operators.length == ((pos.length − 1)/2)
Result: operators
pos ← position;
count ← 0;
i ← 0;
operators ← {};
while count < pos.length do
    if pos[count] > threshold then
        operators[i] ← getOperators(pos[count + 1]);
        i ← i + 1;
    end
    count ← count + 2;
end
return operators;
```

Algorithm 1: Selecting the operators for selected features using array representation

3.3.3 Pair Representation

In this proposed representation which we call it *pair representation*, each particle is initialised with an array of float numbers except the last feature since there are no other features will be operated with the last selected feature. The array will have same length of the number of features in the given dataset. However, each number of the array can decide both whether the feature is selected or not, and which operators will be used if the feature is selected. The general structure can be seen in Figure 3.3.

During the evolution of PSO, the algorithm will first look at whether the feature is selected or not. If the feature is selected, an operator will be decided according to the same value which decide the feature selection. The feature selection and function operator selection will not terminate until the last value in the array of a particle has been evaluated. To be noticed, the value range for feature selection and function operator selection are not same. The value range of the feature selection will be always twice to the value range of the function operator one. Algorithm 2 shows the operator selection process when using the pair representation.

```
Data: position // the position should not be null
Data: operators // operators.length == pos.length − 1
Result: operators
pos ← position;
count ← 0;
i ← 0;
operators ← {};
while count < pos.length do
    if pos[count] > threshold then
        operators[i] ← getOperators(pos[count]);
        i ← i + 1;
    end
    count ← count + 1;
end
return operators;
```

Algorithm 2: Selecting the operators for selected features using pair representation

3.3.4 Binary Classification

In the proposed algorithm, the binary classification performance as the fitness value is used to evaluate the constructed feature. Because of only binary classification problems considered in the work for both representation, the proposed algorithm sets the threshold to "0". An instance is classified to class 1 when the value of the constructed feature is larger than 0. Otherwise, it is classified to class 2. The purpose for having an independent classification process without any classification algorithm is to increase the speed of fitness evaluation so that we can have shorter time for running the program.

3.4 Experiment Design

In order to evaluate the performance of the proposed feature construction algorithm, a set of experiments have been conducted using different binary benchmark datasets. The binary benchmark datasets are chosen from UCI machine learning repository [2]. The details of the seven datasets used in this report are shown in Table 3.1. The number of features is ranged from 14 to 500. The number of instances is from 351 to 4400. The instances in each dataset will be divided into 70% as a training set and 30% as a testing set [14, 20].

According to [17], the parameters of the proposed algorithm are set as follows: $w = 0.7298$, $c_1 = c_2 = 1.49618$, $v_{max} = 0.1$, $v_{min} = -0.1$, $domain_{min} = 0.0$ as the minimum position of a particle in each dimension, $domain_{max} = 1.0$ as the maximum position of a particle in each dimension, population size is 30, $threshold = 0.5$ and the maximum number

Table 3.1: Datasets

Dataset	No. of Features	No. of Classes	No.of Instances
Australian	14	2	690
Ionosphere	34	2	351
WBCD	30	2	569
Hillvalley	100	2	606
Musk1	166	2	476
Semeion	256	2	1593
Madelon	500	2	4400

of iterations is 100. The fully connected topology is used in the experiments. For each dataset, the proposed algorithm has been independently run 50 times. A single high-level newly constructed feature will be produced in each run. The function operators set used in this work are "+", "-", "*" and "/" (protected division).

For the array representation, the value range for each operator are:

"+": [0.0, 0.25)

"-": [0.25, 0.5)

"*": [0.5, 0.75)

"/": [0.75, 1]

For the pair representation, the value range for each operator are:

"+": [0.5, 0.625)

"-": [0.625, 0.7)

"*": [0.7, 0.825)

"/": [0.825, 1]

We designed the operator value range for each representation is based on the concerns that every operator is equally important.

Three different classifiers are used to test the classification performance of the newly constructed high-level feature. The three classifiers are naive bayes (NB), K-nearest neighbour (KNN) where K = 5 and decision trees (DT).

For each data set, we will conduct the classification performance evaluation using the three classifiers on the original dataset. The results will be used as a baseline for comparison with the constructed feature. For each representation of the algorithm, we will test the constructed feature by the three different classifiers and get the best classification performance, the average classification performance and the standard deviation. Then the classification performance of the combination of the original features and the newly constructed high-level feature is evlauated by the three classifiers to get the best classification performance, the average classification performance and the standard deviation of the classification performance.

3.5 Experimental Results

The experimental results for using the array representation is shown in Table 3.2. The experiment results for using the pair representation is shown in Table 3.3. In both tables,

Table 3.2: Result of array representation

Data set	no. Features	Methods	DT			KNN			NB		
			Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
australian	14	All	85.99			70.05			85.51		
		CF	86.96	84.79	1.48E-2	86.47	66.22	18.5E-2	76.33	55.54	5.34E-2
		CFOrg	87.92	85.79	1E-2	86.96	78.88	4.39E-2	88.89	86.63	60.7E-4
ionosphere	34	All	86.67			83.81			28.57		
		CF	85.71	76.04	4.57E-2	84.76	75.92	5.24E-2	85.71	82.38	1.36E-2
		CFOrg	91.43	86.08	3.88E-2	87.62	84.4	1.14E-2	28.57	28.57	2.78E-16
wbcd	30	All	92.98			92.98			90.64		
		CF	95.91	92.19	2E-2	95.91	91.54	2.62E-2	61.4	61.4	2.22E-16
		CFOrg	97.08	93.45	1.01E-2	95.91	92.99	1.07E-2	90.64	90.64	2.22E-16
hillvalley	100	All	62.09			56.59			52.2		
		CF	98.35	95.27	3.08E-2	98.35	94.43	44.4E-2	47.8	47.8	0E0
		CFOrg	98.63	95.4	3.05E-2	98.35	91.73	46.8E-2	52.47	52.2	3.85E-4
musk1	166	All	71.33			83.92			42.66		
		CF	66.43	58.14	4.54E-2	67.13	57.41	4.33E-2	60.84	59.33	54.8E-4
		CFOrg	72.03	71.29	32.5E-4	86.71	66.76	13E-2	72.73	72.73	3.33E-16
semeion	265	All	93.31			96.44			90.79		
		CF	91.42	90.02	25.5E-4	89.96	89.2	79.2E-2	91.42	90.02	27.8E-4
		CFOrg	96.23	93.44	54.7E-4	96.65	96.45	8.87E-4	91.84	90.83	30.1E-4
madelon	500	All	76.79			70.9			49.49		
		CF	62.82	53.84	18.3E-2	55.9	52.3	56.7E-2	52.95	49.71	72.8E-4
		CFOrg	77.95	76.81	25E-4	70.9	53.38	58E-2	55.51	55.5	5.29E-4

"All" means only the original features are used for the classification; "CF" means only the single constructed feature is used for the classification; "CFOrg" means the combination of constructed feature and original features is used for the classification. The symbol "Best", "Avg", "Std" represent the best classification accuracy, the average classification performance and the standard deviation of the classification performance.

3.5.1 Using the Array Representation

The experimental results of the array representation are shown in the Table 3.2. The experimental results by using only the constructed feature based on the DT classifier, the experimental results show that there are 3 out of 7 datasets can improve the classification performance. When using KNN classifier show that the classification performance of 4 out of 7 datasets can be improved by using only the constructed feature. An interesting observation from the result shows that the dataset with less than 100 features can benefit from using the constructed feature for classification performance. When the number of features is over 100, the classification is hard to benefit from only using the single constructed feature. Comparing with the DT classifier and KNN classifier, the result using NB is that 4 out of 7 datasets have the classification performance benefited from only using the constructed feature but all the dataset with over 100 features can benefit from only using the constructed feature.

The experimental results of using DT classifier show that the constructed feature can improve the classification performance in all datasets if combined with the original features and the results are 6 out of 7 datasets for KNN classifier. The classification performance using the NB classifier is slightly worse compared with the DT classifier and KNN classifier, which is 5 out of 7 datasets.

Table 3.3: Result of pair representation

Data set	no. Features	Methods	DT			KNN			NB		
			Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
australian	14	All	85.99			70.05			85.51		
		CF	85.99	66.79	36.2E-2	70.53	62.04	55.2E-2	66.18	54.66	2.98E-2
		CFOrg	86.96	86.01	12.3E-2	74.88	69.48	59.2E-2	87.92	85.65	89.7E-4
ionosphere	34	All	86.67			83.81			28.57		
		CF	83.81	74.22	19.7E-2	82.86	73.79	58E-2	85.71	81.2	2.92E-2
		CFOrg	90.48	86.91	78.1E-4	86.67	84.94	61E-2	28.57	28.57	2.78E-16
wbcd	30	All	92.98			92.98			90.64		
		CF	95.32	83.22	16.7E-2	95.32	85.4	46.4E-2	61.4	61.4	2.22E-16
		CFOrg	95.32	93.18	70.8E-4	95.32	92.79	49.3E-2	91.23	90.67	11.5E-4
hillvalley	100	All	62.09			56.59			52.2		
		CF	76.92	54.39	7.23E-2	75	54.43	23E-2	47.8	47.8	0E0
		CFOrg	78.3	62.92	4.21E-2	75	54.95	6.39E-2	52.2	52.2	0E0
musk1	166	All	71.33			83.92			42.66		
		CF	67.83	58.86	15.2E-2	62.94	55.74	1.38E0	60.84	59.37	61.4E-4
		CFOrg	71.33	71.24	43.4E-4	83.92	79.08	1.41E0	72.73	72.73	3.33E-16
semeion	265	All	93.31			96.44			90.79		
		CF	89.96	89.96	5.55E-16	90.79	82E-2	5.55E-16	89.96	89.96	5.55E-16
		CFOrg	93.31	93.31	5.55E-16	96.44	96.44	8.88E-16	90.79	90.79	4.44E-16
madelon	500	All	76.79			70.9			49.49		
		CF	59.36	50.69	2.09E-2	53.33	50.43	37.9E-2	49.49	49.49	0E0
		CFOrg	77.31	76.67	93.8E-4	70.9	58.36	20E-2	55.51	55.51	3.33E-16

3.5.2 Using the Pair Representation

The experimental results for using the pair representation are shown in the Table 3.3. When using the DT classifier, the result shows that the number of the dataset whose classification performance can benefit from only using the constructed feature is 2 out of 7 datasets. When using the KNN classifier, the number increases to 3 out of 7 datasets. The classification performance of the NB classifier is 2 out of 7 datasets.

When using the DT classifier, the classification performance can be improved on 5 out of the 7 datasets by using the combination of the original features and the constructed feature. When using the KNN classifier with the combination of the constructed feature and original features, the number of datasets with improved classification performance is 4 out of 7. When the NB classifier is used, the number increased by 1 and 5 out of 7 datasets can improve the classification performance by using the combination of the constructed feature and the original features.

3.6 Discussion

We have two general observations from the experimental results above by comparing the two representation. Firstly, the constructed feature can improve the classification performance if we combined the constructed feature with the original features. This observation is applied for both representations. Therefore, we can make a conclusion that the proposed algorithm for both representation cannot achieve the goal of reducing the original features. Secondly, we found that both representations cannot construct a single effective feature for the datasets with more than 100 features. Thirdly, for both representations, the classification performance of the DT classifier is better than the other two classifiers.

3.6.1 Limitation on Feature Selection

One of the most important steps for feature construction tasks is to select a set of features from the original features. The set of the selected features are to be used as the input for constructing a new feature. The hidden relationship between features can be diverse. PSO has a good nature for feature selection problems. In our PSO algorithm for selecting the features from the original dataset, we represent each original feature by an ordered array of float numbers. Therefore, the relationship can be omitted if two features are not neighbours to each other. Also, the number of appearance of each feature is fixed and it is always 1 or 0. The problem is that some feature may be worth to be repeatedly used for constructing a new feature.

3.6.2 Limitation on Operator Selection

The selection of the function operators by PSO is the most difficult part of the work. There are still shortages of function operator selection although this work has successfully evolved the function operators using PSO. Firstly, the mapping function for the division function is slight larger compared with other operators. The reason is that the interval for the division include the side value of the maximum domain number on the right hand side. This slightly difference can increase the opportunity of the selection of the division function. Secondly, if we compare the two representations, we can find that the array method have larger range for each operator. The result by using the array representation is slightly better than the pair representation. Therefore, we doubt that the mapping range may affect the quality of the constructed feature.

3.6.3 Efficiency Improvement

Compared with [21], the proposed algorithm can efficiently construct a reasonably good quality of new features. The computation cost by the new algorithm can be hugely reduced for the function operator selection. Considering a dataset with n features and m candidate operators,ehe maximum number of evaluations for selecting operators in [21] is m^{n-1}. The proposed method using a hashtable based method. Therefore, the worst cost for operator selection is $1 * (n-1)$. Therefore, we can see that the proposed algorithm is more efficient than the previous work although the average classification performance is slightly worse compared than the our previous work. In Table 3.4, we compare the array representation, the pair representation with the previous results using only the constructed feature. In most of case, the previous results are slightly better than the new algorithms.

In Table 3.4 and Table 3.5, we comparing three different algorithms. The symbol "All" represents using all the original features only, "Array" means the array representation, "Pair" means the pair representation and "Gecco" means the algorithm proposed in [21].

In Table 3.4, we compare the three algorithms by the classification performance of only using the single constructed feature. When using the DT classifier, the array representation can achieve better classification performance in 2 out of 7 datasets compared with other two algorithm. When using the KNN classifier, only the "madelon" dataset can achieve better classification performace compared with others two algorithms and, 2 out of 7 can achieve equally good classification performance with the "Gecco" one. When using the NB classifier, there are 3 out of 7 datasets can achieve better classification performance by using the array representation. One observation we found is that the array representation may be good at constructing useful feature for larger dataset because the constructed feature by the array representation achieves better classification performance than the other two algorithms in "madelon" dataset which is the largest dataset in our experiment.

