Intelligent Data Analysis 16 (2012) 305–325
DOI 10.3233/IDA-2012-0525
IOS Press

An intelligent data pre-processing of complex datasets

Shuzlina Abdul-Rahman[a], Azuraliza Abu Bakar[a,*] and Zeti-Azura Mohamed-Hussein[b]
[a]*Department of Science and System Management, Faculty of Information Science and Technology, The National University of Malaysia, Bangi, Selangor, Malaysia*
[b]*School of Biosciences & Biotechnology, Faculty of Science & Technology, The National University of Malaysia, Bangi, Selangor, Malaysia*

Abstract. Pre-processing plays a vital role in classification tasks, particularly when complex features are involved, and this demands a highly intelligent method. In bioinformatics, where datasets are categorised as having complex features, the need for pre-processing is unavoidable. In this paper, we propose a framework for selecting the discriminatory features from protein sequences prior to classification by integrating the filter and wrapper approaches. Several state-of-the-art multivariate filters were explored in the first phase to remove the unwanted features that contributed to noise, while particle swarm optimisation (PSO) with support vector machine (SVM) was adopted in the wrapper phase to produce the most optimal features. Several PSO variants were investigated in the wrapper phase to compare the most suitable PSO variants for the problem domain. The results of both phases were analysed based on classification accuracy, number of selected features, modelling time and area under the curve on the main dataset and, five benchmark machine learning datasets of similar complexity. The higher classification accuracy of the proposed framework was highly reliable with an improvement over the filter phase and the use of full features despite using smaller features.

Keywords: Classification, data mining, feature selection, machine learning, optimisation, particle swarm optimisation

1. Introduction

Increasingly sophisticated technology has led to more storage and manipulation of complex data. The characteristics of complex datasets include, but are not limited to, high dimensionality, unstructured and semi-structured data, temporal and spatial patterns, and heterogeneity [1]. Some categories and typical examples of these datasets include spatial data (maps, VLSI chip layouts), biology data (gene and protein sequences), web data (text, http logs), and multimedia data (video clips, voice). These kinds of datasets pose a challenge to the data mining community and require an advance method that can handle the anatomy and representation of data more efficiently. In bioinformatics for example, there are still a number of challenges to be addressed despite many advancements. One main challenge is to identify the most relevant subset of data in a particular classification. The presence of many irrelevant and redundant features allows for overfitting and less cost-effective models [2], which necessitates the selection of highly discriminatory features prior to mining the dataset.

*Corresponding author: Azuraliza Abu Bakar, Department of Science and System Management, Faculty of Information Science and Technology, The National University of Malaysia, 43600 Bangi, Selangor, Malaysia. Tel.: +603 89216794; E-mail: aab@ftsm.ukm.my.

Previous research realised that data pre-processing is of considerable importance in most classification tasks [3–5]. Feature transformation and feature selection are some forms of data pre-processing tasks that have been commonly used in the literature [4,6–8]. These forms are generally aim to change the data into a simpler dimension that enable the machine learning algorithms to learn faster. Feature construction and feature extraction are two models of feature transformation. The former is defined as a process that discovers missing information about the relationship between features and adds the space of features by inferring or creating new features [9]. Feature extraction is defined as a process that extracts a set of new features from the original features based on some transformation functions and usually only the transformed features are used [10]. From the transformation view, feature construction usually expands the feature space that causes the number of features to be larger than the original features whereas feature extraction reduces the feature space that causes the number of features to be smaller than the original features. Feature selection (FS) takes another view whereby no new features will be generated, instead only a subset of original features is selected from the original features by removing the irrelevant and redundant features. Therefore, it reduces the feature space and at the same time does not change the semantics of the original features. These are the merits of FS that make it preferable over the others.

Generally, FS methods are classified into two models: filter and wrapper, depending on the evaluation measures that they use in distinguishing the different class labels. The former utilises the intrinsic properties of the data to select subsets of features independent of the classifier, while the latter utilises a learning machine to assess subsets of features based on their performance. The filter category can be further divided into two groups: the feature weighting or univariate approach that evaluates and ranks the features individually and the feature subsets or multivariate approach that evaluates the goodness of each subset using certain evaluation criteria. The wrapper approach to FS has attracted more attention than the filter approach because this approach is seen as a stochastic optimisation that attempts to generate better solutions by employing prior knowledge gained from a previous population after the raw features are filtered. However, the higher computational complexity of the wrapper method suggests that the feature space should be pre-reduced using the filter model prior to applying the wrapper approach [2]. Additionally, extensive searching using the wrapper approach suffers from overfitting, particularly in datasets with many irrelevant features and fewer instances [11]. Generally, a common drawback of the filter approach is that it ignores any interactions with the induction algorithm, and most proposed techniques are univariate (i.e., they ignore feature dependencies), whereas the wrapper approach suffers from the common weaknesses of higher overfitting and increased demand on computational resources. In most complex applications, the number of features ranges from moderate (in tens of features) to high dimensional data (in hundreds of features) and therefore the searching of feature subsets become an NP-hard combinatorial problem. The framework of the filter and wrapper approaches is seen as a complement when searching is performed. As mentioned in [12], "*... the feature subset selection problem requires complex function evaluations which are often not available in closed analytical form or exhibits a nonlinear relationship with the space of feature subset*". Furthermore, the capability of filter feature selection algorithms (FSAs) could be exploited by integrating them with a meta-heuristic method, namely particle swarm optimisation (PSO). Such a combination will be examined in this study.

One key contribution of this study is the proposal of a multivariate filter approach and a meta-heuristic approach using a PSO algorithm with support vector machine (SVM) classifier applied to protein sequences. The aim of this approach is to increase the classification accuracy while generating the most discriminatory feature subsets by making use of the strengths of both filter and wrapper approaches. In implementing this work, rigorous comparisons were carried out in both phases. In the filter phase, three state-of-the-art multivariate filters were tested using a Pectin Lyase-like (PLL) dataset,

a protein sequence dataset, and five benchmark UCI datasets of similar complexity for comparison. Our complexity definition is based on moderate to high dimensionality of features, mixed features type (numeric, categorical, and both), unstructured, and semi-structured domains with two classes or more. In the wrapper phase we employed PSO algorithms using three state-of-the-art PSO variants. The use of PSO is motivated by two factors. First, compared to a genetic algorithm, the operation of PSO does not involve crossover and mutation; thus, it is computationally inexpensive, in terms of both memory and runtime [13]. Second, unlike other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance global and local exploration abilities [14].

The remainder of this paper is organised as follows. Section 2 reviews literature on feature selection algorithms. Sections 3 and 4 present the fundamentals behind the FSAs and PSO algorithms. Section 5 presents the proposed framework of filter and wrapper approach. In Sections 6 and 7, the experimental setup and results are discussed, and in Section 8, we offer a conclusion.

2. Feature selection algorithms

FSAs have been applied in various fields, including bioinformatics [15,16], signal processing [17,18], text categorisation [19], image retrieval [20] and pattern recognition [21]. There are several studies that report the overview of state-of-the-art FS methodologies [4,5,22]. In [4] for example, a three-dimensional categorising framework for FSAs is presented. The three dimensions framework was presented based on search strategies (further categorised into complete, sequential and random), evaluation criteria (further categorised into filter, wrapper and hybrid), and data mining task (further categorised into classification or clustering). The benefits of FSAs are manifold, but the most important is prediction because it improves the model performance and avoids the overfitting issue [4,5]. Employing FSAs also produces an effective model because it is able to reduce memory and the time of learning while doing the processing task. Consequently, one would understand the data better by identifying the relevant factors. A typical FSAs process involves four steps namely subset generation, subset evaluation, stopping criterion, and result validation [4,22].

Past literature that focused on filter methods mentioned that identifying features independently, or using the so-called univariate approach, was insufficient [23] because features that are useless by themselves can be useful together [5]. Among the approach's merits, the concept of relevance and redundancy become the focus of current literature. Relevance, as defined by Gutlein [23], *"... is how crucial the value of a feature for predicting the resulting class value*." while "*two features are considered redundant to each other if they are completely correlated"*. Some existing FSAs that have been shown utilised these concepts effectively include correlation feature selection (CFS) introduced by Hall [24] and fast correlation feature selection (FCBF) introduced by Liu and Yu [25]. CFS algorithms issue high merits to subsets that include features that are highly associated to the class feature but have low association to its member. This algorithm exploits heuristic searches such as best first, forward and backward elimination search, with a preference for low redundancy subsets. Similar to CFS, FCBF algorithms consider the relevance of feature to the class, but the feature must also exhibit non-redundancy to the other relevant features. Both algorithms have been successfully used for reducing the feature space, but the capability of these algorithms could be further exploited by integrating with more advanced meta-heuristic approaches, such as the particle swarm optimisation algorithm.

