The Journal of Supercomputing (2022) 78:19725–19753
https://doi.org/10.1007/s11227-022-04650-w

Hybrid feature selection based on SLI and genetic algorithm for microarray datasets

Sedighe Abasabadi[1] · Hossein Nematzadeh[1] · Homayun Motameni[1] · Ebrahim Akbari[1]

Accepted: 8 June 2022 / Published online: 30 June 2022

Abstract

One of the major problems in microarray datasets is the large number of features, which causes the issue of "the curse of dimensionality" when machine learning is applied to these datasets. Feature selection refers to the process of finding optimal feature set by removing irrelevant and redundant features. It has a significant role in pattern recognition, classification, and machine learning. In this study, a new and efficient hybrid feature selection method, called $\mathrm{Ga}_{\mathrm{rank\&rand}}$, is presented. The method combines a wrapper feature selection algorithm based on the genetic algorithm (GA) with a proposed filter feature selection method, SLI-γ. In $\mathrm{Ga}_{\mathrm{rank\&rand}}$, some initial solutions are built regarding the most relevant features based on SLI-γ, and the remaining ones are only the random features. Eleven high-dimensional and standard datasets were used for the accuracy evaluation of the proposed SLI-γ. Additionally, four high-dimensional well-known datasets of microarray experiments were used to carry out an extensive experimental study for the performance evaluation of $\mathrm{Ga}_{\mathrm{rank\&rand}}$. This experimental analysis showed the robustness of the method as well as its ability to obtain highly accurate solutions at the earlier stages of the GA evolutionary process. Finally, the performance of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ was also compared to the results of GA to highlight its competitiveness and its ability to successfully reduce the original feature set size and execution time.

Keywords Hybrid feature selection · Genetic algorithm · High-dimensional datasets

✉ Hossein Nematzadeh
hossein.nematzadeh@iau.ac.ir

[1] Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

1 Introduction

Feature selection essentially aims to reduce both the redundancy and dimension of high-dimensional datasets during the classification process. Feature selection is also a process of finding optimal feature set by adding relevant and additional features. Feature selection algorithms are used to speed up the learning process of classifiers and to improve the classification accuracy. In general, feature selection methods are categorized into five types: filter, wrapper, embedded, ensemble, and hybrid. In filter methods, the redundant and unrelated features are removed, and an optimal feature subset is selected. The filter methods do not apply the classification algorithms to feature selection processes; therefore, these methods have low computational cost and typically show the best performance in case of high-dimensional datasets. On the contrary, the wrapper methods rely on the predictive performance of a classifier to evaluate the selected feature sets quality. It has two steps: First, a subset of features is selected; second, the selected features are evaluated. Afterward, the two steps are repeated until the fulfilment criteria are met. The highest learning performance feature set is returned as the selected feature. The wrapper methods suffer from high computational cost; however, they finally provide better results. On the other hand, the embedded methods are made up of the filter and wrapper methods, which integrate feature selection in the model learning with the learning algorithm. As they do not evaluate the feature set repeatedly, they are much more efficient than the wrapper methods. Ensemble methods build on the assumption that combining the output of multiple feature selection methods is better than using the output of any single method. Creating a feature selection ensemble involves applying different feature selection methods, each providing their output and then combining the outputs of the single models [1]. The hybrid methods combine two or more feature selection algorithms to shape a new plan to workout problem. These methods take the advantage of the sub-algorithms and, as a consequence, are stronger compared with traditional methods. In this study, a novel hybrid feature selection method is introduced through combining our proposed filter method and GA to eliminate the redundant and irrelevant samples and reduce the final feature set size. The experiments conducted in this study revealed the potency of the proposed feature selection methods by comparing their classification accuracy with other existing feature selection methods. Three different classifiers were applied to the selected datasets to test the validity of the proposed algorithm. In the following, the main contributions of our work are described in detail.

- **Novelty**

  (1) Introducing a rank-based filter feature selection method, SLI-γ, minimizing feature redundancy and maximizing relevancy between features and class label.
  (2) Introducing a novel hybrid approach based on our proposed filter method and GA in a way to be applicable to high-dimensional problems. The aim is

to identify the optimal subset of relevant features, maximize classification accuracy, and minimize the size of final feature set and the execution time.

- **Effectiveness**
  The hybrid algorithm capitalizes on the advantages of the SLI-γ and GA. The features selected by our algorithm show more accurate identification rates compared with the existing feature selection approaches. The proposed algorithm maximizes the classification accuracy and, at the same time, minimizes the number of selected features and the execution time of GA when applied to the feature selection problems.
- **Robustness**
  Three different classifiers were tested in this study on the selected datasets. All classifiers produced stable classification accuracy.
  To achieve the objectives defined for the current paper, a GA-based wrapper feature selection method is combined with a ranking-based filter feature selection method.

The rest of this paper is organized as follows: Sect. 2 reviews the related work. Then, Sect. 3 presents the background of the above-mentioned algorithms. Next, Sect. 4 discusses the proposed filter method and the proposed hybrid algorithm. Afterward, Sect. 5 presents and analyzes the experimental results. Finally, the conclusions of the research are given in Sect. 6.

2 Related work

Many researchers have shown the effectiveness of feature selection methods in removing unrelated features and improving the learning performance. Research in this field can be divided into two categories:

1. Studies that have examined the efficiency of the filter, wrapper, embedded, and ensemble feature selection methods.
2. Studies that have used a hybrid feature selection method to achieve an optimized feature subset.

A number of the studies that have used filter, wrapper, embedded, and ensemble methods to determine a practical feature subset are discussed in the following.

Maleki et al. [2] used GA to select the features. In their proposed method, the data are first preprocessed; then, the lost data are deleted or replaced with the appropriate data. The uncompleted rows of datasets were filled with the software function to use the average of other values. GA was then applied to this dataset to find the best combination of features, which provided the most relevant features. Pardo et al. [3] analyzed the comparability of six advanced feature selection methods: Interact, CFS, Chi-square, ReliefF, Infogain, and MRMR. Accordingly, they introduced new criteria based on accuracy, execution time, and stability. The results indicated that the filter methods are the most scalable methods of selecting the features, while the Interact, ReliefF, and MRMR methods are the most accurate ones. Gu et al. [4] studied the scalability of modern feature selection algorithms. They compared the performance and power of these algorithms in selecting related features and removing irrelevant features without allowing the number of features to increase. For the scalability analysis, they set new evaluation criteria based on the selection accuracy, execution time, and stability of selected features. Abasabadi et al. [1] introduced a heterogeneous ensemble feature selection method with automatic thresholding capability. To create diversity in the ensemble, they used three filter methods, i.e., ReliefF, Mutual Congestion, and a proposed filter method called Sorted Label Inference. They used the non-dominated sorting method to aggregate the ranking lists, which allows automatic thresholding in the proposed ensemble method. The experimental evaluation of the method using six different high-dimensional datasets and two standard datasets showed the efficiency of the proposed method. Pardo et al. [5] used the filter and embedded methods instead of the usual ensemble method. Their study aimed to diversify and increase the order of the feature selection process to take advantage of the strengths of individual selectors and overcome their weaknesses. Depending on how the data was distributed and the feature selection method was used, two methods were employed, which were tested with SVM classifier. The experimental evaluation of the methods using seven different datasets showed the competence of the proposed ensemble.

Some researchers proposed the use of the hybrid feature selection methods to remove unrelated features and to achieve high accuracy. A number of these studies are discussed below.

Sadeghian et al. [6] proposed the Information Gain binary Butterfly Optimization Algorithm (IG-bBOA) to overcome the S-bBOA constraints. In the first phase of this algorithm, 80% of the irrelevant and redundant features are removed using the Minimal Redundancy–Maximal New Classification Information (MRMNCI) method. Then, in the second phase, the best feature subset is selected using IG-bBOA. Finally, a similarity-based ranking method is used to select the final features subset. The experimental results on six standard datasets showed the efficiency of the proposed method in improving the classification accuracy and selecting the minimum number of features in most cases. Nematzadeh et al. [7] presented a hybrid feature selection method to be applied to large binary medical datasets. Their proposed method consists of three parts. First, the whale algorithm is used to remove half of unrelated features. Next, in the second stage, the remaining features are ranked based on a frequency-based method, called mutual congestion. Finally, a majority vote with a threshold of 10 is applied to the best set of features obtained by the selection of the features. Jain et al. [8] proposed a hybrid feature selection method for sentiment classification. They used GA and a combination of filter methods such as IG, CHI Square, and GINI Index. First, they acquired features from filter methods and then implement the UNION SET Operation to achieve the reduced feature set. Afterward, GA was used to raise the feature set further. They used an ensemble method based on the error rate. To test their proposed hybrid feature selection and ensemble classification approach, they considered four support vector machine (SVM) classifier variants. They applied their proposed method to the UCI ML Datasets. The experimental results showed that their proposed method performed best in all datasets.

Although GA can accurately select meaningful features, it suffers from the execution time problem and also its convergence is highly dependent on the initial population. The results of filter methods should be used as the initial population for GA to reduce the execution time and select the optimal feature subset.

