El-Mageed *et al. Journal of Big Data* (2024) 11:46
https://doi.org/10.1186/s40537-024-00902-z

Journal of Big Data

RESEARCH Open Access

Gene selection via improved nuclear reaction optimization algorithm for cancer classification in high-dimensional data

Amr A. Abd El-Mageed[1*], Ahmed E. Elkhouli[2], Amr A. Abohany[3] and Mona Gafar[4,5]

*Correspondence: amr.atef@commerce.sohag.edu.eg

[1] Department of Information Systems, Sohag University, 82511 Sohag, Egypt
[2] Department of Biomedical Engineering, Faculty of Electrical Engineering, Menofia University, 32951 Menofia, Egypt
[3] Department of Information Systems, Faculty of Computers and Information, Kafrelsheikh University, 33516 Kafrelsheikh, Egypt
[4] Department of Computer Engineering and Information, College of Engineering in Wadi Alddawasir, Prince Sattam Bin Abdulaziz University, 16278 Kharj, Saudi Arabia
[5] Machine Learning and Information Retrieval Department, Artificial Intelligence, Kafrelsheikh University, 33516 Kafrelsheikh, Egypt

Abstract

RNA Sequencing (RNA-Seq) has been considered a revolutionary technique in gene profiling and quantification. It offers a comprehensive view of the transcriptome, making it a more expansive technique in comparison with micro-array. Genes that discriminate malignancy and normal can be deduced using quantitative gene expression. However, this data is a high-dimensional dense matrix; each sample has a dimension of more than 20,000 genes. Dealing with this data poses challenges. This paper proposes RBNRO-DE (Relief Binary NRO based on Differential Evolution) for handling the gene selection strategy on (rnaseqv2 illuminahiseq rnaseqv2 un edu Level 3 RSEM genes normalized) with more than 20,000 genes to pick the best informative genes and assess them through 22 cancer datasets. The *k*-nearest Neighbor (*k*-NN) and Support Vector Machine (SVM) are applied to assess the quality of the selected genes. Binary versions of the most common meta-heuristic algorithms have been compared with the proposed RBNRO-DE algorithm. In most of the 22 cancer datasets, the RBNRO-DE algorithm based on *k*-NN and SVM classifiers achieved optimal convergence and classification accuracy up to 100% integrated with a feature reduction size down to 98%, which is very evident when compared to its counterparts, according to Wilcoxon's rank-sum test (5% significance level).

Keywords: RNA sequencing (RNA-Seq), Micro-array, High-dimensionality, Cancer biomark, Meta-heuristic, Nuclear reaction optimization (NRO), Relief algorithm, Differential evolution (DE), Gene selection

Introduction

DNA contains our recipe, "our genetic code". Although each cell's DNA is the same, each tissue structure is distinct and has a unique function, as DNA expresses which genes in a cell are active and which are not engaged through a mechanism called RNA transcription. This RNA is then converted into a protein responsible for cell structure and function. Therefore, analyzing a transcriptome profile is our method for determining the genetic changes in each cell from which we can evaluate diseases' biomarkers. Differential expression analysis aims to discover quantitative changes in expression levels through statistical analysis to classify genes whose expression levels vary under different conditions, which helps us understand diseases and control them.

In this manner, Gene expression Profiling technologies have been significantly developed. There are two leading popular technologies: the hybridization-based technique "micro-array", which is elder, and the next-generation sequencing-based "RNA-Seq" [1]. Both techniques are meant to quantify gene expression for statistical analysis and classification. The quantification data based on the next-generation sequencing-based RNA-Seq technique is chosen in this paper because it can detect RNA quantification levels more accurately than micro-array data. This reason is not the only advantage of the RNA-Seq technique but also because the previous technique has many limitations that have been overcome thanks to the next-generation sequencing-based technology [2], which is the base of the RNA-seq method as mentioned above. One primer obstacle in the micro-array was the reliance upon existing sequencing knowledge that limited the detection range; this obstacle is no longer a problem in RNA-seq as it requires no previous knowledge and makes our dynamic detection range wide. That choice helps our results' accuracy and the set of genes we get and gives us a close understanding of the disease's accurate biomarker.

Lyu et al. [3] presented the scope of determining cancer genetic biomarkers depending on RNA-Seq gene expression data; it worked on normalized-level3 RNA-Seq gene expression data of 33 tumor types in Pan-Cancer Atlas, which we have also worked on in this paper. However, it was noted that the researchers in paper [3] used mixed samples of non-tumor samples as if they were all tumors. Therefore, we made a code to separate samples based on their type for binary classification and more accurate tumor data. It is noteworthy that every record of data is comprised of a set of 20531 genes "features", which includes an abundance of extraneous genes and extra information.

The curse of dimensionality [4] is a popular challenge as a result of the evolutionary era of data availability, which leads to progress in Feature Selection (FS) algorithms and techniques. Generally, FS techniques follow four approaches: filter approach, wrapper approach, embedded approach, and hybrid approach [5, 6]. All these approaches aim to select the best features to distinguish the classes, which are, in our case, the informative genes related to their tumor.

The filter approach depends on the single relationship of each gene using statistical scores to represent the strength, which achieves high accuracy and selects the best group of genes. However, working on each gene separately discards the reality of the interrelationships between genes, and it can be trapped in a local optimum. It is also worth mentioning that the filter approach includes sub-types univariate and multivariate; the main difference is that the multivariate considers correlation in its rank. Examples of filter approach are t-test [7], Fisher score [8], signal-to-noise ratio [9], information gain [10], and Relief [11].

The wrapper approach can be seen as an exploration of all possible subsets, and the principle is to create and test a subset of genes. A particular classifier determines the output of a given subset, and the classification algorithm is used many times for each evaluation. This approach achieves higher performance than the filter approach because of the reality that it uses a classification algorithm that guides the learning process. However, that classifier requires high computational cost and slows the process, especially with our high-dimensional data.

A metaheuristic is a higher-level procedure or heuristic used in computer science and mathematical optimization to find, generate, or select a heuristic (partial search algorithm) that may offer a good enough solution to an optimization problem, particularly when there is incomplete or imperfect information or limited computing power. A subset of solutions that would otherwise be too numerous to be fully enumerated or otherwise investigated is sampled by metaheuristics. Metaheuristics may only make a few generalizations about the optimization problem, making them useful for various issues. Metaheuristics do not guarantee that a globally optimal solution can be found for a class of problems, unlike optimization algorithms and iterative techniques. Numerous metaheuristics use stochastic optimization, meaning that the outcome depends on the collection of generated random variables. Metaheuristics are generally more effective than optimization algorithms, iterative techniques, or basic heuristics in combinatorial optimization because they search a much more extensive range of feasible solutions. As a result, they are advantageous strategies for optimization issues. Several publications and research papers have been released on the issue. Meta-heuristic approaches can successfully address the FS problem among several wrapper solutions. Stochastic techniques may produce optimum (or nearly optimal) answers quickly, and academics have begun to use them. These techniques have many benefits, such as flexibility regarding dynamic changes, the ability to self-organize without requiring specific mathematical properties, and the capacity to evaluate multiple solutions simultaneously. For that, meta-heuristic algorithms have attracted researchers' attention for tackling optimization problems. Several meta-heuristic-based algorithms for solving the FS issue have recently been developed [12]. These algorithms yield trustworthy (near-optimal) solutions at a drastically decreased computational cost.

Evolutionary Approaches (EA), Swarm intelligence (SI) approaches, and Physics-based Approaches (PHA) are the classes of metaheuristic approaches. SI approaches are a group inspired by swarms and animals' behavior habits [13]. Multiple SI methods have been proposed in the literature and above. They have obtained reliable outcomes in a broad range of optimization issues, such as Particle Swarm Optimization (PSO) [14], Artificial Bee Colony (ABC), [15], Sparrow Search Algorithm (SSA) [16], Grey Wolf Optimization (GWO) [17], Bat Algorithm (BA) [18], Wheel Optimization Algorithm (WOA) [19], Grasshopper Optimization Algorithm (GOA) [20], Sailfish Optimizer (SFO) [21], Bird Swarm Algorithm (BSA) [22], and Harris Hawks Optimization (HHO) [23]. The EA approaches are designed by simulating biological evolutionary patterns such as mutation, crossover, and choice. Genetic Algorithm (GA) [24], Differential Evolution (DE) [25], COVIDOA Optimization Algorithm [26], invasive tumor growth optimizer [27], and biogeography-based optimizer [28] are significant EA-based metaheuristic methods that have demonstrated their effectiveness in multiple optimization areas. PHA has been created using the rules of physics found in nature techniques, including SA [29], Gravitational Search Algorithm (GSA) [30], Atom Search Optimization (ASO) [31], and Henry Gas Solubility Optimization (HGSO) [32].

The embedded approach uses a learning algorithm to choose the relevant genes, directly interacting with the classification; the FS algorithm is integrated as part of the learning algorithm. The learning model is trained using an initial feature set to establish a criterion for measuring the rank values of features. The main objective is to reduce

the computation time for reclassifying different subsets, which is done in wrapper methods by incorporating the FS into the training process. The most common embedded techniques are tree algorithms like Random Forest (RF). Some embedded methods perform feature weighting based on regularization models with objective functions that minimize fitting errors and, in the meantime, force the feature coefficients to be small or precisely zero. These Methods are the LASSO [33] with the L1 penalty, Ridge with the L2 penalty for constructing a linear model, and Elastic Net [34]. Examples of the embedded approach are SVM based on Recursive Feature Elimination (SVM-RFE) [35], RF [36], and the First Order Inductive Learner (FOIL) rule-based feature subset selection algorithm.

The hybrid approach is designed to combine the filter and wrapper approaches to achieve the advantage of each and maximize each approach's benefits. The feature space dimension space is first reduced using a filter approach, which may produce numerous candidate subsets with moderate complexity. Then, a wrapper is used as a learning strategy to determine the best candidate subset. The excellent efficiency of filters and the high accuracy of wrappers are typically achieved via hybrid approaches. Many intriguing methodologies, hybrid genetic algorithms [37], hybrid ant colony optimization [38], and mixed gravitational search algorithm [39], have recently been proposed. Practically any combination of filter and wrapper can be used to create a hybrid methodology.

Motivation and contributions

Nuclear Reaction Optimization (NRO) [40] is A brand-new meta-heuristic algorithm for global optimization, which mimics the nuclear reaction process. The proposed NRO algorithm can be divided into two phases, nuclear fission (NFi) and nuclear fusion, in accordance with the definitions of nuclear reaction characteristics (NFu). The nuclear fission phase primarily mimics this mechanism. The Gaussian walk and differential operators between the nucleus and neutron have been used for exploitation and exploration in nuclear fission based on the types of nuclei and the probability of decay following bombardment. The NFu phase primarily mimics the fusion of nuclear reactions. The ionization and fusion processes of the NFu can be included in this phase.

In order to address the Gene Selection (GS) problem, this paper suggests an improved binary version of the NRO algorithm, known as the RBNRO-DE algorithm, which is a promising method and shows precise performance. Initially, there's a chance that the suggested algorithm will avoid local optima and achieve sufficient search accuracy, rapid convergence, and enhanced stability. The suggested RBNRO-DE achieves improved efficacy by obtaining optimal or nearly optimum outcomes for many of the investigated issues, in contrast to state-of-the-art meta-heuristic algorithms. Furthermore, RBNRO-DE uses a transfer function to convert continuous data into discrete values, and it incorporates the Relief algorithm and the DE technique for boosting exploration capacity and the best outcomes found inside the solution space through iterations. The rationality for applying the RBNRO-DE approach in FS is due to the fact that it is easy to understand and create, can handle a wide range of optimization problems and achieves worthwhile outcomes in a reasonable amount of time and lower computational costs; it also utilizes few control parameters. The fundamental contributions of this paper can be presented in the following:

- RNA-seq next-generation sequencing-based level 3 data is pre-processed.
- The proposed NRO algorithm is a novel type of metaheuristic algorithm that has not been applied before to RNA-Seq gene expression data. Thus, its ability to resolve this issue has not been examined.
- NRO is modified and then re-created to develop a binary version called the RBNRO-DE algorithm.
- For improving the feature space exploration capacity and enhancing the acquired optimal outcomes, the proposed RBNRO-DE algorithm embeds a Relief algorithm and a DE technique with the binary version of the NRO algorithm. This embedding enhances the algorithm's performance by producing a new population that maintains the fundamental structure but has more appropriate positions.
- As GS has a broad search space, it frequently leads to the issue of being trapped in local optima in most current algorithms. The RBNRO-DE can efficiently explore large spaces to locate optima or near optima solutions while avoiding falling into local optima.
- The final results are estimated based on various performance metrics, including mean of fitness rate, mean of accuracy rate, and mean of features count selected.
- The influence of the proposed RBNRO-DE algorithm using the two suggested classifiers (k-NN and SVM) is compared with its peers of literature algorithms.
- The proposed RBNRO-DE algorithm is evaluated on 22 different types of cancer datasets, and the results are displayed.
- The selected genes are conducted with cancer-type bio-mark.

Structure

The rest of the paper consists of five sections as follows: "Related work" section discusses the literature of FS with genome data; then "Background details" section analyzes and elaborates the base concepts of the presented methodology background; after that in "Proposed relief binary NRO based on DE (RBNRO-DE) for gene selection" section provides a detailed explanation for the proposed RBNRO-DE algorithm, which is the improved version of NRO and its parameters to handle GS; "Experimental results and discussion" section presents the experimental results and comparisons with some competitive algorithms; and finally "Conclusion and future work" section contains the work conclusion and suggestions for future research.

Related work

This section will demonstrate the literature on researchers' techniques to handle the high-dimensionality of genome data for accurate classification. Deleting irrelevant genes plays an essential role in the performance of classification algorithms, so selecting genes is a necessary step before using any Machine Learning (ML), deep learning (DL) algorithms, or other classification methods. For this consideration, we have studied some related studies in this scope to reach the goal of RNA-seq classification for cancer detection.

Li et al. [41] had an interest in finding tumors' biomarkers; they worked on the pan-cancer public data set for 31 different types. GA/K-NN was the method they used to extract the genes. In this method, they carried out multiple iterations of subsets of genes and then asses

the accuracy with the *k*-NN algorithm. Using the resultant accuracy, they chose the best set of features. This method, with 90% success, has been used across 31 types of cancer.

Lyu et al. [3] presented work to find specific cancer biomarkers; they depended on the importance of genes to their contribution to the classification. They followed these steps: pre-processing the data and applying tumor-type classification using a convolutional neural network. After that, they generated heat maps for each class to pick out the genes corresponding to pixels with top intensities in the heat maps and finally validate the pathways of selected genes. In pre-processing, they used a variance threshold of 1.19 to delete the gene expression levels that had not changed as the GS step, which reduced the number of genes to 10381 of 19531, which is a filtering approach and The final accuracy they got was 95.59. Although the accuracy was good, it can still be much better, which can be achieved using a better FS approach to reduce that dimensionality.