There are some recent works that attempt to improve the sequential forward selection algorithm. Gutlein [26] in his work introduced the linear forward selection (LFS) algorithm search strategy that aims to reduce the number of subset evaluations in each forward selection step. The LFS algorithm

is useful for high dimensionality datasets and reduces the risk of overfitting by focusing only the top k-rank features. They [26] have demonstrated that CFS with LFS is significant to a full forward selection method when tested on 12 high dimensionality datasets when tied with naïve Bayes and C4.5 classifiers. Another interesting approach can be seen in Choo et al. [27] who proposed a fitness-rough algorithm (i.e., an integration of statistical and rough set methods) for features selection. It is a simple but effective approach to eliminate irrelevant features. Nevertheless, it is not suitable for non-quantitative datasets due to a higher percentage of information loss. Zhao and Liu [28], for example, introduced INTERACT, an attempt to improve the FCBF method that uses backward elimination. This method is a combination of the symmetrical uncertainty and c-contribution that manages to maintain and improve the classifier's accuracy. It discards features with no or low c-contribution. In addition, the INTERACT manages the interaction between the features by designing an exclusive hashing data structure to render the issue of feature ranking. This method was compared with several other existing FS and showed a competitive performance.

Several past studies that evaluate the FS method empirically include [8,29]. In [8], five FS methods: ReliefF, random forest feature selector, sequential forward selection, sequential backward selection, and Gini index were compared with several classifiers. Their results showed that ReliefF and random forest enabled the classifiers to achieve the highest increase in classification accuracy on the average while reducing the number of unnecessary features. An extensive research on FS methods was done by Jeffery et al. [29] to identify the differentially expressed genes in microarray data. Ten FS methods were compared on nine different microarray datasets and reported that the empirical bayes t-statistic performed well across the variation of instance sizes. Although the study is limited to two classes of domains, the study emphasizes the FS choice, the size of features, the instances sizes and the noise in the dataset are greatly affect the classifier performance. All these studies suggest the benefits of FS that could improve the classifier's performance.

In the wrapper method, the meta-heuristic algorithm such as genetic algorithms [30], artificial immune systems [31], ant colony optimisation [32], and particle swarm optimisation [33] are some of the popular algorithms. Soto et al. [30] employed a genetic algorithm with various forms of non-linear fitness function, namely decision trees, k-nearest neighbours (k-NN) and a polynomic non-linear function. The study was performed on 73 molecular descriptors for predicting hydrophobicity using an aggregation of neural networks in a chemo-informatics domain. The results were quite promising but rather limited in their domain. Furthermore, Secker et al. [31] employed an artificial immune system in solving clustering problems for protein function prediction, but their experiments were less practical as their system required a highly solution space and time. The use of ant colony optimisation was proposed [32] for text categorisation, and the performance was compared to the genetic algorithm and the several statistical filters method. Interestingly, their results on two Reuter's datasets were superior compared to their counterparts. Additionally, the algorithm of particle swarm optimisation was performed [33] for predicting protein function using naive Bayes and Bayes network classifiers on a G-protein-coupled receptors and enzymes dataset. These datasets are based on four kinds of proteins signatures or motifs. The predictive accuracy of the proposed method outperformed the baseline algorithm that use full feature sets. In another study, Lin et al. [34] used PSO to search for the optimal values for SVM and the developed approach was called PSO+SVM. The classification rate of their approach was better than grid search and had a similar result to GA+SVM. Wang et al. [13] proposed a novel method based on rough sets and PSO in which the PSO was employed to find reducts with fewer features. Their study demonstrated that PSO is able to find minimal reducts efficiently compared to GA-based approaches and several established rough set reduction algorithms.

Another popular form of the FS method is hybrid methods that take advantage of both wrapper and filter methods. According to Liu and Yu [4], this approach usually exploits the independent measure for selecting the *"... best subsets for a given cardinality and uses the mining algorithm to select the final best subset among the best subsets across different cardinalities"*. Previous studies on this approach include those of Uncu and Turksen [35], and Yang et al. [36]. Uncu and Turksen [35] proposed an integration of filter and wrapper methods using k-NN for evaluating the features. This study employed the following independent FSAs during the pre-selection stage to select significant features: k-NN sequential forward, k-NN sequential backward, correlation coefficients and functional dependency concept. Subsequently, these four methods employed k-NN with an exhaustive search strategy to select the best input combination. However, their proposed method would be overfitting if higher numbers of features were involved, as the wrapper approach was employed in the first stage and rather limited because it tested mathematical functions only.

Yang et al. [36] is closely related to our study but limited to microarray datasets. Their experiments were compared on filter, wrapper, and hybrid method (filter and wrapper). The hybrid method effectively improved the performance and selected fewer feature subsets. However, we could hardly compare these results to our study because our protein sequences dataset has a different representation than microarray datasets. Compared to protein sequence datasets, microarray datasets usually involve very high dimensionality features (in more than thousands and ten thousand of features) with small numbers of instances, simply known as the "curse of dimensionality" in which the dimensionality usually exceeds the number of instances. For e.g., these kinds of datasets have a minimum of 5000 genes (features) whereas the instance is less than 100. This kind of dataset is not comparable to other complex dataset having moderate to high feature dimension with large instances. Protein sequences, however, have highly dimensional features (in hundreds of features) with moderate (in hundreds and thousands of instances) to large numbers of instances (in more than ten thousands of instances). Therefore, it is similar to UCI datasets after the feature extraction process was performed on its original data representation. Due to these limitations we would like to confirm the suitability of the hybrid approach over the protein sequences dataset and some other complex datasets of similar complexity in particular on different kinds of original data representation, feature's dimension, and the instances sizes. Additionally, for the PSO implementation, we proposed to employ the hamming distance method to map the real number of velocity whereas Yang et al. [28] used the sigmoid function. Theoretically, the hamming distance method would be faster because only a single value (the distance) is used to compare with each bit of velocity position.

In this study, we designed and tested the integration of multivariate filters with a meta-heuristic approach for a classification problem following a hybrid approach. This paper extends previous work [37] in two ways. First, instead of focusing on one bioinformatics dataset, it compares five benchmark UCI datasets of similar complexity which have been used in [11,38,39]. Second, our study explores three variants of the PSO algorithms, whereas previous work focused on only a single PSO variant. In this study, we selected three multivariate approaches: correlation feature selection with best search strategy (CFS-BS), correlation feature selection with linear forward search strategy (CFS-LFS), and fast correlation based filter (FCBF). The performance of these multivariate filters methods was compared with datasets without FS based on the criteria of classification accuracy, selected features, modelling time and area under the curve (AUC). All these methods applied SVM for evaluating the selected features. Out of the three multivariate filter FS methods, the one that yielded the most competitive result was selected as the outcome of the first stage, and its feature was used in the second stage. For the wrapper approach, we have selected three variants of the PSO algorithm, which we named PSO-1, PSO-2 and PSO-3. The next section describes the fundamental theory behind the selected methods for both phases: the filtering phase (i.e. multivariate filters) and the wrapper phase (i.e. particle swarm optimisation algorithm).

3. Multivariate filters

3.1. Correlation feature selection with best search (*CFS-BS*)

Correlation feature selection (CFS) [24] is a filter method that originates from statistical methods that evaluates subset of features instead of individual features, thus belongs to the family of multivariate filter. It selects feature subsets with a higher degree of correlation to the target class and a lower degree of inter-correlation to each other using a heuristic score, $merit_s$. The subset with the highest merit indicates the higher correlation between the features subset to the target class and the lower the inter-correlation among them. The $merit_s$ or the score of a feature subset S that contains k features is calculated using Eq. (1).

$$Merits_s = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{1}$$

where,

$Merit_s$ is the heuristic score of a feature subset S containing k features
$\overline{r_{cf}}$ is the average feature-class correlation
$\overline{r_{ff}}$ is the average feature-feature inter-correlation

Prior to applying the $merit_s$, a symmetrical uncertainty (SU) is utilised to measure the degree of correlation between discrete features X and Y. Equation (2) gives the formula for SU. The values of feature-class and feature-feature correlations are calculated using SU prior to search the feature subset space. The numeric features are usually discretised using the minimum description length principle (MDLP) method [40]. The study uses CFS with best first search (CFS-BS) to rank the features according to the heuristic score with Eq. (1).

$$SU = 2.0 \times \left[\frac{H(X) + H(Y) - H(X,Y)}{H(X) + H(Y)}\right] \tag{2}$$

where

$H(X)$ is the marginal entropy for discrete features X
$H(Y)$ is the marginal entropy for discrete features Y
$H(X,Y)$ is the joint entropy of X and Y

3.2. Correlation feature selection with linear forward search (*CFS-LFS*)

This kind of multivariate filter adopts the same concept as CFS but with a different search strategy and we labelled it as Correlation Feature Selection-Linear Forward Search (CFS-LFS). The difference lies in the way searching is done and it offers a slightly different approach than the traditional sequential forward selection (SFS) algorithm whereby a simple hill-climbing search method is executed. In the traditional SFS algorithm, the total of evaluations expands quadratically with the number of features, N. For each step, the total number of evaluations is similar to the number of remaining features not yet taken into account. This kind of approach is computationally extensive in particular for datasets that have a large number of features. For example, for N number of features, the sequence of subset evaluations would be N, $N-1, N-2, \ldots$, and so forth. Therefore, the upper bound on the number of evaluations is, $\Sigma_{i=0}(N-i) = 1/2$ x $N\ (N+1)$.