3 Preliminaries

In this section, the basic concepts exploited within the proposed method, i.e., filter feature selection method, genetic algorithm (GA), and artificial neural network (ANN), are reviewed.

3.1 Filter feature selection methods

Filter methods are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, features are selected based on their scores in various statistical tests for their correlation with outcome variable. ReliefF, SVM-Rank, Information Gain, MRMR, and Fisher score are well-known filter methods.

3.2 Genetic algorithm

Genetic algorithm (GA) was first introduced by John H. Holland in the 1960s and was developed later by Holland and his students in the 1960s and 1970s [9]. The basic idea behind GA is to evolve a generation of possible candidate solutions to a problem using several operators. At first, an initial population is created using a random process. Next, reproduction, crossover, and mutation operators are used to generate successive populations from the initial population. Reproduction is a process based on the fitness function of each string, which identifies how "good" a feature is. Thus, features with higher fitness values have a bigger probability of contributing offspring to the next generation. Crossover is a process in which members of the last population are mated at random in the mating pool. Therefore, combining two parents results in the generation of a pair of offspring that hopefully contains improved fitness values. The mutation is a genetic operator used to maintain genetic diversity from one generation of a population of the GA chromosomes to the next. Algorithm 1 shows the pseudo-code of GA.

Algorithm 1: Pseudo-code for GA

```
Input: np: base population size, ni: number of iterations, pc: rate of crossover, pm: rate of mutation
Output: the global best solution
Generate initial population of size np
Evaluate initial population according to the fitness function
While (current_iteration<=ni)
    Breed pc×np new solutions
    Select two parent solutions from current population
    Form offspring's solutions via crossover
    If (rand (0,1) < pm)
        Mutate the offspring's solutions
    End if
    Evaluate each child solution according to the fitness function
    Add offspring to population
    Population size is now MaxPop=np×(1+pc)
    Remove the pc×np least-fit solutions from population
End while
Return the global best solution
```

3.3 Artificial neural network

Artificial neural network (ANN) was first introduced by McCulloch and Pitts [10], which was in fact a simulation of human brain. The term is obtained from biological neural networks that evolve the form of a human brain. Similar to human brain, ANN has neurons associated with one another in several layers of the network. These neurons are known as nodes. ANN is a set of connected input–output network in which weight is associated with each connection. It consists of one input layer, one or more hidden layer(s), and one output layer. Multilayer perceptron is most generally conceded type of ANNs, which is made up of one input and output layers accompanying one or more hidden layers. Neural network learns by adjusting the weight of connection. By updating the weight iteratively, the performance of network is improved. Neurons of the neural network are activated by the weighted sum of inputs, which is shown as a transfer function by Eq. (1) [11].

$$\sum_{i=1}^{n} Wi * Xi + b \tag{1}$$

where Wi ($i = 1, 2, \ldots$) and b are the synaptic weights and bias of the perceptron, respectively. The interconnection weights are optimized during training, until the network reaches the specified level of accuracy [12].

4 The proposed hybrid feature selection method

In this section, a new, simple, and effective hybrid feature selection method, called $\text{Ga}_{\text{rank\&rand}}$, is introduced. The goal is to achieve the smallest subset of features as well as the highest classification accuracy. Figure 1 shows the framework of our proposed hybrid feature selection method.

As shown in this figure, our proposed method uses a two-stage procedure to obtain the optimal feature subset. First, our proposed filter feature selection method, called SLI-γ, was applied to the datasets to rank features based on the predictive ability of each feature. Those at the top are more likely to better classify the samples. Afterward, 1% of best ranked features were selected. At the second stage, to start with the best initial solutions, the GA population was initialized with those 1% of the features at the top of the ranking. GA was applied to dataset containing only 1% of selected features, searching for the best feature subset in the solutions space. To calculate the fitness of each individual, two different classifiers (i.e., KNN and ANN) were used as ANN provides more accurate results and KNN improves the execution time. Algorithm 2 shows the $\text{Ga}_{\text{rank\&rand}}$ hybrid feature selection method.

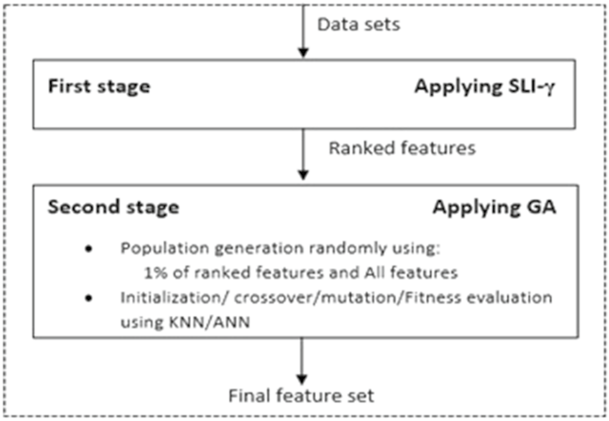
Data sets
First stage
Applying SLI-γ
Ranked features
Second stage
Applying GA
Population generation randomly using:
1% of ranked features and All features
Initialization/ crossover/mutation/Fitness evaluation using KNN/ANN
Final feature set

Fig. 1 Framework of the $\text{Ga}_{\text{rank\&rand}}$ hybrid feature selection method

Algorithm 2: Pseudo-code for $Ga_{rank\&rand}$ hybrid feature selection method

```
Input: Dataset, Maxiteration, Pm, Pc
Output: Final feature set
Generate different population size using 1% of SLI-γ best ranked features randomly
    and Generate others using all features randomly.
While iteration<=Maxiteration
Iteration=iteration+1
Calculate the fitness of each individual using KNN/ANN
Select the individuals according to their fitness
Perform crossover with probability pc
Perform mutation with probability pm
Population= select individuals after crossover and mutation
End while
Return Final Feature Set
```

4.1 Feature ranking by applying SLI-γ

The current paper proposes a filter feature selection method, called SLI-γ, whose schema is illustrated in Fig. 2. The first step is to sort each feature based on its samples' values. Sorting each feature can result in the relocation of each sample and its corresponding label. Then, at the second step, the interference between the relocated labels must be found. Thus, the area where the positive and negative labels intersect must be found. Interference region is the region where after a consequence of positive labels (if the first label is positive), a negative label appears; it continues until the last positive labels appears. The area between these two

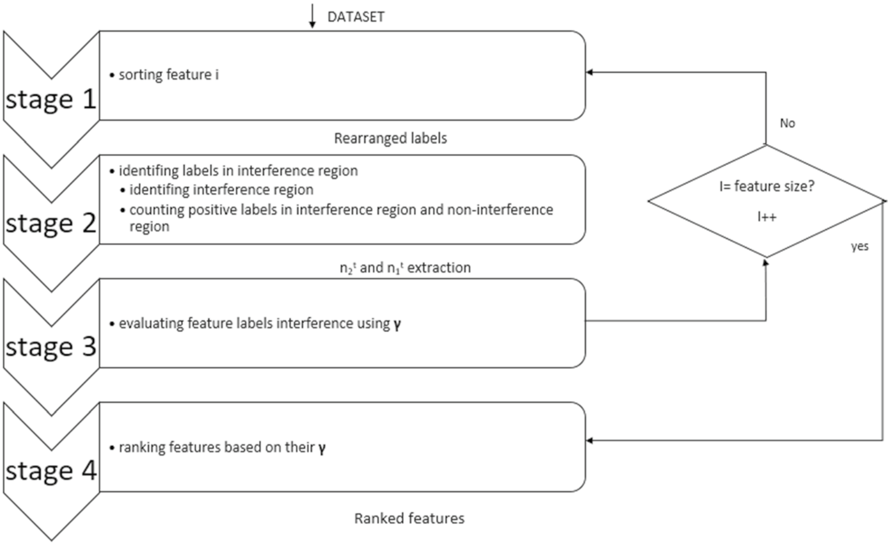
DATASET
stage 1
• sorting feature i
Rearranged labels
stage 2
• identifing labels in interference region
• identifing interference region
• counting positive labels in interference region and non-interference region
n_2^t and n_1^t extraction
stage 3
• evaluating feature labels interference using γ
stage 4
• ranking features based on their γ
Ranked features
No
I= feature size?
I++
yes

Fig. 2 The schema of the proposed filter method

appearances of negative and positive labels is called interference region. Moreover, interference region is the region where after a consequence of negative labels (if the first label is negative), a positive label appears; it continues until the last negative label appears. The area between these two appearances of positive and negative labels is interference region. Figure 3 shows a graphical presentation of the interference area. The green color area shows the interference area. Then, it is time to enumerate the number of positive labels in the interference region and non-interference region. The positive label size in the interference region and the positive label size in the non-interference region are characterized with $\mathrm{n^t}_2$ and $\mathrm{n^t}_1$, respectively. In the example represented in Fig. 3a, the positive label size in the interference region $n^t{}_2$ is 2, while the positive label size in the non-interference region $n^t{}_1$ is 3.