Table 3.4: Comparison table for three methods by single constructed feature

Data set	no. Features	Methods	DT			KNN			NB		
			Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
australian	14	All	85.99			70.05			85.51		
		Array	86.96	84.79	1.48E-2	86.47	66.22	18.5E-2	76.33	55.54	5.34E-2
		Pair	62.82	53.84	18.3E-2	55.9	52.3	56.7E-2	52.95	49.71	72.8E-4
		Gecco	86.96	85.35	1.13E0	86.96	55.16	14.3E0	86.47	62.97	10.2E0
ionosphere	34	All	86.67			83.81			28.57		
		Array	85.71	76.04	4.57E-2	84.76	75.92	5.24E-2	85.71	82.38	1.36E-2
		Pair	83.81	74.22	19.7E-2	82.86	73.79	58E-2	85.71	81.2	2.92E-2
		Gecco	87.62	81.14	3.16E0	87.62	80.53	3.7E0	83.81	77.56	1.71E0
wbcd	30	All	92.98			92.98			90.64		
		Array	95.91	92.19	2E-2	95.91	91.54	2.62E-2	61.4	61.4	2.22E-16
		Pair	95.32	83.22	16.7E-2	95.32	85.4	46.4E-2	61.4	61.4	2.22E-16
		Gecco	95.32	93.31	1.07E0	95.91	92.96	1.36E0	61.4	61.4	35.1E-4
hillvalley	100	All	62.09			56.59			52.2		
		Array	98.35	95.27	3.08E-2	98.35	94.43	44.4E-2	47.8	47.8	0E0
		Pair	76.92	54.39	7.23E-2	75	54.43	23E-2	47.8	47.8	0E0
		Gecco	99.45	99.41	9.53E-2	99.45	99.4	11E-2	49.45	48.5	48.6E-2
musk1	166	All	71.33			83.92			42.66		
		Array	66.43	58.14	4.54E-2	67.13	57.41	4.33E-2	60.84	59.33	54.8E-4
		Pair	67.83	58.86	15.2E-2	62.94	55.74	1.38E0	60.84	59.37	61.4E-4
		Gecco	76.22	65.96	3.24E0	72.03	63.9	3.15E0	59.44	58.43	62.8E-2
semeion	265	All	93.31			96.44			90.79		
		Array	91.42	90.02	25.5E-4	89.96	89.2	79.2E-2	91.42	90.02	27.8E-4
		Pair	89.96	89.96	5.55E-16	90.79	82E-2	5.55E-16	89.96	89.96	5.55E-16
		Gecco	100	100	0E0	89.96	89.96	18.4E-4	100	100	0E0
madelon	500	All	76.79			70.9			49.49		
		Array	62.82	53.84	18.3E-2	55.9	52.3	56.7E-2	52.95	49.71	72.8E-4
		Pair	59.36	50.69	2.09E-2	53.33	50.43	37.9E-2	49.49	49.49	0E0
		Gecco	53.97	53.97	43.6E-4	49.23	49.23	7.69E-4	49.1	49.1	25.6E-4

In Table 3.5, we compare the three algorithms by the classification performance by combining the constructed feature and original features. When using the DT classifier, only 2 out of 7 datasets can achieve better classification performance by using the array reprpesentation. When using the KNN classifier, better result can be found for the array representation. There are 4 out of 7 datasets achieving better classification performance by using the array representation. When using NB classifier, only 2 out of 7 datasets can achieve better classification performance by using array representation.

Seeing from Table 3.4 and Table 3.5, the array representation can produce better result compared with the pair representation. One reason of the results can be that the value range for operator selection is larger compared with the array representation in the array representation. As a consequence, the operator selection for the array representation can better avoid the local optima solution than the pair representation. Future work is needed to be done to futhre investigate and analyse the observation above.

Table 3.5: Comparison table for three methods by constructed and original features

Data set	no. Features	Methods	DT			KNN			NB		
			Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
australian	14	All	85.99			70.05			85.51		
		Array	93.31	93.31	5.55E-16	96.44	96.44	8.88E-16	90.79	90.79	4.44E-16
		Pair	86.96	86.01	12.3E-2	74.88	69.48	59.2E-2	87.92	85.65	89.7E-4
		Gecco	87.44	85.93	66E-2	80.19	73.29	2.28E0	88.41	86.97	43.7E-2
ionosphere	34	All	86.67			83.81			28.57		
		Array	91.43	86.08	3.88E-2	87.62	84.4	1.14E-2	28.57	28.57	2.78E-16
		Pair	90.48	86.91	78.1E-4	86.67	84.94	61E-2	28.57	28.57	2.78E-16
		Gecco	92.38	86.29	3.45E0	91.43	85.54	2.27E0	28.57	28.57	14.3E-4
wbcd	30	All	92.98			92.98			90.64		
		Array	97.08	93.45	1.01E-2	95.91	92.99	1.07E-2	90.64	90.64	2.22E-16
		Pair	95.32	93.18	70.8E-4	95.32	92.79	49.3E-2	91.23	90.67	11.5E-4
		Gecco	97.08	93.81	1.1E0	94.15	92.99	21.9E-2	90.64	90.64	32.7E-4
hillvalley	100	All	62.09			56.59			52.2		
		Array	98.63	95.4	3.05E-2	98.35	91.73	46.8E-2	52.47	52.2	3.85E-4
		Pair	78.3	62.92	4.21E-2	75	54.95	6.39E-2	52.2	52.2	0E0
		Gecco	99.45	99.41	9.53E-2	57.42	56.99	25.9E-2	53.02	52.25	19E-2
musk1	166	All	71.33			83.92			42.66		
		Array	72.03	71.29	32.5E-4	86.71	66.76	13E-2	72.73	72.73	3.33E-16
		Pair	71.33	71.24	43.4E-4	83.92	79.08	1.41E0	72.73	72.73	3.33E-16
		Gecco	77.62	71.54	3.09E0	88.81	70.11	7.95E0	72.73	72.73	27.3E-4
semeion	265	All	93.31			96.44			90.79		
		Array	96.23	93.44	54.7E-4	96.65	96.45	8.87E-4	91.84	90.83	30.1E-4
		Pair	93.31	93.31	5.55E-16	96.44	96.44	8.88E-16	90.79	90.79	4.44E-16
		Gecco	100	100	0E0	96.44	96.44	35.1E-4	94.56	94.56	6.69E-4
madelon	500	All	76.79			70.9			49.49		
		Array	77.95	76.81	25E-4	70.9	53.38	58E-2	55.51	55.5	5.29E-4
		Pair	77.31	76.67	93.8E-4	70.9	58.36	20E-2	55.51	55.51	3.33E-16
		Gecco	76.79	76.79	48.7E-4	72.05	72.05	12.8E-4	55.38	55.38	46.2E-4

3.7 Conclusion

In this chapter, we introduces a new algorithm using PSO for feature construction and binary classification. The proposed PSO based feature construction algorithm can effectively select a combination of features and operators. A relatively good feature can be automatically constructed by the proposed algorithm. The experimental results show that the binary classification can benefit from using the constructed feature.

Currently, our proposed method is only for binary classification problem. In real world, the classification problem can contain multiple classes. Therefore, to exam the ability of solving real-world complex problems, we will design and improve the current algorithm to solve multi-class classification problem using a single constructed feature. And we suppose that the work will not be hard as we need to adjust the fitness function without change the particle representation. Secondly, single constructed feature may not be adequate to solve the real-world complex problem. Therefore, our second task for the future work is to design an algorithm with PSO which can produce multiple constructed features for multi-class classification task.

Chapter 4

Feature Construction For Multi-Class Classification

4.1 Introduction

The first objective of using PSO for feature construction, which was to construct a single high-level feature for binary classification, has been finished. Based on the proposed two representations in the first objective, we will focuse on the second objective of using PSO for feature construction targeting on multi-class classification.

4.2 Chapter Goals

The second objective of the project is to develop a PSO based approach for feature construction in multi-class classification tasks. The newly constructed high-level feature is expected to be able to benefit the multi-class classification performance either used solely or combined with the original features. We will work on the two representations we proposed in the first objective, K-nearest neighbour (KNN) algorithm, decision tree (DT) algorithm and naive bayes (NB) algorithm are used to evaluate the fitness/classification performance of the constructed feature during the evalutionary training process.

Specifically, we will investigate:

- whether the newly constructed single high-level feature can improve the multi-class classification performance, and
- whether combining the constructed feature with the original feature can improve the multi-class classification performance.

4.3 Proposed Approach

In order to use PSO for feature construction for multi-class classification, we use three different algorithms in the PSO training process, DT, KNN and NB to evaluate the classification performance of the constructed feature, which is used as the fitness function in the proposed algorithm. The proposed representations allow PSO to directly evolve function operators for feature construction and construct a single high-level feature for multi-class classification tasks.

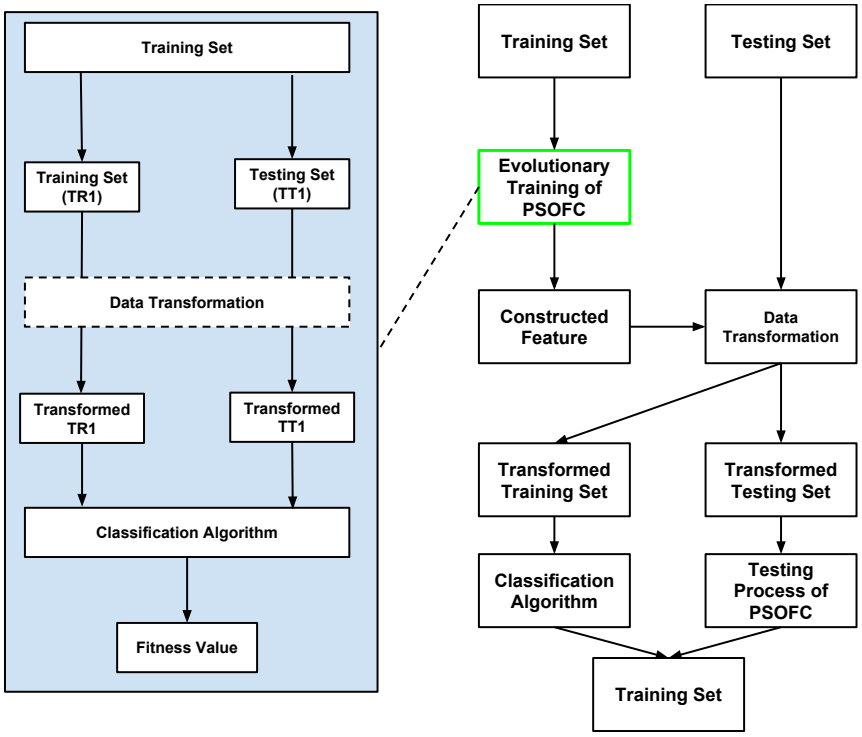

Figure 4.1: OVERALL PROCESS AND THE DETAIL OF EACH EVALUATION IN PSOFC

4.3.1 The Overall Design

In this section, we propose three different PSO based feature construction algorithms. The algorithms will construct one single high-level feature. The three algorithms differ from the learning algorithms and classification algorithms used during the training process and all the three algorithms will use the representations we developed in the first objective, the array representation and the pair representation. The process will start with splitting the original dataset to a training set and a testing set. The training set will be used to construct a high-level feature. Since in this chapter, we target on multi-class datasets rather than only binary-class datasets. The binary classification algorithm we used before is not suitable for this objective. DT, KNN and NB will be used during the evolutionary training process to evaluate the classification performance/fitness of the constructed feature. This is the different part from the first objective. The Figure right handside of 4.1 shows the overall process and the highlighted part is the evolutionary training process of PSOFC. The constructed feature will be used to construct new training and testing sets based on the original training and testing sets. The new training and testing sets will only contain the single newly constructed high-level feature and the correspond class labels. The overall process is same as we did in objective one.

4.3.2 Evolutionary Training of PSOFC

In this section, we will explain how DT, KNN and NB are used for the evalutation during the evolutionary training process of PSOFC. As the left handside of Figure 4.1 shows, we will get a constructed feature during each evolution from each particle to perform a evaluation. In order to evaluate the constructed feature for each particle, we split the original training set to a new training set (TR1) and testing set (TT1). Then according to the constructed feature, we will perform the data transformation on TR1 and TT1 to generate a new training

set (Tranformed TR1) and new testing set (Transformed TT1). The evluation is to perform a classification task using a specific algorithm with Transformed TR1 as training set and Transformed TT1 as testing set. The performance value of the classification is used as the fitness value for each particle. PSO uses the set of fitness values to decide the global best position for the generation.

4.4 Experiment Design

A set of experiments have been conducted using different benchmark datasets to evaluate the performance of the proposed PSO based feature construction algorithm. The used benchmark datasets are chosen from UCI machine learning repository [2]. The details of the 13 datasets are shown in Table 4.1. The number of features is ranged from 13 to 649. The number of instances is from 351 to 4400. The number of classes is ranged from 2 to 16. The instances in each dataset will be divided into 70% as a training set and 30% as a testing set [6, 20].

Table 4.1: Binary-Class Datasets

Dataset	No. of Features	No. of Classes	No.of Instances
Australian	14	2	690
German	24	2	1000
Ionosphere	34	2	351
WBCD	30	2	569
Hillvalley	100	2	606
Musk1	166	2	476
Semeion	256	2	1593
Sonar	60	2	608
Madelon	500	2	4400
Wine	13	3	178
Zoo	17	7	101
Vehicle	18	4	846
Lung	56	3	32
LibrasMovement	90	15	360
Arrhythmia	279	16	452
MultipleFeatures	649	10	2000

The parameters of the proposed algorithm are set as follows [17]: $w = 0.7298$, $c_1 = c_2 = 1.49618$, $v_{max} = 0.1$, $v_{min} = -0.1$ and $domain_{min} = 0.0$ as the minimum position value of a particle in each dimension, $domain_{max} = 1.0$ as the maximum position value of a particle in each dimension, population size is 30 and the maximum number of iterations is 100. The fully connected topology is used in the experiments. For each dataset, the proposed algorithm has been independently run 50 times. A single high-level newly constructed feature will be produced in each run. The function operators set used in this work are "+", "-", "*" and "/" (protected division).

For the array and pair representations, the value range for each operator are shown in the Table 4.2.

Table 4.2: Function Operator Selection

Operator	Array Rep	Pair Rep
"+"	[0.0, 0.25)	[0.5, 0.625)
"-"	[0.25, 0.5)	[0.625, 0.7)
"*"	[0.5, 0.75)	[0.7, 0.825)
"/"	[0.75, 1]	[0.825, 1]

Three different classifiers are used to conduct the classification performance tests to evaluate the goodness of the newly constructed high-level feature.

For each dataset and each newly constructed high-level feature, we will conduct a classification performance test with the three classifiers separately. Then we will combine the newly constructed feature with the original features. We will perform a T-test to see whether the newly constructed feature can improve the classification performance.

4.5 Experimental Results

The experimental results of using the array representation with the training classifier of DT, KNN and NB are shown in Table 4.3 Table 4.4 and Table 4.5. The experiment results of using the pair representation with the training classifier of DT, KNN and NB are shown in Table 4.6, Table 4.7 and Table 4.8. In all the tables, "All" means only the original features are used for classification; "CF" means only the single constructed feature is used for classification; "CFOrg" means the combination of the constructed feature and original features are used for classification. The symbol "Best", "Avg", "Std" represent the best classification accuracy, the average classification performance and the standard deviation of the classification performance.

4.5.1 Using the array representation

Using the DT during Evolutionary Training Process

Table 4.3 shows the experimental results of using the array representation with DT classifier during evolutionary training process. When we using the DT with the single constructed feature, the classification performance is improved on 2 out of the 14 datasets. When we using the KNN, we found that there are 4 out of the 14 datasets with improved classification performance. When we use NB, there are 2 out of the 14 datasets with better classification performance.