Khalifa et al. [42] have followed the paper mentioned above [3] however, They focused on five cancer types of data: lung adenocarcinoma (LUAD), lung squamous cell carcinoma (LUSC), breast invasive carcinoma (BRCA), kidney renal clear cell carcinoma (KIRC) and uterine corpus endometrial carcinoma (UCEC). The total dataset is 2086 rows and 972 columns; each row contains a specific sample and the RPKM RNA-Seq values of a particular gene [43]. They used the hybrid approach for pre-processing the data as they proposed binary particle swarm optimization with design trees (BPSO-DT) algorithm; 615 features out of 971 were chosen as the best features of RNA-seq. The presented results and the performance metrics performed in this research showed that the proposed approach achieved an overall testing accuracy of 96.90%. The comparative results were introduced, and the accuracy achieved in the present work outperforms that of other related work for the testing accuracy for five classes of tumors. Moreover, the proposed approach is less complex and has less time for training.

Xiao et al. [44] evaluated their method on three RNA-Seq gene expression data sets: lung adenocarcinoma, Stomach Adenocarcinoma (STAD), and breast invasive carcinoma. They depended on the DL technique as they used five different classification models followed by the DL model to ensemble each result of the five models and that have made an improvement in all the predictions evaluation as follows: LUAD 99.20%, BRCA 98.4%, and STAD 98.78&.

Liu et al. [45] have investigated genetic data but not in RNA-Seq. However, they used microarray data and followed the hybrid approach as well. Unlike the papers mentioned above, they have worked on each type independently. They used four gene datasets of colon cancer, small-round-blue-cell tumors, leukemia, and lung cancer to evaluate the algorithm's performance. The algorithm depends on Relief as the feature pre-filter to remove the genes with low relevancy with the cancer type. PSO is used as the search algorithm, and finally, the classification accuracy of SVM is used as the evaluation function of the feature subset to get the final optimal gene subset cancer.

Danaee et al. [46] worked on the gene expression data using the power of encoders and decoders of neural networks as they used Stacked Denoising Autoencoder (SDAE) as the FS method. The effectiveness of the extracted representation was then assessed using supervised classification models to confirm the use of the additional features in cancer detection. Finally, by studying the SDAE connection matrices, they discovered a collection of highly interacting genes. They used RNA-seq expression data for both tumor

and healthy breast samples from The Cancer Genome Atlas (TCGA) database for our research. These data comprise 113 healthy samples and 1097 samples with breast cancer. The findings and analyses show that the highly interacting genes may serve as breast cancer indicators that merit further investigation. After training the SDAE, they chose a layer with a low dimension and validation error compared to other encoder stacks. It has four layers that were respectively 15,000, 10,000, 2000, and 500 dimensions thick. The chosen layer's features are fed into the algorithms for classifying data. Deep learning models can, therefore, easily handle vast amounts of input data. Hence, they anticipate this model will perform better and highlight more insightful patterns if additional gene expression data becomes available.

According to the related work, most research with genetic data is at its beginning, and all of the work is trials to conduct and apply the concepts in this promising field. The research literature is filled with experiments on different methods, such as FS and deep learning state-of-the-art techniques. However, due to the very high dimensions of genetic data, there is no perfect technique. FS of genome data detects the link of a gene to its class, which is a critical preprocessing task to overcome the curse of dimensionality and verification of the gene biomarker of cancer. Because of this, the objective of this study is to use a new wrapper approach RBNRO-DE algorithm and apply it for the first time on the RNA-Seq and compare the influence of the algorithm with other FS methods.

Background details

Relief algorithm

Relief algorithm [47, 48] is a highly effective, simple, and rapid filtering method for determining the features associated with each other. The essential idea of this algorithm is to identify features that cause values to be close for identical samples that are near each other and significant for the distinction between the different samples. Therefore, the algorithm relies on the weighted order of features. The higher the feature's weight, the better the feature to classify, and vice versa.

The Relief algorithm begins by selecting a sample at random, after which it investigates two types of nearest samples: one associated with comparable class samples called Near-Hit and the other related to different class samples called Near-Miss. Each feature's weight can be assessed from the values of both Near-Hit and Near-Miss. The features are arranged according to their weights. The features with the highest weights will be chosen in the end. The weight W for the feature A can be measured using the following equation:

$$W_A = \sum_{j=1}^{N} \left(x_A^j - NM(x^j)_A\right)^2 - \left(x_A^j - NH(x^j)_A\right)^2. \quad (1)$$

where W_A is the feature's weight A, x_A^j is the feature's value A for data x^j, and N represents the number of samples. $NH(x^j)$ and $NM(x^j)$ are the closest data points to x^j that belong to the similar same and different classes, respectively.

NRO algorithm

The idea of nuclear reaction arose after finding neutrons derived from boron and nitrogen. This results from research into the interaction of uranium with neutron [49].

Nuclear fission and nuclear fusion are the two processes that make up the nuclear reaction [50]. As shown in Fig. 1, nuclear fission occurs when a heated neutron shells a weighty nucleus and transforms into lighter nuclei as fission outcomes and other molecules. When heated neutrons shell weighty nuclei, new neutrons are produced to shell other weighty nuclei. The nuclear fission chain reaction is the name for this methodology. As a result, a significant amount of power is released, which is relative to the variation in mass between the atom and the masses of the majority of fission fragments.

Nuclear fusion, on the other hand, occurs when a nucleus is warmed until it is in a condition of plasma, where the strong nuclear force causes nuclear particles to get close enough to join together and overcome the Coulomb repulsion force, as seen in Fig. 2.

The nuclear fission process is first used during the presented approach, in which nuclei fragments absorb hot neutrons and then form odd or even-even nuclei. Essential fission products, which might be utilized for exploitation, and subaltern fission products, which can be used for exploration, are two types derived from odd nuclei. The even-even nuclei not present in fission can be sought near the existing positions (current optimal solution). After that, the presented approach utilizes the process of nuclear fusion, whereby the energy generated during nuclear fission is used to heat the nuclei, causing atomic fusion. Some nuclei constrained by the force of Coulomb repulsion slow down the

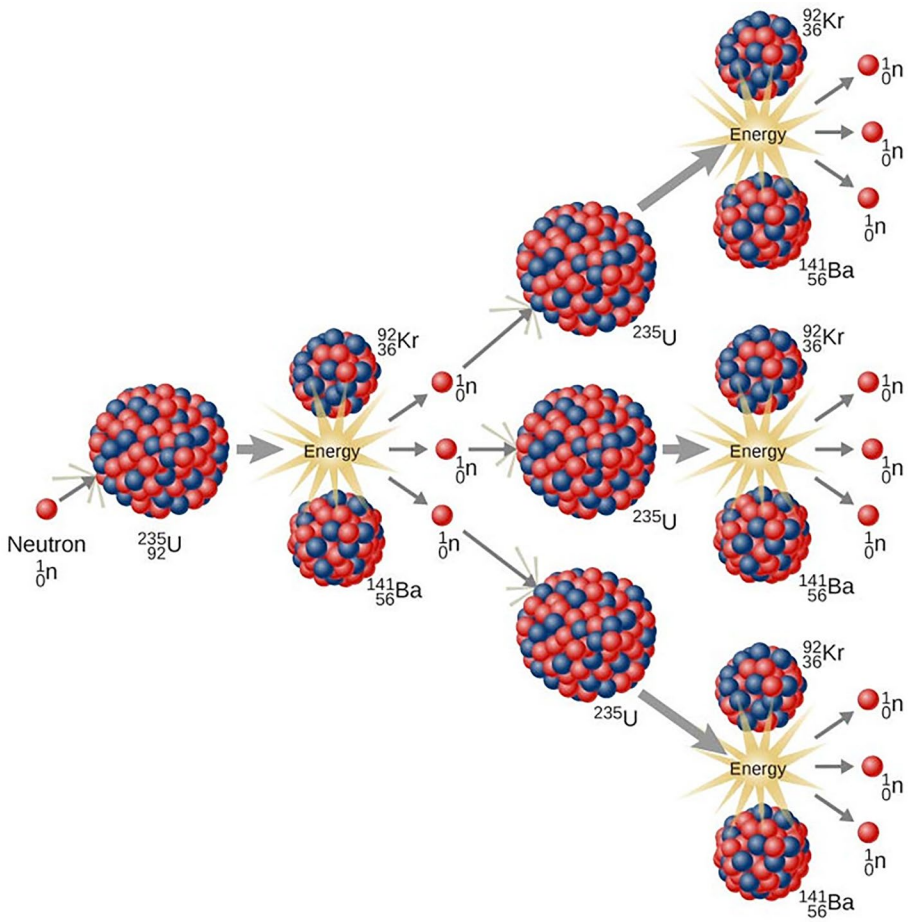

Fig. 1 The nuclear fission process

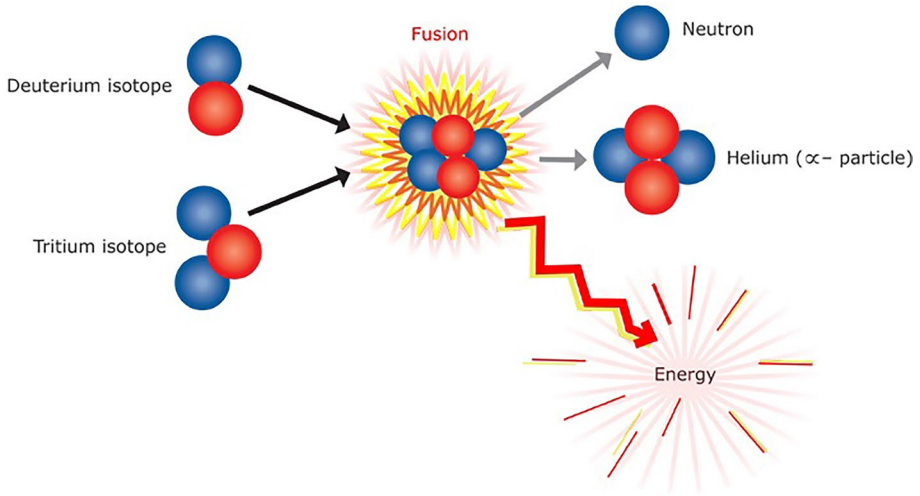

Fig. 2 The nuclear fusion process

upcoming velocity for exploitation or reject one another for exploration. Other nuclei can be explored by overcoming Coulomb repulsion and bonding together by strong nuclear forces. The heated neutron or energy generated in nuclear interaction gives each nucle kinetic energy.

According to the above illustration, a physics-based optimization algorithm known as the NRO algorithm [50] has been developed to mimic the two nuclear reaction processes, namely fission and fusion processes. The nuclear fission process involves the nuclear fission operators comprising of two cases: essential and subaltern fission of odd nuclei and nearby seeking a solution of the even-even nucleus. As for the nuclear fusion process, it has ionization and fusion phases that make up its nuclear fusion operators. Since the NRO algorithm might slip into the local optima trap, a fusion process incorporates a Levy flight methodology to jump out of the local optimal value.

Base processes of NRO algorithm

According to the NRO algorithm, the cycle generated by fission energy and fusion neutrons might be employed to find the most stable nucleus (optimal fitness value). Hence, nuclear fusion can arise from heating lighter nuclei with the energy emitted by nuclear fission. In contrast, nuclear fission can result from shelling the weighty nuclei by thermal neutrons from nuclear fusion. For exploitation and exploration of a search solution area, the NRO algorithm considers nuclear fission and nuclear fusion processes to occur in a closed container where all nuclei interface. The NRO algorithm considers a nucleus characteristic that comprises elements like position, potential energy, nucleus mass number, and charge property, which is a solution in a search solution area. The specific binding energy of each nucleus is assessed as the energy for each mass, which describes the nucleus' stability. The essential processes of the NRO algorithm are depicted below.

1. Nuclear fission process: According to the cycle between nuclear fission and nuclear fusion, it is thought that hot neutrons shelling a weighty nucleus for nuclear fission

may be created by the nuclear fusion of two separate arbitrary nuclei. In order to mathematically model nuclear fission, Gaussian walk [51] is utilized to mimic the various fission elements with diverse cases. In general, two cases can be used to distinguish the attributes of various products. The first case is associated with forming subaltern fission products for exploitation and essential fission products for exploration. These products are created when nuclear fission is applied to odd nuclei. The odd nuclei from which the subaltern fission products are generated are activated for fission utilizing energy emitted by heated neutrons and can be highly steady through β decay. On the other hand, the information on neutron and the present best solution is used by the existing solution to find a more satisfactory solution depending on the Gaussian walk. As for the odd nuclei from which the essential fission products are produced following the absorption of a hot neutron may not be steady because the fission fragment may not afford β decay. In the first case, $rand \leq P_{Fi}$ is correct, where *rand* signifies an arbitrary number distributed uniformly within the range [0, 1], and P_{Fi} is the probability of nucleus fission. For the subaltern fission products of odd nuclei, $rand \leq P_{\beta}$ is correct, where P_{β} is the likelihood of β decay. $rand > P_{\beta}$ is suitable for essential fission products of odd nuclei. The composition process of subaltern and actual fission products of odd nuclei can be expressed as follows:

$$X_i^{Fi} = \left\{ \begin{array}{ll} Gaussian(X_{best}, \sigma_1) + (randn \cdot X_{best} - P_{ne}^{s} \cdot Ne_i), & \text{if } rand \leq P_{\beta}, \\ Gaussian(X_i, \sigma_2) + (randn \cdot X_{best} - P_{ne}^{e} \cdot Ne_i), & \text{if } rand > P_{\beta}, \end{array} \right\} \quad \text{if } rand \leq P_{Fi}, \tag{2}$$

$$\sigma_1 = \left(\frac{log(g)}{g}\right) \cdot |X_i - X_{best}|, \tag{3}$$

$$\sigma_2 = \left(\frac{log(g)}{g}\right) \cdot |X_r - X_{best}|, \tag{4}$$

$$P_{ne}^{s} = round(rand + 1), \tag{5}$$

$$P_{ne}^{e} = round(rand + 2), \tag{6}$$

$$Ne_i = \frac{(X_i + X_j)}{2}. \tag{7}$$

where X_i^{Fi} means the ith fission product nucleus, *randn* means a normally distributed arbitrary number, and X_{best} denotes the present most suitable nucleus. The Gaussian distribution's parameters for subaltern fission products are X_{best} and σ_1, while the parameters of Gaussian distribution for essential fission products are X_i and σ_2, σ_1 and σ_1 signifies the step sizes, g represents the present generation number, X_r means the rth nucleus whose index r is picked randomly from the population of nuclei. Additionally, P_{ne}^{s} represents a mutation factor, indicating that the subaltern fission product can exploit the slighter searching range, whereas P_{ne}^{e} indicates that the essential fission product can exploit the larger searching range, in which *round* is

the closest integer and *rand* is an arbitrary number distributed uniformly within the range [0, 1]. Ne_i is the ith heated neutron, X_i and X_j represent the different random ith nucleus and jth nucleus, respectively. The second case is related to an even-even nucleus, which cannot be activated for fission. The status of the nucleus is altered even if there is no fission. The present nucleus' information might be kept, and it comes from the Gaussian walk. In the second case, $rand > P_{Fi}$ is correct, where P_{Fi} is the prospect of nucleus fission. It is expressed as follows:

$$X_i^{Fi} = \begin{cases} Gaussian(X_i, \sigma_2), & \text{if } rand > P_{Fi}. \end{cases} \tag{8}$$

2. Nuclear fusion process: Whenever nuclei are heated to a plasma shape, they can merge to form nuclei heavier than the initial light nuclei, known as hot nuclear fusion. The nuclear fusion process includes two steps: ionization and fusion steps.