Algorithm 3 shows the pseudocode of the extraction of the interference region. Algorithm 4 shows the extraction pseudocode for $n^t{}_2$ and $n^t{}_1$. In the third step, the frequency of positive labels in each region is determined based on our proposed criterion, γ. To calculate the frequency of positive labels in an area, the interference coefficients of γ is proposed using $n^t{}_2$ and $n^t{}_1$, according to Eq. (2):

$$\gamma = {}^{n_2^t}\!/_{n_1^t} + n_2^t \tag{2}$$

where γ is the number of positive labels in the interference area, divided by the total number of positive labels, which has a value between 0 and 1. The value 1 is obtained when all positive labels are in the interference region. On the other hand, when there are no samples in the interference region, the obtained value would be 0. The best feature is the one with the lowest γ. A lower interference shows more similarity between the classification based on this unique feature and the overall classification based on the whole features. This process is repeated for all features in the dataset. Finally, in the fourth step, all features are sorted by their γ values.

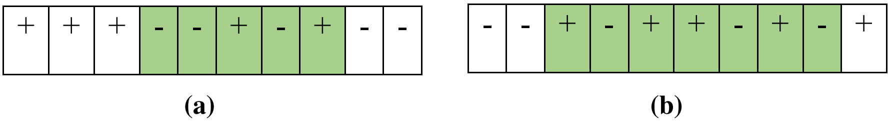

Fig. 3 **a** Interference region when labels start in positive, **b** interference region when labels start in negative

Algorithm 3: Pseudo-code for extracting interference region I

```
Input: feature size
Output: interference region
Initialization:
for each feature do
    if feature. label1 is Positive/Negative then
        store first occurrence of negative/Positive label as s
        store last occurrence of positive/Negative label as t
    end if
end for
Return interference region I[s, t]
```

Algorithm 4: Pseudo-code for extracting n_1^t and n_2^t

```
Input: sample size, inference region I [s, t]
Output: n1t and n2t
Initialization:
$n_1^t$ = s-1; //Positive label size in non-interference region
for each j from I[s t] do
    if feature.labelj is positive then
        $n_2^t$ = $n_2^t$ +1; //Positive label size in interference region
    end if
end for
Return $n_1^t$ and $n_2^t$
```

4.2 Applying GA

In this step, high-ranked features of SLI-γ are given as input to GA, which is explained in detail in the following subsection. To preserve the simplicity and efficiency of the proposed method, the feature subsets are represented as binary vectors. More precisely, the size of the vector is equal to the number of features in dataset. When the ith component of a vector is equal to 1, it means that the subset includes the ith feature; otherwise, the component is set to 0.

4.3 Initial population generation

The initial quantification of population vectors plays an important role in the GA method. In $\text{Ga}_{\text{rank\&rand}}$, some features are selected randomly for initial valuation among 1% of the features with the highest SLI-γ rank. On the other hand, some vectors are generated by the basic method in evolutionary algorithms, i.e., random initial valuation. These vectors certainly produce solutions with less relevant features to reduce the excessive pressure exerted by the first method on some areas.

4.4 Cost function

While running GA, in each iteration, the selected features in the current population are evaluated, and their fitness are ranked based on a KNN-based or ANN-based classification error. With the use of KNN as classifier, lower-cost selected feature sets have higher chance of being transmitted into the next generation. As GA iterates, it reduces the classification error and selects the features with the least fitness, and the smallest error rate is finally selected by GA. With using ANN as classifier, after the initialization, there is a need to assign a fitness value to each feature set in the population. We train each neural network and then evaluate their error with the selection instances. A significant selection error means high cost function. Feature sets with less cost function are more likely to be selected for recombination. The objective of feature subset selection is reducing the dimensionality of the dataset as well as decreasing the classification error to achieve higher accuracy. However, these two objectives are contradictory. The reduction of features leads to lower classification accuracy; thus, the use of classification error as cost function is not enough for obtaining optimal feature set. Here, the contradiction is well addressed by using a single objective function. The cost function is defined as a weighted linear aggregation function that can be calculated using Eq. (3).

$$Z = E \times \left(1 + \beta \times \left(\frac{\mathrm{SF}}{F}\right)\right) \tag{3}$$

where SF and F represent the number of features in the selected feature subset and total number of features, respectively, whereas β is a control parameter used to adjust the weight of the feature set in the fitness function. $\beta \in [0,1]$ is the cost of adding features. Higher β values lead to the selection of less features, and vice versa. The above fitness function is used in conjunction with various evolutionary algorithms to find out the optimal feature subset. The fitness function Z is evaluated using classifier error E in terms of the final reduction of feature set and classification accuracy.

5 Experimental study

In this section, first, the datasets are briefly described. The analysis results of SLI-γ are applied on high-dimensional and standard datasets. Subsequently, our proposed hybrid feature selection technique is analyzed.

5.1 Datasets and parameter setting

The $\mathrm{Ga}_{\mathrm{rank\&rand}}$ was tested using the KNN classifier and the SLI-γ on MATLAB (Ver. 2016) on the HP G9 server with 180 GB of memory and 36 processor CPU e5-2699 v3 2.6 GHz, and Windows Server 2012 operating system. The results are for an average of 100 runs, using the tenfold Monte Carlo cross-validation. SLI-γ was tested on 11 different binary datasets selected from the bioinformatics and biomedical domains, in which Liver, Diabetes, Kidney, and Wisconsin are standard datasets. In contrast, SMK_CAN_187, LeukemiaB, GLI_85, Ovarian, CNS, Colon, and COVID-19 are high-dimensional datasets. COVID-19 is originally a three-label dataset ("no virus," "other virus," and "SC2"); the current study considered "no virus" and "other virus" as one label as "non covid" and SC2 as "covid," so that our proposed method could be applied to the COVID-19 for the purpose of accuracy evaluation. Table 1 shows the specifications of these datasets, which include the number of samples, features and classes. $\mathrm{Ga}_{\mathrm{rank\&rand}}$ was tested on four different high-dimensional binary datasets of LeukemiaB, GLI_85, CNS, and Colon. Meanwhile, the $\mathrm{Ga}_{\mathrm{rank\&rand}}$ tests using ANN classifier were performed by means of MATLAB (Ver. 2016) on the HP G9 server with 40 GB of memory and 24 core CPU e5-2690 v3 2.6 GHz, and Windows Server 2012 operating system.

5.2 Evaluation metric

Accuracy is the criterion used to evaluate the efficiency of the proposed method. Accuracy indicates the correctness of the classification of positive and negative samples. Additionally, true positive (TP) and true negative (TN) are positive and negative samples, respectively, which are correctly classified. False positive (FP) and false negative (FN) are positive and negative samples, respectively, that are not properly classified.

5.3 Classification methods

Implementing feature selection methods on popular classification algorithms is an accepted way to test the performance of any feature selection method [1]. However, class prediction depends on the applied classification algorithm. In this research, several classification methods have been used to test the feature selection method in order to achieve results independent of the classifier. From the available classifiers

Table 1 Datasets used in the study

Dataset	Samples	Features	Classes	References
Colon	62	2000	2	[15]
CNS	60	7129	2	[16]
LeukemiaB	72	7129	2	[17]
Ovarian	253	15,154	2	[18]
GLI_85	85	22,283	2	[19]
SMK_CAN_187	187	19,993	2	[20]
Liver	569	30	2	[21]
Wisconsin	194	34	2	[21]
COVID19	234	15,980	2	[22]
Diabetes	570	17	2	[23]
Kidney	200	29	2	[24]

introduced in the literature, SVM, NB, and DT were selected to be used in this research for the accuracy evaluation of SLI-γ. SVM, a classification method that has received much attention in recent years, is based on the idea of structural risk minimization (SRM). SVM methods have performed extremely well in a wide range of applications and have been used as powerful tools in solving classification problems [3]. SVM in this research uses the linear kernel function with box constraint of 2. Naïve Bayes can achieve relatively good performance on classification tasks. Naïve Bayes greatly simplifies learning by assuming that features are independent given the class variables. In simple terms, a Naïve Bayes classifier assumes that the presence of a particular feature of a class is unrelated to the presence of any other feature. In spite of their naïve design and apparently over-simplified assumptions, Naïve Bayes classifiers have worked quite well in many complex real-world situations [13]. Decision trees are favored prediction tools as they make a model that is easy to clarify. Each leaf node can be given as an if/then rule. The logical rules come after a decision tree similarly bear a resemblance to human reasoning and are allegedly alluring to decision-makers, who is pleasant with models that they can understand. Decision trees are also multivariate that can model a wide range of data distributions. Furthermore, decision trees can handle data of different types without requiring any transformation of the data. Most importantly, decision trees have the capability to break down a complex decision-making process into a collection of simpler decisions, thus providing a solution that is often easier to interpret [14]. Moreover, ANN and KNN were used for the evaluation of the $\text{Ga}_{\text{rank\&rand}}$ accuracy. Table 2 shows the calculated accuracy of each classifier before applying the proposed method to the dataset using the Monte Carlo cross-validation.