In the linear forward search strategy (LFS), instead of using all features during evaluations, it starts by ranking all features and chooses the top-k ranked features as inputs for forward selection. Reported in [26], the LFS has two variants: fixed set and fixed width. The fixed set, as the name implies reduces the number of features to a fixed set of size k. The beginning ranking is carried out by assessing each feature individually based on their scores using a filter or wrapper evaluator. Next, only the k best features are used in the subsequent forward selection while the remaining features are removed. This process decreases the upper bound on the total of evaluations that need to be considered to $\frac{1}{2}$ x $k(k+1)$. Besides, the number of potential subset extensions decreases along the step and indirectly reduces the computation complexity. In the fixed width method, the same initial ranking in fixed set is performed whereby the search starts with the top-k ranked features. The difference exists during the subsequent forward selection, in which the number of subset extensions in each forward selection step is constantly maintained to a fixed width k. This is performed by adding the next best feature in the ranking to the set of expansions. This process result to an increase in the theoretical upper bound for the number of evaluations in the forward search process to N x $\frac{1}{2}$ x $k(k+1)$. Because our initial experiments showed a similar result for each of these models, we omitted the fixed width model and only focused on the fixed set model. As suggested from the literatures, we employed $k = 50$ and $k = 100$ for medium and high dimensionality of features, respectively.

3.3. Fast correlation based filter

Fast correlation-based filter FCBF [38], is another type of multivariate filter to handle features with high dimensionality. FCBF involves two main steps: relevance analysis and redundancy analysis. The relevance analysis measures information entropy to calculate the dependencies of features. Similar to CFS, it uses symmetrical uncertainty (SU) function as in Eq. (2) to calculate the dependence of features, and it finds the best subset using a backward search technique with a sequential search strategy. If a feature has a relevance score below a predefined threshold, then it is considered irrelevant and discarded. The redundancy analysis measures predominant features and remove redundant features among the relevant features. A feature is said meaningful if it is predominant in predicting the target class. The predominant definition is described in greater detail in [38]. Based on the predominant concept, they defined FS as a process that identifies all predominant features to the class and removes the remaining features. To identify the predominant features and discard redundant features among the relevant features, three heuristic functions were proposed as below [38]:

Heuristic 1: (*if* $S^{+}_{Pi} = \{\}$). Treat F_i as a predominant feature, remove all features in S^{-}_{Pi}, and skip identifying redundant peers for them.

Heuristic 2: (*if* $S^{+}_{Pi} \neq \{\}$). Process all features in S^{+}_{Pi} before making a decision on F_i. If none of them becomes predominant, follow Heuristic 1; otherwise only remove F_i and decide whether or not to remove features in S^{-}_{Pi} based on other features in S'.

Heuristic 3: (*starting point*). The feature with the largest $SU_{i,c}$ value is always a predominant feature and can be a starting point to remove other features;

where F_i, is the correlation between a feature ($F_i \in S$). $S_{Pi}(S^{+}_{Pi}, S^{-}_{Pi})$ is the set of all redundant peers for F_i. $SU_{i,c}$ is the symmetrical uncertainty value that measures the correlation between a feature, F_i and the class, c.

4. Particle Swarm Optimisation algorithm (PSO algorithm)

The PSO algorithm falls under the class of complex systems that attempt to exploit Nature's intelligence. In recent years, its ability to solve hard problems efficiently and credibly was clear. This algorithm,

inspired by flocks of birds and shoals of fish, was published by Kennedy and Eberhart [41] to solve non-linear optimisation problems. As such, PSO creates a swarm of candidate solutions in which each potential solution is seen as a particle with a particular rate of change or velocity that operates through the search space. The earlier PSO implementation was meant for continuous search space domain. However, many real world problems can easily be adapted into binary-valued domain. In the area of dimensionality reduction, specifically in the FS problem, each solution of the particle is represented as fixed length binary strings (i.e. $x_i = (x_{i1}, x_{i2}, \ldots, x_{iN})$, in N dimensional search space, where $x_{id} \in$ {0,1}, $i = 1, 2, \ldots, n$ and $d = 1, 2, \ldots, N$). The coordinates x_{id} of these particles have a velocity, $v_i = (v_{i1}, v_{i2,} \ldots, v_{iN})$. Usually the velocity value is limited in a specified range, V_{max}. If the list of features, $f = (f_1, f_2, f_3, f_4, f_5)$ and $n = 4$, a random initialisation of particles in a swarm could be encoded as follows:

$$x_1 = (1, 0, 1, 1, 0); x_2 = (0, 1, 1, 0, 1); x_3 = (1, 1, 0, 0, 1); x_4 = (1, 1, 0, 1, 0)$$

In the above example, the selected features in particle x_1 are f_1, f_3, f_4. If features are $N = 5$, there are 2^5 possible feature subsets, thus the selected features subset in particle x_1 has been reduced to 2^3.

PSO is initialised as a population of particles in which each particle retains its own individual memory or the best position it has visited, as well as a global memory of the best position visited by all particles in the swarm so far. The position of the particle is adjusted by a stochastic velocity, which depends on two forms of distances: the particle's distance from its own best, *pbest* denoted as $p_i = (p_{i1}, p_{i2}, \ldots, p_{iN})$ and the particle's distance from the swarm overall's best, *gbest* denoted as $p_g = (p_{g1}, p_{g2}, \ldots, p_{gN})$. In searching for the optimal solution, each particle updates its velocity and position as follows: Eqs (3) and (4).

$$v_{id}(\textit{new}) = w.v_{id}(\textit{old}) + \varphi_1.U(0,1)(p_{id} - x_{id}(\textit{old})) + \varphi_2.U(0,1)(p_{gd} - x_{id}(\textit{old})) \tag{3}$$

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k) \tag{4}$$

Where w is the inertia weight; φ_1 and φ_2 are the acceleration positive constants known as cognitive learning rate and social learning rate respectively; $U(0,1)$ is a random function within the range [0,1]. The velocity update in Eq. (3) contains three essential parameters for PSO: the momentum component, the cognitive component and the social component. The first component guides how much the particle recalls its previous velocity via its inertial constant, w. The second component guides how much the particle heads towards its personal best via its cognition learning factor, φ_1. The last component attracts the particle towards swarm's best ever position via its social learning factor, φ_2.

Each particle has its own fitness value that needs to be optimised. In FS problem, evaluation of the performance or fitness function of any particles i is usually computed based on the classification accuracy and the length of selected feature subsets. The two parameters, α and β determine the significance of these two principles. In this study, α was set to 0.9 and β was set to 0.1, indicating that classification accuracy had a higher degree than the subset length. In addition, $\gamma_{Fsel}(A)$ is the classification accuracy of the selected feature subset, F_{all} is the initial number of features, and F_{sel} is the length of selected feature subsets. The formula for the fitness function is then defined in Eq. (5) as follows [13]

$$\textit{fitness} = \alpha \cdot \gamma_{F_{sel}}(A) + \beta \cdot \frac{|F_{all}| - |F_{sel}|}{|F_{all}|} \tag{5}$$

In applying PSO, several parameters, including inertia weight (w), cognition learning factor (φ_1), social learning factor (φ_2), velocity limit, population size and number of generations should be determined

properly as these influence PSO performance. However, past studies [42] observed that particles tending to operate from personal and global best position distances caused the velocities to reach large values. This phenomena led to large position updates and indirectly forced the particles to leave the boundaries of their search space. Therefore, Eberhart and Shi [43] suggested velocity clamping within a maximum value in each feature's dimension. Specifically, the maximum velocity, V_{max}, should be set equal to the dynamic range of each dimension. Moreover, in a recent study [44] suggested the maximum velocity should grow with the problem size.