After we combine the original features with the constructed feature, using DT and KNN, we found that on 6 out of the 14 datasets, the classification performance is improved. When we use NB, we found that there are 2 out of the 14 datasets can be improved.

Using the KNN during the Evolutionary Training Process

Table 4.4 shows the experimental results of using the array representation with KNN during Evolutionary Training Process. When using the single constructed feature for classification, there is 1 out of the 14 datasets with improved classification performance. When we use KNN, the number is 6 out of 14. When we use NB, there are 4 out of the 14 datasets with better performance compared with the original classification performance.

When we combine the original features and the constructed feature, the number of datasets with improved classification performance is 6 out of 14. When using KNN, the number is decreased to 3. When we use NB, the number of datasets with improved classification performance is 2 out of 14.

Using the NB during the Evolutionary Training Process

Table 4.5 shows the experimental result of using the array representation with NB. When we use the single constructed feature and DT, there is no datase where the classification performance is improved. When we use KNN, there are 5 out of the 14 datasets. When

Table 4.3: Result of array representation, using DT classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
australian	14	Org	85.99				70.05				85.51			
		CF	84.54	84.54	5.55E-16	-	82.61	82.61	2.22E-16	+	53.62	53.62	4.44E-16	-
		CFOrg	85.02	85.02	7.77E-16	-	70.05	70.05	3.33E-16	=	85.51	85.51	5.55E-16	=
german	24	Org	72.67				68.33				73.33			
		CF	69	69	3.33E-16	-	66	66	0E0	-	70.33	70.33	1.11E-16	-
		CFOrg	71	71	5.55E-16	-	68	68	2.22E-16	-	73	73	1.11E-16	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	61.81	61.81	3.33E-16	-	61.81	61.81	3.33E-16	+	48.08	48.08	5.55E-16	-
		CFOrg	78.02	78.02	4.44E-16	+	56.59	56.59	1.11E-16	=	52.2	52.2	0E0	=
ionosphere	34	Org	86.67				83.81				28.57			
		CF	35.24	35.24	0E0	-	71.43	71.43	3.33E-16	-	73.33	73.33	4.44E-16	+
		CFOrg	86.67	86.67	3.33E-16	=	83.81	83.81	4.44E-16	=	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	80	80	3.33E-16	-	80	80	3.33E-16	+	80	80	3.33E-16	-
		CFOrg	50	50	0E0	-	70	70	1.11E-16	=	90	90	8.88E-16	=
madelon	500	Org	76.79				70.9				50.51			
		CF	58.33	58.33	4.44E-16	-	53.85	53.85	6.66E-16	-	49.62	49.62	6.11E-16	-
		CFOrg	76.79	76.79	9.99E-16	=	70.9	70.9	0E0	=	50.51	50.51	0E0	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	88.15	88.15	0E0	-	88.27	88.27	4.44E-16	-	87.16	87.16	2.22E-16	-
		CFOrg	91.36	91.36	3.33E-16	=	94.94	94.94	4.44E-16	=	90.99	90.99	5.55E-16	=
multiplefeatures	649	Org	98.1				98.63				81.9			
		CF	91.57	91.57	0E0	-	89.07	89.07	7.77E-16	-	83.9	83.9	6.66E-16	+
		CFOrg	98.13	98.13	8.88E-16	+	98.63	98.63	6.66E-16	=	81.9	81.9	1.11E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	58.74	58.74	5.55E-16	-	58.04	58.04	3.33E-16	-	59.44	59.44	4.44E-16	+
		CFOrg	69.23	69.23	4.44E-16	-	83.92	83.92	5.55E-16	=	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-
		CFOrg	93.31	93.31	5.55E-16	=	96.44	96.44	8.88E-16	+	91	91	3.33E-16	+
sonar	60	Org	71.43				76.19				53.97			
		CF	53.97	53.97	6.66E-16	-	55.56	55.56	5.55E-16	-	49.21	49.21	6.11E-16	-
		CFOrg	71.43	71.43	3.33E-16	=	76.19	76.19	0E0	=	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	62.01	62.01	4.44E-16	-	62.8	62.8	4.44E-16	-	61.81	61.81	6.66E-16	-
		CFOrg	84.84	84.84	5.55E-16	+	84.06	84.06	6.66E-16	+	83.27	83.27	4.44E-16	=
wbcd	30	Org	92.98				92.98				90.64			
		CF	89.47	89.47	5.55E-16	-	89.47	89.47	5.55E-16	-	61.99	61.99	0E0	-
		CFOrg	97.08	97.08	6.66E-16	+	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	95.06	95.06	11.1E-16	+	96.3	96.3	5.55E-16	+	54.32	54.32	3.33E-16	-
		CFOrg	97.53	97.53	11.1E-16	+	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=

Table 4.4: Result of array representation, using KNN classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	93.21	93.21	8.88E-16	-	85.75	85.75	2.22E-16	-	93.21	93.21	8.88E-16	-
		CFOrg	95.59	95.59	5.55E-16	=	94.46	94.46	6.66E-16	=	94.46	94.46	6.66E-16	=
australian	14	Org	85.99				70.05				85.51			
		CF	83.57	83.57	4.44E-16	-	83.57	83.57	4.44E-16	+	53.62	53.62	4.44E-16	-
		CFOrg	83.57	83.57	4.44E-16	-	70.05	70.05	3.33E-16	=	86.47	86.47	4.44E-16	+
german	24	Org	72.67				68.33				73.33			
		CF	70.33	70.33	1.11E-16	-	67.33	67.33	4.44E-16	-	71.67	71.67	5.55E-16	-
		CFOrg	70.67	70.67	1.11E-16	-	67.67	67.67	3.33E-16	-	76	76	2.22E-16	+
hillvalley	100	Org	62.09				56.59				52.2			
		CF	59.07	59.07	4.44E-16	-	60.16	60.16	4.44E-16	+	48.08	48.08	5.55E-16	-
		CFOrg	89.29	89.29	5.55E-16	+	56.59	56.59	1.11E-16	=	52.2	52.2	0E0	=
ionosphere	34	Org	86.67				83.81				28.57			
		CF	35.24	35.24	0E0	-	39.05	39.05	55.5E-18	-	71.43	71.43	3.33E-16	+
		CFOrg	86.67	86.67	3.33E-16	=	83.81	83.81	4.44E-16	=	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	90	90	8.88E-16	=	80	80	3.33E-16	+	80	80	3.33E-16	-
		CFOrg	90	90	8.88E-16	=	70	70	1.11E-16	=	90	90	8.88E-16	=
madelon	500	Org	76.79				70.9				50.51			
		CF	59.74	59.74	2.22E-16	-	58.08	58.08	3.33E-16	-	49.49	49.49	0E0	-
		CFOrg	76.79	76.79	9.99E-16	=	70.9	70.9	0E0	=	50.51	50.51	0E0	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	88.4	88.4	0E0	-	88.4	88.4	0E0	-	87.04	87.04	8.88E-16	-
		CFOrg	91.36	91.36	3.33E-16	=	94.81	94.81	7.77E-16	-	90.99	90.99	5.55E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	41.26	41.26	4.44E-16	-	41.26	41.26	4.44E-16	-	58.04	58.04	3.33E-16	+
		CFOrg	73.43	73.43	4.44E-16	+	83.92	83.92	5.55E-16	+	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-
		CFOrg	93.31	93.31	5.55E-16	=	96.44	96.44	8.88E-16	+	90.79	90.79	4.44E-16	=
sonar	60	Org	71.43				76.19				53.97			
		CF	65.08	65.08	5.55E-16	-	61.9	61.9	4.44E-16	-	49.21	49.21	6.11E-16	-
		CFOrg	74.6	74.6	5.55E-16	+	77.78	77.78	4.44E-16	+	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	76.77	76.77	2.22E-16	-	77.76	77.76	5.55E-16	-	62.2	62.2	4.44E-16	-
		CFOrg	87.01	87.01	5.55E-16	+	84.06	84.06	6.66E-16	+	83.27	83.27	4.44E-16	=
wbcd	30	Org	92.98				92.98				90.64			
		CF	91.23	91.23	4.44E-16	-	94.15	94.15	11.1E-16	+	61.99	61.99	0E0	-
		CFOrg	95.91	95.91	6.66E-16	+	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	86.42	86.42	7.77E-16	-	88.89	88.89	6.66E-16	+	54.32	54.32	3.33E-16	-
		CFOrg	92.59	92.59	2.22E-16	+	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=
zoo	17	Org	93.33				80.95				98.1			
		CF	84.76	84.76	4.44E-16	-	85.71	85.71	4.44E-16	+	81.9	81.9	2.22E-16	-
		CFOrg	93.33	93.33	8.88E-16	=	80.95	80.95	5.55E-16	=	98.1	98.1	0E0	=

we use NB, there is only 1 out of the 14 datasets where the classification performance is improved.

When we combine the original features with the single constructed high-level feature and use DT, there are 6 out of the 14 datasets with improved classification performance. When we use KNN, there are 3 out of the 14 datasets with improved classification performance. When we use NB, there are 2 out of the 14 datasets with improved classification performance.

Table 4.5: Result of array representation, using NB classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	91.97	91.97	3.33E-16	-	91.74	91.74	7.77E-16	-	93.21	93.21	8.88E-16	-
		CFOrg	95.59	95.59	5.55E-16	=	94.46	94.46	6.66E-16	=	94.34	94.34	9.99E-16	-
german	24	Org	72.67				68.33				73.33			
		CF	70.67	70.67	1.11E-16	-	62.33	62.33	6.66E-16	-	70	70	1.11E-16	-
		CFOrg	72	72	4.44E-16	-	68	68	2.22E-16	-	71.67	71.67	5.55E-16	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	47.8	47.8	0E0	-	47.25	47.25	5.55E-16	-	48.08	48.08	5.55E-16	-
		CFOrg	62.09	62.09	3.33E-16	=	56.59	56.59	1.11E-16	=	53.02	53.02	4.44E-16	+
ionosphere	34	Org	86.67				83.81				28.57			
		CF	28.57	28.57	2.78E-16	-	27.62	27.62	2.78E-16	-	73.33	73.33	4.44E-16	+
		CFOrg	86.67	86.67	3.33E-16	=	83.81	83.81	4.44E-16	=	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	80	80	3.33E-16	-	80	80	3.33E-16	+	80	80	3.33E-16	-
		CFOrg	70	70	1.11E-16	-	70	70	1.11E-16	=	90	90	8.88E-16	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	88.4	88.4	0E0	-	87.78	87.78	2.22E-16	-	86.91	86.91	3.33E-16	-
		CFOrg	91.36	91.36	3.33E-16	=	94.94	94.94	4.44E-16	=	90.99	90.99	5.55E-16	=
multiplefeatures	649	Org	98.1				98.63				81.9			
		CF	82.13	82.13	2.22E-16	-	81.9	81.9	1.11E-16	-	83.83	83.83	1.11E-16	+
		CFOrg	98.1	98.1	11.1E-16	=	98.63	98.63	6.66E-16	=	81.9	81.9	1.11E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	40.56	40.56	55.5E-18	-	40.56	40.56	55.5E-18	-	59.44	59.44	4.44E-16	+
		CFOrg	71.33	71.33	0E0	=	83.92	83.92	5.55E-16	=	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-
		CFOrg	93.31	93.31	5.55E-16	=	96.44	96.44	8.88E-16	+	91	91	3.33E-16	+
sonar	60	Org	71.43				76.19				53.97			
		CF	52.38	52.38	5.55E-16	-	52.38	52.38	5.55E-16	-	50.79	50.79	6.66E-16	-
		CFOrg	66.67	66.67	3.33E-16	-	76.19	76.19	0E0	=	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	76.18	76.18	2.22E-16	-	70.08	70.08	0E0	-	63.39	63.39	6.66E-16	-
		CFOrg	84.65	84.65	5.55E-16	=	83.86	83.86	6.66E-16	=	82.68	82.68	5.55E-16	-
wbcd	30	Org	92.98				92.98				90.64			
		CF	67.84	67.84	5.55E-16	-	67.25	67.25	3.33E-16	-	62.57	62.57	0E0	-
		CFOrg	92.98	92.98	1.11E-16	=	92.98	92.98	1.11E-16	=	90.06	90.06	3.33E-16	-
wine	13	Org	87.65				76.54				82.72			
		CF	77.78	77.78	4.44E-16	-	79.01	79.01	6.66E-16	+	54.32	54.32	3.33E-16	-
		CFOrg	87.65	87.65	6.66E-16	=	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=
zoo	17	Org	93.33				80.95				98.1			
		CF	93.33	93.33	8.88E-16	=	73.33	73.33	4.44E-16	-	93.33	93.33	8.88E-16	-
		CFOrg	93.33	93.33	8.88E-16	=	80.95	80.95	5.55E-16	=	98.1	98.1	0E0	=

4.5.2 Using the Pair Representation

Using the DT during the Evolutionary Training Process

Table 4.6 shows the experimental results of using the pair representation with DT. The experimental results by using the only constructed feature based on the DT show that only the *wbcd* dataset where the classification performance can be improvedy. When using KNN, 5 out of the 14 datasets can improve the classification performance. The number of the datasets where classification performance can be improved by only using the constructed feature slightly decrease to 4 out of the 14 datasets when using the NB. An interesting observation here is that the results indicate the constructed feature by using DT as the training algorithm can perform better when using KNN and NB.

The experimental results by using the combination of constructed feature and original features are shown in Table 4.6. When we use DT, there are 12 out of the 14 datasets where the classification performance can be improved. When we use KNN, there are 5 out of the 14 datsets where the calssification performance can be improved. When we use NB, there are 5 out of the 14 datasets where the classification performance can be improved. Therefore, when using the combination of the constructed feature and original feature, the new datasets can gain better performance by using the DT as the testing classifier.

Using the KNN during the Evolutionary Training Process

Table 4.7 shows the experimental results for only using the pair representation with KNN. When using the DT, there is no dataset where classification performance can be improved by only using the constructed feature. When using the KNN and NB, the number of the datasets where the classification performance can be improved by the single constructed feature is 5 out of the 15 datasets.

Table 4.7 shows the experimental results for using the combination of the constructed feature with original features are shown. When using DT, 13 out of the 15 datasets where the classification performance can be improved. When using KNN, the number is 6 out of the 15. When using NB, there are 5 out of 15 datasets where the classification performance can be improved.

Using the NB during the Evolutionary Training Process

Table 4.8 shows the experimental results for using the pair representation with NB. When using the single constructed feature for the classification test, there are 2 out of the 15 datasets where the classification performance can be improved when using DT. The number is increased to 6 when using KNN. When we use NB, there are 4 out of the 15 datasets where the classification performance can be improved.

When using the combination of the constructed feature wiht original features, 10 out of the 15 datasets where the classification performance can be improved by using DT. When using KNN, there are 5 out of the 15 datasets where the classification performance can be improved. When using NB, there are 6 out of the 15 datsets where classification performance can be improved.