5.4 Numerical results and discussion

Regarding the performance analysis of the proposed methods, the effectiveness of the SLI-γ needs to be discussed first. Thus, in Sect. 5.4.2 it is experimentally shown to what extent this algorithm should be used to discard irrelevant features. Then, in

Table 2 Accuracy results of classifiers for datasets without applying the feature selection methods

Datasets	NB (%)	SVM (%)	DT (%)
LeukemiaB	98	97	84
Liver	93	97	92
Ovarian	97	100	97
CNS	63	64	57
Colon	56	82	74
GLI_85	80	90	78
SMK_CAN_187	61	73	59
Wisconsin	67	71	69
COVID-19	59	82	73
Diabetes	89	80	93
Kidney	100	100	100

Sect. 5.4.3, it is experimentally shown that why GA should be used as an evolutionary algorithm to discard the irrelevant features. Moreover, the convergence analysis of these algorithms is discussed. In Sect. 5.4.4, $\mathrm{Ga}_{\mathrm{rank\&rand}}$ is applied to the benchmark datasets to identify the best features in the proposed method. Moreover, the $\mathrm{Ga}_{\mathrm{rank\&rand}}$ method is compared with GA.

5.5 Evaluation of SLI-γ

In this section, a comprehensive experiment was performed on four standard datasets and seven high-dimensional medical datasets to evaluate SLI-γ in terms of accuracy. The performance of SLI-γ was first evaluated and compared with that of three other rankers. Experiments indicated that SLI-γ is able to effectively eliminate the unrelated features. Tables 3, 4 and 5 show that SLI-γ increased the classification accuracy of all datasets compared to the data presented in Table 2. For accuracy comparison, we have used four feature sets for each dataset, and the highest accuracy on each feature set of each dataset among all rankers is bold. More specifically, in Colon, CNS, Diabetes, Kidney, GLI_85, Liver, and Ovarian, SLI-γ increased accuracy with least number of features with all classifiers in comparison with the data presented in Table 2.

In CNS, Colon, and COVID19, SLI-γ increased accuracy with least number of features in DT in comparison with other rankers. In GLI_85 and SMK_CAN_187, SLI-γ increased accuracy with least number of features in NB and SVM in comparison with other rankers. In Liver, LeukemiaB, and Ovarian, SLI-γ increased accuracy with least number of features in SVM in comparison with other rankers. In LeukemiaB, SLI-γ increased accuracy with the least number of features with SVM and DT in comparison with Table 2. In SMK_CAN_187 and Wisconsin, SLI-γ increased accuracy with least number of features with NB and DT in comparison with Table 2. In Wisconsin, SLI-γ increased accuracy with least number of features in NB and DT in comparison with other rankers. In COVID19, SLI-γ increased accuracy with the least number of features in SVM and DT in comparison with other rankers. SLI-γ seems to be a powerful ranking for high-dimensional and standard datasets. As indicated clearly by the obtained results, SLI-γ increased the classification accuracy of all datasets compared to the results presented formerly in Table 2. Therefore, SLI-γ creates a better model with NB, DT, and SVM. In addition, it increases accuracy in all classifiers. As a result, SLI-γ is used as a ranker in this paper.

Number of features: Abasabadi et al. [1] introduced an automatic thresholding feature selection method, called ATFS, which specifies the threshold value for each dataset. To determine the number of features for testing and comparison of our proposed filter method on benchmark datasets, the ATFS method results were used for these datasets. In Colon, the ATFS method resulted in 14 features as the best and smallest final feature subset. With this threshold value, SLI-γ increased the accuracy compared to other filter methods with all classifiers. In CNS, 8 features were obtained where SLI-γ surpassed other filter methods by NB and SVM. However, in SMK_CAN_187, 9 features were obtained where SLI-γ outperformed other rankers with all classifiers. In GLI-85, 6 features were selected by ATFS, where SLI-γ

Table 3 Comparison of SLI-γ with common rankers using NB classifier

Datasets	Rankers	Feature sets			
Colon		F_SET1 = 14	F_SET2 = 34	F_SET3 = 105	F_SET4 = 304
	ReliefF	0.76	0.72	**0.85**	**0.79**
	SVM-RANK	0.45	0.55	0.58	0.54
	IG	0.64	0.55	0.52	0.54
	SLI-γ	**0.79**	**0.79**	0.71	0.64
CNS		F_SET1 = 8	F_SET2 = 16	F_SET3 = 96	F_SET4 = 430
	ReliefF	0.6	0.64	**0.75**	**0.64**
	SVM-RANK	0.51	0.52	0.59	0.62
	IG	0.51	0.52	0.57	0.54
	SLI-γ	**0.68**	**0.71**	0.56	0.56
GLI_85		F_SET1 = 6	F_SET2 = 15	F_SET3 = 149	F_SET4 = 604
	ReliefF	0.87	0.86	0.89	0.87
	SVM-RANK	0.57	0.75	0.81	0.83
	IG	0.74	0.74	0.78	0.81
	SLI-γ	**0.91**	**0.93**	**0.91**	**0.91**
Liver		F_SET1 = 5	F_SET2 = 10	F_SET3 = 13	F_SET4 = 27
	ReliefF	0.88	0.9	0.9	0.93
	SVM-RANK	0.86	0.91	0.92	0.93
	IG	**0.95**	**0.95**	**0.94**	0.93
	SLI-γ	0.94	0.93	0.93	**0.94**
LeukemiaB		F_SET1 = 4	F_SET2 = 10	F_SET3 = 103	F_SET4 = 345
	ReliefF	0.9	0.94	**0.97**	**1**
	SVM-RANK	0.67	0.61	0.77	0.91
	IG	0.66	0.61	0.89	0.91
	SLI-γ	**0.96**	**0.95**	0.93	0.94
Ovarian		F_SET1 = 3	F_SET2 = 6	F_SET3 = 193	F_SET4 = 690
	ReliefF	**0.97**	**0.97**	**0.98**	**0.97**
	SVM-RANK	0.79	0.8	0.86	0.87
	IG	0.63	0.67	0.84	0.85
	SLI-γ	**0.97**	**0.97**	**0.98**	**0.97**
SMK_CAN_187		F_SET1 = 9	F_SET2 = 21	F_SET3 = 142	F_SET4 = 536
	ReliefF	**0.67**	0.66	**0.69**	**0.67**
	SVM-RANK	0.55	0.56	0.62	0.6
	IG	0.59	0.66	0.62	0.62
	SLI-γ	**0.67**	**0.68**	**0.69**	**0.67**
Wisconsin		F_SET1 = 6	F_SET2 = 12	F_SET3 = 18	F_SET4 = 21
	ReliefF	0.76	**0.72**	0.66	**0.68**
	SVM-RANK	0.72	0.7	**0.68**	**0.68**
	IG	0.73	0.7	0.67	0.66
	SLI-γ	**0.78**	**0.72**	**0.68**	**0.68**

Table 3 (continued)

Datasets	Rankers	Feature sets			
COVID19		F_SET1 = 10	F_SET2 = 18	F_SET3 = 29	F_SET4 = 47
	ReliefF	**0.55**	**0.55**	**0.57**	**0.59**
	SVM-RANK	0.51	0.51	0.53	0.55
	IG	0.51	0.54	0.52	0.56
	SLI-γ	0.51	0.5	0.49	0.5
Diabetes		F_SET1 = 4	F_SET2 = 8	F_SET3 = 12	F_SET4 = 16
	ReliefF	0.80	0.82	**0.89**	**0.89**
	SVM-RANK	0.69	0.84	0.88	**0.89**
	IG	0.7	0.75	0.82	**0.89**
	SLI-γ	**0.87**	**0.89**	0.88	**0.89**
Kidney		F_SET1 = 7	F_SET2 = 14	F_SET3 = 21	F_SET4 = 28
	ReliefF	0.91	0.93	0.97	**1**
	SVM-RANK	0.65	0.73	0.83	**1**
	IG	0.56	0.65	0.79	**1**
	SLI-γ	**1**	**1**	**1**	**1**

obtained the best accuracy by all classifiers. In LeukemiaB, with only 4 features that were selected by ATFS, SLI-γ with all classifiers achieved the best accuracy. In COVID19, 10 features were obtained by ATFS, where SLI-γ surpassed other filter methods by DT and SVM. In Wisconsin, with 6 features introduced by ATFS, SLI-γ received the best accuracy for all classifiers. In Ovarian with only 3 features presented by ATFS, SLI-γ improved accuracy with all classifiers. In Liver, SLI-γ surpassed all rankers with 5 features presented by ATFS. In diabetes, the ATFS ensemble method resulted in 4 features as the best and smallest final feature subset. With this threshold value, SLI-γ increased the accuracy compared to other filter methods with SVM, NB, and DT. In Kidney, the ATFS ensemble method resulted in 7 features as the best and smallest final feature subset. With this threshold value, SLI-γ increased the accuracy compared to other filter methods with all classifiers.

5.5.1 Removal rate analysis

After ranking the features based on their relevancy using SLI-γ, it is time to acquire a unique final feature set size for all datasets. Ten experiments were done on each dataset with different discarding rates: 99, 90, 80, 70, 60, 50, 40, 30, 20, and 10%. We used three different classifiers (SVM, NB, and DT) to acquire the best results. As shown in Fig. 4, the best accuracy with most classifiers was achieved when the SLI-γ method discarded 99% of the all features. It means the SLI-γ algorithm could effectively identify relevant features by discarding a huge number of irrelevant features correctly.