Inertia weight is another important parameter that can control and reduce the importance of V_{max}. From the literature, we found that there are numerous strategies for setting the inertia weight (w) such as a nonlinear decreasing inertia weight, a random inertia weight, a constant inertia weight, a time-varying inertia and a fuzzy inertia weight [13,42,43,45–47]. The nonlinear decreasing weight inertia usually decreases within the range of 0.9 and 0.4 during a run; the formula for this is shown in Eq. (6). This high value of inertia weight in the beginning allows for exploration, and a much lower value of inertia weight allows the swarm to be more exploitative. In addition, this strategy usually employs constant coefficients; $\varphi_1 = \varphi_2 = 2$. However, Eberhart and Shi [43] mentioned that this cannot guarantee a good result for tracking a nonlinear dynamic system. Their concern brings us to explore the random inertia strategy in which inertia weight is defined as a random number within the range of 0.5 and 1.0, resulting in a mean value of 0.75. Besides the nonlinear decreasing weight and the random inertia weight, past studies [45] also mentioned that the use of constriction coefficients, γ may improve the overall performance. The role of constriction coefficients is similar to the inertia weight except that it is multiply with the whole parameters as shown in Eq. (7). The aim of the constriction coefficients is to avoid the particles deducing very large values and to manage convergence without the need for velocity clamping [48]. Nevertheless, later studies [49] suggested that it is still useful to limit v_{max} to the dynamic range of each dimension of x_{max}. In this study, the value for the acceleration coefficients, φ was set to 4.1, and the constriction coefficient, γ, was approximately 0.7298 that can be derived using Eq. (8).

$$weight_{new} = \frac{(weight_{old} - 0.4) * (iter_{max} - iter_{cur})}{iter_{max} + 0.4} \tag{6}$$

$$v_{id}(new) = \gamma(v_{id}(old) + \varphi_1 U(0,1)(p_{id} - x_{id}(old)) + \varphi_2 U(0,1)(p_{gd} - x_{id}(old))) \tag{7}$$

where,

$$\varphi = \varphi_1 + \varphi_2 > 4 \text{ and } \gamma = \frac{2}{|\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}|} \tag{8}$$

Generally, two methods in interpreting the velocity of the binary PSO exist: the hamming distance and the sigmoid function. In the hamming distance method, the distance is calculated based on the number of different bits between two particles which basically correspond to the difference between their positions [13]. Let p_{gbest} = [1 0 1 1 1], p_i = [0 1 1 0 1]. The difference between these two particle's position: $p_{gbest} - p_i$ = [1 −1 0 1 0] in which the value of one implies that this bit should be selected but is not selected while the value of negative one implies that this bit should not be selected but is selected. The distance is calculated based on the difference between total number of ones and total number of negative ones [13]. The positive and negative differences in this value allow particles to be more explorative in the searching space. Then the particle's bit is transformed into a requisite 0 and 1 by comparing to this distance value. Whereas, the velocity that employs the sigmoid function is transformed into a requisite 0 or 1 by comparing to a uniformly random value in the interval [0.0, 1.0].

Algorithm 1

Input:

n: the swarm size; φ_1, φ_2: positive constants

w: inertia weight; *U(0,1)*: random number (0,1)

Max_Vel: maximum velocity of particles

Max_Gen: maximum generation

Max_Fit: maximum fitness value

Output: gbest: Global best position

Begin

Initial Pbest(*i*) = 0; Gbest = 0; Iter = 0;

/* Create and randomly initialize particles' position and velocities on *N* dimensions*/

While Iter < Max_Gen and Gbest < Max_Fit

for each particle *i*=1,…,*n* do

/*Calculate the performance *f* of each particle to its best performance so far */

if *f*(i) > Pbest(i) then

Pbest(i) = *f*(i); $p_{id} = x_{id}$; *d* = 1,. . .,*N;*

if *f*(i) > Gbest then

Gbest = *f*(i); gbest = i;

for each particle *i*=1,…,*n* do

for each dimension *d*=1,…,*N* do

/* Calculate the distance between the current particle & lbest and

current particle with gbest using Hamming distance method*/

$v_{id}(new) = w.v_{id}(old) + \varphi_1.U(0,1)(p_{id} - x_{id}(old)) + \varphi_2.U(0,1)(p_{gd} - x_{id}(old))$

/* Update velocity where p_i is the best position visited so far by x_i,

p_g is the best position visited so far by any particle */

if $v_{id} > Max\_Vel$ then $v_{id} = Max\_Vel$ /* Limit the velocity's magnitude *

if $v_{id} > -Max\_Vel$ then $v_{id} = -Max\_Vel$

if $v_{ij} \leq distance(gbest)$ then /*Update position following gbest */

$x_{ij} = rand()$; /* the bits of the particle are randomly change, different from gbest */

else

for every *k* do /*change the bits outside the different bits between the particle and gbest */

if ($gbest_k == x_{ik}$)

$x_{ik} = rand()$;

Iter=Iter+1;

Output gbest

End

Fig. 1. Pseudo-code for the binary PSO using hamming distance method.

Algorithm 2

Input:

n: the swarm size; φ_1, φ_2: positive constants;

w: inertia weight; $U(0,1)$: random number (0,1)

Max_Vel: maximum velocity of particles

Max_Gen: maximum generation

Max_Fit: maximum fitness value

Output: gbest: Global best position

Initial Pbest(i) = 0; Gbest = 0; Iter = 0;

/* Create and randomly initialize particles' position and velocities on N dimensions*/

While Iter < Max_Gen and Gbest < Max_Fit

for each particle i=1,…,n do

/*Calculate the performance f of each particle to its best performance so far */

if $f(i)$ > Pbest(i) then

Pbest(i) = $f(i)$;

if $f(i)$ > Gbest then

Gbest = $f(i)$; gbest = i;

for each particle i=1,…,n do

for each dimension d=1,.., N do

$v_{id}(new) = w.v_{id}(old) + \varphi_1.U(0,1)(p_{id} - x_{id}(old)) + \varphi_2.U(0,1)(p_{gd} - x_{id}(old))$ /*PSO-2 */

$v_{id}(new) = \gamma[v_{id}(old) + \varphi_1.U(0,1)(p_{id} - x_{id}(old)) + \varphi_2.U(0,1)(p_{gd} - x_{id}(old))]$ /*PSO-3 */

/* Update velocity where p_i is the best position visited so far by x_i,

p_g is the best position visited so far by any particle */

If $v_{id}(k+1) \in (V_{min}, V_{max})$ then

$v_{id}(k+1) = max(min(V_{max}, v_{id}(k+1)), V_{min})$ /*Limit the velocity's magnitude */

$$S(v_{id}(new)) = \frac{1}{1+e^{-v_{id}(new)}}$$

If ($U(0,1) < S(v_{id}(new)$ then $x_{id}(new) = 1$ else $x_{id}(new) = 0$ /*Update position*/

Iter=Iter+1;

Output gbest

Fig. 2. Pseudo-code for the binary PSO using sigmoid function.

In this paper, we are interested to explore the PSO algorithms with the nonlinear decreasing inertia weight, the random inertia weight and the constriction coefficient as it offers different kind of control to the velocity update. In this paper, the PSO following the nonlinear decreasing weight with hamming distance method is designated as PSO-1, while the PSO that follows the random inertia weight and constriction coefficient using sigmoid function is designated as PSO-2 and PSO-3 respectively. Figures 1 and 2 describe the pseudo code of these three PSO variants (adapted from [13]). The details of other parameter settings in these variants are presented in the experimental setup section.

5. The proposed filter-wrapper approach

The goal of this study is to find the best combination of FS approaches for the classification problem from complex domains. Specifically, we propose to examine the FS framework created by integrating multivariate filters and meta-heuristic wrapper approaches in a complex classification problem. The proposed method avoids the issue of overfitting by filtering the potential significant input features prior to identifying the best input combination with the wrapper approach. This process is basically composed of three main stages: the discretisation stage, the filter stage and the wrapper stage.

5.1. Discretisation stage

Discretisation is a process of quantising continuous attributes [50] that prove to be important to guarantee more accurate and faster learning. Discretisation methods have been developed according to several taxonomies which generally fall into supervised and unsupervised discretisation. Till date there are three kind of taxonomies reported in literatures [51–53]. The most recent taxonomy [51] proposed four levels of data discretisation taxonomy based on hierarchical approach: (1) hierarchical and non-hierarchical, (2) splitting, merging, and combination, (3) supervised, unsupervised, and combinations, (4) binning, statistics, entropy, and etc. In this study, we employed entropy discretisation specifically a minimum description length principle (MDLP), to descretise the dataset with numeric values. MDLP, proposed by Fayyad and Irani [40], was suggested as one of the most successful supervised discretisation method due to smaller error rates and less modelling time [50]. It has also been widely used in complex domains [54,55]. As a confirmation, our initial experiment on the selected datasets also shows a better performance with the MDLP method.

5.2. The integration approach – Filter and wrapper

The proposed approach for this study adopted the filter-wrapper approach (Fig. 3). In the filter phase, we examined the multivariate filters using WEKA [56] and evaluated them using SVM. The justification of the classifier was mentioned in the previous paper [57]. In WEKA, the SVM classifier was implemented by the sequential minimal optimisation (SMO). For both phases, SVM used the normalised values whereby all the discretised value are normalised by default [56] in the filter phase and *svm-scale* function was used in the wrapper phase. For each dataset, we randomly split into a training set (90%) and a testing set (10%). In the wrapper phase, the aim was to further refine the features by only selecting the most optimum features for the classification task. The implementation of PSO was done in Java and NetBeans IDE 6.9 environment. The optimum features produced are classified using SVM algorithm in libSVM [58] using radial basis function (RBF) kernel's function. The use of RBF kernel involved the searching of two optimum parameters, C (cost) and γ (gamma), within the range 2^{-5} and 2^{15} for C while the range 2^{-15} and 2^{3} for γ [59]. In the filter phase, the searching of these two parameters was handled by the WEKA SVM (SMO) within the default ranges and this was performed on the dataset with 10-fold cross validation. Whereas, the searching of these parameters in the wrapper phase was performed using the grid search method, a commonly used method to get the optimum parameter of C and γ, on the training set (90%) with 10-fold cross validation method. This process produces a set of optimum parameters which was used to retrain the training data. Subsequently, the model file was used to predict the testing set (10%).