4.6 Discussion

In this section, we will give a discussion based on the experimental results above. Firstly, we will compare some of the experimental results with the previous experimental result of PSO based feature construction for binary classification. We found that the experimental results

Table 4.6: Result of pair representation, using DT classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
australian	14	Org	85.99				70.05				85.51			
		CF	74.4	57.5	8.45E-2	-	72.46	55.63	8.32E-2	-	60.39	53.78	98.7E-4	-
		CFOrg	89.37	86.14	58.3E-4	=	70.05	70.05	3.33E-16	=	86.96	85.15	74.7E-4	-
german	24	Org	72.67				68.33				73.33			
		CF	71.33	68.65	5.78E-2	-	70.33	57.85	14.7E-2	-	70.67	69.96	33.8E-4	-
		CFOrg	74.33	72.43	1.08E-2	=	68.67	67.81	37.2E-4	-	75.33	72.79	1.03E-2	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	60.44	50.82	2.24E-2	-	56.87	50.61	2.27E-2	-	52.2	48.2	1.01E-2	-
		CFOrg	70.33	60.43	2.55E-2	-	56.59	56.59	1.11E-16	=	53.3	52.12	58.2E-4	=
ionosphere	34	Org	86.67				83.81				28.57			
		CF	76.19	55.16	16.9E-2	-	75.24	57.58	14.8E-2	-	73.33	71.54	1.15E-2	+
		CFOrg	89.52	86.8	73.8E-4	=	84.76	83.89	25.8E-4	+	28.57	28.57	2.78E-16	-
lung	56	Org	90				70				90			
		CF	90	83.6	12.9E-2	-	100	69	21.3E-2	=	100	83.4	7.9E-2	-
		CFOrg	90	89.8	1.4E-2	=	70	70	1.11E-16	=	90	89.6	1.96E-2	=
madelon	500	Org	76.79				70.9				50.51			
		CF	60.64	51.58	2.91E-2	-	56.15	50.84	2.11E-2	-	49.74	49.51	10.9E-4	-
		CFOrg	76.92	76.8	1.79E-4	=	70.9	70.9	0E0	=	50.51	50.51	0E0	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	89.01	87.81	45.9E-4	-	88.89	87.81	39.4E-4	-	87.16	87	8.49E-4	-
		CFOrg	92.22	91.32	23.5E-4	=	94.94	94.92	4.01E-4	-	90.99	90.99	5.55E-16	=
multiplefeatures	649	Org	98.1				98.63				81.9			
		CF	89.6	82.36	1.44E-2	-	89.03	82.31	1.32E-2	-	83.83	81.81	29.4E-4	-
		CFOrg	98.13	98.1	46.7E-6	=	98.63	98.63	6.66E-16	=	81.9	81.9	1.11E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	64.34	54.78	7.88E-2	-	63.64	51.13	8.94E-2	-	60.84	59.51	54.6E-4	+
		CFOrg	74.13	71.38	39.2E-4	=	83.92	83.92	5.55E-16	=	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.88	55.6E-4	-	89.96	89.55	2.81E-2	-
		CFOrg	93.51	93.26	24.2E-4	=	96.44	96.41	7.67E-4	+	92.89	90.85	49.4E-4	=
sonar	60	Org	71.43				76.19				53.97			
		CF	65.08	50.1	4.42E-2	-	63.49	50.51	4.69E-2	-	50.79	48.35	1.59E-2	-
		CFOrg	71.43	70.73	2.06E-2	-	77.78	76.54	73E-4	+	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	76.38	67.18	3.46E-2	-	74.02	66.08	3.32E-2	-	62.8	61.92	27E-4	-
		CFOrg	86.81	85.04	65.1E-4	+	84.06	84	8.84E-4	+	83.46	83.25	9.02E-4	=
wbcd	30	Org	92.98				92.98				90.64			
		CF	93.57	74.75	14.4E-2	-	94.15	73.39	14.5E-2	-	62.57	61.93	29.2E-4	-
		CFOrg	96.49	93.22	68.2E-4	+	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	83.95	72.12	7.05E-2	-	83.95	72.72	7.23E-2	-	54.32	52.62	92E-4	-
		CFOrg	91.36	87.9	85.5E-4	+	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=

Table 4.7: Result of pair representation, using KNN classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	93.33	91.87	2.66E-2	-	93.21	88.3	3.42E-2	-	93.21	93.21	8.88E-16	-
		CFOrg	95.59	95.59	5.55E-16	=	94.46	94.46	6.66E-16	=	94.57	94.55	5.54E-4	+
australian	14	Org	85.99				70.05				85.51			
		CF	81.16	57.2	8.58E-2	-	80.68	55.18	8.17E-2	-	60.39	53.65	2.36E-2	-
		CFOrg	88.89	86.06	43.2E-4	=	70.05	70.05	3.33E-16	=	86.47	85.51	66.9E-4	=
german	24	Org	72.67				68.33				73.33			
		CF	71.67	69.33	1.34E-2	-	71	59.55	12.6E-2	-	70.67	69.86	49.9E-4	-
		CFOrg	75	72.17	1.03E-2	-	68.67	67.81	38.3E-4	-	74.67	72.47	1.09E-2	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	54.12	50.38	1.7E-2	-	55.77	50.58	2.67E-2	-	53.02	48.15	1.12E-2	-
		CFOrg	74.73	61.09	2.95E-2	-	56.59	56.59	3.85E-4	=	53.57	52.2	39.6E-4	=
ionosphere	34	Org	86.67				83.81				28.57			
		CF	82.86	56.99	15.7E-2	-	78.1	56.55	14.7E-2	-	75.24	71.81	1.31E-2	+
		CFOrg	90.48	86.99	1.14E-2	=	84.76	83.94	33E-4	+	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	90	79.4	22.8E-2	-	100	65	20.4E-2	=	100	81.8	7.12E-2	-
		CFOrg	90	89.8	1.4E-2	=	70	70	1.11E-16	=	90	89.6	1.96E-2	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	89.01	87.8	52.7E-4	-	89.26	87.9	53.8E-4	-	87.16	86.98	7.47E-4	-
		CFOrg	92.35	91.29	37E-4	=	95.19	94.92	6.05E-4	-	90.99	90.99	5.55E-16	=
multiplefeatures	649	Org	98.1				98.63				81.9			
		CF	89.43	82.25	1.12E-2	-	88.97	82.23	1.02E-2	-	82	81.76	4.46E-4	-
		CFOrg	98.13	98.1	66.7E-6	=	98.63	98.63	6.66E-16	=	81.9	81.9	1.11E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	65.73	57.96	5.35E-2	-	62.24	55.47	7.57E-2	-	60.84	59.37	52.8E-4	+
		CFOrg	76.22	71.43	68.5E-4	=	83.92	83.92	5.55E-16	=	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.75	1.44E-2	-	89.96	89.95	2.93E-4	-
		CFOrg	94.14	93.24	35.1E-4	=	96.44	96.39	8.93E-4	+	92.68	90.97	54E-4	+
sonar	60	Org	71.43				76.19				53.97			
		CF	58.73	50.03	4.11E-2	-	63.49	50.38	5.4E-2	-	50.79	47.81	1.21E-2	-
		CFOrg	73.02	71.17	1.43E-2	=	79.37	76.54	1.2E-2	+	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	74.02	66.33	3.07E-2	-	73.03	65.45	2.83E-2	-	62.8	61.89	23.3E-4	-
		CFOrg	86.42	84.92	64.9E-4	+	84.06	83.96	9.81E-4	+	83.46	83.24	7.51E-4	-
wbcd	30	Org	92.98				92.98				90.64			
		CF	92.4	70.21	15E-2	-	91.81	69.06	15.7E-2	-	62.57	61.98	29.8E-4	-
		CFOrg	97.08	93.25	82E-4	+	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	86.42	71.93	7.9E-2	-	86.42	73.7	7.82E-2	-	54.32	52.89	67E-4	-
		CFOrg	90.12	87.83	60.5E-4	=	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=
zoo	17	Org	93.33				80.95				98.1			
		CF	91.43	83.89	4.38E-2	-	90.48	82.88	3.35E-2	+	87.62	81.92	2.01E-2	-
		CFOrg	95.24	93.52	53.9E-4	+	80.95	80.95	5.55E-16	=	99.05	98.06	37.9E-4	=

Table 4.8: Result of pair representation, using NB classifier

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	93.33	90.7	3.5E-2	-	93.33	88.84	3.63E-2	-	93.21	93.21	8.88E-16	-
		CFOrg	95.59	95.59	5.55E-16	=	94.46	94.46	6.66E-16	=	94.8	94.56	6.65E-4	+
australian	14	Org	85.99				70.05				85.51			
		CF	78.74	62.11	7.63E-2	-	77.29	61.73	6.88E-2	-	55.07	53.79	35.6E-4	-
		CFOrg	88.41	86.09	43.2E-4	=	70.05	70.05	3.33E-16	=	86.96	85.25	90.2E-4	-
german	24	Org	72.67				68.33				73.33			
		CF	70.67	69.36	1.63E-2	-	70.33	58.59	13.5E-2	-	70.33	69.9	36.1E-4	-
		CFOrg	73.67	72.13	1.34E-2	-	68.67	67.81	32.8E-4	-	76.33	72.79	1.21E-2	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	54.12	50.38	2.02E-2	-	54.95	50.48	2.08E-2	-	53.57	49.93	2.18E-2	-
		CFOrg	67.03	61.62	1.57E-2	-	56.59	56.59	1.11E-16	=	53.57	52.36	53E-4	+
ionosphere	34	Org	86.67				83.81				28.57			
		CF	76.19	56.27	18.1E-2	-	72.38	58.59	16E-2	-	78.1	71.77	1.34E-2	+
		CFOrg	89.52	86.9	72.7E-4	+	84.76	83.85	18.7E-4	=	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	90	75.2	23.7E-2	-	100	67.8	24.1E-2	=	100	83.4	7.9E-2	-
		CFOrg	90	89	5E-2	=	70	70	1.11E-16	=	90	89.8	1.4E-2	=
madelon	500	Org	76.79				70.9				50.51			
		CF	56.79	51.09	2.42E-2	-	53.59	50.47	1.83E-2	-	49.74	49.51	12.3E-4	-
		CFOrg	76.79	76.79	9.99E-16	=	70.9	70.9	0E0	=	50.51	50.51	0E0	=
movementlibras	90	Org	91.36				94.94				90.99			
		CF	89.01	87.83	53.1E-4	-	89.01	87.9	47.6E-4	-	87.16	86.99	9.24E-4	-
		CFOrg	92.35	91.35	26.1E-4	=	95.19	94.94	4.61E-4	=	90.99	90.99	5.55E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	59.44	56.36	7.02E-2	-	60.14	55.12	7.78E-2	-	60.84	59.38	46E-4	+
		CFOrg	71.33	71.3	19.6E-4	=	83.92	83.92	5.55E-16	=	42.66	42.66	55.5E-18	=
semeion	265	Org	93.31				96.23				90.79			
		CF	89.96	89.96	5.55E-16	-	89.96	89.96	5.55E-16	-	89.96	89.9	24E-4	-
		CFOrg	98.54	93.46	90.3E-4	=	96.44	96.43	5.68E-4	+	92.68	91.03	56.4E-4	+
sonar	60	Org	71.43				76.19				53.97			
		CF	65.08	48.76	4.67E-2	-	73.02	50.25	6.08E-2	-	50.79	47.94	1.27E-2	-
		CFOrg	74.6	71.52	49.3E-4	=	77.78	76.44	58.2E-4	+	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	77.17	68.28	4.62E-2	-	76.38	67.28	4.06E-2	-	62.8	62	21.7E-4	-
		CFOrg	86.22	84.74	60.4E-4	=	84.06	83.96	9.84E-4	+	83.66	83.13	30.4E-4	-
wbcd	30	Org	92.98				92.98				90.64			
		CF	94.15	72.65	19.3E-2	-	95.32	70.4	19.9E-2	-	62.57	62.02	33.9E-4	-
		CFOrg	95.91	93.25	72.3E-4	+	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	95.06	74.62	5.69E-2	-	96.3	75.23	6.55E-2	=	54.32	52.86	68.5E-4	-
		CFOrg	92.59	87.95	97.4E-4	+	76.54	76.54	3.33E-16	=	82.72	82.72	4.44E-16	=
zoo	17	Org	93.33				80.95				98.1			
		CF	91.43	84.02	3.21E-2	-	88.57	80.65	3.2E-2	=	88.57	81.26	2.48E-2	-
		CFOrg	95.24	93.37	26.7E-4	=	80.95	80.95	5.55E-16	=	99.05	98.17	25.8E-4	+

we get for the second objective is worse than the previous experimental results when only using the single constructed high-level feature. During the experiments, we found that the newly constructed high-level feature is mathematically significantly huge, for example 1.5 * 10E150. Therefore, we look back at the training process. The training process consists two major parts, the feature selection and functional operator selection. In all the dataset, there is no such significant huge number in any features. Therefore, those huge number in our constructed feature is not caused in the feature selection since the feature selection cannot affect the constructed feature by select a particular feature. Secondly, Therefore, we will also investigate the potential causes for the huge number. In particular, we will investigate the operator selections for each experiment.

In order to investigate the operator selection for the proposed PSOFC, we collect and count the average number of the four operators being used during 50 runs of each dataset. In Figure 4.2, Figure 4.3 and Figure 4.4, we show the average percentage of the four operators used for the final constructed high-level feature by using the array representation. Figure 4.5, Figure 4.6 and Figure 4.7 shows the average percentage of the four operators being used for the final constructed high-level feature. The blue, red, green and purple bars represent the operator "+", "-", "*" and "/" respectively.

4.6.1 The operator selection when using array representation

Seeing from the Figure 4.2, Figure 4.3 and Figure 4.4, we found that the operator "+" get the least chance to be selected. The operator "-" and "*" has the better chance to be selected compared with other two operators. By reviewing the operators selection we describe in the previous section, we found that the values for selecting operator "-" and "*" are around 0.5 from 0.0 to 1.0. Therefore, we can make the conclusion that during the PSO, most of the particles are moving around the center of the searching space since we defined that the scope is from 0.0 to 1.0 inclusive on each dimension for the operator selection. Since the operator "*" get more chance to be selected for using array representation, it is understandable that the constructed feature will produce some significant huge number sometimes.

Besides, there are some datasets whose operator selection produce interesting results. In the Figure 4.2, we can find that the operators used for the dataset semeion is very even. The average number of appearance for each of the the four operators are almost 25 %. In Figure 4.3, the datasets of *vehicle*, *ionosphere* and *australian* does not use the operator "+" at all and the dataset wine does not use the operator "/". In Figure 4.4, the *wine* dataset only uses the / operators and the *vehicle* dataset only uses the "*" operators. Therefore, we found that our operator selection strategy in the array representation is relatively fair but the operator "-" and "*" are preferred by the PSO.