Table 4 Comparison of SLI-γ with common rankers using SVM classifier

Datasets	Rankers	Feature sets			
Colon		F_SET1$_{=}$14	F_SET2 = 34	F_SET3 = 105	F_SET4 = 304
	ReliefF	0.69	0.69	0.82	**0.84**
	SVM-RANK	0.66	0.74	**0.85**	0.81
	IG	0.8	0.71	0.77	**0.84**
	SLI-γ	**0.83**	**0.84**	0.83	**0.84**
CNS		F_SET1 = 8	F_SET2 = 16	F_SET3 = 96	F_SET4 = 430
	ReliefF	0.58	0.72	**0.75**	**0.77**
	SVM-RANK	0.45	0.53	0.53	0.61
	IG	0.63	0.59	0.68	0.62
	SLI-γ	**0.71**	**0.73**	0.62	0.66
GLI_85		F_SET1$_{=}$ 6	F_SET2 = 15	F_SET3 = 149	F_SET4 = 604
	ReliefF	0.9	0.83	0.92	**0.93**
	SVM-RANK	0.52	0.73	0.85	0.91
	IG	0.69	0.69	0.86	0.89
	SLI-γ	**0.92**	**0.91**	**0.95**	**0.93**
Liver		F_SET1 = 5	F_SET2 = 10	F_SET3 = 13	F_SET4 = 27
	ReliefF	0.91	0.95	0.96	**0.97**
	SVM-RANK	0.88	**0.97**	**0.97**	**0.97**
	IG	**0.94**	**0.97**	**0.97**	**0.97**
	SLI-γ	**0.94**	0.95	0.95	**0.97**
LeukemiaB		F_SET1 = 4	F_SET2 = 10	F_SET3 = 103	F_SET4 = 345
	ReliefF	0.93	0.92	**0.98**	**0.99**
	SVM-RANK	0.65	0.59	0.88	0.93
	IG	0.64	0.6	0.9	0.96
	SLI-γ	**0.96**	**0.94**	**0.98**	**0.99**
Ovarian		F_SET1 = 3	F_SET2 = 6	F_SET3 = 193	F_SET4 = 690
	ReliefF	**0.97**	**0.97**	**1**	**1**
	SVM-RANK	0.77	0.85	0.99	1
	IG	0.62	0.67	0.97	0.98
	SLI-γ	**0.97**	**0.97**	**1**	**1**
SMK_CAN_187		F_SET1 = 9	F_SET2 = 21	F_SET3 = 142	F_SET4 = 536
	ReliefF	**0.67**	0.65	0.65	**0.69**
	SVM-RANK	0.57	0.54	**0.69**	0.65
	IG	0.59	0.63	0.64	0.68
	SLI-γ	**0.67**	**0.66**	0.63	**0.69**
Wisconsin		F_SET1 = 6	F_SET2 = 12	F_SET3 = 18	F_SET4 = 21
	ReliefF	0.51	0.56	0.57	0.58
	SVM-RANK	0.58	**0.71**	**0.63**	**0.68**
	IG	**0.63**	0.59	0.53	0.58
	SLI-γ	0.58	0.57	0.58	0.63

Table 4 (continued)

Datasets	Rankers	Feature sets			
COVID19		F_SET1 = 10	F_SET2 = 18	F_SET3 = 29	F_SET4 = 47
	ReliefF	0.54	0.52	0.55	0.6
	SVM-RANK	0.63	0.63	0.65	0.63
	IG	0.55	0.59	0.57	0.61
	SLI-γ	**0.76**	**0.76**	**0.74**	**0.73**
Diabetes		F_SET1 = 4	F_SET2 = 8	F_SET3 = 12	F_SET4 = 16
	ReliefF	0.76	0.79	0.79	**0.82**
	SVM-RANK	0.66	0.72	0.78	**0.82**
	IG	0.46	0.53	0.65	**0.82**
	SLI-γ	**0.82**	**0.83**	**0.80**	**0.82**
Kidney		F_SET1 = 7	F_SET2 = 14	F_SET3 = 21	F_SET4 = 28
	ReliefF	**1**	**1**	**1**	**1**
	SVM-RANK	0.55	0.98	**1**	**1**
	IG	0.54	0.9	0.94	**1**
	SLI-γ	**1**	**1**	**1**	**1**

5.5.2 Wrapper method selection

To select the appropriate evolutionary algorithm, a comparison was made between GA, wall optimization algorithm (WOA), and gray wolf optimization (GWO). Figure 5 shows the results of this comparison, indicating that GA was clearly more successful than the other methods in terms of finding the optimal solution for fewer iterations in all datasets. For that reason, among evolutionary algorithms, GA was used in this study to create a hybrid method.

5.5.3 Performance evaluation of the proposed hybrid method

This section evaluates the performance of the proposed hybrid method, $\mathrm{Ga}_{\mathrm{rank\&rand}}$. Table 6 shows the different parameters of the proposed method.

To implement $\mathrm{Ga}_{\mathrm{rank\&rand}}$, the SLI-$\gamma$ filter method was first applied to datasets. Then 1% of the features with the highest SLI-γ rank were selected to generate the initial population. A portion of initial population was generated by the basic method of GA, and the others were randomly selected from among 1% of the features with the highest rank to improve the initial solutions. With part of the population randomly selected from all features, the pressure exerted by the related features on some areas will decrease.

5.5.3.1 Performance evaluation of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using ANN as classifier Figures 6 and 7 show the results obtained for $\mathrm{Ga}_{\mathrm{rank\&rand}}$ on all datasets using ANN as classifier. Two experiments were done for the accuracy evaluation of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using

Table 5 Comparison of SLI-γ with Common Rankers using DT Classifier

Datasets	Rankers	Feature sets			
Colon		F_SET1$_{=}$14	F_SET2 = 34	F_SET3 = 105	F_SET4 = 304
	ReliefF	0.74	0.74	0.79	0.77
	SVM-RANK	0.6	0.6	0.63	0.62
	IG	0.59	0.7	0.69	0.74
	SLI-γ	**0.83**	**0.81**	**0.81**	**0.82**
CNS		F_SET1 = 8	F_SET2 = 16	F_SET3 = 96	F_SET4 = 430
	ReliefF	0.53	0.64	0.59	0.57
	SVM-RANK	0.52	0.5	0.5	0.51
	IG	0.59	0.59	0.54	0.53
	SLI-γ	**0.71**	**0.72**	**0.63**	**0.59**
GLI_85		F_SET1$_{=}$ 6	F_SET2 = 15	F_SET3 = 149	F_SET4 = 604
	ReliefF	0.8	0.79	**0.88**	**0.87**
	SVM-RANK	0.57	0.69	0.72	0.74
	IG	0.67	0.69	0.75	0.7
	SLI-γ	**0.86**	**0.84**	0.83	0.83
Liver		F_SET1 = 5	F_SET2 = 10	F_SET3 = 13	F_SET4 = 27
	ReliefF	0.91	0.91	0.91	0.92
	SVM-RANK	0.84	**0.94**	**0.94**	0.92
	IG	0.93	0.93	0.93	**0.93**
	SLI-γ	**0.93**	0.92	0.92	**0.93**
LeukemiaB		F_SET1 = 4	F_SET2 = 10	F_SET3 = 103	F_SET4 = 345
	ReliefF	0.89	**0.9**	**0.9**	0.9
	SVM-RANK	0.72	0.67	0.78	**0.92**
	IG	0.63	0.58	0.76	0.8
	SLI-γ	**0.91**	**0.90**	0.89	0.89
Ovarian		F_SET1 = 3	F_SET2 = 6	F_SET3 = 193	F_SET4 = 690
	ReliefF	**0.97**	**0.96**	**0.98**	**0.98**
	SVM-RANK	0.73	0.78	0.86	0.95
	IG	0.6	0.67	0.82	0.81
	SLI-γ	**0.97**	**0.96**	**0.98**	**0.98**
SMK_CAN_187		F_SET1 = 9	F_SET2 = 21	F_SET3 = 142	F_SET4 = 536
	ReliefF	**0.63**	0.62	0.58	0.59
	SVM-RANK	0.52	0.54	0.58	0.58
	IG	0.54	0.54	0.58	0.61
	SLI-γ	**0.63**	**0.63**	**0.63**	**0.65**
Wisconsin		F_SET1 = 6	F_SET2 = 12	F_SET3 = 18	F_SET4 = 21
	ReliefF	0.63	0.63	0.63	**0.69**
	SVM-RANK	0.63	**0.68**	0.67	0.67
	IG	0.63	0.64	0.65	0.64
	SLI-γ	**0.69**	**0.68**	**0.72**	**0.69**

Table 5 (continued)

Datasets	Rankers	Feature sets			
COVID19		F_SET1 = 10	F_SET2 = 18	F_SET3 = 29	F_SET4 = 47
	ReliefF	0.57	0.55	0.57	0.58
	SVM-RANK	0.66	0.65	0.70	**0.71**
	IG	0.60	0.61	0.60	0.59
	SLI-γ	**0.72**	**0.74**	**0.72**	**0.71**
Diabetes		F_SET1 = 4	F_SET2 = 8	F_SET3 = 12	F_SET4 = 16
	ReliefF	0.8	0.89	**0.94**	**0.93**
	SVM-RANK	0.66	0.72	0.87	**0.93**
	IG	0.79	0.84	0.9	**0.93**
	SLI-γ	**0.88**	**0.90**	0.92	**0.93**
		F_SET1 = 7	F_SET2 = 14	F_SET3 = 21	F_SET4 = 28
Kidney	ReliefF	**1**	**1**	**1**	**1**
	SVM-RANK	0.94	0.96	**1**	**1**
	IG	0.91	0.93	0.94	**1**
	SLI-γ	**1**	**1**	**1**	**1**