Table 1
Protein features and its description

Features/Label	Description	#Features
Amino Acids ($f_1 - f_{20}$)	Percentage of Amino Acid Compositions (AAC)	20
Mol weight (f_{21})	The molecular weight of the protein	1
AA Size, (f_{22})	Number of residues in each protein	1
Charged, (f_{23})	Physiochemical properties	1
Aliphatic, (f_{24})	Physiochemical properties	1
Aromatic, (f_{25})	Physiochemical properties	1
Polar, (f_{26})	Physiochemical properties	1
Neutral, (f_{27})	Physiochemical properties	1
Hydrophobic, (f_{28})	Physiochemical properties	1
+ve charged, (f_{29})	Percentage of positively charged residues in the protein	1
−ve charged, (f_{30})	Percentage of negatively charged residues in the protein	1
Tiny, (f_{31})	Physiochemical properties	1
Small, (f_{32})	Physiochemical properties	1
Large, (f_{33})	Physiochemical properties	1
Dpc ($f_{34-}f_{433}$)	Percentage of Dipeptide Composition (DPC)	400
Total	Total number of features	433

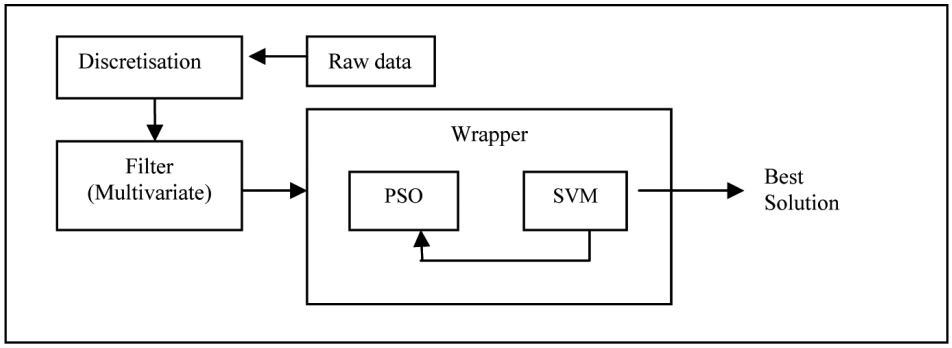

Fig. 3. Diagram for the proposed Filter-Wrapper approach.

6. Experimental setup

The experiment was performed on protein sequences from the pectin lyase-like (PLL) superfamily collected from UniProt databases. The functional information was extracted from Pfam, a large collection of protein families, available at http://pfam.sanger.ac.uk/search [60]. The classification of PLL into various subfamilies was done on the basis of amino acid compositions extracted from COPid, a composition-based protein identification web server available at http://www.imtech.res.in/raghava/copid/ [61]. The initial data set had 1074 sequences belonging to seven subfamilies of PLL; however, classes contains insignificant members were excluded in this study, resulting in 859 proteins in four subfamilies (classes). These subfamilies were pertactin (128 instances), glyco_hydro_28 (326 instances), pectinesterase (204 instances) and pectate_lyase_C (201 instances).

A total of 433 features from three categories (amino acid composition (AAC), physico-chemical composition (PCC), and dipeptide composition (DPC)) were extracted from each protein sequence. In addition, the AAC category was composed of 20 features, the PCC category was composed of 13 features, and DPC category was composed of 400 features. The use of these feature categories was motivated by past studies [62–64], and Table 1 describes the details of these feature sets. The complexity of data can be justified based on several factors such as the feature's dimension (FD) or number of features

Table 2
Feature summary

Dataset	Description	Domain	Data type	Feature type	Features	Instances	Classes	Past usage
DS1	USCensus90[red]	Economic data	Multivariate	Categorical	67	2500	3	[38,39]
DS2	Coil2000[red]	Insurance data	Multivariate	Categorical, Numeric	85	1941	2	[11,38,39]
DS3	Promoters	DNA data	Sequence	Nominal	228	106	2	[38,39]
DS4	Arrhythmia	ECG data	Multivariate	Categorical, Numeric	279	452	16	[11,38,39]
DS5	Multi-features	Multi-features	Multivariate	Categorical, Numeric	649	2000	10	[38,39]
DS6	Bioinformatics	Protein data	Sequence	Numeric	433	672	4	

Table 3
PSO parameter settings

Parameters	*PSO-1 [13]*	*PSO-2 [43]*	*PSO-3 [48]*
Inertia weight	nonlinear decreasing [1.4,0.4]; Formula as in Eq. (6)	random inertia weight [0.5, 1.0]	constriction coefficient; constant value = 0.7928 formula as in Eq. (8)
Constant coefficient	$\varphi_1 = 2$; $\varphi_2 = 2$; $\varphi_1 + \varphi_2 = 2$	$\varphi_1 = \varphi_2 = 1.494$	$\varphi_1 = \varphi_2 = 2.05$; $\varphi = \varphi_1 + \varphi_2$
Velocity update	Hamming Distance	Sigmoid function	Sigmoid function
Velocity limit	$V_{max} = [1, (1/3) * N)]$	$V_{max} = [-6, +6]$	$V_{max} = [-6, +6)]$
Control parameter	$\alpha = 0.9$, $\beta = 0.1$	$\alpha = 0.9$, $\beta = 0.1$	$\alpha = 0.9$, $\beta = 0.1$
Max generation	100	100	100
Population size	30	30	30

(medium to very high dimensions), the number of instances, and the type of data (semi-structured or/and unstructured), and heterogeneity [1]. These kinds of dataset usually exist in domains such as in biological data, world-wide web data, time series data, spatial data, and graphical data. Therefore, a complex dataset can be defined as having any two of these factors. However, the definition on the number of instances is quite subjective. A small number of instances (less than hundred) with high dimensionality (thousands or more) can also be considered as a complex dataset. The microarray dataset usually contains thousand of genes with limited number of instances [11,65,66]. Previously, Kudo and Sklansky [21] defined dimensional category as follow: low ($0 < \text{FD} \leqslant 19$), moderate ($20 \leqslant \text{FD} \leqslant 49$), high ($50 \leqslant \text{FD}$). We improvised this definition after analysing the profile of datasets used in [11] and proposed our new definition on the feature's dimension as follow: low ($0 < \text{FD} \leqslant 49$), moderate ($50 \leqslant \text{FD} \leqslant 99$), high ($100 \leqslant \text{FD} \leqslant 999$), very high ($1000 \leqslant \text{FD}$).

Apart from the PLL dataset, we obtained five other datasets from the benchmark UCI repository with similar complexity. These datasets are USCensus90 (DS1), Coil2000 (DS2), Promoters (DS3), Arrhythmia (DS4), and Multi-Features (DS5) whereby the selection was based on several criteria, which involved unstructured data (DS3) and multivariate type of data (DS1, DS2, DS4, DS5), and both moderate and high dimensional datasets. For DNA data, its raw representation was preprocessed by transforming each position into four binary attributes, one for each nucleotide [11]. The original number of features for DS3 was 57 features. For large datasets such as DS1 and DS2, the instances of the data were removed using *weka.filters.supervised.instance.StratifiedRemoveFolds* features and these datasets were marked with the tag [red]. In evaluating the effectiveness over the real world data, the classification accuracy on the selected features subsets was considered as an indirect measure [39] using SVM from WEKA [56]. The details of these datasets are presented in Table 2. Table 3 describes the three types of PSO parameter settings mentioned in Section 4. Most of the settings followed closely to the past studies [13,43,48] except for the last two parameters which were based on our preliminary experiments on the selected benchmark UCI datasets.

Table 4
Comparison of SVM classification accuracy and the selected number of features from full features (FF), correlation feature selection-best search (CFS-BS), correlation feature selection-linear forward search (CFS-LFS), and fast based correlation filter (FCBF) on 10-fold cross validation. The W/T/L summarises the wins/ties/losses (at p-value < 0.05) over FF over all datasets

Dataset	Accuracy (%)				Selected number of features (feature length, (%))			
	FF	CFS-BS	CFS-LFS	FCBF	FF	CFS-BS	CFS-LFS	FCBF
DS1	93.40	90.04^{L}	90.00^{L}	90.00^{L}	67	3.27(4.9)	3.27(4.9)	3.28(4.9)
DS2	94.02	94.02^{T}	94.02^{T}	94.02^{T}	85	7.34(8.6)	7.34(8.6)	4.04(4.8)
DS3	93.20	93.17^{L}	93.20^{T}	92.89^{L}	228	7.99(3.5)	7.99(3.5)	6.36(2.8)
DS4	70.29	61.74^{L}	61.60^{L}	54.53^{L}	279	18.51(6.6)	18.51(6.6)	7.60(2.7)
DS5	96.11	98.85^{W}	98.87^{W}	98.58^{W}	649	147.51(22.7)	131.59(20.3)	129.28(19.9)
DS6	93.03	97.69^{W}	97.78^{W}	97.50^{W}	433	52.25(12.1)	49.72(11.5)	43.42(10)
Average	90.01	89.25	89.25	87.92	290.2	39.48(9.7)	36.40(9.2)	32.33(7.5)
W/T/L		2/1/3	2/2/2	2/1/3		6/0/0	6/0/0	6/0/0

7. Experimental results and discussion

This section empirically evaluates the performance of filter and wrapper phases by comparing several multivariate FS algorithms and three variants of PSO algorithms for information loss. Information loss is defined as the quantity of information lost in the process of data mining that degrades the classifier performance due to dimension reduction, size reduction or missing values [27]. This study adopts classification accuracy as the measure of information loss; the higher the classification accuracy, the lower the information loss from the selected FSAs. Apart from the classification accuracy, some other principles, such as the selected number of features, the modelling time and the area under curve (AUC), are also compared.