   - The ionization step: It supposes that nuclear fission causes the emission of thermal ionization energy, which yields the motion of a nucleus. Differential operators can be involved in the ionization step. Firstly, each nucleus is rated given its fitness function level, starting with the biggest and ending with the smallest. For exploitation, the nucleus with a higher fitness function value is kept for guiding, whereas the nucleus with a lower fitness function value is utilized for exploration.

     In the ionization step, when $rand > Pa_i$, where Pa_i is a probability value of the nucleus's ionization and illustrates that the higher possibility value means a better nucleus, the ionization step can be described as mathematically, to enhance the exploration's quality, as follows:

$$X_{i,d}^{Ion} = \left\{ \begin{array}{ll} X_{r1,d}^{Fi} + rand \cdot (X_{r2,d}^{Fi} - X_{i,d}^{Fi}), & \text{if } rand \leq 0.5, \\ X_{r1,d}^{Fi} - rand \cdot (X_{r2,d}^{Fi} - X_{i,d}^{Fi}), & \text{if } rand > 0.5, \end{array} \right\} \text{ if } rand > Pa_i, \tag{9}$$

$$Pa_i = \frac{rank(fit(X_i^{Fi}))}{N}. \tag{10}$$

     where $X_{i,d}^{Ion}$ is the dth variable of ith ion after ionization. The dth variables of $r1$th, $r2$th and ith fission nuclei are represented by $X_{r1,d}^{Fi}$, $X_{r2,d}^{Fi}$ and $X_{i,d}^{Fi}$, respectively, and *rand* implies an arbitrary number between 0 and 1. Pa_i denotes a probability value of nucleus's ionization, $fit(X_i^{Fi})$ is the fitness function value of X_i^{Fi}, $rank(fit(X_i^{Fi}))$ means the rank of X_i^{Fi} in the population, and N is the overall number of nuclei. In contrast, when $rand \leq Pa_i$, the thermal fission's energy can't ionize the more stable nucleus. As a result, $X_{i,d}^{Fi}$ is adjusted to improve the exploitation's performance using the following formula:

$$X_{i,d}^{Ion} = \begin{cases} X_{i,d}^{Fi} + round(rand) \cdot rand \cdot (X_{worst,d}^{Fi} - X_{best,d}^{Fi}), & \text{if } rand \leq Pa_i. \end{cases} \tag{11}$$

     where $X_{worst,d}^{Fi}$ and $X_{best,d}^{Fi}$ mean the dth variable for the worst and better fission product nucleus, respectively. The algorithm is sometimes susceptible to falling into the trap of local optimal conditions, where two solutions are almost identical,

and the difference item might be zero. In this case, the search strategy is considered the most challenging part. Therefore, finding an algorithm-optimized approach for supporting the current solution in leaping out of a local optimal solution and investigating the global optimum is critical. This approach is called Levy flight distribution [52]. About Eq. (9), which was formed for improving the exploration in the ionization step, this equation can be applied appropriately when $X_{r2,d}^{Fi}$ is not equal to the value of $X_{i,d}^{Fi}$. However, in case the value of $X_{r2,d}^{Fi}$ is equal to the value of $X_{i,d}^{Fi}$. The Levy flight distribution approach should be employed to avoid a locally optimal solution as follows:

$$X_{i,d}^{Ion} = X_{i,d}^{Fi} + \Big(\alpha \otimes Levy(\beta)\Big)_d \cdot \Big(X_{i,d}^{Fi} - X_{best,d}^{Fi}\Big), \tag{12}$$

$$Levy(\beta) = \frac{\mu}{|\nu|^{1/\beta}}, \tag{13}$$

$$\mu = N(0, \sigma_\mu^2), \quad \nu = N(0, \sigma_\nu^2), \tag{14}$$

$$\sigma_\mu = \left(\frac{\Gamma(1+\beta)\sin(\Pi\beta/2)}{\Gamma[(1+\beta)/2]\beta 2^{(\beta-1)/2}}\right)^{1/\beta}, \quad \sigma_\nu = 1. \tag{15}$$

where α is a scale factor whose value is determined by the problem's scales ($\alpha = 0.01$), and $Levy(\beta)$ denotes the Levy flight step size. μ and ν are calculated from the normal distribution $N(0, \sigma_\mu^2)$, and $N(0, \sigma_\nu^2)$ respectively, and $\beta = 1.5$. As for Eq. (11), which was formed for improving the exploitation in the ionization step, this equation can be applied appropriately when $X_{worst,d}^{Fi}$ is not equal to the value of $X_{best,d}^{Fi}$. However, in case the value of $X_{worst,d}^{Fi}$ is equal to the value of $X_{best,d}^{Fi}$, then the Levy flight distribution approach should be utilized as follows:

$$X_{i,d}^{Ion} = X_{i,d}^{Fi} + \Big(\alpha \otimes Levy(\beta)\Big)_d \cdot \Big(UB_d - LB_d\Big). \tag{16}$$

- The fusion step: It attempts to combine an ion with information from different ions and modify the status of the ions. Initially, all ions acquired from the ionization are ranked given their fitness function levels, starting with the largest and ending with the lowest. In the fusion step, if $rand > Pc_i$,where Pc_i is a probability value of the ith ion, the ions of two light nuclei defeat the Coulomb repelling force and are fused through a robust nuclear force. Additional differential operators are used in the fusion stage to simulate the collision and fusion and boost the variety of the nuclei population to allow for more effective exploration. This situation can be depicted mathematically through the following equation:

$$X_i^{Fu} = \begin{cases} X_i^{Ion} + rand \cdot (X_{r1}^{Ion} - X_{best}^{Ion}) + rand \cdot (X_{r2}^{Ion} - X_{best}^{Ion}) \\ \quad -e^{-norm(X_{r1}^{Ion} - X_{r2}^{Ion})} \cdot (X_{r1}^{Ion} - X_{r2}^{Ion}), & \text{if } rand > Pc_i, \end{cases} \tag{17}$$

$$Pc_i = \frac{rank(fit(X_i^{Ion}))}{N}. \tag{18}$$

where X_i^{Fu} is the ith product of fusion, X_i^{Ion} represents the current ion, X_{r1}^{Ion} and X_{r2}^{Ion} denote the $r1$th and $r2$th ions, respectively, in which $r1$ and $r2$ are unlike.The difference expression $(X_{r1}^{Ion} - X_{best}^{Ion})$ is used to describe a portion of fusion process, the expression $(X_{r2}^{Ion} - X_{best}^{Ion})$ utilizes the difference to clarify another part's information of fusion, and the final expression $(X_{r1}^{Ion} - X_{r2}^{Ion})$ means that ions defeat the Coulomb repelling force. The exponential coefficient seeks to accomplish an equilibrium between exploration and exploitation. Pc_i stands for a probability value of nucleus's fusion, $fit(X_i^{Ion})$ is the fitness function value of X_i^{Ion}, and $rank(fit(X_i^{Ion}))$ stands for the rank of X_i^{Ion} in the population. On the other hand, when $rand \leq Pc_i$, ions cannot defeat the Coulomb force and fail to be fused by a nuclear force. The Coulomb force may lessen the approach speed or repel the opposing motion if fusion does not occur. The mathematical formula is recommended as follows:

$$X_i^{Fu} = \left\{ \begin{array}{l} X_i^{Ion} - 0.5 \cdot \left(\sin(2\Pi \cdot freq \cdot g + \pi) \cdot \frac{G_{max} - g}{G_{max}} + 1 \right) \cdot (X_{r1}^{Ion} - X_{r2}^{Ion}), \\ \text{if } rand > 0.5, \\ X_i^{Ion} - 0.5 \cdot \left(\sin(2\Pi \cdot freq \cdot g + \pi) \cdot \frac{g}{G_{max}} + 1 \right) \cdot (X_{r1}^{Ion} - X_{r2}^{Ion}), \\ \text{if } rand \leq 0.5, \end{array} \right\} \quad \text{if } rand \leq Pc_i. \tag{19}$$

where $freq$ denotes the sine function's frequency, g represents the present generation number, G_{max} is the permissible maximum generation number, X_{r1}^{Ion} and X_{r2}^{Ion} represent the $r1$th and $r2$th ions, respectively, with distinct indexes. In the first row of Eq. (19), the state where the Coulomb force might lower the approach speed used the non-adaptive sine adjustment to exploit the solution space and converge to the optimal solution. The case in which the two ions repulse and are far from each other to explore is in the second row of Eq. (19). The Levy flight distribution approach is applied to enhance the algorithm's capability to avoid getting stuck into a local optimum in the fusion step. In case of the value of $X_{r1}^{Ion} = X_{r2}^{Ion}$ in the fusion step, then the Levy flight distribution approach should be utilized for avoiding a locally optimal solution as follows:

$$X_i^{Fu} = X_i^{Ion} + \alpha \otimes Levy(\beta) \otimes (X_i^{Ion} - X_{best}^{Ion}). \tag{20}$$

The fission nucleus with the best fitness function value in the present generation should be saved as guiding information for the following process. While the fusion nucleus with the best fitness function value should be the globally acquired best solution. The individuals outside the search boundary are reformed using the boundary control approach.

Suggested classifiers

k-NN classifier

The k-NN [53, 54] is a pattern classification algorithm, which is used to predict whether new sample instances will belong to one or another class based on which class the cases closest to it belong to for making a decision [55]. The k-NN is a wrapper for generating classification rules from training samples. Then, by computing the distances amongst the new un-classified instance and its closest k-training

neighbours, it tries to locate the cases in the training set most comparable to the new instances in the test set. Finally, depending on the training process, a novel instance is classified according to the most significant category likelihood.

However, while training *k*-NN, the option of k is fundamental and the sole factor to consider when categorizing a novel test set; therefore, it is picked after a series of trial and error runs. The *k*-NN classifier (k = five [56, 57]) with the Euclidean distance metric was utilized to assess the feature subsets in the literature experiments.

SVM classifier

The greatest margin hyper-planes in the space can be found using the SVM [58] to accurately classify training instances into different classes. SVM can analyze high-dimensional data with a fast training period and minimal computational resources, even with a few training examples.

SVM employs a margin maximization strategy to avoid assessing the distributions linked to the statistics of distinct classes in the hyper-dimensional space. It creates hyper-planes to produce resolution boundaries for linear or nonlinear classification. Since the classes cannot be divided along a straight line in the nonlinear classification, SVM makes the data linearly separable by using the so-called kernel function [59] as a scalar product. SVM is used in a variety of industries, including bioinformatics [60], face detection [61], image classification [62], and text categorization [63].

Proposed relief binary NRO based on DE (RBNRO-DE) for gene selection

As one of the most valuable uses of RNA-Seq gene expression data is disease classification, ML algorithms may be misled by the high dimensionality of data. Therefore, an enhanced version of NRO called RBNRO-DE, which indicates a Relief Binary NRO based on DE, is proposed to ignore irrelevant genes and identify the minor relevant genes' subsets from the classification process.

The main characteristic of RBNRO-DE is that it achieves the best accuracy with the most minor subset of features. Two main phases constitute the proposed RBNRO-DE. Firstly, a pre-processing phase uses the Relief algorithm to identify the relevant features by computing a weight for every feature to describe its relationship and then ignoring the irrelevant features with the lowest weights. The second phase includes applying the binary NRO algorithm combined with the DE technique to determine the most relevant and non-redundant features. When solving large-scale problems, the NRO algorithm is susceptible to the local optimal trap. To prevent this, the DE technique is included in the NRO algorithm.

The stages required for the proposed RBNRO-DE to be able to handle the GS strategy include filtration, initialization, position improvement depending on the NRO algorithm, binary conversion, fitness estimation, and hybridization with DE. The following subsections describe these stages.

Filtration of features

As illustrated in subsection "Relief algorithm", the Relief algorithm is used to pre-process the population by filtering the features and choosing the relevant features. The weight of each feature is first evaluated by Eq. (1), and then the weights are ordered from the largest to the smallest weights to determine relevance for the classification process. By concentrating only on the relevant features and minimizing the initial search space, the Relief algorithm supports the NRO algorithm to obtain better features faster.

Initialization of nuclei population

The suggested BRNO initiates by randomly producing a population of N nuclei. Each nucleus represents a potential solution within its restricted lower and upper limits, depicted by a D dimensions vector equal to the original dataset's feature count. The randomly generated position of each nucleus is employed in this randomly initialized step, which is confined within the $[-1, 1]$ range at each variable of the position vector.

Improvement and adjustment of position

Positions are improved using equations linked to the NRO algorithm presented in Subsection "NRO algorithm". These equations are repeated until a certain stopping condition is fulfilled. This paper's acceptable stopping condition for suitably assessing the proposed algorithm's quality is the maximum number of generations G_{max}.

Some nuclei may be outside the search space's boundaries when optimizing the position utilizing the NRO algorithm. This paper offers a procedure for enhancing these worthless nuclei by adjusting them to an arbitrary position inside the permitted boundaries. By randomly varying the optimal position, this procedure will improve the exploitation of the NRO algorithm. This procedure can be expressed as follows:

$$X_{i,d}^{adjust} = \begin{cases} X_{i,d}, & \text{if } X_d^{LB} \leq X_{i,d} \leq X_d^{UB} \\ rand(X_d^{LB}, X_d^{UB}), & \text{if } X_d^{LB} > X_{i,d} \ \ or \ \ X_{i,d} > X_d^{UB}. \end{cases} \tag{21}$$

where $X_{i,d}^{adjust}$ refers to the proper product nucleus, $X_{i,d}$ is the value that surpasses the variable's boundaries, X_d^{LB} denotes the lower boundary of product nuclei, X_d^{UB} denotes the upper boundary of product nuclei. An arbitrary value between X_d^{LB} and X_d^{UB} is returned through $rand(X_d^{LB}, X_d^{UB})$ with regular distribution.

Continuous to binary conversion

The nuclei positions are represented as continuous (real) values in the NRO. Therefore, they can't be utilized directly for the GS binary problem. To fit in with the binary character of GS, a binary conversion strategy for transforming the continuous (real) values of the nucleus' positions into binary values is required. At the same time, the original algorithm's structure is preserved.

In the binary vector, the continuous (real) values of the relevant selected features are expressed by ones, whereas zeros express the continuous values of the irrelevant

unselected features. The mathematical formulation to transform the continuous nucleus position X_i^g to a binary position $(X_i^g)_{bin}$, at each generation g, is as follows:

$$(X_i^g)_{bin} = \begin{cases} 1 \text{ if } \mathbf{X}_i^g > \delta, \\ 0 \text{ otherwise.} \end{cases} \tag{22}$$

where δ represents a random threshold value within the range [0, 1]. This essential binary conversion strategy implies that if $(X_i^g)_{bin}$ is bigger than δ. It changes from its continuous value to the binary "one" (selected features for the classification process). In contrast, the continuous value of $(X_i^g)_{bin}$ has been adjusted to the binary "zero" if it is less than *delta*. (unselected feature for the classification process).