4.6.2 The Operator selection when using pair representation

In Figure 4.5, Figure 4.6 and Figure 4.7, we calculate and present the average percentage of the appearance of each of the four operators used by the pair representation with DT, KNN and NB during the ETP. The boundary for the each operator selection dimension in the pair representation is from 0.5 to 1.0 inclusive. We found that the operator "+" is the most likely operators which can get selected by PSO. The boundary for the operators "+" is from 0.5 to 0.625 exclusive. By comparing the operator selection using the array representation, we can make similar conclusion which is that the more close to 0.5, the higher the chance the operator can be selected. Also, When we use the pair representation, there is no such operators being abandoned by any datasets. However, the high percentage of the "+" operator makes us concerning the fairness of the operator selection for using the pair representation.

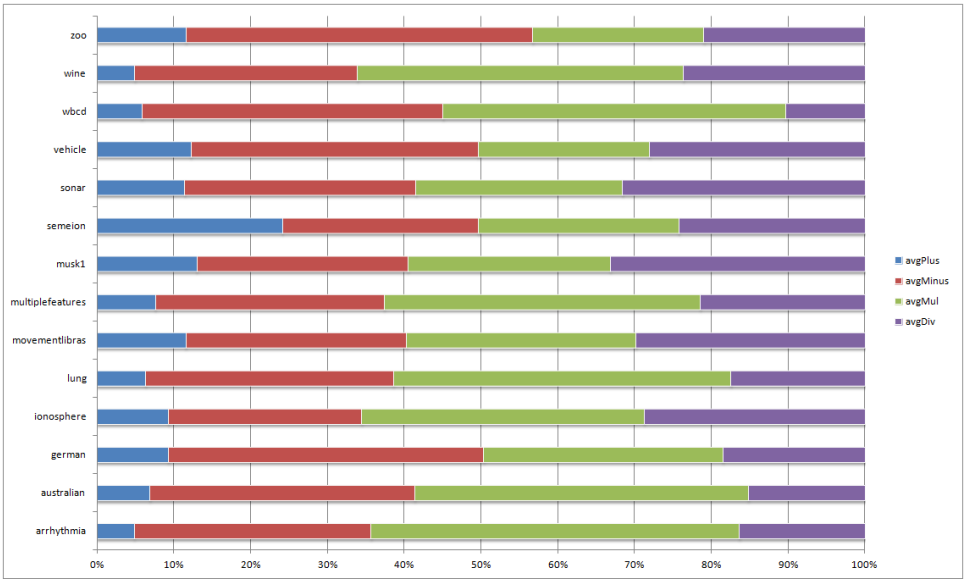

Figure 4.2: ARRAY REPRESENTATION WITH DT

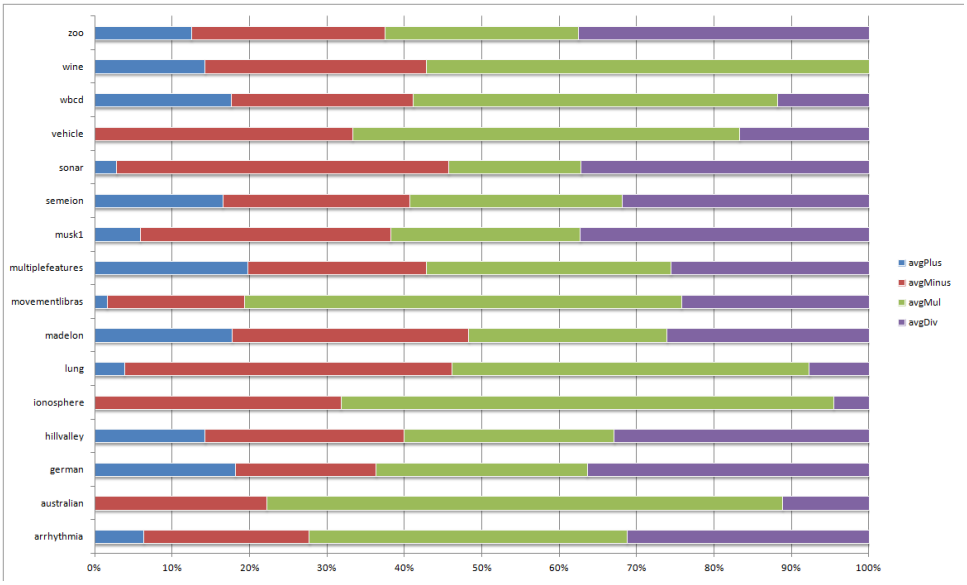

Figure 4.3: ARRAY REPRESENTATION WITH KNN

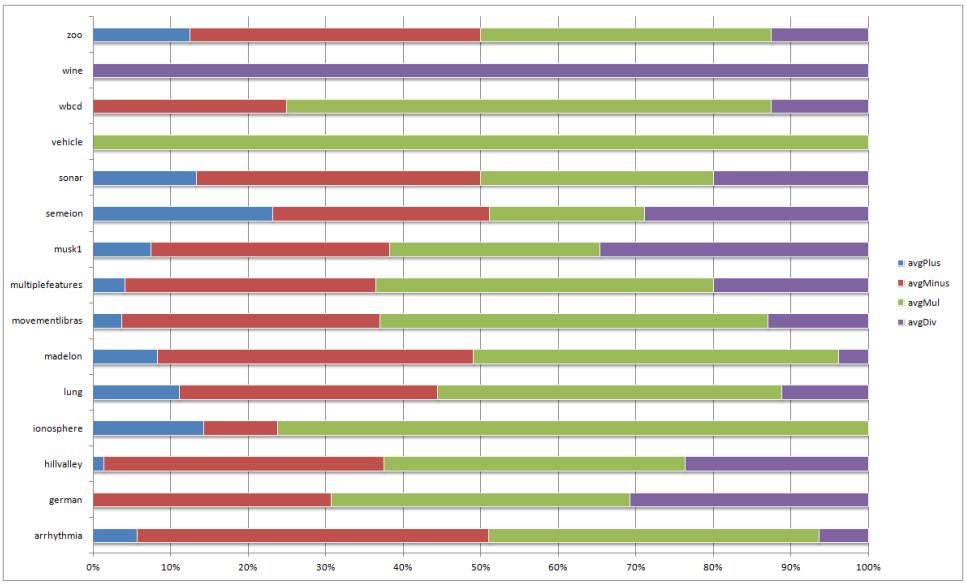

Figure 4.4: ARRAY REPRESENTATION WITH NB

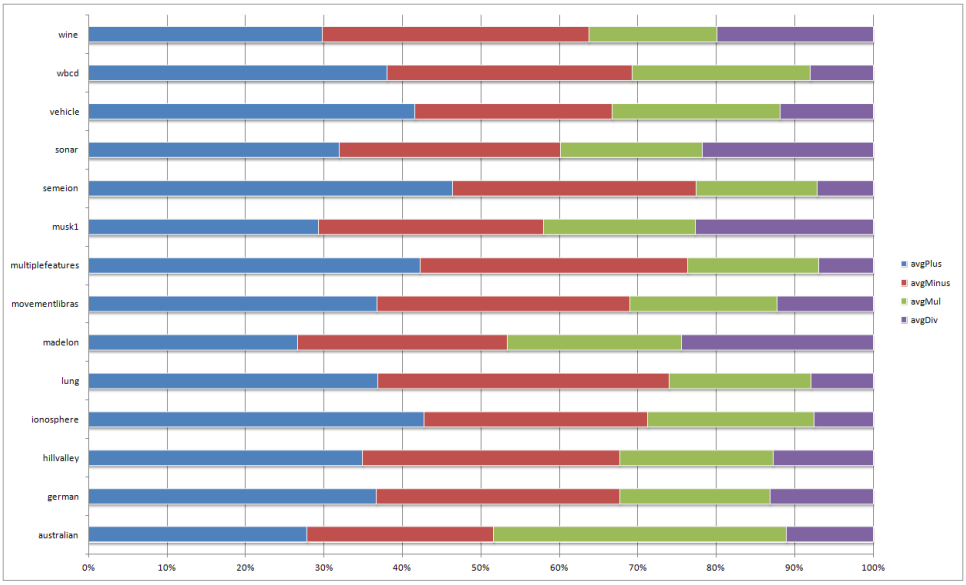

Figure 4.5: PAIR REPRESENTATION WITH DT

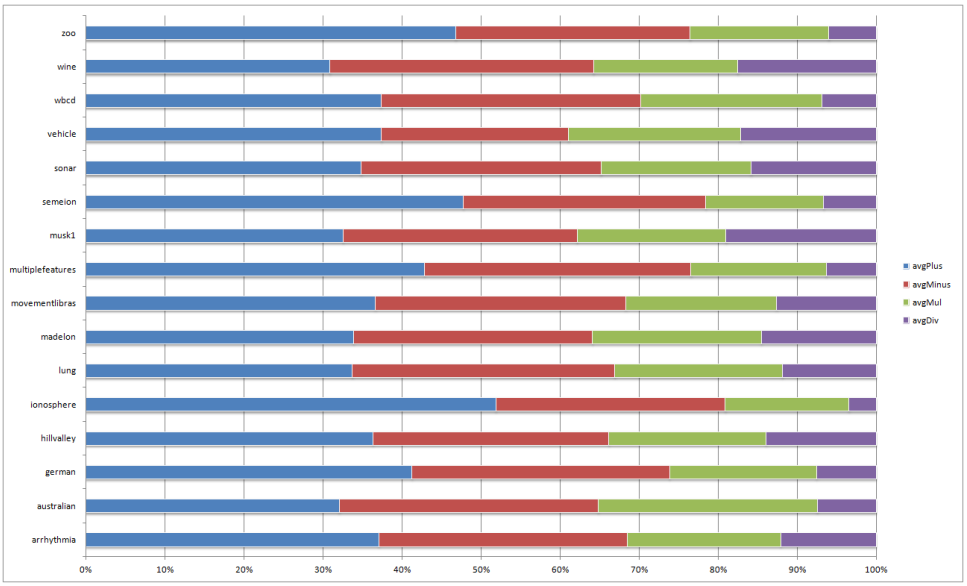

Figure 4.6: PAIR REPRESENTATION WITH KNN

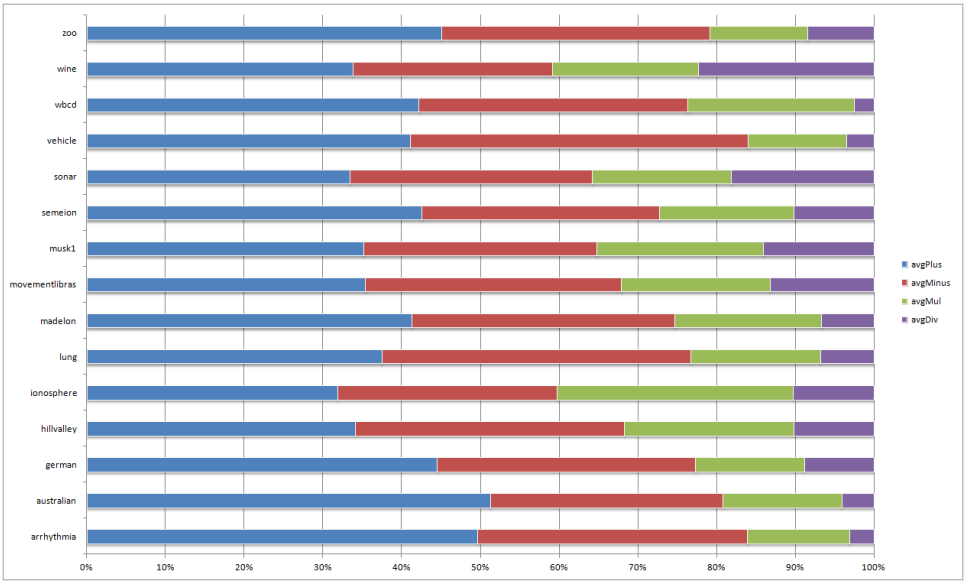

Figure 4.7: PAIR REPRESENTATION WITH NB

4.7 Conclusion

In this chapter, we propose a way for feature construction using PSO and targeting on multi-class classification. The main difference between the proposed method in this chapter and objective one is that DT, KNN and NB classification algorithms are used during the evolutionary training process for PSOFC. The experimental results show that the proposed method can construct a high-level feature. The classification performance can be improved for some datasets if we combine the constructed feature with the original features.

We investegate the function operator selection for the proposed PSOFC method. We find that the operators with the range close to 0.5 are more likely to be used. Therefore, we will keep exploring a optimised way to do the function operator selection in the future plan.

Chapter 5

Multiple Features Construction Using PSO

5.1 Introduction

In the second objective, we have developed a PSO based approach to construct a single high-level feature for multi-class classification problems. The experimental results show that the single high-level feature cannot improve the classification performance in many datasets. When the number of original features in the dataset is large, we may need multiple new high-level features that are collaborated to improve the classification performance. In this chapter, we will develop a new PSO based approach to constructing multiple new high-level features for classification.

5.2 Chapter Goals

In this chapter, our overall goal is to develop a new PSO based approach to constructing multiple high-level features for classification. We developed the multi-feature construction approach based on the array representation from the first objective. We expect that each high-level features can distinguish one class from all other classes in the dataset. Therefore the number of constructed features equals to the number of classes. The new set of high-level constructed features are used for classification and the classification performance is measured by Decision Tree classifier (DT), K-nearest neighbour classifier (KNN) and Naive Bayes classifier (NB).

Specially, we will investigate:

- whether the newly constructed high-level features can improve the classification performance.
- whether the newly constructed high-level features with original features together can improve the classification performance.

5.3 Proposed Approach

In this section, the proposed approach will be presented. The developed approach is different from the previous approaches for single feature construction we introduced in the first and second objective. In order to construct multiple features, we develop the approach based on the PSO with n-swarms, where n is the number of classes of the dataset. Each

swarm will construct one high-level feature for each class. For example, if a dataset contains three classes, the developed approach will produce three new high-level features.

5.3.1 The Overall Design

In Figure 5.1, the overall process of the multi-feature construction using PSO is shown. The objective of this process is to produce a new set of the constructed high-level features. The evolutionary training process is the major difference from the first and second objectives. The first and second objectives use one single swarm to construct one single high-level feature. In order to construct multiple features, multiple swarms are used. The original dataset is splitted into a training set and a testing set at the beginning. The training set is going to be used as the input of the multi-feature construction algorithm to construct high-level multiple features. According to the number of classes in the dataset, n swarms will be initialised and the training set will be feed into the n swarms for the evolutionary training of multi-Feature Construction using PSO (MPSOFC) where n is the number of classs. All the swarms use the same training set and each swarm will produce one high-level feature regarding of one classes. There will be n new high-level features constructed. The following step is to transform the original dataset to a new dataset. We transform the testing set into a new set of data based on the constructed features. Now, a new transformed dataset based on new high-level constructed features is produced. The next stage is to test the classification performance of the newly constructed features, i.e. the transformed dataset..