7.1. Filter phase

Tables 4 and 5 record the results of the filter phase in which the last two rows in each table summarise the average values of each measurement and the results of all datasets in terms of its wins/ties/losses at p-value < 0.05 (indicated by the letters W/T/L) over the full features set. In addition to average measurement, we emphasise wins/ties/losses because the average criteria would be susceptible to outliers [67]. In general, it can be seen that all these FSAs produced a good trend in most of the datasets. We defined a good trend as when the reduction of features maintained or improved the classification accuracy of the SVM. This result can be seen in CFS-LFS, for example, in which four out of the six datasets produced a good trend with two wins and two ties. For both CFS-BS and FCBF, the results were competitive with two wins and one tie if compared to the full features set. Overall, out of the six datasets, two datasets were statistically significant (i.e., wins) over the full set of features (DS5 and DS6). This result indicated the need of the wrapper phase to further optimise the selected features.

On average, all these FSAs achieve competitive classification accuracy over the full features, but they achieve significant reduction of dimensionality by selecting only a small number of features from the full feature set. This can be seen from the results of selected number of features. The average numbers of selected features for the three FSAs were 39.48 (CFS-BS), 36.40(CFS-LFS), and 32.33(FCBF). In the DS6 (Bioinformatics) for example, the percentage of the selected number of features was only 12.1% (CFS-BS), 11.5% (CFS-LFS), and 10% (FCBF) and yet the classification accuracy is improved from 93.03% to 97.69% (CFS-BS), 97.78% (CFS-LFS), and 97.5% (FCBF). These results highlighted that not all features were necessary to achieve high classification accuracy. Except for the DS4, the proposed FSAs were able to minimise information loss (within 3% allowance) in all datasets based on classification

Table 5
Comparison of average modelling time and ROC on full features (FF), correlation feature selection-best search (CFS-BS), correlation feature selection-linear forward search (CFS-LFS), and fast based correlation filter (FCBF) on 10-fold cross validation. The W/T/L summarises the wins/ties/losses in modelling time and AUC (at p-value < 0.05) for FF over all datasets

Dataset	*Modelling Time* (*seconds*)				*AUC*			
	FF	CFS-BS	CFS-LFS	FCBF	FF	CFS-BS	CFS-LFS	FCBF
DS1	12.43	1.72^{W}	1.63^{W}	1.59^{W}	1.00	1.00^{T}	1.00^{T}	1.00^{T}
DS2	6.31	0.69^{W}	0.65^{W}	0.39^{W}	0.50	0.50^{T}	0.50^{T}	0.50^{T}
DS3	0.13	0.15^{L}	0.11^{W}	0.08^{W}	0.93	0.93^{T}	0.93^{T}	0.93^{T}
DS4	7.38	8.2^{L}	6.94^{W}	5.76^{W}	0.73	0.60^{L}	0.59^{L}	0.50^{L}
DS5	40.60	5.18^{W}	3.38^{W}	2.50^{W}	0.92	1.00^{W}	1.00^{W}	1.00^{W}
DS6	253.46	87.40^{W}	62.90^{W}	63.57^{W}	0.99	1.00^{W}	1.00^{W}	1.00^{W}
Average	53.39	17.22	12.60	12.32	0.85	0.84	0.84	0.82
W/T/L		4/0/2	6/0/0	6/0/0		2/3/1	2/3/1	2/3/1

accuracy compared to the full features. Reviewing the complexity measurement, the use of FSAs was able to decrease the degree of complexity of search space from 2^{290} to 2^{36} (290 is the average feature length for all datasets, and 36 is the average feature length of the three FSAs).

Further comparison was made on the modelling time and AUC on full features as well as all of the FSAs. Modelling time was defined as the elapsed time during the classification on the training datasets while the AUC was defined as the probability of the classifier in ranking a randomised positive instance above the negative instance with a perfect condition of "1". From the experiments, all FSAs showed a significant reduction of modelling time over the full features set. For modelling time, both CFS-LFS and FCBF won in all the datasets, with FCBF being the fastest and CFS-BS being the slowest. In terms of AUC, all three FS algorithms gave a similar number of wins, ties and losses over the full features set. However, except for the DS1, DS5 and DS6 datasets, the other three datasets demanded further action, as their AUC values were less than "1".

To sum up, the experimental results of the filter phase employing the multivariate FSAs verified the need of the wrapper phase in optimising its selected feature based on the evaluation criteria mentioned above. In selecting the most suitable FSAs for the next phase, we chose CFS-LFS because it maintained and improved the classification accuracy over the other FSAs on four out six datasets and gained similar number of wins to FCBF with the least number of losses.

7.2. Wrapper (Optimisation) phase

Table 6 compares the results of each PSO variant for ten runs based on classification accuracy, number of selected features and modelling time on the DS6 or the Bioinformatics (PLL) dataset with the results from the filter phase using a multivariate (MV) approach. The comparison was analysed in terms of the average value and t-test (p-value < 0.05). From the results, we could observe that the fewer selected features did not influence the classification accuracy. In this dataset, for example, although the selection of feature was about 40% (average number of selected features in PSO-1, PSO-2, and PSO-3 over the number of selected features in MV method) from the first phase, the accuracy results among the PSO variants were better than the MV method. This indicated that the selected features from the first phase still contain noise that could be further enhanced with an optimisation method.

A validation using a t-test also showed that this result of selected length was statistically significant, with all the three variants of PSO having a p-value less than 0.05. However, in terms of modelling time, both MV+PSO-2 and MV+PSO-3 methods took a longer time than the MV phase. The longer time for

Table 6
Comparison of average classification accuracy, length and modelling time on multivariate (MV), MV+PSO-1, MV+PSO-2 and MV+PSO-3 methods over ten runs. The last row records the significance level at p-value < 0.05

PLL	Accuracy				Number of selected features				Time (seconds)			
	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3
run1	97.80	100.0	100.0	100.0	49.4	31	18	13	3.35	7	275	614
run2	98.06	100.0	100.0	100.0	49.8	26	18	12	3.36	1	1044	1056
run3	97.93	100.0	100.0	100.0	50.0	28	18	11	3.38	10	622	1054
run4	97.80	100.0	100.0	100.0	49.5	27	18	14	3.32	1	590	1169
run5	97.54	100.0	100.0	100.0	50.3	30	19	14	3.31	12	195	1711
run6	97.92	100.0	100.0	100.0	49.2	25	18	15	3.33	11	812	1152
run7	97.67	99.3	100.0	100.0	49.6	26	21	13	3.39	6	275	901
run8	97.54	100.0	100.0	100.0	49.7	29	19	18	3.49	3	942	1095
run9	97.67	99.3	100.0	100.0	50.2	27	18	12	3.50	3	590	932
run10	97.93	100.0	100.0	100.0	49.5	29	18	15	3.40	3	797	1012
avg	97.78	99.9	100.0	100.0	49.7	27.8	18.5	13.7	3.38	5.7	614.2	1069.6
p-value		0.00^{+}	0.00^{+}	0.00^{+}		0.00^{+}	0.00^{+}	0.00^{+}		1.00^{+}	0.00^{-}	0.00^{-}

Table 7
Comparison of average classification accuracy, number of selected features, and modelling time between the FF(full features), multivariate (MV), MV+PSO-1, MV+PSO-2 and MV+PSO-3 using a support vector machine classifier

	Accuracy (%)					*Number of selected features*					*Modelling time (seconds)*				
	FF	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3	FF	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3	FF	MV	MV+ PSO-1	MV+ PSO-2	MV+ PSO-3
DS1	93.4	90.0	92.4^{W}	92.4^{W}	92.4^{W}	67	3	2^{W}	2^{W}	2^{W}	12.4	1.6	4.4^{L}	10.6^{L}	7.8^{L}
DS2	94.0	94.0	94.1^{W}	94.1^{W}	94.1^{W}	85	7.3	1.7^{W}	1.3^{W}	1^{W}	6.31	0.6	15.5	33.2^{L}	87^{L}
DS3	93.2	93.2	98.5^{W}	100^{W}	100^{W}	228	7.9	3.5^{W}	3^{W}	3^{W}	0.13	0.1	8.1	6.6^{L}	4.3^{L}
DS4	70.3	61.6	78.0^{W}	81.1^{W}	80.9^{W}	279	18.5	11.2^{W}	10.7^{W}	10.7^{W}	7.38	6.9	225	968.5^{L}	580^{L}
DS5	96.1	98.9	98.8	99.5^{W}	99.6^{W}	649	132	66.1^{W}	62.2^{W}	36.3^{W}	254	62.9	1703	15068^{L}	16080^{L}
DS6	93.0	97.8	99.9^{W}	100^{W}	100^{W}	433	49.6	27.8^{W}	18.5^{W}	13.7^{W}	40.6	3.4	5.7	614.2^{L}	1069^{L}
Avg.	90.0	89.2	93.6	94.5	94.5	290	36.3	18.7	16.3	11.1^{W}	53.4	12.6	327	2784	2972
W/T/L			5/0/1	6/0/0	6/0/0			6/0/0	6/0/0	6/0/0			0/0/1	0/0/6	0/0/6

The symbols W/T/L indicate the wins, ties or losses over the multivariate method (MV) at $p < 0.05$.

these variants is caused by the way the particle's position is updated. In MV+PSO-1, the new particle is updated based on hamming distance method while in MV+PSO-2 and MV+PSO-3, the new is updated based on sigmoid function. The details of this implementation can be found in Figs 1 and 2.