Estimation of fitness function

Two clashing goals should be considered to estimate the goodness of a solution and reach the optimal solution: maximizing the accuracy of classification from the classifiers (k-NN and SVM classifiers) while searching for the shortest size of elected features, and this enhances the algorithm's predictive capacity. The fitness function will be used to balance the size of selected features and the accuracy of (k-NN and SVM) classifiers since accuracy may be impaired if the size of selected features is reduced more than desired. To minimize the two goals, the fitness function will focus on reducing the error rate of classification instead of the accuracy, as follows:

$$fit = w_1 \times Err_{rate} + w_2 \times \frac{|feat_{elected}|}{|D|}, \quad w_1 \in [0,1], \; w_2 = 1 - w_1. \tag{23}$$

where Err_{rate} reflects the rate of classification error from the (k-NN and SVM) classifiers, $feat_{elected}$ signifies the selected features' length, and D indicates the dataset's overall feature count. The weight parameters w_1 and w_2 refer to the significance of classification accuracy and the length of the elected features, respectively. Based on the comprehensive trials executed in prior research [64, 65], w_1 is assigned to 0.99, and w_2 equals 0.01. Minimizing the error rate of classification Err_{rate} (maximizing classification accuracy) is given more preference than shortening the length of the elected features $feat_{selected}$, which suggests that w_1 should be given more weight than w_2.

Embedding of the DE approach

One of the most influential and straightforward stochastic, population-based trial-and-error approaches for acquiring the preferable solution to complicated optimization problems is DE [25]. The DE approach requires few control parameters, is simple to learn and use, and can handle a variety of optimization problems while producing valuable results quickly and at a reduced computational cost. DE depends on three primary stages: mutation, crossover, and selection, as follows:

- Mutation stage: It is also known as a differential mutation. With each iteration, this stage aims to create a mutated vector υ_i for each solution vector. To create the

mutated vector υ_i, three distinct nominee vectors $X_{r_1}, X_{r_2}, X_{r_3}$ are randomly selected from the range [1, population size]. The difference between two of the nominee vectors X_{r_2}, X_{r_3} is then estimated. The third nominee vector X_{r_1} is then added to after this difference is multiplied by a mutation weighting factor (W_M) within the range [0, 1] [66]. The following is a mathematical representation of υ_i:

$$\overrightarrow{\upsilon}_i = \overrightarrow{X}_{r_1} + W_M(\overrightarrow{X}_{r_2} - \overrightarrow{X}_{r_3}) \tag{24}$$

- Crossover stage: DE uses the crossover stage to enhance population diversity after the differential mutation stage. Combining values from the target vector X_i and the mutated vector υ_i yields creating an offspring vector u_i. The binary crossover is characterized as the most popular and straightforward crossover search operator in DE, which is mathematically expressed as:

$$u_{i,d} = \begin{cases} \upsilon_{i,d}, & \text{if } rand \leq C_R \quad or \quad d = j_{rand}, \\ X_{i,d}, & \text{otherwise.} \end{cases} \tag{25}$$

  where $j_{rand} \in [1, 2, \ldots, D_X]$ is a uniformly distributed arbitrary number to guarantee that the mutated vector includes at least one dimension. Crossover rate C_R is employed to determine the likelihood of each element being crossed; it is often set to a high value ($C_R = 0.9$). It is evident from Eq. (25) that C_R and *rand* are compared. u_i is derived from υ_i if the value of *rand* is less than or equal to the value of C_R. If not, X_i is used to infer u_i.
- Selection stage: Eventually, the selection stage is performed, as illustrated in Eq. (26), in which the target vector's fitness function $fit(X_i)$ and the corresponding offspring vector's fitness function $fit(u_i)$ are compared, and the fitness function with the lowest value is retained, and the best possible solution is ready for the next generation.

$$X_i = \begin{cases} u_i, & \text{if } fit(u_i) < fit(X_i) \\ X_i, & \text{otherwise.} \end{cases} \tag{26}$$

  u_i is set to X_i if $fit(u_i)$ yields a value that is smaller than $fit(X_i)$. If not, the previous target vector X_i remains in place.

After illustrating the main stages of DE, the pseudo-code for these stages is presented in Algorithm 1.

Algorithm 1 The main stages of DE

Input:
X – the current individuals' positions
$fit(X)$ – the current fitness function
N – overall number of individuals (population size)
D – dimensions of individuals' position
C_R – crossover rate
W_M – mutation weighting factor
Output:
u – the improved individuals' positions
$fit(u)$ – the improved fitness function, which will be minimized
Start
  Rank the individuals ascendingly according to $fit(X)$;
  for individual $i = 1 : N$ **do**
    Select three individuals' positions $X_{r_1}, X_{r_2}, X_{r_3}$ randomly;
    Generate a random number $rand$ within the range $[0, 1]$;
    Pick randomly an integer number $j_{rand} \in [1, 2, ..., D_X]$;
    for $d = 1 : D$ **do**
      if $rand \leq C_R$ or $d = j_{rand}$ **then**
        Estimate the mutated vector $v_{i,d}$ based on Eq. (24);
        $u_{i,d} \leftarrow v_{i,d}$;
      else
        $u_{i,d} \leftarrow X_{i,d}$;
    $fit(u_i) \leftarrow$ Evaluate the fitness function for u_i;
    if $fit(u_i) < fit(X_i)$ **then** ▷ Improve the prior position if the current new position is better than it
      $X_i \leftarrow u_i$; $fit(X_i) \leftarrow fit(u_i)$;
End

The exhaustive RBNRO-DE algorithm

Finally, after describing the steps of the suggested RBNRO-DE algorithm in the preceding Subsections to handle the GS strategy, Algorithm 2 provides the pseudo-code for the proposed RBNRO-DE algorithm. In addition, Fig. 3 includes a flowchart of the proposed RBNRO-DE algorithm to show its essential steps.

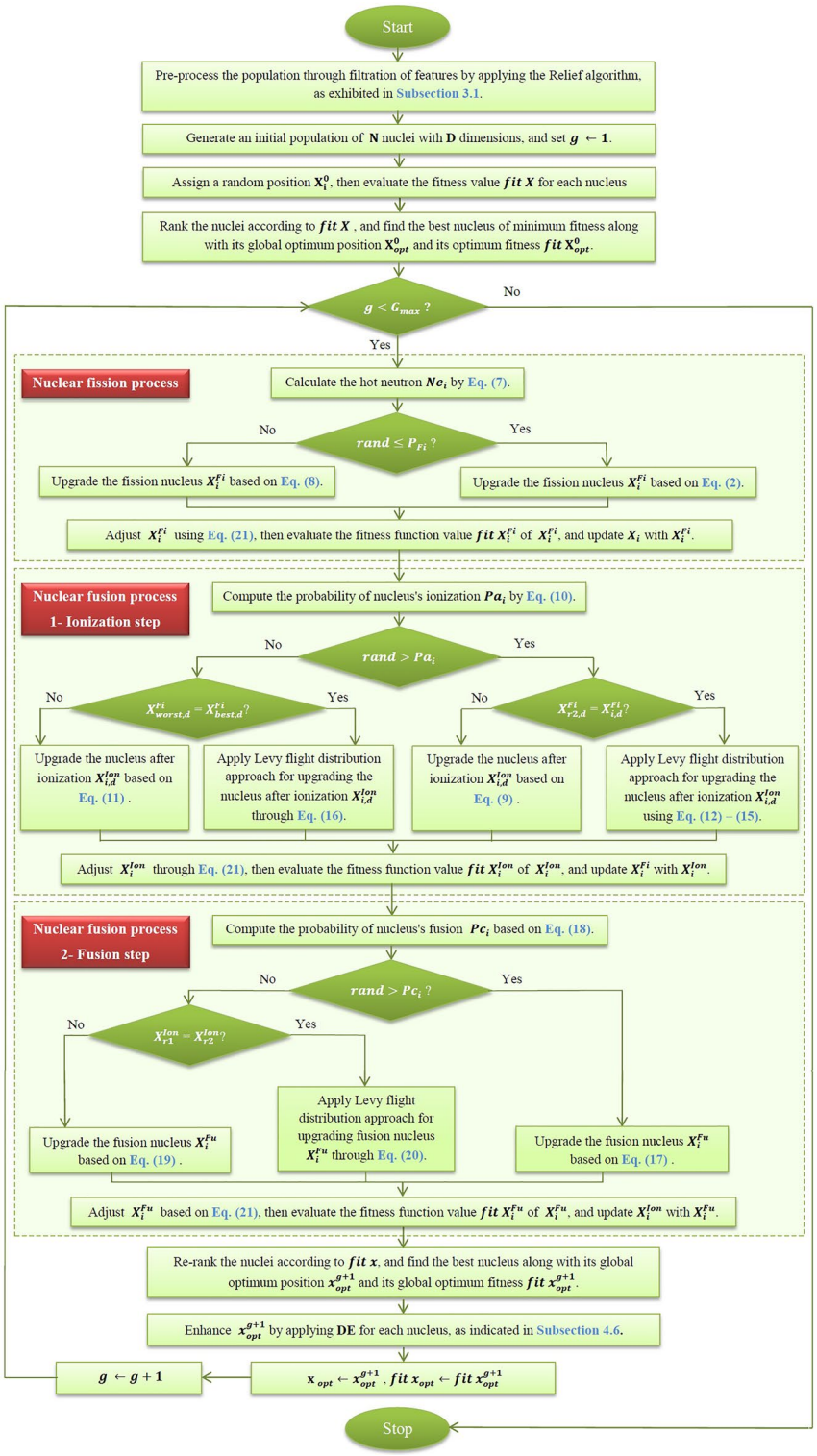

Fig. 3 Flowchart of the proposed RBNRO-DE algorithm

Algorithm 2 The proposed RBNRO-DE algorithm

Input:
N – overall number of nuclei (population size)
G_{max} – maximum generations' number
D – dimensions of nuclei' status
P_{Fi} – the nucleus fission's probability
P_{β} – the β decay's probability
LB – lower boundaries of product nuclei
UB – upper boundaries of product nuclei
C_R – crossover rate
W_M – mutation weighting factor
Output:
X_{opt} – the global optimum nucleus found during the search (the global best solution)
$fit(X_{opt})$ – the global optimum fitness found, which will be minimized
Start
Apply the Relief algorithm to pre-process the population through filtering the features and choosing the relevant features, as exhibited in Subsection 3.1;
Initialize a population of N nuclei;
Assign a random position X_i^0 for each nucleus in the population;
Evaluate the fitness values $fit(X)$ for each nucleus;
Rank the nuclei ascendingly according to their fitness function values $fit(X)$;
Locate the best nucleus of minimum fitness with its global optimum position X_{opt}^0 and its fitness value $fit(X_{opt}^0)$ among all nuclei in the initial population;
$g \leftarrow 1$; ▷ Current generation number
while $g < G_{max}$ **do**
For all current product nuclei X in the whole population, apply the mathematical equations of the nuclear fission process, as explained in the first point of Subsection 3.2.1, to obtain a population of upgraded fission product nuclei X^{Fi};
Adjust each upgraded product nuclei found by the end of the nuclear fission process X^{Fi}, using Eq. (21);
$fit(X^{Fi}) \leftarrow$ Evaluate the new upgraded fitness function value for X^{Fi};
if $fit(X^{Fi}) < fit(X)$ **then**
$X \leftarrow X^{Fi}$; $fit(X) \leftarrow fit(X^{Fi})$;
Apply the mathematical formulas of the ionization step of the nuclear fusion process, as illustrated in the first item of the second point in Subsection 3.2.1, to every fission product nucleus X^{Fi} in the population found so far by the end of the nuclear fission process, for getting a population of upgraded nuclei after ionization X^{Ion};
Adjust each product nucleus located through the ionization step of the nuclear fusion process X^{Ion}, based on Eq. (21);
$fit(X^{Ion}) \leftarrow$ Evaluate the new upgraded fitness function value for X^{Ion};
if $fit(X^{Ion}) < fit(X)$ **then**
$X \leftarrow X^{Ion}$; $fit(X) \leftarrow fit(X^{Ion})$;
For each population of product nuclei after ionization X^{Ion} discovered so far through the ionization step of the nuclear fusion process, use the mathematical formulas of the fusion step of the nuclear fusion process, as defined in the second item of the second point in Subsection 3.2.1 for obtaining a population of upgraded fusion product nuclei X^{Fu};
Adjust each product nucleus located through the fusion step of the nuclear fusion process X^{Fu}, through Eq. (21);
$fit(X^{Fu}) \leftarrow$ Evaluate the new upgraded fitness function value for X^{Fu};
if $fit(X^{Fu}) < fit(X)$ **then**
$X \leftarrow X^{Fu}$; $fit(X) \leftarrow fit(X^{Fu})$;
Re-rank the nuclei ascendingly through their fitness function values $fit(X)$;
Locate the best nucleus with its global optimum position X_{opt}^{g+1} and its global optimum fitness value $fit(X_{opt}^{g+1})$;
Enhance X_{opt}^{g+1} by applying the DE technique for each nucleus, as indicated in Subsection 4.6;
$X_{opt} \leftarrow X_{opt}^{g+1}$;
$fit(X_{opt}) \leftarrow fit(X_{opt}^{g+1})$;
$g \leftarrow g + 1$;
End

Experimental results and discussion

The experimental results for the proposed RBNRO-DE algorithm and its peers are presented in this section. The models are evaluated using training and testing datasets. The final findings are derived using the evaluation metrics' average value. The datasets used to verify the efficacy of the proposed model are described in Subsection "Dataset description", the parameters that are utilized in working environments are presented in Subsection "Parameters setting", the evaluation criteria are shown

Table 1 Description of the datasets used in this study

#	Cancer	No. of tumour samples	No. of normal samples
1	Bladder urothelial carcinoma (BLCA)	408	19
2	Thyroid carcinoma (THCA)	501	59
3	Cervical and endocervical cancers (CESC)	304	3
4	Cholangiocarcinoma (CHOL)	36	9
5	Colon adenocarcinoma (COAD)	458	41
6	Esophageal carcinoma (ESCA)	184	13
7	Glioblastoma multiforme (GBM)	153	5
8	Thymoma (THYM)	120	2
9	Head and neck squamous cell carcinoma (HNSC)	520	44
10	Kidney chromophobe (KICH)	66	25
11	Kidney renal clear cell carcinoma (KIRC)	533	72
12	Kidney renal papillary cell carcinoma (KIRP)	290	32
13	Liver hepatocellular carcinoma (LIHC)	371	50
14	Lung adenocarcinoma (LUAD)	515	59
15	Lung squamous cell carcinoma (LUSC)	501	51
16	Pancreatic adenocarcinoma (PAAD)	178	4
17	Uterine corpus endometrial carcinoma (UCEC)	176	35
18	Pheochromocytoma and paraganglioma (PCPG)	179	3
19	Rectum adenocarcinoma (READ)	94	10
20	Sarcoma (SARC)	259	2
21	Skin cutaneous melanoma (SKCM)	103	1
22	Stomach adenocarcinoma (STAD)	415	37

Table 2 The main parameters of the ML classifiers

Classifier	Parameters
k-NN	The Euclidean distance metric $k = 5$ [56, 57]
SVM	Polynomial kernel = 2

in Subsection "Evaluation criteria", and experimental results analysis is explained in Comparison results of the proposed RBNRO-DE against popular ML classifiers.