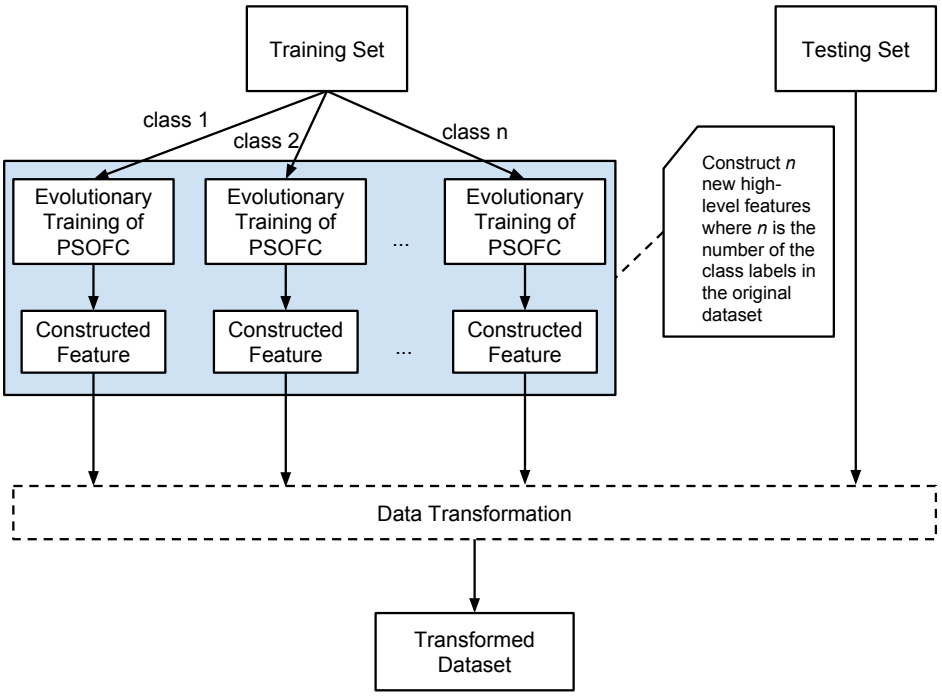

Figure 5.1: Overall Process for Multi-Feature Construction Using PSO

5.3.2 Multi-Feature Construction

Constructing multiple new high-level features is considered as a more difficult task than constructing single new high-level features. In this objective, the number of constructed features is the number of classes. The advantages are:

- The number of newly constructed high-level features is easily defined since it equals to the number of the class labels.
- The boundary for each constructed high-level feature is clearly defined since the feature is to classify whether the instance belongs to a particular class or not. Therefore, the multi-feature construction objective is divided into several single feature construction tasks. Each single featrue construction will be conducted by an individual swarm.
- Since the objective for each constructed high-level feature is to identify whether the instance belongs to the class or not and each class is distinguish to each other, the constructed high-level features set can be better protected from introducing new noise features.

According to above, the new approach will need to measure how good the constructed feature is to the corresponding class during the evolutionary process of PSO. The following section will discuss about the fitness function of the developed approach.

5.3.3 The Fitness Function

In order to measure the goodness of a candidate constructed feature, we will use a class interval entropy based algorithm for the fitness function in this objective and the original idea is from [15]. Since each constructed feature corresponds to one class, we intend to calculate the goodness of the candidate construct feature based on how good it is for classifying whether the instance belongs to the corresponding class.

5.3.4 The Class Interval

A class interval is defined with a lower value and an upper value denoted as $(lower, upper)$. In Figure 5.2, an example feature x for an interval I to class "+" is presented. The example shows that there are two class labels ("+", "−") in the dataset in total. An interval I is defined by lower and upper boundary. Within the interval, majority of instances belong to class "+".

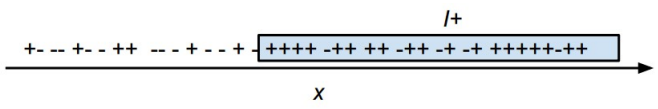

Figure 5.2: an Example Feature x and an Interval I for Class +

When we define the class interval for a particular class in a dataset based on a particular feature, three different situations can be found: a) the interval contains purely the corresponding class, which is show in Figure 5.3. b) the interval contains a mixture of majority of the corresponding class and some other classes, which is show in Figure 5.2. c) the interval contains a mixture of the corresponding class and other classes. The number of the corresponding class cannot definitely dominate the class interval. Specifically, the situation c is easier to happen for multi-class dataset which Figure 5.4 is an example for such situation.

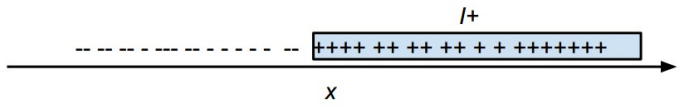

Figure 5.3: an Example Interval Contains Purely the Corresponding Class

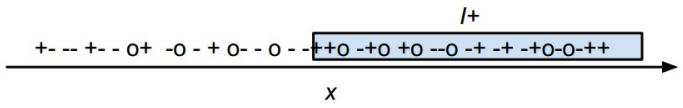

Figure 5.4: an Example Interval Contains Noiesy Class Labels

5.3.5 The Overall Process of the Fitness Function

In order to construct a new high-level feature for a specific class in a dataset, we will use that construct high-level feature as an input to find and cosntruct the class interval and calculate the purity of the class interval for the specific class. The overall process of the fitness function for construct a single high-level feature for a specific class is presented in Figure 5.5. The fitness function of the evolutionary process starts with decoding the selected features and the corresponding evolved operators from the particle to obtain the constructed feature. We use the array representation developed in the first objective. The decode method can be found in Section 3.3.2 of Chapter 3. The next step is to transform the Training set into SubTraining set and a SubTesting set. There is a single high-level constructed feature in the Transformed Training and Testing Set. The Transformed Training Set is used to by the Class Interval Finding method to find the corresponding class label i. Class label i is predefined at beginning of the evolutionary, where i is one of the class labels from the dataset. For each swarm, the class label i is fixed for all the iterations. There cannot be any two swarms obtain same class label i since each swarm is to construct one high-level feature corresponding to one class. The output from the Class Interval Finding function will be an interval ($lower$, $upper$) which will be defined in Section 5.3.6. A Class Purity Calculation function (described in Section 5.3.7) then will conduct the class interval calculation based on the discovered class interval and the Transformed Testing Set. The Class Interval Purity is used as the goodness of the constructed feature.

5.3.6 The Class Interval Finding

The function for finding a class interval towarding to a specific class label has been presented in Algorithm 3. The algorithm needs two inputs, one is the transformed data set called $data$ and the other one is the class label c for the interval. In the transformed data set, it contains one constructed feature and the corresponding class label for each instance in the transformed dataset. The algorithm will start with initialise two arrays $left$ and $right$. The $left$ is to select the lower boundary of the class interval and the $right$ is to select the upper interval. The array $left$ and $right$ contains one initial value the $POSITIVE\_INFINITY$ and the $NEGATIVE\_INFINITY$. The next step is to construct the array $left$ and $right$ to construct the class interval. For each instance in $data$, $data[i]$ is the ith constructed feature and $data[i].class$ is the corresponding class label.

In order to find the lower and upper boundaries for the class interval according to [15], the algorithm finds 99% of instances with class label c that falls in the middle of the range of all the instances in $data$. The 1% instances with extreme values is excluded. The process for this is shown in the two $IF - ELSE$ clause. The two array $left$ and $right$ can grow up to

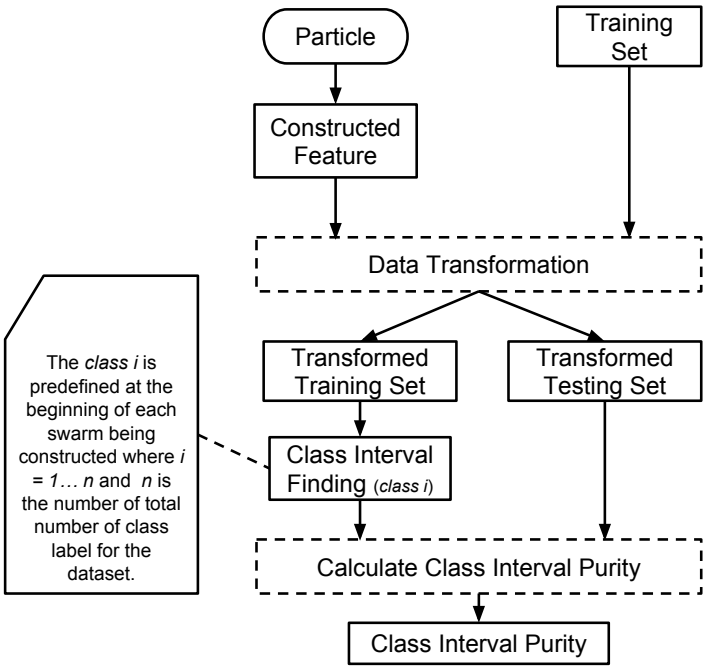

Figure 5.5: Construct a Single High-Level Feature for a Specific Class Label

data.classes.size/200 elements where *data.classes.size* is the total number of instances with the corresponding class label c. "*data.classes.size*/200" is to define the number of 0.5% of instances and therefore, the $left$ and *right* will exclude the 1% instance with extreme value. When the size of $left$ and *right* exceeds the number *data.classes.size*/200, the maximum and the minimum value of the dataset will be deducted from $left$ and *right*. The $FOR - EACH$ loop will finally fill up the $left$ and *right* array. The class interval will be constructed based on the max value in $left$ as lower value and minimum value in *right* as the upper value.

5.3.7 Calculation of Purity of the Interval

The algorithm for calculating the fitness of the single candidate constructed high-level feature is shown in Algorithm 4. The algorithm takes three inputs, the transformed training set as *dataTrain*, the transformed testing set as *dataTest* and the target class label for the constructed feature as c. The first step of the class interval purity calculation is to initialise a class interval for class c based on the training data set *dataTrain* and a empty set c' for storing all the instance in the *dataTest* falling into the interval. The process to construct the set c' is the first *FOR* loop. The second loop in the algorithm is to calculate the percentages of each class from the original dataset which is falling into the class interval of class c.

The last $FOREACH$ loop is to calculate the class interval purity for class c based on the Equation 5.1, where C is the the set of class labels in original dataset, I_c is the class interval for a particular class label c and X is one of the instance falling into the class interval I_c

$$H(C) = -\sum_{i=1}^{L} p(c_i) \log_b p(c_i) \tag{5.1}$$

The equation for class interval purity is based on Shannons Entropy [15]. *Shannons Entropy* is one of the most common way to measuring entropy of an object. The original *Shannons Entropy* can be found in Equation 5.2.

```
Data: data //the transformed training dataset which contains a single constructed
      feature
Data: c //the target class label
Result: (l, u) //the class interval for class c based on the transformed set data
left ← {POSITIVE_INFINITY} ;
right ← {NEGATIVE_INFINITY} ;
for i ← 0 to data.size do
    if data[i].class == c then
        if left.size >= data.classes.size/200 then
            left ← left − left.Max;
        end
        left ← left ∪ data[i];
    end
    if data[i].class == c then
        if right.size >= data.classes.size/200 then
            right ← right − right.Min;
        end
        right ← right ∪ data[i];
    end
end
(l, u) ← (left.Max, Right.min);
return (l, u);
```

Algorithm 3: Class Interval Finding Algorithm

$$H(I_c) = -\sum_{c \in C} p(c|X \in I_c) \log_2 p(c|X \in I_c) \tag{5.2}$$

In our project, the entropy of the class interval is to represent the purity of the class interval. The class interval with higher purity can be referred to Figure 5.3 in Section 5.3.6 and the entropy should be lower. And the class interval with lower purity can be referred to Figure 5.4 and the entropy is higher.

5.4 Experimental Design

We conduct a set of experiments using different benchmark datasets to evaluate the performance of the proposed PSO based multi-feature construction approach. The used benchmark datasets are chosen from UCI machine learning repository [2]. The details of the 15 datasets are shown in Table 5.1. The number of features is ranged from 13 to 649. The number of instances is from 32 to 2000. The number of classes is ranged from 2 to 16. The instances in each dataset will be divided into 70% as a training set and 30% as a testing set [6, 20].

The parameters of the proposed algorithm are set as follows [17]: $w = 0.7298$, $c_1 = c_2 = 1.49618$, $v_{max} = 0.1$, $v_{min} = -0.1$ and $domain_{min} = 0.0$ as the minimum position value of a particle in each dimension, $domain_{max} = 1.0$ as the maximum position value of a particle in each dimension, population size is 30 and the maximum number of iterations is 100. The fully connected topology is used in the experiments. For each dataset, the proposed algorithm has been independently run for 50 times. A set of high-level newly constructed

Data: $dataTrain$ //the transformed training dataset which contains a single constructed feature
Data: $dataTest$ //the transformed testing dataset which contains a single constructed feature
Data: c //the target class label
Result: $fitness$ //the fitness value of the constructed feature
$interval \leftarrow findInterval(dataTrain, c)$;
$c' \leftarrow \{\}$;
for $i \leftarrow 0$ **to** $dataTest.size$ **do**
  if $dataTest[i] \in c$ **then**
    $c' \leftarrow c' \cup c$;
  end
end
foreach $label \in data.classes$ **do**
  if $dataTest[i] \in c$ **then**
    $p_{label} \leftarrow \frac{|\{i:i \in \{1,2,...,|c'|, c'|i| = label\}\}|}{|c'|}$;
  end
end
$fitness \leftarrow 0$;
foreach $label \in data.classes$ **do**
  $fitness \leftarrow fitness - p_{label} log(p_{label})$;
end
return $fitness$;

Algorithm 4: Fitness for Cosntructed Feature Calculation Algorithm

features will be produced in each run. The function operators set used in this chapter are "+", "-", "*" and "/" (protected division).

For the array, the value range for each operator are shown in the Table 5.2.

Three different classifiers (DT, KNN and NB) are used to conduct the classification performance tests to evaluate the performance of the newly multiple constructed high-level features.

For each dataset, we conduct a classification performance test with the three classifiers using the newly multiple constructed high-level features. Then we combine the set of newly multiple constructed features with the original features for classification. We perform a T-test to see whether the newly constructed features can improve the classification performance.

5.4.1 Cross Validation Experiments

We conduct a 10-folds cross validation experiment based on the experiment we talked above to avoid the problem caused by a small number of instances. Firstly, we split the original dataset into 10 equally sets. The cross-validation process will be then process 10 times. Each of the subset of the 10 sets will be used exactly once as the validation data. Then we will get 10 validation result and using the average value as a single estimation for the classification performance of the dataset. One of the advantages is that all the instances from the dataset will be used as training and testing data so that the overtraining problem can be minimized.