7.3. Discussion of overall results

The experimental results for the six datasets are summarised in Table 7. We analysed the results based on the classification accuracy, number of selected features and modelling time over the use of full features (FF), the multivariate (MV) filter, and the filter and wrapper (MV+PSOs). The last two rows summarise the average results of each criteria and the wins/ties/losses (at p-value <0.05) over the MV method. The overall results showed that the proposed method produced a good trend in most of the datasets. We defined a good trend as when the selected number of features or the reduction of features produced higher classification accuracy than the use of the MV method.

From Table 7, we can see that the MV+PSO-1 produced a good trend in most datasets with five wins and one loss in the classification accuracy. The MV+PSO-1 method also preserved its information loss in all datasets in which the classification accuracy increased, despite the fewer features (except in the

DS5). However, within an allowance of less than 1%, the information loss in the DS5 dataset was considered small and insignificant. Additionally, MV+PSO-1 greatly reduced the unwanted features in all dataset with an average of 50% feature reduction from the features in the MV method. As for modeling time, although on average the MV+PSO-1 took more time than the MV method, but it was statistically insignificant in all the datasets except for DS1 dataset. Both MV+PSO-2 and MV+PSO-3 methods gave similar average classification accuracy and outperformed the MV method with an average of 5.3% difference while MV+PSO-1 outperformed the MV method with an average of 4.4% difference. In addition, both MV+PSO-2 and MV+PSO-3 methods gave all wins over the MV method in classification accuracy that implies the information loss is preserved. Similar to MV+PSO-1, both MV+PSO-2 and MV+PSO-3 methods greatly attained smaller numbers of selected features, with an average of more than 50% feature reduction from the features in the MV method.

Overall, the above results demonstrated the superiority of the integrated method over the MV method from several aspects. The proposed method has produced a good trend in most datasets in terms of higher classification accuracy with a fewer number of selected features. In DS6 (Bioinformatics) dataset for example in the first phase, although the selected number of features was about 12% from the original features, the classification accuracy was improved more than 4%. Further improvement in this dataset could be seen in the second phase, where the selected number of the features was within the range of 3–6% from the original features. On average, the classification accuracy of the proposed method was about 5% higher than the single approach, despite using less number of features. The high number of wins in the proposed method further emphasised the reliability of the proposed method. In most cases, the higher classification accuracy is significantly better than the MV method, which also translates more information is preserved. However, to accommodate the risk of losing information on a particular domain, the expert involvement would benefit the final analysis. In this study, the information loss is indicated by the improvement of accuracy despite the reduce features. The accuracy results among the three PSO variants were comparatively similar with a difference of less than 1%. This was also true for the number of selected features in which only 50% of the features from the single method was used. Among the three PSO-variants, PSO-3 obtained the least number of selected features but requires the longest modelling time which is almost nine times higher than the PSO-1. The faster modelling time in MV+PSO-1 and the less number of selected features explained that a hamming distance method is a better choice. Additionally, there is no statistical difference in the modelling time between the MV+PSO-1 with the MV method. In terms of complexity, measurements on all the selected datasets illustrated that the complexity of the search space was reduced from 2^{290} to 2^{36} in the first phase and 2^{36} to 2^{15} in the second phase.

8. Conclusion

This study introduced a framework of dimensionality reduction by integrating a multivariate filter with a meta-heuristic algorithm, specifically the PSO algorithm, to attain the classification problems on complex datasets. The need of the proposed method is justified with the increase of accuracy despite the smaller number of features on most selected datasets. A significant decrease of the selected features subset in both phases highlighted the importance of pre-processing to guarantee the discriminatory features for classification. The accuracy and the selected number of features among the three variants of PSO were comparatively similar except on modelling time. Our experimental results conclude the need of data pre-processing particularly for complex datasets prior to classification via the integration of FSAs approach. A future continuation of our work is to improve the efficiency of the velocity's

updates that influence the particle's position. In the current implementation, we followed the standard PSO algorithms using the identified velocity and inertia weight settings. We believe the improvement of the velocity's update would enhance the performance of the proposed method.

Acknowledgments

Shuzlina Abdul-Rahman was funded by the Ministry of Science and Technology Innovation, Malaysia (UKM-MGI-NDB0005-2007) and Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia.

References

[1] S. Bandyopadhyay and U. Maulik, Knowledge Discovery and Data Mining, in *Advanced Method for Knowledge Discovery from Complex Data*, S. Bandyopadhyay, U. Maulik, L.B. Holder and D.J. Cook, eds, USA: Springer, 2006.
[2] Y. Saeys, I. Inza and P. Larranaga, A review of feature selection techniques in bioinformatics, *Bioinformatics* **23** (1 October, 2007), 2507–2517.
[3] A. Jain and D. Zongker, Feature selection: Evaluation, application, and small sample performance, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19** (1997), 153–158.
[4] H. Liu and L. Yu, Toward integrating feature selection algorithms for classification and clustering, *IEEE Transactions on Knowledge and Data Engineering* **17** (2005), 491–502.
[5] I. Guyon and A. Elisseff, An introduction to variable and feature selection, *Journal of Machine Learning Research* **3** (2003), 1157–1182.
[6] E. Corchado, X. Wu, E. Oja, Á. Herrero, B. Baruque, L. Shafti and E. Pérez, Feature construction and feature selection in presence of attribute interactions, in *Hybrid Artificial Intelligence Systems*, vol. 5572: Springer Berlin / Heidelberg, 2009, pp. 589–596.
[7] A. Arauzo-Azofra and J.M. Benitez, Empirical study of feature selection methods in classificaiton, in *Empirical Study of Feature Selection Methods in Classification*, 2008.
[8] Luka, ehovin and B. Zoran, Empirical evaluation of feature selection methods in classification, *Intell Data Anal* **14** (2010), 265–281.
[9] J. Wnek and R.S. Michalski, Hypothesis-driven constructive induction in AQ17-HCI: A method and experiments, *Machine Learning* **14** (1994), 139–168.
[10] L. Huan and M. Horoshi, *Feature Extraction, Construction and Selection. A Data Mining Perspective*: Kluwer Academic Publishers Group, 1998.
[11] M. Gutlein, E. Frank, M. Hall and K. Andreas, Large-scale attribute selection using wrappers, in *Proc IEEE Symposium on Computational Intelligence and Data Mining*, 2009.
[12] A. Unler and A. Murat, A discrete particle swarm optimization method for feature selection in binary classification problems, *European Journal of Operational Research* (2010), 528–539.
[13] X. Wang, J. Yang, X. Teng, W. Xia and R. Jensen, Feature selection based on rough sets and particle swarm optimization, *Pattern Recognition Letters* **28** (2007).
[14] M.A. Abido, Optimal power flow using tabu search algorithm, *Electric Power Components & Systems* **30** (2002), 469–483.
[15] A. Al-Shahib, R. Breitling and D. Gilbert, Franksum: New feature selection method for protein function prediction, *International Journal Of Neural Systems* **15** (2005), 259–275.
[16] M. Hauskrecht, R. Pelikan, M. Valko and J.L. Weiler, *Feature Selection and Dimensionality Reduction in Genomics and Proteomics*: Springer, 2006.
[17] E. Guldogan and M. Gabbouj, Feature selection for content-based image retrieval, *Signal, Image and Video Processing,* vol. Volume 2, Number 3 / September, 2008.
[18] S. Nemati, R. Boostani and M.D. Jazi, A Novel Text-Independent Speaker Verification System Using Ant Colony Optimization Algorithm in *ICISP2008*, Berlin, Heidelberg, France, 2008, pp. 421–429.
[19] W. Shang, H. Huang, H. Zhu, Y. Lin, Y. Qu and Z. Wang, A novel feature selection algorithm for text categorization, *Expert Systems with Applications* **33** (2007), 1–5.
[20] H. Song, X. Li and P. Wang, Adaptive feature selection and extraction approaches for image retrieval based on region, *Journal of Multimedia* **5**(1) (2010), 85–92.
[21] M. Kudo and J. Sklansky, Comparison of algorithms that select features for pattern recognition, *Pattern Recognition Letters* **33** (2000), 25–41.