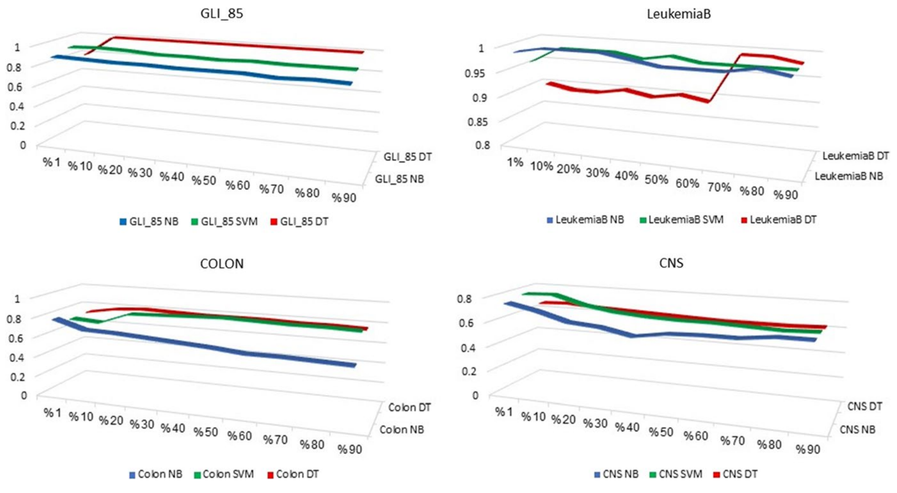

Fig. 4 Classification accuracy by applying different numbers of selected features

ANN. In the first experiment, 1% of the best ranked features were obtained from SLI-γ; then, GA was forced to generate its initial population from the selected features. The results show that using high ranking features to generate the initial population of GA reduces the classifier error (increases the classification accuracy) on all datasets. In the second experiment, 50% of the initial population of GA was generated using 1% of high-ranking features of SLI-γ, and the remaining

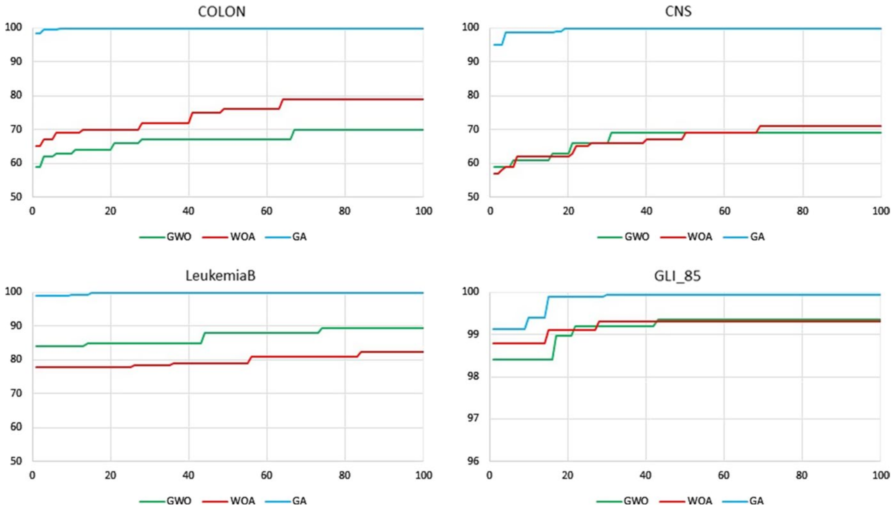

Fig. 5 Convergence of evolutionary feature selection methods

Table 6 Adjustment of parameters for experiments

GA(Parameters)	Values
β	0.52
Npop	10
Pc	0.8
Pm	0.02
Cost function	KNN
K	1
Cost function	ANN
#Hidden layers	1
#Neurons	10
Activation function	Sigmoid
Performance function	MSE
Training algorithm	Levenberg Marquardt
Training/validation/test	70% samples/15% samples/15% samples

50% of the population was produced from all features using basic initial population generation of GA. The results show that applying the proposed method on all datasets reduces the classifier errors.

5.5.3.2 Performance evaluation of Ga$_{\text{rank\&rand}}$ using KNN as classifier Figure 8 shows the results obtained from Ga$_{\text{rank\&rand}}$ for all datasets using KNN as classifier. 100, 90, 80, 70, 60, and 50% of the initial population of GA were generated using 1% of high-ranking features of SLI-γ, along with the random population of GA on datasets.

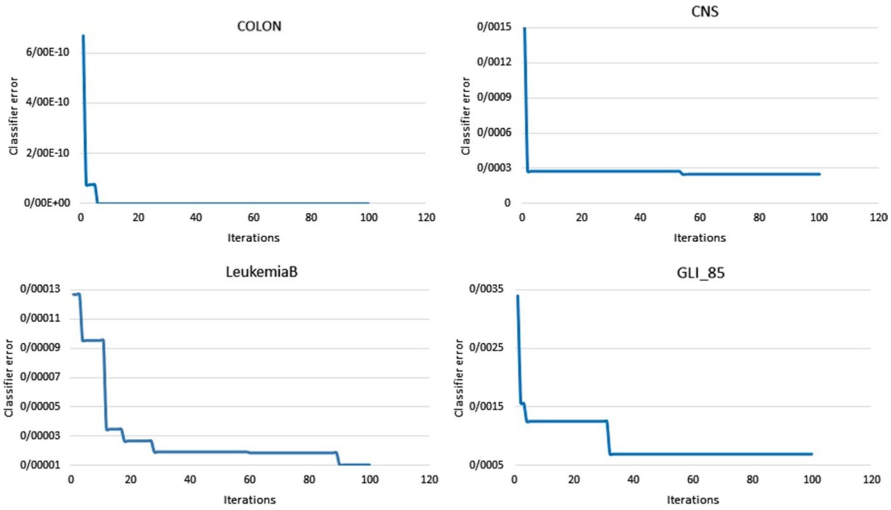

Fig. 6 Applying Ga$_{\text{rank\&rand}}$ to the first experiment using ANN

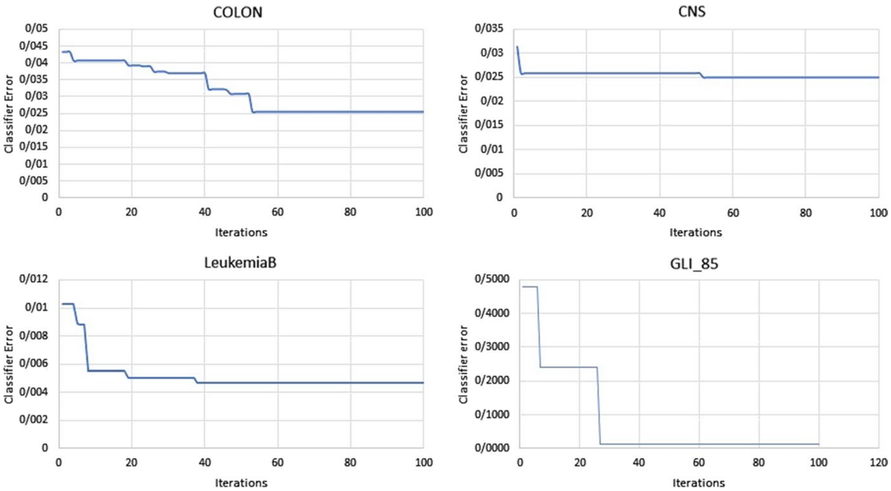

Fig. 7 Applying Ga$_{\text{rank\&rand}}$ to the second experiment using ANN

In Colon and CNS the highest accuracy was achieved when 80% of the initial population of GA were generated using 1% of high-ranking features of SLI-γ, along with 20% of the random population of GA. In GLI_85, the highest accuracy was achieved when 60% of the initial population of GA were generated using 1% of high-ranking features of SLI-γ, along with 40% of the random population of GA. In LukemiaB, the highest accuracy was achieved when 50% of the initial population of GA were generated using 1% of high-ranking features of SLI-γ, along with 50% of the random