Dataset description

Extensive experiment techniques and other wrapper algorithms are conducted on 22 gene expression datasets. The data used is the normalized-level3 RNA-Seq gene expression data of 22 tumor types in Broad Institute. It is publicly found and obtainable in [67]. We followed the whole process applied in paper [3] and noticed the difference between the data used in the mentioned paper from GitHub and the numbers written in the mentioned paper, which is copied from the site. The data from the site was a mixture of tumor and normal samples, while it was used as a tumor as a whole in the mentioned paper. Therefore, we investigated the data closely. First of all, the

site contains different forms of the same data that we chose to work on; we explored the data and found these challenges:

- Some Genes are named with ID but without symbol.
- Some Genes are not found in the annotation file.
- Samples are mixed up between normal and tumor and other staff.

As a result, we needed some pre-processing to separate and identify samples to get normal samples versus tumor samples that could be used in binary classification and to facilitate the process of FS. We faced the mentioned challenges as follows:

- We searched the annotation file for the found ID and got the gene symbol.
- We have compared with the annotation file, so more than 100 genes are removed.
- depending on the samples report, we separated each row depending on the sample type in an Excel sheet for binary classification.

Furthermore, the Relief algorithm, described in subsection "Relief algorithm", is employed for pre-processing by computing the weight for each feature in the dataset, and the weights are then sorted from biggest to smallest. Finally, the features with small weights are eliminated. After applying the Relief algorithm on the 22 gene expression datasets, we found that just 500 features had the largest weights. For that, the remaining irrelevant features with small weights were ignored, and these 500 relevant features were only chosen for use in the FS process. The Relief algorithm can eliminate features that are irrelevant to classification.

After pre-processing, the resulting file became clean enough for use in the FS process. Still, unlike paper [3], which provided multi-classification of all cancer types, we worked on each type separately to be more specific. Table 1 shows a detailed list of all 22 tumour types and the corresponding number of samples.

Parameters setting

The proposed RBNRO-DE algorithm has been compared with binary conversions of distinct meta-heuristic algorithms, which include Binary SSA (BSSA) [68], Binary ABC (BABC) [15], Binary BA (BBA) [69], Binary PSO (BPSO) [70], Binary WOA (BWOA) [57], Binary GWO (BGWO) [17], Binary GOA (BGOA) [20], Binary SFO (BSFO) [71], Binary BSA (BBSA) [22], Binary ASO (BASO) [31], Binary HHO (BHHO) [23], and Binary HGSO (BHGSO) [32]. The main parameters of the ML classifiers suggested in this paper are depicted in Table 2.

To ensure a fair comparison between different meta-heuristic algorithms, each method was subjected to thirty separate experiments on each dataset due to their stochastic nature. The resulting performance measures, which include accuracy, fitness, selected features, and standard deviation, were based on the average results of these experiments. To maintain consistency across all methods, each experiment's population size and maximum number of iterations were set to 10 and 100, respectively. Furthermore, the number of attributes in the datasets used in this study indicated the problem size. To enable

Table 3 Configurations of parameter for all algorithms

Algorithm	Parameter
All algorithms	Run's number = 30
	Maximum number of iterations $G_{max} = 100$
	Population size $N = 10$
	Dimensionality D = The number of attributes in the used benchmarks
	Lower boundaries LB
	Upper boundaries UB
Proposed RBNRO-DE [25, 40]	Probability of nucleus fission P_{Fi} =uniformly distributed random number
	Probability of β decay P_{β} =uniformly distributed random number
	A scale factor $\alpha = 0.01$
	$\sigma_v = 1$
	$\beta = 1.5$
	Sine function's frequency $freq = 0.05$
	Mutation weighting factor $W_M = 0.85$
	Crossover rate $C_R = 0.9$
BSSA	Number of scroungers $SD = 0.1*N$
	Number of producers $PD = 0.2*N$
	Maximum number of generations in LSA = 20
	Safety threshold $ST = 0.8$
BABC	Number of employed bees = 16
	Number of scout bees = 3
	Number of onlooker bees = 4
BPSO	Inertia weight ($\omega_{\max} = 0.9\omega_{\min} = 0.4$)
	Acceleration coefficients ($c_2 = c_1 = 1.2$)
BBA	Loudness $A = 0.8$
	Lower and upper pulse frequencies = 0, 10
	Pulse emission rate $r = 0.95$
BWOA	a is linearly reduced from 2 to 0
	$b = 1.0$
	$p = 0.5$
BSFO	Ratio between sardines and sailfish $pp = 0.1$
	$\varepsilon = 0.0001$
	$A = 1$
BHHO	Rabbit energy $E \in [-1, 1]$
BGWO	a is linearly reduced from 2 to 0
BGOA	$C_{\min} = 0.00004$ and $C_{\max} = 1$
BBSA	Frequency of flight $ff = 10$
	Followed coefficient $fl = 0.5$
	Effect on birds' vigilance behaviors ($a_1 = a_2 = 1.0$)
	Acceleration coefficients ($c_1 = c_2 = 1.5$)
	Probability of foraging for food $p = 0.8$
BASO	Multiplier weight $\beta = 0.2$
	Depth weight $\alpha = 50$
BHGSO	Number of clusters = 2
	$l_1 = 5E-03, l_2 = 1E+02$, and $l_3 = 1E-02$
	$\alpha = \beta = 0.1$and $K = 1.0$

individuals to search within a continuous search space, the domain was set to $[-1, 1]$, allowing them to explore a relatively wide but constrained search range.

In the presented framework, the optimality degrees of the outcomes are confirmed using a 10-fold cross-validation method to assure the reliability of the values received. This involves a data-splitting strategy that employs random sampling without replacement to distribute training and testing groups. Each benchmark is divided into two separate subsets through this method. Specifically, 80% of the benchmark data is randomly selected without replacement for training, ensuring that each data point is unique and not duplicated in the training set. The remaining 20% of the data is also uniquely chosen for testing. This approach ensures that the training subset is utilized to learn the ML classifier through optimization while the testing subset is employed to assess the performance of the chosen features. By using random sampling without replacement, we ensure that there is no overlap between training and testing data, thus maintaining the integrity of the evaluation process. Each method's remaining parameters are set considering the original variants and the data presented in their first publications. Standard configurations for all techniques and parameter settings for each method are shown in Table 3. Python is utilized in the computing environment to execute the runs with an Intel processor core i7, 16 GB of RAM, and an NVIDIA GTX 1050i GPU.

Evaluation criteria

To assess the performance of the proposed RBNRO-DE algorithm compared to others, each approach is independently verified 30 times in each dataset to validate the results statistically. To this end, the following standard performance measures for the FS problem are utilized.

- Average accuracy (AVG_{Acc}): this metric is the rate of correct data classification and is obtained by executing the algorithm independently 30 times, and is computed as follows:

$$AVG_{Acc} = \sum_{k=1}^{30} \sum_{r=1}^{m} match(PL_r, AL_r) \tag{27}$$

  where m represents the size of the samples in the testing dataset, PL_r and AL_r are the classifier output labels of the predicted and actual class labels for sample r, respectively, and $match(PL_r, AL_r)$ represents a comparison discriminant function. If $PL_r == AL_r$, then $match(PL_r, AL_r) = 1$; otherwise, $match(PL_r, AL_r) = 0$.
- Average fitness value (AVG_{Fit}): this metric measures the average fitness value obtained by executing the proposed algorithm independently 30 times, which defines the synergy between minimizing the error rate of classification and reducing the number of selected features. The lower value represents the better solution, which is evaluated in terms of fitness as follows:

$$AVG_{Fit} = \frac{1}{30} \sum_{k=1}^{30} f_*^k \tag{28}$$

Table 4 Classification accuracy values of the proposed RBNRO-DE against popular ML classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	*k*-NN	SVM	DT	RF	XGBoost
BLCA	**1.0000**	**1.0000**	0.9651	0.9884	0.9884	0.9419	0.9767
CESC	0.9839	**1.0000**	0.9839	0.9677	0.9839	0.9839	0.9839
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.8889
COAD	**1.0000**	**1.0000**	0.9848	0.9848	0.9848	0.9848	0.9697
ESCA	0.9744	**0.9915**	0.9744	0.9744	0.9487	0.9744	0.9744
GBM	**1.0000**	**1.0000**	0.9688	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	0.9938	**1.0000**	0.9558	0.8673	0.9469	0.9381	0.9469
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	0.9983	**1.0000**	0.9835	0.9752	0.8843	0.9256	0.9752
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9765	0.9647	0.9765
LUAD	0.9913	**0.9922**	0.9826	0.9826	0.9652	0.9913	0.9826
LUSC	**1.0000**	**1.0000**	0.9730	0.9820	0.8919	0.9730	0.9820
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	0.9811	**0.9962**	0.9811	0.9811	0.9811	0.9811	0.9811
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9524	**1.0000**	**1.0000**
STAD	**1.0000**	0.9900	0.9333	0.9333	0.8556	0.9111	0.9333
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9643	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	0.9992	**1.0000**	0.9250	0.9250	0.9250	0.9500	0.9500
Ranking (W|T|L)	0|15|7	**7|14|1**	0|10|12	0|11|11	0|8|14	0|10|12	0|9|13

where f_*^k represents the optimal fitness value obtained in the kth run.

- Average size of selected features (AVG_{Feat}): this metric represents the average size (or FS ratio) of the number of features selected by executing the algorithm independently 30 times and is determined as:

$$AVG_{Feat} = \frac{1}{30}\sum_{k=1}^{30}\frac{|d_*^k|}{|D|} \tag{29}$$

where $|d_*^k|$ is the absolute number of selected features in the best solution for the k_th run, and $|D|$ is the absolute total number of features in the original dataset.

- Standard deviation (STD): corresponding to the above results, the final average results obtained from the 30 independent runs of each algorithm on every dataset are evaluated in terms of stability as follows:

$$STD = \sqrt{\frac{1}{29}\sum_{k=1}^{30}(Y_*^k - AVG_Y)^2} \tag{30}$$

where Y denotes the metric to be measured, $Y * k$ represents the value of the metric Y in the k_th run, and AVG_Y is the average of the metric over 30 independent runs.

The results presented in the following tables are the average values over 30 independent runs in terms of the fitness value (AVG_{Fit}), classification accuracy (AVG_{Acc}), and the

Table 5 Fitness values of the proposed RBNRO-DE against popular ML classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	*k*-NN	SVM	DT	RF	XGBoost
BLCA	0.0029	**0.0025**	0.0445	0.0215	0.0215	0.0676	0.0330
CESC	0.0185	**0.0025**	0.0260	0.0419	0.0260	0.0260	0.0260
CHOL	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.1200
COAD	**0.0025**	**0.0025**	0.0250	0.0250	0.0250	0.0250	0.0400
ESCA	0.0279	**0.0113**	0.0354	0.0354	0.0608	0.0354	0.0354
GBM	**0.0025**	**0.0025**	0.0409	0.0100	0.0100	0.0100	0.0100
HNSC	0.0089	**0.0025**	0.0538	0.1414	0.0626	0.0713	0.0626
KICH	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
KIRC	0.0047	**0.0025**	0.0264	0.0345	0.1245	0.0836	0.0345
KIRP	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
LIHC	**0.0025**	0.0026	0.0100	0.0100	0.0333	0.0449	0.0333
LUAD	0.0112	**0.0103**	0.0272	0.0272	0.0444	0.0186	0.0272
LUSC	0.0026	**0.0025**	0.0368	0.0278	0.1170	0.0368	0.0278
PAAD	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
PCPG	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
READ	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
SARC	0.0212	**0.0067**	0.0287	0.0287	0.0287	0.0287	0.0287
SKCM	**0.0025**	**0.0025**	0.0100	0.0100	0.0571	0.0100	0.0100
STAD	**0.0029**	0.0125	0.0760	0.0760	0.1530	0.0980	0.0760
THCA	**0.0025**	**0.0025**	0.0100	0.0100	0.0454	0.0100	0.0100
THYM	**0.0025**	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100
UCEC	0.0038	**0.0025**	0.0842	0.0842	0.0842	0.0595	0.0595
Ranking (W|T|L)	2|11|9	**9|11|2**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

number of selected features (AVG_{Feat}). The experimental results are closely analyzed and discussed in the subsequent subsections, where bold numbers indicate the best-required results.

Comparison results of the proposed RBNRO-DE against popular ML classifiers

This section introduced a comparison of the proposed RBNRO-DE with *k*-NN and SVM classifiers and the most popular ML classifiers [*k*-NN, SVM, Decision Tree (DT), RF, and eXtreme Gradient Boosting (XGBoost)] in terms of classification accuracy, fitness values, precision, recall, and F1-score.

Table 4 shows the results of the proposed RBNRO-DE with *k*-NN and SVM classifiers compared with other popular ML classifiers regarding the classification accuracy values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 7 out of 22 datasets and identical results in 14 datasets as other techniques. The proposed RBNRO-DE with *k*-NN is ranked second by yielding identical results in 15 datasets as other techniques. It should also be noted that SVM ranked third by yielding identical results in 11 datasets as others, while *k*-NN ranked fourth by yielding identical results in 10 datasets as other techniques. Finally, DT is ranked last by yielding identical results in 8 datasets as other techniques.

Table 5 shows the results of the proposed RBNRO-DE with *k*-NN and SVM classifiers compared with other popular ML classifiers regarding fitness values. The empirical

Table 6 Precision values of the proposed RBNRO-DE against popular ML classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	*k*-NN	SVM	DT	RF	XGBoost
BLCA	**1.0000**	**1.0000**	0.9634	0.9875	0.9875	0.9405	0.9753
CESC	0.9839	**1.0000**	0.9839	0.9836	0.9839	0.9839	0.9839
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	0.9828	0.9828	0.9828	0.9828	0.9661
ESCA	0.9737	**0.9912**	0.9737	0.9737	0.9487	0.9737	0.9737
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	0.9929	**1.0000**	0.9510	0.8661	0.9596	0.9327	0.9505
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	**1.0000**	**1.0000**	0.9903	0.8879	0.9204	0.9810
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9733	0.9605	0.9733
LUAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9899	**1.0000**	**1.0000**
LUSC	**1.0000**	**1.0000**	0.9895	**1.0000**	0.9468	0.9794	0.9896
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	0.9811	**0.9962**	0.9811	0.9811	0.9811	0.9811	0.9811
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	0.9888	0.9294	0.9294	0.9024	0.9277	0.9294
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9800	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Ranking (W|T|L)	1|17|4	**4|17|1**	0|14|8	0|14|8	0|10|12	0|12|10	0|12|10

results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 9 out of 22 datasets and identical results in 11 datasets as the proposed RBNRO-DE with *k*-NN. The proposed RBNRO-DE with *k*-NN is ranked second by achieving the best results in 2 out of 22 datasets and identical results in 11 datasets as the proposed RBNRO-DE with SVM. Finally, it should also be noted that other methods did not achieve optimal results on any of the datasets used regarding fitness values.