[22] M. Dash and H. Liu, Feature selection for classification, *Intelligent Data Analysis: An International Journal* **1** (1997), 131–156.
[23] M. Guetlin, Large scale attribute reduction using wrappers, in *Computer Science*. vol. Diploma of Computer Science: Freiburg University, 2006.
[24] M. Hall, Correlation-based Feature Subset Selection for Machine Learning, in *Department of Computer Science*. vol. PhD: University of Waikato 1999.
[25] L. Yu and H. Liu, Efficient feature selection via analysis of relevance and redundancy, *Machine Learning Research* **5** (2004), 1205–1224.
[26] M. Gutlein, E. Frank, M. Hall and A. Karwath, Large-scale attribute selection using wrappers, in *Proceedings of IEEE Symposium on Computational Intelligence and Data Mining*, 2009, pp. 332–339.
[27] Y.-H. Choo, A.A. Bakar and A.R. Hamdan, The fitness-rough: A new attribute reduction method based on statistical and rough set theory, *Intelligent Data Analysis* **12** (2008), 73–87.
[28] Z. Zhao and H. Liu, Searching for interacting features in subset selection, *Intelligent Data Analysis: An International Journal* **13** (2009), 207–228.
[29] I.B. Jeffery, D.G. Higgins and A.C. Culhane, Comparison and evaluation of methods for generating differentially expressed gene lists from microarray data, *BMC Bioinformatics* **7** (2006).
[30] A.J. Soto, R.L. Cecchini, G.E. Vazquez and I. Ponzoni, A Wrapper-Based feature selection method for admet prediction using evolutionary computing, in *EvoBIO 2008*, 2008, pp. 188–199.
[31] A. Secker, M.N. Davies, A.A. Freitas, J. Timmis, E. Clark and D.R. Flower, An artificial immune system for evolving amino acid clusters tailored to protein function prediction, in *7th International Conference on Artificial Immune Systems* Phuket, Thailand 2008.
[32] M.H. Aghdam, N. Ghasem-Aghaee and M.E. Basiri, Text feature selection using ant colony optimisation, *Expert Systems with Applications* **36** (2009), 6843–6853.
[33] E.S. Correa, A.A. Freitas and C.G. Johnson, Particle swarm for attribute reduction in bayesian classification to protein function prediction, *Journal of Artificial Evolution and Applications* **2008** (2008).
[34] S.-W. Lin, K.-C. Ying, S.-C. Chen and Z.-J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, *Expert Systems with Applications* **35** (2008), 1817–1824.
[35] Ö. Uncu and I.B. Türksen, A novel feature selection approach: Combining feature wrappers and filters, *Information Sciences* **177** (2007), 449–466.
[36] C.-S. Yang, L.-Y. Chuang, C.-H. Ke and C.-H. Yang, A hybrid feature selection method for microarracy classification, *IAENG International Journal of Computer Science* **35** (2008).
[37] A.-R. Shuzlina, M.H. Zeti-Azura and B. Azuraliza Abu, Multivariate filter and pso in protein function classification, in *2010 International Conference of Soft Computing and Pattern Recognition* (*SoCPar 2010*), University of Cergy Pontoise, Paris, France, 2010.
[38] L. Yu and H. Liu, Feature selection for high-dimensional data: a fast correlation-based filter solution, in *Proceedings of the Twentieth International Conference on Machine Learning* (*ICML-2003*), Washington, DC, 2003.
[39] L. Yu and H. Liu, Efficient feature selection via analysis of relevance and redundancy, *Journal of Machine Learning Research* **5** (2004), 1205–1224.
[40] U. Fayyad and K. Irani, Multi-interval discretization of continuos-valued attributes for classification learning, in *Thirteenth International Joint Conference on Artificial Intelligence*, 1993, pp. 1022–1227.
[41] J. Kennedy and R. Eberhart, *Swarm Intelligence*, San Francisco: Morgan Kaufmann Publishers, 2001.
[42] B. Samantha and C. Nataraj, Application of particle swarm optimization and proximal support vector machines for fault detection, *Swarm Intelligence,* 2009.
[43] R.C. Eberhart and Y. Shi, Tracking and optimizing dynamic systems with particle swarms, in *In Proceedings of the 2001 IEEE congres on evolutionary computing*, New York, 2001, pp. 94–100.
[44] D. Sudholt, Runtime Analysis of Binary PSO, in *The Genetic and Evolutionary Computation Conference* (*GECCO-2008*), Atlanta, Georgia USA, 2008.
[45] R. Poli, J. Kennedy and T. Blackwell, Particle Swarm Optimisation: An Overview, *Swarm Intelligence* **2007** (2007), 33–57.
[46] A. Ratnaweera, S.K. Halgamuge and H.C. Watson, elf-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation* **8** (2004), 240–255.
[47] Y. Shi and R.C. Eberhart, Fuzzy Adaptive particle swarm optimization, in *Proceedings of the 2001 IEEE congres on evolutionary computation*, New York, 2001, pp. 101–106.
[48] M. Clerc and J. Kennedy, The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space, *IEEE Transactions on Evolutionary Computation* (2002), 58–73.
[49] R.C. Eberhart and Y. Shi, Comparing inertia weights and constriction factors in Particle Swarm Optimization, in *In Proceedings of the Congress on Evolutionary Computation*, New York, 2000, pp. 84–88.

[50] H. Liu, F. Hussain, C.L. Tan and M. Dash, Discretizaton: An enabling technique, *Data Mining and Knowledge Discovery* **6** (2002), 393–423.
[51] A.B. Azuraliza, A.O. Zulaiha and M.S. Nor-Liyana, Building a new taxonomy for data discretization techniques, in *2nd Conference on Data Mining and Optimization*, Selangor, Malaysia, 2009.
[52] M.R. Chmielewski and J.W. Grzymala-Busse, Global discretization of continuos attributes as preprocessing for machine learning, *International Journal of Approximate Reasoning* **15** (1996), 319–331.
[53] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*: Morgan Kaufmann, 2000.
[54] B.J. Lee, M.S. Shin, Y.J. Oh, H.S. Oh and K.H. Ryu, Identification of protein functions using a machine-learning approach based on sequence-derived properties, *Proteome Science* **7** (2009).
[55] U. Syed and G. Yona, Enzyme function prediction with interpretable models, *Computational Systems Biology* (2007), 1–33.
[56] I.H. Witten and E. Frank, *Data Mining – Practical Machine Learning Tools and Techniques*, (2nd Edition), Morgan Kaufmann Publishers, 2005.
[57] S.A. Rahman, Z.A. Mohamed Hussein and A.A. Bakar, Data mining framework for protein function prediction, in *Information Technology, 2008. ITSim 2008. International Symposium on*, 2008, pp. 1–5.
[58] C.C. Chang and C.J. Lin, LIBSVM: a library for support vector machines, 2001.
[59] C.-W. Hsu, C.-C. Chang and C.-J. Lin, A Practical Guide to Support Vector Classification, Department of Computer Science, National Taiwan University, Taiwan 2003.
[60] R.D. Finn, J. Tate, J. Mistry, P.C. Coggill, J.S. Sammut, H.R. Hotz, G. Ceric, K. Forslund, S.R. Eddy, E.L. Sonnhammer and A. Bateman, The Pfam protein families database, *Nucleic Acids Research* (2008), D281–D288.
[61] M. Kumar, V. Thakur and G.P. S. Raghava, COPid: Composition based protein identification, *Silico Biology* **8** (2008).
[62] B.J. Lee, H.G. Lee, M.S. Shin and K.H. Ryu, Classification of Ligase Functin Based on Multi-parametric Feature Extracted from Protein Sequences, in *International Conference on Computational Science and Its Application*, Perugia, Italy, 2008, pp. 1096–1106.
[63] S. Umar and Y. Golan, Enzyme function prediction with interpretable models, *Computational Systems Biology,* 2007, pp. 1–33.
[64] A. Al-Shahib, R. Breitling and D. Gilbert, Predicting protein function by machine learning on amino acid sequences – A critical evaluation, *BMC Genomic* **8** (2007), 78.
[65] I.B. Jeffery, D.G. Higgins and A.C. Culhane, Comparison and evaluation of methods for generating differentially expressed gene lists from microarray data, *BMC Bioinformatics* **7** (2006).
[66] C.-S. Yang, L.-Y. Chuang, C.-H. Ke and C.-H. Yang, A hybrid feature selection method for microarray classification, *IAENG International Journal of Computer Science* **35** (2008).
[67] J. Demsar, Statistical comparisons of classifiers over multiple data sets, *Journal of Machine Learning Research* (2006), 1–30.