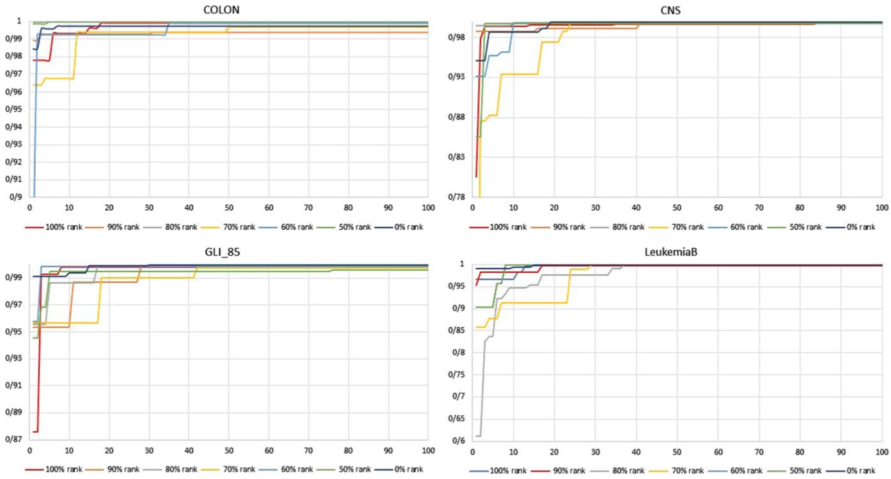

Fig. 8 Applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ to the datasets using KNN

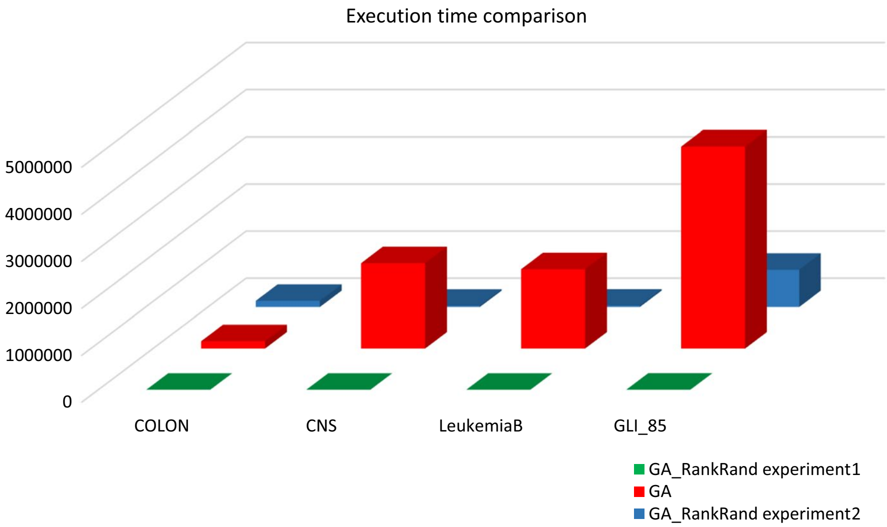

Fig. 9 Comparison of the execution time of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ with that of GA using ANN

population of GA. The results indicated that using the SLI-γ best-ranked features for initial population generation increases the classification accuracy.

5.5.3.3 Execution time comparison between GA and $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using ANN This subsection compares the GA execution time on all datasets with that of $\mathrm{Ga}_{\mathrm{rank\&rand}}$:

- 100% of initial population generation using 1% of SLI-γ ranked features
- 50% of initial population generation using 1% of SLI-γ ranked features

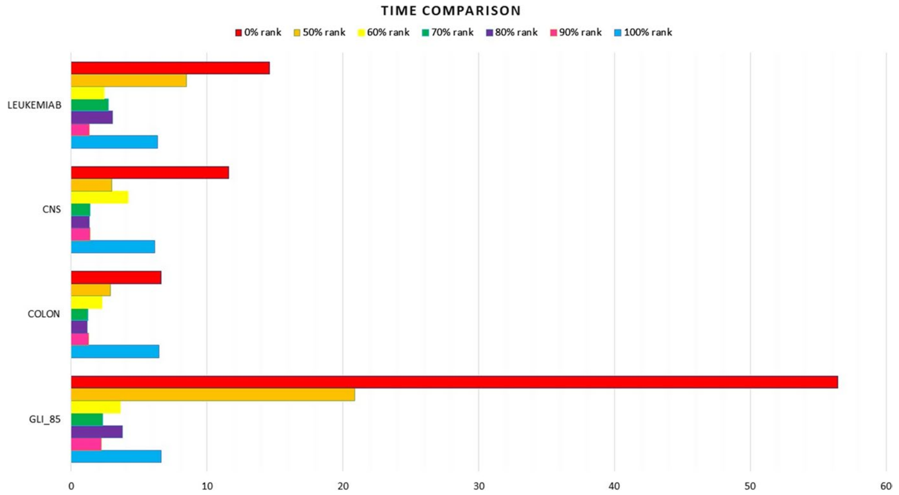

Fig. 10 Comparison of the execution time using KNN

As can be seen in Fig. 9, applying the $Ga_{rank\&rand}$ method to datasets using the ANN classifier led to a significant reduction in the algorithm execution time. In addition, the shortest execution time was obtained by generating 100% of the initial population by 1% of the best-ranked features. The results indicated that using the SLI-γ best-ranked features for initial population generation significantly reduce the execution time.

5.5.3.4 Comparison of the execution time between GA and proposed hybrid methods using KNN For the execution time comparison using the KNN classifier, 100, 90, 80, 70, 60, and 50% of the initial population of GA was generated using 1% of high-ranking features of SLI-γ, along with the random population of GA on datasets. The highest execution time was recorded when the whole population was generated randomly using all features (as in classic GA) in all datasets. As can be seen in Fig. 10, applying the $Ga_{rank\&rand}$ method to datasets led to a significant reduction in the algorithm execution time.

5.5.3.5 Performance comparison of GA and proposed hybrid methods using ANN In Table 7, $Ga_{rank\&rand}$ is compared with the basic GA in terms of the initial population size, final feature set size, and classification error. In CNS, using 1% of the best features of SLI-γ led to a significant reduction in the final feature set size, execution time, and classifier errors. Applying $Ga_{rank\&rand}$ (with generating 50% of the population from 1% of the best-ranked features) to CNS dataset resulted in 40% reduction in classifier errors and a 0.1 reduction in the number of final features set size. Furthermore, applying $Ga_{rank\&rand}$ (with generating all population from 1% of the best-ranked features) to CNS dataset resulted in 0.01 reduction in classifier errors and number of final features. Applying $Ga_{rank\&rand}$ (with generating 50% of the population from 1% of the best-ranked features) to the Colon data-

Table 7 Comparison of the proposed hybrid methods with Genetic Algorithm

	Datasets	Number of features	Population size	Classifier error	Number of selected features	Classification accuracy
$Ga_{rank\&rand}$ with 100% ranked features	CNS	7129	**71**	**0.000251**	**29**	**0.999749000**
	Colon	2000	**20**	**1.37E−19**	**11**	**1.00000000**
	GLI_85	22,283	**224**	**0.000693**	**98**	**0.999307000**
	LeukemiaB	7129	**70**	**0.000010121**	**29**	**0.999989879**
GA	CNS	7129	7129	0.05981	3556	0.940190000
	Colon	1007	2000	2000	0.030484	0.969516000
	GLI_85	22,283	22,283	0.052313	10,182	0.947687000
	LeukemiaB	7129	7129	0.017156	3547	0.982844000
$Ga_{rank\&rand}$ With 50% ranked feature	CNS	7129	7129	0.025071	361	0.974929000
	Colon	0.025671	2000	2000	1004	0.974329000
	GLI_85	22,283	22,283	0.013089	1163	0.986911000
	LeukemiaB	7129	7129	0.004668	359	0.995332000

set resulted in 0.7% reduction in classifier errors and also a magnificent reduction in the number of final feature set size. In addition, applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ by generating all population from 1% of the best-ranked features significantly decreased the classifier errors and also reduced the number of final features to 0.01 of the selected features of the basic GA. Applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ (with generating 50% of the population from 1% of the best-ranked features) to GLI_85 reduced the classifier errors, execution time, and final feature set size. However, applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ (with generating 100% of the population from 1% of the best-ranked features) to the GLI_85 dataset reduced the classifier errors and also reduced the number of final features to 0.001 GLI_85 features. Applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ (with generating 50% of the population from 1% of the best-ranked features) to the LeukemiaB dataset resulted in 27% reduction in the classifier errors and 0.1 reduction in the number of final features. Additionally, applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ (with generating 100% of the population from 1% of the best-ranked features) to LeukemiaB dataset considerably reduced the classifier errors in the basic GA and also reduced the number of final features to 0.01 of the selected features of the basic GA. As it is shown in Table 8, applying $\mathrm{Ga}_{\mathrm{rank\&rand}}$ to all datasets caused a magnificent increase in classification accuracy.

Finally, $\mathrm{Ga}_{\mathrm{rank\&rand}}$ with generating 50 and 100% of the population from 1% of the best-ranked features is a hybrid method suitable to large-dimensional datasets. $\mathrm{Ga}_{\mathrm{rank\&rand}}$ with generating 100% of the population from 1% of the best-ranked features achieved high accuracy with the fewest features and least execution time.