Table 6 shows the results of the proposed RBNRO-DE with *k*-NN and SVM classifiers compared with other popular ML classifiers regarding the precision values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 4 out of 22 datasets and identical results in 17 datasets as other techniques. The proposed RBNRO-DE with *k*-NN is ranked second by achieving the best results in 1 out of 22 datasets and identical results in 17 datasets as other techniques. It should also be noted that SVM and *k*-NN ranked third by yielding identical results in 14 datasets as others, while RF and XGBoost ranked fourth by yielding identical results in 12 datasets as other techniques. Finally, DT is ranked last by yielding identical results in 10 datasets as other techniques.

Table 7 shows the results of the proposed RBNRO-DE with *k*-NN and SVM classifiers compared with other popular ML classifiers regarding the recall values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 2 out of 22 datasets and identical results in 20 datasets as other

Table 7 Recall values of the proposed RBNRO-DE against popular ML classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	*k*-NN	SVM	DT	RF	XGBoost
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	0.9836	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.8333
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
ESCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
GBM	**1.0000**	**1.0000**	0.9677	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9794	**1.0000**	0.9897
KICH	**1.0000**	**1.0000**	**1.0000**	0.9808	**1.0000**	**1.0000**	**1.0000**
KIRC	0.9981	**1.0000**	0.9808	0.9808	0.9904	**1.0000**	0.9904
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	0.9901	**0.9911**	0.9802	0.9802	0.9703	0.9901	0.9802
LUSC	**1.0000**	**1.0000**	0.9792	0.9792	0.9271	0.9896	0.9896
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9524	**1.0000**	**1.0000**
STAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9367	0.9747	**1.0000**
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9800	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9700	**1.0000**	**1.0000**
UCEC	0.9991	**1.0000**	0.9189	0.9189	0.9189	0.9459	0.9459
Ranking (W|T|L)	0|19|3	**2|20|0**	0|17|5	0|16|6	0|13|9	0|18|4	0|16|6

techniques. The proposed RBNRO-DE with *k*-NN is ranked second by achieving identical results in 19 datasets as other techniques. It should also be noted that RF ranked third by yielding identical results in 18 datasets as others, while *k*-NN ranked fourth by yielding identical results in 17 datasets as other techniques. Finally, DT is ranked last by yielding identical results in 13 datasets as other techniques.

Table 8 shows the results of the proposed RBNRO-DE with *k*-NN and SVM classifiers compared with other popular ML classifiers regarding the F1-score values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 7 out of 22 datasets and identical results in 14 datasets as other techniques. The proposed RBNRO-DE with *k*-NN is ranked second by achieving the best results in 1 out of 22 datasets and identical results in 14 datasets as other techniques. It should also be noted that SVM ranked third by yielding identical results in 11 datasets as others, while *k*-NN and RF ranked fourth by yielding identical results in 10 datasets as other techniques. Finally, DT is ranked last by yielding identical results in 8 datasets as other techniques.

Comparison results of different versions of the proposed RBNRO-DE

This section introduced a comparison between different versions of the proposed RBNRO-DE (RBNRO-DE with *k*-NN, RBNRO-DE with SVM, and RBNRO-DE with

Table 8 F1-score values of the proposed RBNRO-DE against popular ML classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	*k*-NN	SVM	DT	RF	XGBoost
BLCA	**1.0000**	**1.0000**	0.9814	0.9937	0.9937	0.9693	0.9875
CESC	0.9919	**1.0000**	0.9919	0.9836	0.9919	0.9919	0.9919
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9091
COAD	**1.0000**	**1.0000**	0.9913	0.9913	0.9913	0.9913	0.9828
ESCA	0.9867	**0.9956**	0.9867	0.9867	0.9737	0.9867	0.9867
GBM	**1.0000**	**1.0000**	0.9836	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	0.9964	**1.0000**	0.9749	0.9282	0.9694	0.9652	0.9697
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	0.9990	**1.0000**	0.9903	0.9855	0.9364	0.9585	0.9856
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9865	0.9799	0.9865
LUAD	0.9950	**0.9955**	0.9900	0.9900	0.9800	0.9950	0.9900
LUSC	**1.0000**	**1.0000**	0.9843	0.9895	0.9368	0.9845	0.9896
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	0.9905	**0.9981**	0.9905	0.9905	0.9905	0.9905	0.9905
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9756	**1.0000**	**1.0000**
STAD	**1.0000**	0.9943	0.9634	0.9634	0.9193	0.9506	0.9634
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9800	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	0.9995	**1.0000**	0.9577	0.9577	0.9577	0.9722	0.9722
Ranking (W|T|L)	1|14|7	**7|14|1**	0|10|12	0|11|11	0|8|14	0|10|12	0|9|13

XGBoost) in terms of classification accuracy, fitness values, and number of selected features.

Table 9 shows the results of different versions of the proposed RBNRO-DE (RBNRO-DE with *k*-NN, RBNRO-DE with SVM, and RBNRO-DE with XGBoost) regarding classification accuracy values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 4 out of 22 datasets and identical results in 14 datasets as other versions. The proposed RBNRO-DE with XGBoost is ranked second by achieving the best results in 2 out of 22 datasets and identical results in 13 datasets as other versions. Finally, RBNRO-DE with *k*-NN is ranked third by achieving the best results in 1 out of 22 datasets and identical results in 14 datasets as other versions.

Table 10 shows the results of different versions of the proposed RBNRO-DE (RBNRO-DE with *k*-NN, RBNRO-DE with SVM, and RBNRO-DE with XGBoost) regarding the fitness values. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 8 out of 22 datasets and identical results in 11 datasets as other versions. The proposed RBNRO-DE with *k*-NN is ranked second by achieving the best results in 1 out of 22 datasets and identical results in 11 datasets as other versions. Finally, RBNRO-DE with XGBoost is ranked third by achieving the best results in 2 out of 22 datasets.

Table 9 Classification accuracy values of the proposed RBNRO-DE based on *k*-NN, SVM and XGBoost classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	RBNRO-DE with XGBoost
BLCA	**1.0000**	**1.0000**	**1.0000**
CESC	0.9839	**1.0000**	0.9839
CHOL	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**
ESCA	0.9744	0.9915	**1.0000**
GBM	**1.0000**	**1.0000**	**1.0000**
HNSC	0.9938	**1.0000**	0.9735
KICH	**1.0000**	**1.0000**	**1.0000**
KIRC	0.9983	**1.0000**	0.9917
KIRP	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**
LUAD	0.9913	0.9922	**1.0000**
LUSC	**1.0000**	**1.0000**	0.9910
PAAD	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**
SARC	0.9811	0.9962	0.9811
SKCM	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	0.9900	0.9556
THCA	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**
UCEC	0.9992	**1.0000**	0.9646
Ranking (W|T|L)	1|14|7	**4|14|4**	2|13|7

Table 11 shows the results of different versions of the proposed RBNRO-DE (RBNRO-DE with *k*-NN, RBNRO-DE with SVM, and RBNRO-DE with XGBoost) regarding the number of selected features. The empirical results show that the proposed RBNRO-DE with SVM is ranked first by achieving the best results in 8 out of 22 datasets and identical results in 10 datasets as other versions. The proposed RBNRO-DE with *k*-NN is ranked second by achieving the best results in 4 out of 22 datasets and identical results in 10 datasets as other versions. Finally, RBNRO-DE with XGBoost did not achieve the best results in any one of the utilized datasets. Therefore, the experimental results in this research will be conducted using *k*-NN and SVM classifiers due to their superiority and efficiency, as described in the following subsections.

Comparison results of the proposed RBNRO-DE against other state-of-the-art meta-heuristic algorithms

To demonstrate the dominance of RBNRO-DE over other counterparts in literature, the best performing RBNRO-DE algorithm with the two suggested classifiers, *k*-NN, and SVM, is compared with other state-of-the-art meta-heuristic algorithms executed in identical situations. The comparison with RBNRO-DE incorporates binary versions of some optimization algorithms, such as BSSA, BABC, BBA, BPSO, BWOA, BGWO, BGOA, BSFO, BBSA, BASO, BHHO, and BHGSO. Note that the 22 original gene

Table 10 Fitness values of the proposed RBNRO-DE based on *k*-NN, SVM and XGBoost classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	RBNRO-DE with XGBoost
BLCA	0.0029	**0.0025**	0.0030
CESC	0.0185	**0.0025**	0.0170
CHOL	**0.0025**	**0.0025**	0.0100
COAD	**0.0025**	**0.0025**	0.0100
ESCA	0.0279	**0.0113**	0.0300
GBM	**0.0025**	**0.0025**	0.0100
HNSC	0.0089	**0.0025**	0.0283
KICH	**0.0025**	**0.0025**	0.0100
KIRC	0.0047	**0.0025**	0.0112
KIRP	**0.0025**	**0.0025**	0.0100
LIHC	0.0025	0.0026	**0.0020**
LUAD	0.0112	0.0103	**0.0020**
LUSC	0.0026	**0.0025**	0.0129
PAAD	**0.0025**	**0.0025**	0.0100
PCPG	**0.0025**	**0.0025**	0.0100
READ	**0.0025**	**0.0025**	0.0100
SARC	0.0212	**0.0067**	0.0197
SKCM	**0.0025**	**0.0025**	0.0100
STAD	**0.0029**	0.0125	0.0470
THCA	**0.0025**	**0.0025**	0.0200
THYM	**0.0025**	**0.0025**	0.0100
UCEC	0.0038	**0.0025**	0.0030
Ranking (W|T|L)	1|11|10	**8|11|3**	2|0|20

expression datasets are first subjected to the Relief algorithm, and the 500 relevant features with the biggest weights are only chosen for use in the FS process. Subsequently, the suggested RBNRO-DE and the other state-of-the-art meta-heuristic algorithms are implemented only on these 500 pertinent features.

Comparisons based on the suggested k-NN classifier

Table 12 reveals the results of the proposed RBNRO-DE compared with other optimizers based on the *k*-NN classifier regarding the classification accuracy values evaluated under the same implementation conditions. The empirical results show that the proposed RBNRO-DE and BSFO scored the best in only one dataset. It should also be noted that all competitive algorithms yielded identical results in 20 datasets as RBNRO-DE with *k*-NN.

Table 13 reveals the average fitness and STD values of the proposed RBNRO-DE algorithm with its other peers based on *k*-NN under identical implementation requirements. The proposed RBNRO-DE with *k*-NN classifier demonstrates higher quality than different algorithms. By investigating Table 13, the results reveal that *k*-NN-based RBNRO-DE produced the least fitness values and competitive STD over all datasets. Furthermore, all the used datasets are large-scale, which verifies that the proposed RBNRO-DE can consistently execute on all datasets regardless of the size of the dataset. Currently, the

Table 11 The number of extracted features by the proposed RBNRO-DE based on *k*-NN, SVM, and XGBoost classifiers

Benchmark	RBNRO-DE with *k*-NN	RBNRO-DE with SVM	RBNRO-DE with XGBoost
BLCA	147	**125**	300
CESC	**125**	**125**	238
CHOL	**125**	**125**	210
COAD	**125**	**125**	149
ESCA	**125**	140	300
GBM	125	**124**	195
HNSC	139	**125**	225
KICH	**125**	**125**	214
KIRC	153	**127**	386
KIRP	**125**	**125**	150
LIHC	**125**	130	233
LUAD	130	**127**	212
LUSC	131	**125**	400
PAAD	**125**	**125**	200
PCPG	**125**	**125**	195
READ	**125**	**125**	216
SARC	**125**	146	190
SKCM	**125**	**125**	182
STAD	144	**129**	267
THCA	**125**	126	222
THYM	**125**	**125**	140
UCEC	151	**127**	188
Ranking (W|T|L)	4|10|8	**8|10|4**	0|0|22

proposed RBNRO-DE can be positively inferred to be promising, with a demonstrated ability to balance exploitation and exploration in the search space on iterations and escape from local optima. While standard algorithms may evolve, trapping it.

Table 14 shows the number of extracted features using the proposed RBNRO-DE and its other counterparts for training the *k*-NN classifier. The proposed RBNRO-DE surpassed the other algorithms in all datasets regarding the number of extracted features. Furthermore, the RBNRO-DE's capability to identify the most informative features is attributable to the ability to search within feasible regions while considering improved classification accuracy.

Table 15 displays the average precision values of the proposed RBNRO-DE algorithm with *k*-NN and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean precision values for 3 datasets. Alternatively, BABC, BPSO, BGWO, BGOA, BSFO, and BHHO achieved identical results as the proposed RBNRO-DE in 19 datasets, while BWOA performed similarly in 18 datasets. BASO and BHGSO ranked fourth by achieving identical results as the proposed RBNRO-DE in 17 datasets. Finally, BBA yielded identical results as the proposed RBNRO-DE in 15 datasets, ranking it last among all methods.

Table 12 The proposed RBNRO-DE scores with *k*-NN and its peers in terms of mean values of classification accuracy

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Mean	**1.0000**	0.9930	0.9977	0.9934	0.9891	0.9950	0.9938
	STD	**0.0000**	0.0057	0.0047	0.0058	0.0029	0.0058	0.0058
CESC	Mean	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	**0.9938**	0.9914	0.9914	0.9912	0.9912	0.9917	0.9912
	STD	0.0041	0.0016	0.0016	**0.0000**	**0.0000**	0.0022	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	0.9983	0.9928	0.9939	0.9923	0.9917	0.9939	0.9923
	STD	0.0033	0.0028	0.0037	0.0021	**0.0000**	0.0037	0.0021
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUAD	Mean	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUSC	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	STD	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SKCM	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	STD	**1.0000**	0.9959	0.9985	0.9956	0.9889	0.9963	0.9952
	Mean	**0.0000**	0.0054	0.0038	0.0054	**0.0000**	0.0052	0.0055
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	**0.9992**	0.9850	0.9875	0.9767	0.9692	0.9842	0.9767
	STD	**0.0045**	0.0122	0.0125	0.0062	0.0106	0.0120	0.0062
Ranking	W|T|L	**1|20|1**	0|18|4	0|18|4	0|18|4	0|18|4	0|18|4	0|18|4

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Mean	0.9930	**1.0000**	0.9907	0.9942	0.9938	0.9888
	STD	0.0057	**0.0000**	0.0047	0.0058	0.0058	0.0021

Table 12 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Mean	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	0.9912	0.9923	0.9912	0.9914	0.9912	0.9912
	STD	**0.0000**	0.0030	**0.0000**	0.0016	**0.0000**	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	0.9937	**0.9986**	0.9934	0.9931	0.9926	0.9917
	STD	0.0035	0.0031	0.0033	0.0031	0.0025	**0.0000**
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUAD	Mean	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUSC	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	STD	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SKCM	STD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	Mean	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	STD	0.9967	**1.0000**	0.9922	0.9952	0.9963	0.9889
	Mean	0.0051	**0.0000**	0.0051	0.0055	0.0052	**0.0000**
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	0.9775	0.9875	0.9783	0.9792	0.9758	0.9683
	STD	0.0075	0.0125	0.0085	0.0093	**0.0045**	0.0111
Ranking	W|T|L	0|18|4	**1|20|1**	0|18|4	0|18|4	0|18|4	0|18|4

Table 13 The proposed RBNRO-DE scores with *k*-NN and its peers in terms of mean values of fitness