Table 5.1: Binary-Class Datasets

Dataset	No. of Features	No. of Classes	No.of Instances
Australian	14	2	690
German	24	2	1000
Ionosphere	34	2	351
WBCD	30	2	569
Hillvalley	100	2	606
Musk1	166	2	476
Semeion	256	2	1593
Sonar	60	2	608
Wine	13	3	178
Zoo	17	7	101
Vehicle	18	4	846
Lung	56	3	32
LibrasMovement	90	15	360
Arrhythmia	279	16	452
MultipleFeatures	649	10	2000

Table 5.2: Function Operator Selection

Operator	Array Rep
"+"	[0.0, 0.25)
"-"	[0.25, 0.5)
"*"	[0.5, 0.75)
"/"	[0.75, 1]

5.5 Experimental Results

The experimental results for the developed approach based on the array representation are shown in Table 5.3 and Table 5.4. In both tables, "All" means only the original features are used for classification; "CF" means only the multiple constructed features are used for classification; "CFOrg" means the combination of multiple constructed features and original features is used for classification. The symbol "Best", "Avg" and "Std" represent the best classification accuracy, the average classification performance and the standard deviation of the classification performance.

5.5.1 Experimental Results of 70/30 for Training/Testing

In Table 5.3, the experimental results for using the developed PSO based multi-feature construction approach. For the classification performance of using only the constructed features, the results are shown in the CF rows. When we use DT, there are only one out of the 15 datasets where the classification performance is better than only using the original features for the classification. When we use KNN, the number is increased to two out of the 15 datasets. When we use NB, the number is two out of the 15 datasets.

When we combined the constructed features and the original features for classification, the experimental results are shown in CFOrg rows. We found that there are two datasets getting better classification performance compared with only using the original features when using DT. When using KNN, the number is five out of the 15 datasets. When using NB, the number is reduced to four out of the 15 datasets.

One observation is that the best classification performance very differs from the average classification performance. For example, when using KNN for dataset of "ionosphere", the *avg* value for *CF* is 37.64 and the *best* value is 80. By reading the standard deviation

of CF from the experimental results, we can imply that the classification performance of each run is not stable at a small range of values. For the datasets with increased classification performance for any category with any classification algorithm using either only the constructed features or the combination of constructed features and original features, we found that the number of features is varied. When using DT, only the dataset with features number less than 20 (2 out of four datasets) can benefit from using the constructed fetures. When using KNN, 4 datasets (out of 7 datasets) with features number less than 50 can improve the classification performance by using the constructed features and one dataset (out of 3 datasets) with more than 200 features can benefit from using the constructed features. When using NB, 4 (out of 7 datasets) with features number less than 50 can be benefit from the constructed features. Three datasets (out of 5 datasets) can be benefit from using the constructed features. One dataset with more than 200 features can improve the classification performance by using the constructed featurs.

5.5.2 Experimental Results of 10-Folds Cross Validation

In Table 5.4, the experimental results of using 10 folds cross validation are shown. For the classification performance based on the constructed high-level features, there are one out of 14 datasets with the classification performance better than only using the original features when we use DT. When using KNN, there are two datasets out of the 14 datasets with better classification performance compared with the classification performance based on the original features. When using NB, the number is two out of the 14 datasets. When using NB, the number is three out of the 14 datasets.

When we combine the constructed features and the original features for classification, the experimental results are shown in CFOrg rows. When using DT, there are 4 out of the 14 datasets with better classification performance compared with the classification performance by only using the original features. When using KNN, the number increased to 8 out of the 14 datasets. When using NB, the number is 5 out of the 14 datasets.

We found that there are only four out of 14 datasets cannot improve the classification performance by using the constructed as the only features or part of the combination of constructed features and original features through any defined classification algorithm.

5.6 Discussion

In this section, we will discuss the experimental results. Firstly, we will investigate deeply at the quality of the multiple constructed features through discussing the classification performance by only using the multiple constructed features. Secondly, we will discuss whether the multiple constructed features are a strong set of new high-level features to replace the original features or they are better to be combined with the original features for classification. In order to examine the relationship between the multiple constructed features and the original features, we will discuss about the comparison of the experimental results of only using the multiple constructed features, and the combination dataset of original features and multiple constructed features. At last, we will discuss the advantages and disadvantages of the developed approach including the discussion of class interval finding method and the class interval purity calculating function.

5.6.1 The Quality of the multiple constructed Features

In Figure 5.6, we present the experimental result of average testing accuracy, the training accuracy and the classification performance using the original features only. The data in

Table 5.3: Result of the Original Design

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	93.33	88.57	3.32E-2	-	93.44	88.2	2.71E-2	-	87.22	86.44	47.6E-4	-
		CFOrg	95.7	95.53	25.3E-4	=	94.46	94.46	6.66E-16	=	94.46	94.2	14.7E-4	-
australian	14	Org	85.99				70.05				85.51			
		CF	85.99	56.35	14.3E-2	-	80.68	50.21	8.57E-2	-	81.64	52	9.87E-2	-
		CFOrg	86.96	85.05	3.97E-2	=	70.05	70.05	3.33E-16	=	86.47	85.83	47.9E-4	+
german	24	Org	72.67				68.33				73.33			
		CF	71	61.66	12.1E-2	-	70.67	60.13	9.6E-2	-	71	58.19	15.2E-2	-
		CFOrg	75.33	69.31	3.69E-2	-	68.67	67.99	26.6E-4	-	73	70.82	1.16E-2	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	55.77	52.55	2.16E-2	-	56.59	51.32	3.42E-2	-	48.35	48.12	18.5E-4	-
		CFOrg	56.59	53.15	1.87E-2	-	56.59	56.58	6.52E-4	=	53.02	52.61	21.5E-4	+
ionosphere	34	Org	86.67				83.81				28.57			
		CF	71.43	39.75	13.7E-2	-	80	37.64	14.5E-2	-	81.9	73.05	2.37E-2	+
		CFOrg	90.48	86.67	1.98E-2	=	84.76	83.98	36.6E-4	+	28.57	28.57	2.78E-16	=
lung	56	Org	90				70				90			
		CF	90	66.6	24.2E-2	-	90	56.8	23.1E-2	-	100	78	9.59E-2	-
		CFOrg	90	81	18.8E-2	-	70	70	1.11E-16	=	90	84.4	4.96E-2	-
movementlibras	90	Org	91.36				94.94				90.99			
		CF	89.26	87.88	58.1E-4	-	90.99	88.86	78E-4	-	88.4	87.74	22.1E-4	-
		CFOrg	90.86	88.68	92.4E-4	-	95.19	94.84	17.7E-4	-	91.48	91.17	10.8E-4	+
multiplefeatures	649	Org	98.1				98.63				81.9			
		CF	90.47	85.87	2.29E-2	-	93.73	86.1	3.16E-2	-	85.57	82.51	87.9E-4	+
		CFOrg	98.67	94.4	2.82E-2	-	98.63	98.63	6.66E-16	=	81.9	81.9	1.11E-16	=
musk1	166	Org	71.33				83.92				42.66			
		CF	60.84	43.48	6.72E-2	-	61.54	44.2	7.43E-2	-	61.54	59.36	2.36E-2	+
		CFOrg	71.33	71.33	0E0	=	83.92	83.92	5.55E-16	=	41.96	41.96	2.78E-16	-
sonar	60	Org	71.43				76.19				53.97			
		CF	73.02	54.79	6.83E-2	-	65.08	53.11	5.27E-2	-	53.97	50.22	1.87E-2	-
		CFOrg	79.37	68.19	6.76E-2	-	80.95	76.7	1.21E-2	+	53.97	53.97	6.66E-16	=
vehicle	18	Org	84.65				83.86				83.27			
		CF	83.86	75.5	4.96E-2	-	84.65	77.63	3.65E-2	-	62.99	62.35	27.2E-4	-
		CFOrg	87.8	80.71	5.6E-2	-	84.06	84.03	6.4E-4	+	83.27	83.24	6.4E-4	-
wbcd	30	Org	92.98				92.98				90.64			
		CF	98.25	82.97	15.7E-2	-	98.25	87.68	13.1E-2	-	63.16	62.3	74.2E-4	-
		CFOrg	98.25	83.59	15.6E-2	-	92.98	92.98	1.11E-16	=	90.64	90.64	2.22E-16	=
wine	13	Org	87.65				76.54				82.72			
		CF	97.53	89.38	5.65E-2	+	98.77	92.86	3.97E-2	+	58.02	54.89	1.42E-2	-
		CFOrg	97.53	89.98	5.03E-2	+	76.54	76.54	3.33E-16	=	85.19	83.46	69.8E-4	+
zoo	17	Org	93.33				80.95				98.1			
		CF	95.24	89.9	4.3E-2	-	95.24	93.05	2.01E-2	+	97.14	82.88	6.47E-2	-
		CFOrg	95.24	94.19	1.13E-2	+	80.95	80.95	5.55E-16	=	100	98.46	75.9E-4	+
semeion	265	Org	93.31				96.23				90.79			
		CF	92.05	88.44	6.48E-2	-	90.59	87.79	6.84E-2	-	89.75	40.56	28.8E-2	-
		CFOrg	94.35	93.1	72.9E-4	=	96.44	96.36	10.2E-4	+	91.42	89.28	98.9E-4	-

Figure 5.6 is based on the experimental results of 10-folds cross validation. In Fiture 5.6, the horizontal axis is the number features and the right vertical axis is the classification performance. The blue bar is the testing accuracy, The red line and green line are the training accuracy and the classification performance by only using the original features.

The first observation we have from the experimental results is that the apparent difference of training and testing accuracy by only using the constructed features. At the beginning of each multi-feature classification, the original dataset is splitted into a *Training Set* and *Testing Set* as we discussed in Section 5.3.1.

We found that the training accuracy is much higher than the testing accuracy in the

Table 5.4: Result of 10 Folds Cross Validation

Data set	no. Features	Methods	DT				KNN				NB			
			Best	Avg	Std	T-test	Best	Avg	Std	T-test	Best	Avg	Std	T-test
arrhythmia	278	Org	95.59				94.46				94.46			
		CF	92.01	90.26	86.7E-4	-	90.49	88.54	93.6E-4	-	86.32	85.97	17.6E-4	-
		CFOrg	95.16	94.64	33.9E-4	-	94.14	94.14	12.2E-16	-	94.18	94.07	5.96E-4	-
australian	14	Org	85.99				70.05				85.51			
		CF	67.97	50.21	4.48E-2	-	53.33	46.9	2.54E-2	-	60.43	52.88	3.06E-2	-
		CFOrg	85.65	84.27	66.3E-4	-	70.87	70.87	2.22E-16	+	85.65	84.97	23.7E-4	-
german	24	Org	72.67				68.33				73.33			
		CF	68.2	59.31	5.21E-2	-	64.7	56.91	4.83E-2	-	55.2	43.95	5.73E-2	-
		CFOrg	74	71	1.32E-2	-	68	67.53	18.7E-4	-	72.1	70.35	84.2E-4	-
hillvalley	100	Org	62.09				56.59				52.2			
		CF	61.64	51.64	2.11E-2	-	57.43	50.92	1.53E-2	-	51.07	50.65	28E-4	-
		CFOrg	61.31	52.04	2.11E-2	-	56.6	56.52	1.62E-4	-	50.5	50.5	5.55E-16	-
ionosphere	34	Org	86.67				83.81				28.57			
		CF	56.17	44.43	4.62E-2	-	56.51	44.22	4.88E-2	-	73.48	69.77	1.55E-2	+
		CFOrg	90.01	87.26	3.02E-2	=	83.48	83.16	14.3E-4	-	35.93	35.93	2.22E-16	+
lung	56	Org	90				70				90			
		CF	75	63.35	6.7E-2	-	80.83	60.12	6.45E-2	-	76.67	60	9.54E-2	-
		CFOrg	78.33	70.22	7.09E-2	-	80.83	77.63	93.3E-4	+	80.83	78.77	1.62E-2	-
musk1	166	Org	71.33				83.92				42.66			
		CF	49.12	44.74	1.6E-2	-	52.54	45.3	2.7E-2	-	61.57	58.41	1.21E-2	+
		CFOrg	75.63	73.9	1.27E-2	+	86.32	86.32	0E0	+	43.45	43.45	2.78E-16	+
semeion	265	Org	93.31				96.23				90.79			
		CF	90.08	88.25	3.41E-2	-	90.14	87.7	2.65E-2	-	67.27	39.18	10.1E-2	-
		CFOrg	95.23	93.83	1.56E-2	+	97.24	97.04	10.5E-4	+	91.97	91.24	36.4E-4	+
sonar	60	Org	71.43				76.19				53.97			
		CF	60.95	53.11	4E-2	-	62.9	52.86	3.61E-2	-	59.69	55.13	2.15E-2	+
		CFOrg	74.98	70.95	2.08E-2	=	81.76	80.11	76.6E-4	+	47.48	45.88	72.2E-4	-
vehicle	18	Org	84.65				83.86				83.27			
		CF	78.95	73.98	1.9E-2	-	78.43	75.26	1.52E-2	-	63	62.35	28.2E-4	-
		CFOrg	83.57	78.58	1.99E-2	-	82.27	82.27	5.55E-16	-	81.62	81.49	5.16E-4	-
wbcd	30	Org	92.98				92.98				90.64			
		CF	87.89	77.68	5.32E-2	-	91.57	80.63	5.42E-2	-	66.26	65.53	28.1E-4	-
		CFOrg	91.92	79.34	5.47E-2	-	93.31	93.31	1.11E-16	+	88.4	88.23	2.46E-4	-
wine	13	Org	87.65				76.54				82.72			
		CF	94.73	88.77	3.35E-2	+	95.12	91.26	1.96E-2	+	70.94	67.46	1.38E-2	-
		CFOrg	95.14	91.21	3.39E-2	+	80.52	80.17	7.26E-4	+	87.23	85.98	71.1E-4	+
zoo	17	Org	93.33				80.95				98.1			
		CF	93.77	90.6	1.64E-2	-	94.05	90.48	1.74E-2	+	90.6	82.3	3.13E-2	-
		CFOrg	97.14	95.53	1.03E-2	+	83.82	83.6	23.8E-4	+	98.91	97.65	52.5E-4	-
movementlibras	90	Org	91.36				94.94				90.99			
		CF	89.67	88.68	34.8E-4	-	91.04	89.66	43.7E-4	-	88.93	88.57	19.9E-4	-
		CFOrg	91.37	90.03	52.3E-4	-	96.89	96.62	9.69E-4	+	92.07	91.86	10.4E-4	+

three group of data. This phenomena in machine learning is called *overfitting* and it can be caused by several different factors [4]. For the developed approach, one potential reason for overfitting can be the training data chosen strategy although we have used 10-folds cross validation to eliminate such potential factor. From the experimental results, we can also exclude the possibility for overfitting that is the less features from the dataset, the lower the performance. The classification performance is not improved when the features number increase.

Another observation we have is that the training accuracy is better than the classification performance by only using the original features. Especially when using NB, there is only one dataset obtains worse classification performance than using the original features.