5.5.3.6 Comparison of the Accuracy and Execution time of GA and $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using KNN

For accuracy evaluation of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using KNN, 1% of SLI-γ high-ranked features was used for the generation of 100, 90, 80, 70, 60, 50, and 0% of the ini-

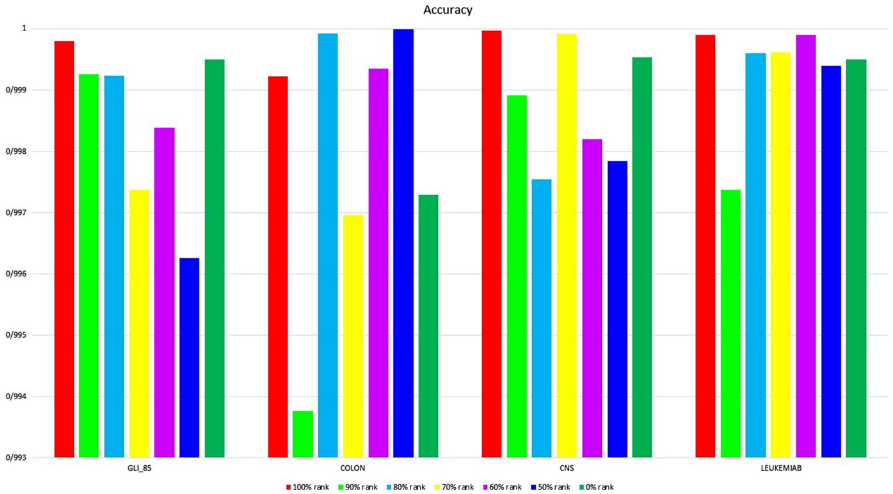

Fig. 11 Accuracy comparison of different percentages of population generation using KNN

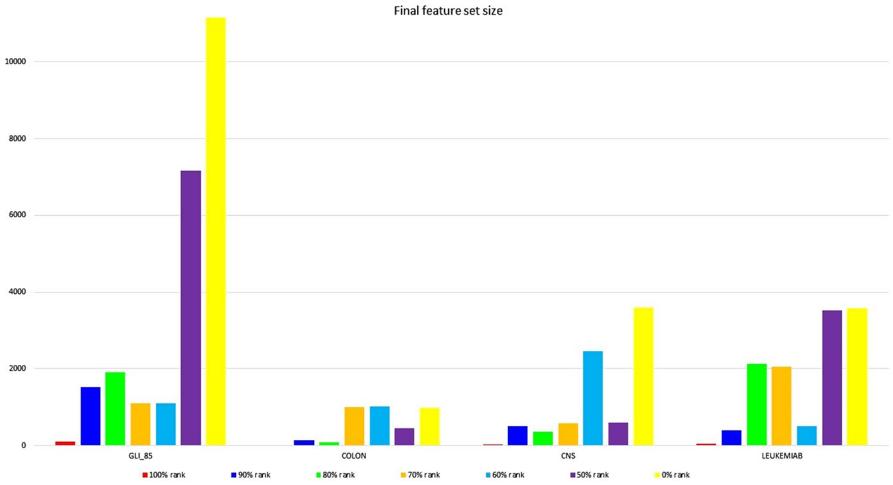

Fig. 12 Final feature set size comparison in $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using KNN

tial population. Though, generating all population using only 1% of ranked features resulted in the highest classification accuracy in GLI_85, LeukemiaB, and CNS. In Colon, the highest accuracy was achieved when 50% of GA population was generated using only 1% of high ranked features. On the other hand, the lowest accuracy was recorded when 90% of GA population was generate using 1% of ranked features. As shown in Fig. 11, $\mathrm{Ga}_{\mathrm{rank\&rand}}$ significantly improved accuracy in all datasets.

For final feature set size comparison of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ using KNN, We used 1% of most relevant features of each dataset, for generation of 100, 90, 80, 70, 60 and 50%, of the initial population. As it is shown in Fig. 12, when all initial population generated randomly as in GA, it causes in larger feature set size in all datasets. On the other hand, generating all population using only 1% of ranked features resulted in the smallest final feature set size in all datasets. As it shown in Fig. 12, $\mathrm{Ga}_{\mathrm{rank\&rand}}$ significantly reduced the final feature set size in all datasets compared with GA.

6 Conclusion

This paper proposed a hybrid feature selection method, called $\mathrm{Ga}_{\mathrm{rank\&rand}}$, which combined a proposed filter feature selection (SLI-γ) and the wrapper GA-based feature selection approach. Initially, SLI-γ was used to remove 99% of irrelevant features in the first phase. Then, GA used the most relevant features calculated by SLI-γ to optimize the first phase solutions. This paper used 11 well-known datasets (including 7 high-dimensional datasets and 4 standard datasets) for the evaluation of the measurement criteria. The experimental results showed that SLI-γ was able to produce better results than the other rankers (which were considered in this study) on all the datasets. Moreover, SLI-γ had a significant impact on the performance of GA in terms of the classification accuracy and the number of selected features. Furthermore, the execution time of $\mathrm{Ga}_{\mathrm{rank\&rand}}$ decreased significantly when 1% of the best ranked features were selected for the GA population generation. As future work, we considered using filter methods as fitness function in other state-of-the-art evolutionary algorithms, which guarantees less complexity and execution time in wrapper methods.

Data Availability Statement The datasets analyzed during the current study are available from the corresponding author on reasonable request.

References

1. Abasabadi S, Nematzadeh H, Motameni H, Akbari E (2021) Automatic ensemble feature selection using fast non-dominated sorting. Inform Syst 100:101760
2. Maleki N, Zeinali Y, Niaki ST (2021) A k-NN method for lung cancer prognosis with the use of a genetic algorithm for feature selection. Expert Syst Appl 164:113981
3. Seijo-Pardo B, Bolón-Canedo V, Alonso-Betanzos A (2017) Testing different ensemble configurations for feature selection. Neural Process Lett 46(3):857–880
4. Gu S, Cheng R, Jin Y (2018) Feature selection for high-dimensional classification using a competitive swarm optimizer. Soft Comput 22(3):811–822
5. Seijo-Pardo B et al (2017) Ensemble feature selection: homogeneous and heterogeneous approaches. Knowl Based Syst 118:124–139.
6. Sadeghian Z, Akbari E, Nematzadeh H (2021) A hybrid feature selection method based on information theory and binary butterfly optimization algorithm. Eng Appl Artif Intell 97:104079
7. Nematzadeh H, Enayatifar R, Mahmud M, Akbari E (2019) Frequency based feature selection method using whale algorithm. Genomics 111(6):1946–1955
8. Jain A, Jain V (2022) Sentiment classification using hybrid feature selection and ensemble classifier. J Intell Fuzzy Syst 42(2):659–668
9. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
10. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5(4):115–133
11. Zhang Z (2018) Artificial neural network. In Multivariate time series analysis in climate and environmental research. Springer, Cham, pp. 1–35.
12. Rajni Bala, Dharmender Kumar (2017) Classification using ANN: a review. Int J Comput Intell Res 13(7):1811–1820. ISSN 0973-1873.
13. Novakovic J (2010) The impact of feature selection on the accuracy of naïve bayes classifier. In: 18th Telecommunications forum TELFOR 2010 Nov 23, vol 2, pp. 1113–1116.
14. Li M, Vanberkel P, Zhong X (2022) Predicting ambulance offload delay using a hybrid decision tree model. Socioecon Plann Sci 1(80):101146
15. Alon U et al (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc Natl Acad Sci 96(12):6745–6750
16. Pomeroy, Scott L. et al (2002) Prediction of central nervous system embryonal tumor outcome based on gene expression. Nature 415(6870):436–442.
17. Golub, Todd R et al (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439):531–537.
18. Petricoin III, Emanuel F et al (2002) Use of proteomic patterns in serum to identify ovarian cancer. Lancet 359(9306):572–577
19. Cooper, Brendan G et al (2017) The Global Lung Function Initiative (GLI) Network: bringing the world's respiratory reference values together. Breathe 13(3):e56–e64.

20. Spira, Avrum et al (2007) Airway epithelial gene expression in the diagnostic evaluation of smokers with suspect lung cancer. Nat Med 13(3):361–366.
21. Alirezanejad M et al (2020) Heuristic filter feature selection methods for medical datasets. Genomics 112(2):1173–1181
22. Mick E, Kamm J, Pisco AO, Ratnasiri K, Babik JM, Calfee CS, Castañeda G, DeRisi JL, Detweiler AM, Hao S, Kangelaris KN, Kumar GR, Li LM, Mann SA, Neff N, Prasad PA, Serpa PH, Shah SJ, Spottiswoode N, Tan M, Christenson SA, Kistler A, Langelier C (2020) Upper airway gene expression differentiates COVID-19 from other acute respiratory illnesses and reveals suppression of innate immune responses by SARS-CoV-2. medRxiv [Preprint]. 2020 May 19:2020.05.18.20105171. https://doi.org/10.1101/2020.05.18.20105171. Update in: Nat Commun 2020 11(1):5854. PMID: 32511476; PMCID: PMC7273244.
23. Islam, MM Faniqul, et al (2020) Likelihood prediction of diabetes at early stage using data mining techniques. Computer vision and machine intelligence in medical image analysis. Springer, Singapore, pp 113–125.
24. Islam MA, Akter S, Hossen MS, Keya SA, Tisha SA, Hossain S (2020) Risk factor prediction of chronic kidney disease based on machine learning algorithms,' 2020. In: 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, pp. 952–957. https://doi.org/10.1109/ICISS49785.2020.9315878.