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Mean	**0.0029**	0.0111	0.0071	0.0112	0.0147	0.0094	0.0107
	STD	**0.0002**	0.0053	0.0044	0.0054	0.0024	0.0055	0.0054
CESC	Mean	**0.0185**	0.0199	0.0202	0.0205	0.0198	0.0200	0.0202
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
CHOL	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
COAD	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
ESCA	Mean	**0.0279**	0.0293	0.0296	0.0299	0.0293	0.0294	0.0297
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
GBM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
HNSC	Mean	**0.0089**	0.0125	0.0129	0.0133	0.0128	0.0124	0.0131
	STD	0.0038	0.0016	0.0014	**0.0001**	0.0003	0.0021	**0.0001**
KICH	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
KIRC	Mean	**0.0047**	0.0111	0.0104	0.0122	0.0120	0.0102	0.0120
	STD	0.0030	0.0027	0.0034	0.0019	0.0002	0.0034	0.0019
KIRP	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
LIHC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0042
	STD	0.0002	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
LUAD	Mean	**0.0112**	0.0126	0.0129	0.0131	0.0124	0.0127	0.0129
	STD	0.0002	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	**0.0001**
LUSC	Mean	**0.0026**	0.0040	0.0043	0.0045	0.0038	0.0041	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	**0.0001**	0.0002
PAAD	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
PCPG	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
READ	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
SARC	Mean	**0.0212**	0.0226	0.0229	0.0232	0.0226	0.0227	0.0230
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
SKCM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
STAD	Mean	**0.0029**	0.0083	0.0062	0.0093	0.0147	0.0081	0.0095
	STD	0.0003	0.0050	0.0036	0.0051	0.0002	0.0050	0.0051
THCA	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
THYM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
UCEC	Mean	**0.0038**	0.0192	0.0171	0.0278	0.0354	0.0201	0.0277
	STD	**0.0044**	0.0120	0.0122	0.0061	0.0100	0.0119	0.0062
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Mean	0.0114	0.0044	0.0131	0.0103	0.0108	0.0159
	STD	0.0054	**0.0002**	0.0044	0.0055	0.0054	0.0018

Table 13 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Mean	0.0202	0.0196	0.0197	0.0202	0.0203	0.0206
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
CHOL	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
COAD	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
ESCA	Mean	0.0296	0.0290	0.0291	0.0296	0.0298	0.0300
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	0.0002
GBM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
HNSC	Mean	0.0132	0.0115	0.0127	0.0129	0.0132	0.0136
	STD	**0.0001**	0.0027	0.0002	0.0015	**0.0001**	**0.0001**
KICH	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
KIRC	Mean	0.0107	0.0058	0.0104	0.0112	0.0117	0.0129
	STD	0.0032	0.0027	0.0030	0.0028	0.0022	**0.0001**
KIRP	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
LIHC	Mean	0.0043	0.0036	0.0038	0.0043	0.0042	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
LUAD	Mean	0.0129	0.0122	0.0124	0.0129	0.0130	0.0134
	STD	**0.0001**	**0.0001**	0.0002	**0.0001**	**0.0001**	**0.0001**
LUSC	Mean	0.0043	0.0037	0.0039	0.0043	0.0043	0.0048
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
PAAD	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
PCPG	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
READ	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
SARC	Mean	0.0229	0.0223	0.0224	0.0229	0.0230	0.0233
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
SKCM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
STAD	Mean	0.0080	0.0043	0.0117	0.0094	0.0084	0.0157
	STD	0.0048	0.0003	0.0048	0.0052	0.0049	**0.0001**
THCA	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
THYM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
UCEC	Mean	0.0269	0.0167	0.0256	0.0252	0.0286	0.0367
	STD	0.0072	0.0119	0.0083	0.0091	0.0045	0.0105
Ranking	W|T|L	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Table 14 The number of extracted features by the proposed RBNRO-DE and its peers for training the *k*-NN

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Features number	**147**	210	238	235	196	218	229
	STD	012.2759	020.5005	019.0668	017.1231	023.8320	018.2740	019.4349
CESC	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
CHOL	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
COAD	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
ESCA	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
GBM	Features number	**125**	195	213	225	194	200	213
	STD	006.5201	011.1676	005.4017	**003.1107**	009.6552	009.0478	008.5052
HNSC	Features number	**139**	203	223	227	203	211	218
	STD	014.6040	009.2619	011.9564	**003.9131**	013.2363	007.6528	007.4139
KICH	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
KIRC	Features number	**153**	201	220	228	192	210	216
	STD	027.5450	009.6266	011.7362	008.2710	008.7334	014.0754	010.1085
KIRP	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
LIHC	Features number	**125**	197	214	226	193	201	212
	STD	007.9387	007.7803	005.2248	**003.3599**	009.1664	010.0249	009.3500
LUAD	Features number	**130**	197	216	223	191	205	213
	STD	008.1948	008.9805	004.7380	006.1586	008.6065	010.5715	006.1465
LUSC	Features number	**131**	199	216	223	189	204	213
	STD	007.3182	008.7702	003.5448	004.8854	010.6856	007.1954	009.7011
PAAD	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
PCPG	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
READ	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
SARC	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
SKCM	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
STAD	Features number	**144**	215	236	243	188	221	235
	STD	013.0348	018.4264	013.9429	016.2897	010.2689	019.2425	022.6314
THCA	Features number	**125**	195	213	225	194	200	214
	STD	006.8977	011.2002	005.2890	**003.0955**	009.6552	009.8098	008.5935
THYM	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
UCEC	Features number	**151**	216	238	236	242	220	231
	STD	016.5202	015.4128	015.8235	010.1127	034.1306	011.2574	011.2706
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Features number	226	219	195	228	232	237
	STD	014.8147	**011.4022**	013.1673	014.5967	018.9006	014.1368

Table 14 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Features number	213	182	186	213	216	233
	STD	005.6588	005.1941	006.4890	005.8784	006.0241	005.6373
CHOL	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
COAD	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
ESCA	Features number	213	182	186	212	219	230
	STD	005.6588	005.1941	006.4890	006.9363	004.6814	007.6373
GBM	Features number	213	181	185	211	212	233
	STD	005.7236	006.1079	006.4015	006.4446	004.9872	004.3855
HNSC	Features number	221	195	196	220	224	241
	STD	005.3150	019.4685	008.2621	007.7075	005.6832	006.2125
KICH	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
KIRC	Features number	222	223	193	219	219	234
	STD	015.1793	027.3403	017.3569	013.8249	012.7844	**004.2059**
KIRP	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
LIHC	Features number	214	181	189	213	212	232
	STD	004.8886	004.6332	006.8903	005.6941	004.8061	004.9711
LUAD	Features number	216	181	192	215	218	238
	STD	**004.4297**	005.1339	008.5025	006.1261	004.7123	004.9420
LUSC	Features number	216	183	195	216	215	241
	STD	**003.4398**	005.1635	006.4049	005.5422	006.1627	004.6596
PAAD	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
PCPG	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
READ	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
SARC	Features number	213	182	186	213	216	232
	STD	005.6588	005.1941	006.4890	006.3000	004.9778	003.4000
SKCM	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
STAD	Features number	233	216	202	231	237	237
	STD	014.2471	012.5628	013.9951	014.0445	015.3500	**005.8622**
THCA	Features number	213	182	187	211	212	232
	STD	005.6886	005.2624	005.8750	006.4446	005.0773	004.7843
THYM	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
UCEC	Features number	229	215	207	228	235	267
	STD	011.6593	029.4144	010.8403	010.3170	**006.1427**	030.5487
Ranking	W|T|L	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Table 15 The proposed RBNRO-DE scores with *k*-NN and its peers in terms of mean precision values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9996	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9989	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
ESCA	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**	**0.9737**
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**0.9929**	0.9647	0.9654	0.9650	0.9635	0.9669	0.9666	0.9657	0.9626	0.9654	0.9654	0.9610	0.9629
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**1.0000**	0.9977	0.9974	0.9984	0.9987	0.9977	0.9980	0.9954	0.9964	0.9971	0.9974	0.9967	0.9980
LUSC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9979	**1.0000**	0.9997	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9790	0.9993
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	0.9518	0.9518	0.9518	0.9487	0.9518	0.9518	0.9518	0.9518	0.9518	0.9518	0.9900	0.9509
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9991	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9333	0.9991
Ranking (W|T|L)	**3|19|0**	0|19|3	0|19|3	0|19|3	0|15|7	0|19|3	0|18|4	0|19|3	0|19|3	0|19|3	0|19|3	0|17|5	0|17|5

Table 16 The proposed RBNRO-DE scores with *k*-NN and its peers in terms of mean recall values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
ESCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	0.9952	0.9945	0.9948	0.9959	0.9928	0.9931	0.9942	0.9976	0.9945	0.9945	0.9979	0.9973
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**0.9981**	0.9904	0.9904	0.9904	0.9897	0.9904	0.9904	0.9904	0.9904	0.9904	0.9904	0.9901	0.9904
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**1.0000**	0.9924	0.9927	0.9917	0.9914	0.9924	0.9921	0.9947	0.9937	0.9931	0.9927	0.9934	0.9921
LUSC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9958	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9993	**1.0000**	0.9931	0.9927
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9975	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9962	0.9975
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9937	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9964	**1.0000**
Ranking (W|T|L)	**3|19|0**	0|19|3	0|19|3	0|19|3	0|16|6	0|19|3	0|19|3	0|19|3	0|19|3	0|18|4	0|19|3	0|16|6	0|17|5

Table 17 The proposed RBNRO-DE scores with *k*-NN and its peers in terms of mean F1-score values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9998	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**	**0.9919**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9994	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
ESCA	**1.0000**	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**0.9964**	0.9797	0.9797	0.9797	0.9794	0.9797	0.9797	0.9797	0.9797	0.9797	0.9797	0.9798	0.9797
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**0.9990**	0.9952	0.9952	0.9952	0.9948	0.9952	0.9952	0.9952	0.9952	0.9952	0.9952	0.9950	0.9952
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**	**0.9950**
LUSC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9969	**1.0000**	0.9998	**1.0000**	**1.0000**	0.9997	**1.0000**	0.9963	0.9960
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**	**0.9905**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	0.9753	0.9753	0.9753	0.9724	0.9753	0.9753	0.9753	0.9753	0.9753	0.9753	0.9718	0.9736
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9964	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9964	0.9996
Ranking (W|T|L)	**4|18|0**	0|18|4	0|18|4	0|18|4	0|14|8	0|18|4	0|17|5	0|18|4	0|18|4	0|17|5	0|18|4	0|16|6	0|16|6

Table 18 The proposed RBNRO-DE scores with SVM and its peers in terms of mean values of classification accuracy

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CESC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	**0.9915**	0.9752	0.9769	0.9744	0.9744	0.9769	0.9744
	STD	0.0121	0.0046	0.0077	**0.0000**	**0.0000**	0.0077	**0.0000**
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9992	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.0029	**0.0000**	**0.0000**
LUAD	Mean	**0.9922**	0.9913	0.9913	0.9913	0.9913	0.9913	0.9913
	STD	0.0026	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	Mean	0.9962	0.9862	0.9887	0.9862	0.9811	0.9893	0.9868
	STD	0.0075	0.0083	0.0092	0.0083	**0.0000**	0.0093	0.0086
SKCM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	Mean	**0.9900**	0.9889	0.9889	0.9889	0.9893	0.9889	0.9889
	STD	0.0033	**0.0000**	**0.0000**	**0.0000**	0.0020	**0.0000**	**0.0000**
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Ranking	W|T|L	**2|19|1**	0|19|3	0|19|3	0|19|3	0|19|3	0|19|3	0|19|3

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**

Table 18 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	0.9752	0.9778	0.9752	0.9752	0.9744	0.9744
	STD	0.0046	0.0087	0.0046	0.0046	**0.0000**	**0.0000**
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUAD	Mean	0.9913	0.9916	0.9913	0.9913	0.9916	0.9913
	STD	**0.0000**	0.0016	**0.0000**	**0.0000**	0.0016	**0.0000**
LUSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	Mean	0.9855	**0.9987**	0.9836	0.9855	0.9843	0.9811
	STD	0.0080	0.0047	0.0064	0.0080	0.0070	**0.0000**
SKCM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	Mean	0.9889	0.9893	0.9889	0.9889	0.9889	0.9889
	STD	**0.0000**	0.0020	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Ranking	W|T|L	0|19|3	1|19|2	0|19|3	0|19|3	0|19|3	0|19|3

Table 19 The proposed RBNRO-DE scores with SVM and its peers in terms of mean values of fitness

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
CESC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
CHOL	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
COAD	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
ESCA	Mean	**0.0113**	0.0284	0.0272	0.0299	0.0293	0.0269	0.0297
	STD	0.0118	0.0046	0.0075	**0.0001**	0.0002	0.0075	0.0002
GBM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
HNSC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
KICH	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
KIRC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0038	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
KIRP	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
LIHC	Mean	**0.0026**	0.0041	0.0044	0.0045	0.0050	0.0043	0.0044
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	0.0028	**0.0001**	0.0002
LUAD	Mean	**0.0103**	0.0125	0.0129	0.0131	0.0125	0.0126	0.0129
	STD	0.0025	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
LUSC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
PAAD	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
PCPG	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
READ	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
SARC	Mean	0.0067	0.0178	0.0157	0.0184	0.0226	0.0148	0.0176
	STD	0.0073	0.0080	0.0088	0.0080	0.0002	0.0090	0.0082
SKCM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
STAD	Mean	**0.0125**	0.0149	0.0153	0.0155	0.0145	0.0150	0.0153
	STD	0.0031	0.0002	**0.0001**	**0.0001**	0.0020	0.0002	0.0002
THCA	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
THYM	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0039	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
UCEC	Mean	**0.0025**	0.0039	0.0043	0.0045	0.0038	0.0040	0.0043
	STD	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002	0.0002	0.0002
Ranking	W|T|L	**21|0|1**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**

Table 19 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
CHOL	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
COAD	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
ESCA	Mean	0.0288	0.0258	0.0283	0.0288	0.0298	0.0300
	STD	0.0045	0.0082	0.0046	0.0044	**0.0001**	0.0002
GBM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
HNSC	Mean	0.0043	0.0037	0.0037	0.0042	0.0042	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
KICH	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
KIRC	Mean	0.0043	0.0036	0.0038	0.0043	0.0043	0.0046
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	0.0002
KIRP	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
LIHC	Mean	0.0045	0.0037	0.0040	0.0045	0.0045	0.0050
	STD	**0.0001**	**0.0001**	0.0002	**0.0001**	**0.0001**	0.0002
LUAD	Mean	0.0129	0.0120	0.0123	0.0129	0.0126	0.0133
	STD	**0.0001**	0.0014	**0.0001**	**0.0001**	0.0015	**0.0001**
LUSC	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
PAAD	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
PCPG	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
READ	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
SARC	Mean	0.0187	**0.0057**	0.0200	0.0187	0.0200	0.0233
	STD	0.0077	0.0044	0.0061	0.0077	0.0067	**0.0001**
SKCM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
STAD	Mean	0.0153	0.0143	0.0147	0.0152	0.0153	0.0156
	STD	**0.0001**	0.0018	**0.0001**	**0.0001**	**0.0001**	**0.0001**
THCA	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
THYM	Mean	0.0043	0.0036	0.0037	0.0042	0.0042	0.0047
	STD	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
UCEC	Mean	0.0043	0.0037	0.0038	0.0042	0.0043	0.0047
	STD	**0.0001**	**0.0001**	0.0002	**0.0001**	**0.0001**	**0.0001**
Ranking	W|T|L	0|0|22	1|0|21	0|0|22	0|0|22	0|0|22	0|0|22