In conclusion, the quality of the constructed features is not strong enough to replace the original features for better classification performance. However, the high performance of

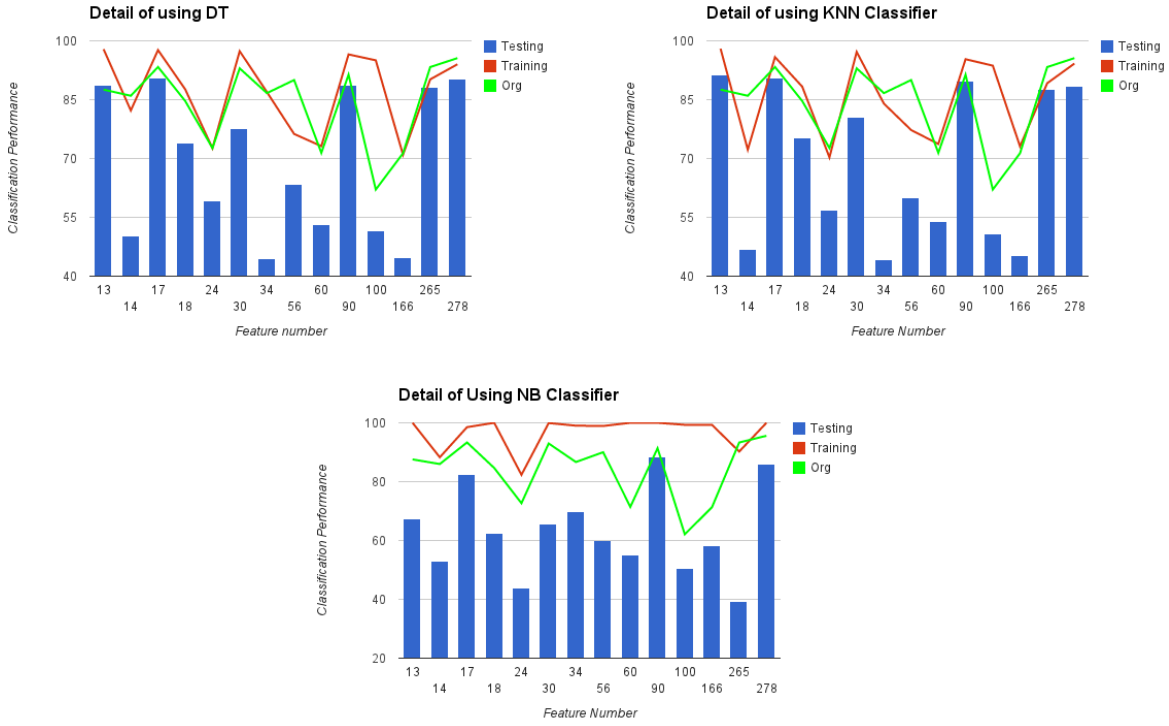

Figure 5.6: the Experimental Result of Testing Accuracy, the Training Accuracy and the Classification Performance Using the Original Features Only

the training accuracy shows the potential of replacing the original features by the newly constructed high-level features. Besides, the constructed features can be more effective for classification by using NB.

5.6.2 The Relationship Between the Multiple Constructed Features and Original Features

In Figure 5.7, we present the experimental results to compare the classification performance by only using the multiple constructed features, the combination of the constructed features and the original features, and only using the original features. In Figure 5.7, the horizontal axis is the number of features and the right vertical axis is the classification performance. The blue bar is the classification performance by only using the constructed features. The red line is the classification performance by using the combination of the constructed features and the original features. The green line is the classification performance by only using the original features.

Seeing from the figures, we found that the classification performance is lower when we use the combination of the constructed features and the original features than only using the constructed features. When using KNN, there are two datasets out of the 14 datasets. When using NB, there are also four out of the 14 datasets. Therefore the classification perfor-

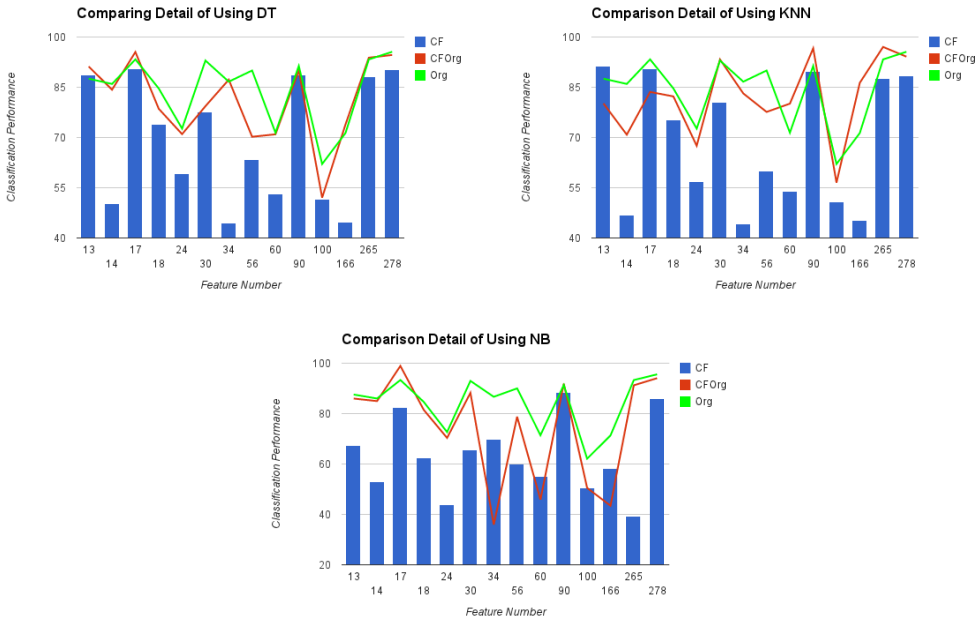

Figure 5.7: the Experimental Result to Compare the Classification Performance by Only Using the Multiple Constructed Features, the Combination of the Constructed Features and the Original Features, and Only Using the Original Features

mance dropped when we combine with the original features since the combination includes a larger number of features with redundancy. Although the constructed features are not informative enough for datasets to replace the original features, the classification performance is better than only using the original features compared with using the combination of the constructed features and the original features in some cases.

5.6.3 The Difference from the GP Method

In [15], a GP based multi-feature construction method was developed. The method uses the class interval finding and the purity of the class interval as the goodness of the constructed feature. Compared with the approach developed in this project, there are following differences:

- we introduce the class interval finding method and class interval purity calculating method into PSO for multi-feature construction. The purity of the class interval is used to evaluate the candidate solution in each iteration.
- when we evaluate an individual particle, we first split the data set into a training and testing set. The training set is used for the class interval discovery and the testing set

is to calculate the purity of the class interval.

- the testing datasets cover larger range of number of features. In [15], the number of features from the datasets are from 4 to 13 compared with a range from 13 to 649 in this project.

The Advantages of the Developed Method

The class interval finding method as part of the developed multi-feature construction approach protects from introducing more noise features since each constructed high-level feature is targeting on individual class. Each class will represent different functionality during the classification. Therefore, the constructed features are properly constructed with explicit objective. Besides, the developed method with class interval finding method is easy to be implemented. We develop the multi-feature construction approach based on the approach of the first and second approach without changing the overall evolutionary process for each swarm except the fitness function. Only one new parameter is added to the evolutionary process, which is the class label for the class interval.

The Limitation of the Developed Method

The class interval finding method provides an easy way to develop the multi-feature construction using PSO. However, we also find some limitations from the experimental results:

- the discovery of constructed high-level features for the developed approach only stays at the individual feature level, which means that each feature is targeted on each individual class label but the algorithm does not discover the interrelationship between the constructed features during the multi-feature construction. Therefore, the relationship between the multiple constructed high-level feature is not established. To simplify, the evolution process for each swarm is independent to each other so that the constructed features is not highly interrelated.
- the classification performance for using both only the constructed features and the combination of constructed features and original features is lower than using the original features on the majority datasets.

5.7 Conclusion

The experimental results show the potential of PSO for multi-feature construction in classification. The developed approach can construct multiple high-level features. The developed approach can reduce the dimension of the dataset without losing too much accuracy although the classification performance still cannot be improved in most cases. One shortage of the developed approach is that the constructed features are not considered as a whole.. During the features construction, they are constructedindividually by different swarms. Therefore, one of the future plans is to develop a way to introduce the interrelationship between the constructed features.

Chapter 6

Conclusion and Future Plan

In this project, PSO has been used for feature construction. In order to construct useful feature(s) by PSO, we first proposed two different representations of array and pair representation to use PSO for function operator selection. We have conducted the experiments for both array and pair representation. The experimental results show that the proposed approach can construct one high-level feature for binary classification and the classification performance can be improved by using the combination of the constructed feature and original features for some datasets. We discussed that the proposed representations have limitations. For both representation, the interval defined for each candidate operator are not even since there always is a candidate operator obtaining a larger interval than others. We have compared the experimental results with [21]. We found that the feature construction efficiency is improved since the proposed approach can construct a single high-level feature much faster than [21] without losing much classification performance.

The proposed representations are also used for the second and third objective in the project. In the second objective of the project, we developed an PSO based approach of feature construction approach for multi-class classification. The proposed method are implemented based on both the array and pair representation. The experimental results show that the proposed approach cannot always construct high-level feature to improve the classification performance for multi-class dataset. By comparing the experimental results, we found that the constructed feature using the array representation is better at improving the classification performance than using the pair representation. We found that one of reasons is that the array representation can be more efficient on operator selection than using pair representation. Using one single constructed feature for multi-class classification problem is challenging, therefore, we developed a PSO based approach to construct multiple features for classification problem.

In order to efficiently and effectively construct multiple new high-level features, we developed a PSO based approach using class interval purity to evaluate the constructed feature. The class interval finding and purity calculating approach is originally from [15]. Different from [15] which using GP to construct features, we using PSO based approach to construct multiple high-level features. We conducted the experiments for the proposed approach using the datasets with larger number of features compared with [15]. The experimental results show that the set of constructed features can improve the classification performance for some datasets. We found that overfitting problem is one of the major factors that are the drawbacks of the proposed method. However, the experimental results show potential of using PSO for feature construction.

6.1 Futrue Plan

The experimental results in this project show the potential for PSO on the problem of feature construction for classification. Our future plan are shown below:

- Improve the particle encoding mechanism. In this project, we propose two representations for the particle encoding. However, one of the shortages of the array and pair representation are less diversity during the evolutionary process. In order to overcome such shortage, we can use tree based representation for the particle encoding. We can introduce mutation based function to PSO to increase the diversity of candidate solutions which learned from GP.
- In this project, the experimental results of multi-feature construction show that we need more than one constructed high-level feature for classification problem. However, one of the shortage of the developed PSO based multi-feature construction approach is that the discovery of the multiple high-level constructed features only stays at the individual feature level. Therefore, in order to discover the relationship between constructed features, we can introduce one more swarm to evolve and control the overall relationship between the multiple constructed features. We thought we can introduce a weight value to each constructed feature. The weight value will be evolved by the newly introduced swarm.

Bibliography

[1] ALFRED, R. A genetic-based feature construction method for data summarisation. In *Advanced Data Mining and Applications*, C. Tang, C. Ling, X. Zhou, N. Cercone, and X. Li, Eds., vol. 5139 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 39–50.

[2] BACHE, K., AND LICHMAN, M. UCI machine learning repository, 2013.

[3] CHUANG, L.-Y., TSAI, S.-W., AND YANG, C.-H. Improved binary particle swarm optimization using catfish effect for feature selection. *Expert Syst. Appl. 38*, 10 (Sept. 2011), 12699–12707.

[4] DIETTERICH, T. Overfitting and undercomputing in machine learning. *ACM Comput. Surv. 27*, 3 (Sept. 1995), 326–327.

[5] ENGELBRECHT, A. P. *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.

[6] GUO, Y., LI, W., MILEHAM, A., AND OWEN, G. Applications of particle swarm optimisation in integrated process planning and scheduling. *Robotics and Computer-Integrated Manufacturing 25*, 2 (2009), 280 – 288.

[7] KENNEDY, J., AND EBERHART, R. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on* (1995), vol. 4, pp. 1942–1948 vol.4.

[8] KRAWIEC, K. Genetic programming-based construction of features for machine learning and knowledge discovery tasks. *Genetic Programming and Evolvable Machines 3*, 4 (2002), 329–343.

[9] LIM, S. H., WANG, L.-L., AND DEJONG, G. Explanation-based feature construction. In *Proceedings of the 20th international joint conference on Artifical intelligence* (San Francisco, CA, USA, 2007), IJCAI'07, Morgan Kaufmann Publishers Inc., pp. 931–936.

[10] LIU, H., AND MOTODA, H. *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[11] MAYFIELD, E. Using feature construction to avoid large feature spaces in text classification. In *Proceedings of the Genetic and Evolutionary Computation Conference* (2010).

[12] MURTHY, S. K., KASIF, S., AND SALZBERG, S. A system for induction of oblique decision trees. *J. Artif. Int. Res. 2*, 1 (Aug. 1994), 1–32.

[13] NESHATIAN, K., AND ZHANG, M. Genetic programming for performance improvement and dimensionality reduction of classification problems. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on* (2008), pp. 2811–2818.

[14] NESHATIAN, K., ZHANG, M., AND ANDREAE, P. A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming. *Evolutionary Computation, IEEE Transactions on 16*, 5 (2012), 645–661.

[15] NESHATIAN, K., ZHANG, M., AND JOHNSTON, M. Feature construction and dimension reduction using genetic programming. In *Proceedings of the 20th Australian joint conference on Advances in artificial intelligence* (Berlin, Heidelberg, 2007), AI'07, Springer-Verlag, pp. 160–170.

[16] OTERO, F. E. B., SILVA, M. M. S., FREITAS, A. A., AND NIEVOLA, J. C. Genetic programming for attribute construction in data mining. In *Proceedings of the 6th European conference on Genetic programming* (Berlin, Heidelberg, 2003), EuroGP'03, Springer-Verlag, pp. 384–393.

[17] SHI, Y., AND EBERHART, R. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on* (1998), pp. 69–73.

[18] VAFAIE, H., AND DE JONG, K. Feature space transformation using genetic algorithms. *IEEE Intelligent Systems 13*, 2 (Mar. 1998), 57–65.

[19] WANG, X., YANG, J., TENG, X., XIA, W., AND JENSEN, R. Feature selection based on rough sets and particle swarm optimization. *Pattern Recognition Letters 28*, 4 (2007), 459 – 471.

[20] XUE, B., CERVANTE, L., SHANG, L., BROWNE, W., AND ZHANG, M. A multi-objective particle swarm optimisation for filter-based feature selection in classification problems. *Connect. Sci 24*, 2-3 (Sept. 2012), 91–116.

[21] XUE, B., ZHANG, M., DAI, Y., AND BROWNE, W. N. Pso for feature construction and binary classification. In *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference* (New York, NY, USA, 2013), GECCO '13, ACM, pp. 137–144.