Table 20 The number of extracted features by the proposed RBNRO-DE and its peers for training the SVM

Benchmark	Metric	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA
BLCA	Features number	**125**	195	213	225	194	200	213
	STD	006.5286	011.4430	005.1800	**003.1107**	009.6552	009.3014	008.4364
CESC	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
CHOL	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
COAD	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
ESCA	Features number	**140**	195	215	225	194	202	214
	STD	025.9610	011.1676	009.5798	**003.1107**	009.6552	012.3307	008.3526
GBM	Features number	**124**	195	213	225	194	199	214
	STD	006.9048	011.2722	005.3071	**003.1107**	009.4670	010.2752	008.3552
HNSC	Features number	**125**	196	214	225	194	202	214
	STD	005.3109	008.4916	005.2240	**003.1910**	009.5620	010.7559	009.3384
KICH	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
KIRC	Features number	**127**	197	215	225	192	200	213
	STD	006.4339	009.4175	005.0869	004.2348	008.7022	011.5935	007.6397
KIRP	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
LIHC	Features number	**130**	204	220	226	213	214	219
	STD	007.2622	006.9902	006.2901	006.7951	017.6032	007.0566	007.8780
LUAD	Features number	**127**	195	213	225	194	199	214
	STD	008.4892	011.1676	005.3071	**003.1107**	009.6552	009.3690	008.3526
LUSC	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
PAAD	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
PCPG	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
READ	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
SARC	Features number	**146**	203	227	234	194	211	224
	STD	015.8213	020.8620	020.2457	014.4353	009.6552	017.7421	017.3949
SKCM	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
STAD	Features number	**129**	195	213	225	194	199	214
	STD	010.9780	010.2229	005.3765	**003.2912**	009.7883	009.0863	008.6847
THCA	Features number	**126**	195	213	225	194	200	214
	STD	006.0646	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
THYM	Features number	**125**	195	213	225	194	200	214
	STD	006.5072	011.1676	005.3071	**003.1107**	009.6552	009.3014	008.3526
UCEC	Features number	**127**	196	213	225	192	201	213
	STD	005.9258	009.0591	006.3084	004.2668	007.8506	008.2401	007.7356
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	Features number	213	182	186	211	212	232
	STD	005.6588	004.9000	005.7673	006.4446	005.2478	004.4233

Table 20 (continued)

Benchmark	Metric	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
CESC	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
CHOL	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
COAD	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
ESCA	Features number	213	190	186	212	219	230
	STD	006.4381	020.4246	006.4712	008.9800	004.6814	007.6373
GBM	Features number	213	182	186	211	211	232
	STD	005.6926	005.0711	006.5051	006.4446	004.9193	004.1500
HNSC	Features number	214	184	187	211	211	232
	STD	005.2202	004.1162	005.7442	005.6612	004.7940	005.0205
KICH	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
KIRC	Features number	213	182	188	214	215	231
	STD	004.9378	006.3481	005.3084	005.8143	**004.1991**	008.0164
KIRP	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
LIHC	Features number	224	186	201	223	225	250
	STD	**004.1324**	007.3082	007.6577	006.4781	004.8103	007.9722
LUAD	Features number	213	183	186	212	215	233
	STD	005.6588	009.2949	006.3965	006.0970	005.9819	004.3436
LUSC	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
PAAD	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
PCPG	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
READ	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
SARC	Features number	218	225	190	221	221	232
	STD	013.9006	018.0083	013.4113	016.3965	011.7636	**003.7035**
SKCM	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
STAD	Features number	213	184	186	212	215	232
	STD	005.5447	012.1803	006.9193	005.8849	005.1182	004.5803
THCA	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
THYM	Features number	213	182	186	211	212	233
	STD	005.6588	005.1941	006.4890	006.4446	005.0773	004.3626
UCEC	Features number	213	183	189	212	213	234
	STD	005.6656	004.3767	008.7776	004.9845	004.9728	**003.9660**
Ranking	W|T|L	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Table 21 The proposed RBNRO-DE scores with SVM and its peers in terms of mean precision values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9989	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9989	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
ESCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9982	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9921
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	0.9414	0.9417	0.9375	0.9284	0.9414	0.9399	0.9411	0.9417	0.9411	0.9405	0.8455	0.9315
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9994	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9987	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9993
LUSC	**1.0000**	0.9986	0.9986	0.9986	0.9983	0.9966	0.9986	0.9990	0.9986	0.9993	0.9986	1.0814	0.9997
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**1.0000**	0.9994	**1.0000**	0.9994	0.9950	0.9994	**1.0000**	0.9994	**1.0000**	0.9962	0.9994	**1.0000**	0.9912
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**0.9888**	0.9484	0.9511	0.9417	0.9361	0.9473	0.9469	0.9469	0.9518	0.9391	0.9465	0.9520	0.9350
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	16.3217	**1.0000**
UCEC	**1.0000**	0.9991	0.9982	0.9982	0.9930	**1.0000**	0.9982	0.9982	**1.0000**	0.9991	**1.0000**	**1.0000**	0.9939
Ranking (W|T|L)	**3|19|0**	0|17|5	0|18|4	0|17|5	0|12|10	0|18|4	0|18|4	0|17|5	0|19|3	0|17|5	0|18|4	0|18|4	0|15|7

Table 22 The proposed RBNRO-DE scores with SVM and its peers in terms of mean recall values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9994	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9994	**1.0000**
ESCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	**1.0000**	**1.0000**	0.9997	0.9993	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9993	0.9993
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	0.9904	0.9904	0.9904	0.9901	0.9904	0.9904	0.9904	0.9904	0.9904	0.9904	0.9878	0.9904
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9950	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9993	**1.0000**	0.9983	0.9977
LUSC	**1.0000**	0.9910	0.9910	0.9910	0.9913	0.9896	0.9910	0.9906	0.9910	0.9903	0.9910	0.9910	0.9899
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	0.9955	0.9964	0.9838	0.9820	0.9946	0.9910	0.9901	0.9964	0.9847	0.9937	0.9784	0.9793
Ranking (W|T|L)	**3|19|0**	0|19|3	0|19|3	0|18|4	0|16|6	0|19|3	0|19|3	0|19|3	0|19|3	0|18|4	0|19|3	0|16|6	0|17|5

Table 23 The proposed RBNRO-DE scores with SVM and its peers in terms of mean F1-score values

Benchmark	RBNRO-DE	BSSA	BABC	BPSO	BBA	BGWO	BWOA	BGOA	BSFO	BHHO	BBSA	BASO	BHGSO
BLCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9995	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9991	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9997	**1.0000**
ESCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9991	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9982	0.9960
GBM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	0.9698	0.9700	0.9676	0.9625	0.9698	0.9690	0.9697	0.9700	0.9697	0.9694	0.9573	0.9642
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	0.9952	0.9952	0.9952	0.9947	0.9952	0.9952	0.9952	0.9952	0.9952	0.9952	0.9932	0.9952
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LUAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9969	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9997	**1.0000**	0.9988	0.9985
LUSC	**1.0000**	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948	0.9948
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**1.0000**	0.9997	0.9981	0.9997	0.9975	0.9997	**1.0000**	0.9997	**1.0000**	0.9981	0.9997	0.9971	0.9956
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**0.9943**	0.9735	0.9749	0.9699	0.9670	0.9729	0.9727	0.9727	0.9753	0.9686	0.9725	0.9656	0.9664
THCA	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	0.9973	0.9973	0.9909	0.9873	0.9973	0.9945	0.9941	0.9873	0.9918	0.9968	0.9868	0.9864
Ranking (W|T|L)	**5|17|0**	0|16|6	0|16|6	0|16|6	0|12|10	0|16|6	0|17|5	0|16|6	0|17|5	0|15|7	0|16|6	0|13|9	0|14|8

Table 16 displays the average recall values of the proposed RBNRO-DE algorithm with *k*-NN and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean recall values for 3 datasets. Alternatively, BSSA, BABC, BPSO, BGWO, BWOA, BGOA, BSFO, and BBSA achieved identical results as the proposed RBNRO-DE in 19 datasets, while BHHO performed similarly in 18 datasets. BHGSO ranked fourth by achieving identical results as the proposed RBNRO-DE in 17 datasets. Finally, BASO and BBA yielded identical results as the proposed RBNRO-DE in 16 datasets, ranking it last among all methods.

Table 17 displays the average F1-score values of the proposed RBNRO-DE algorithm with *k*-NN and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean F1-score values for 4 datasets. Alternatively, BSSA, BABC, BPSO, BGWO, BGOA, BSFO, and BBSA achieved identical results as the proposed RBNRO-DE in 18 datasets, while BWOA and BHHO performed similarly in 17 datasets. BASO and BHGSO ranked fourth by achieving identical results as the proposed RBNRO-DE in 16 datasets. Finally, BBA yielded identical results as the proposed RBNRO-DE in 14 datasets, ranking it last among all methods.

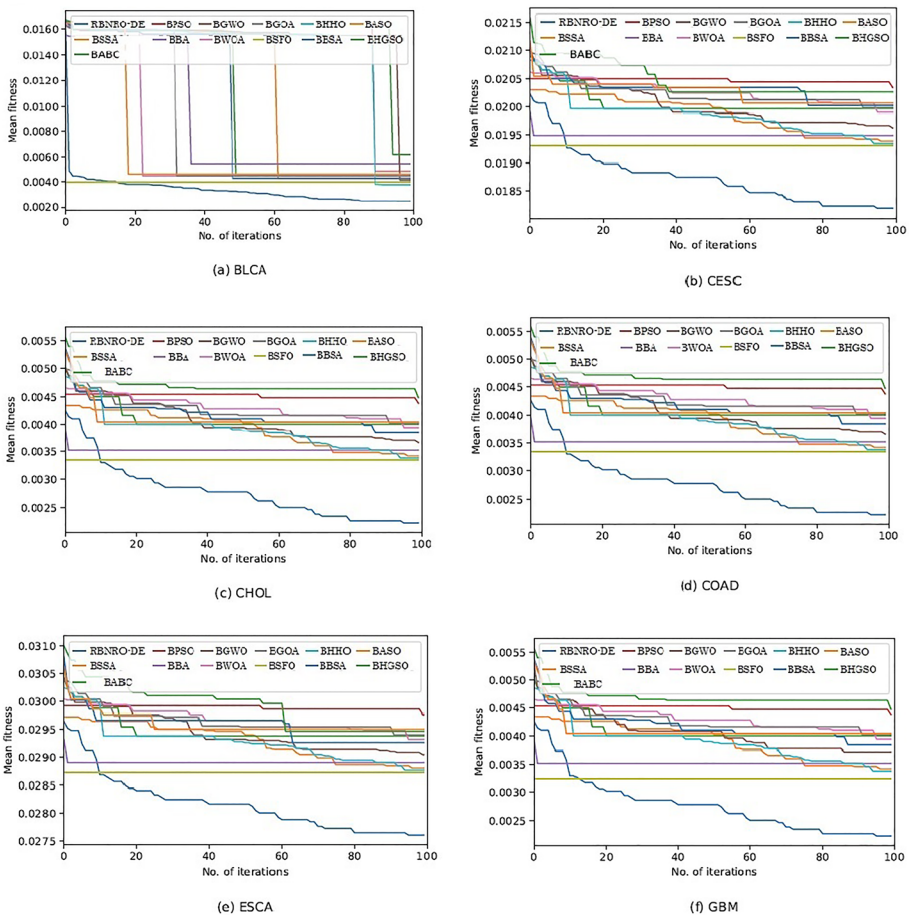

Fig. 4 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on *k*-NN classifier over all datasets

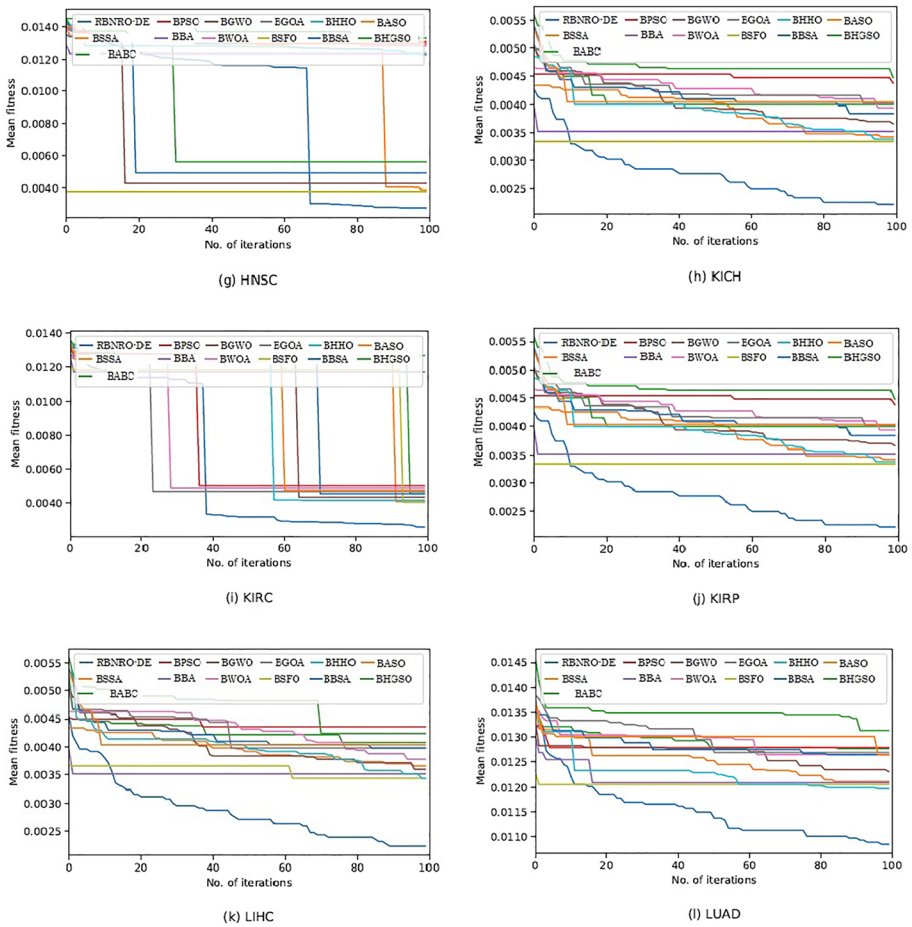

Fig. 5 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on *k*-NN classifier over all datasets (Cont.)

Comparisons based on the suggested SVM classifier

Table 18 shows the results of the proposed RBNRO-DE compared with other optimizers based on the SVM classifier regarding the classification accuracy values that are fairly evaluated under the same implementation conditions. The empirical results show that the proposed RBNRO-DE is ranked first by achieving the best results in 2 out of 22 datasets. BSFO is ranked second with the best results in only one dataset. It should also be noted that all competitive algorithms yielded identical results in 19 datasets as the proposed RBNRO-DE with SVM.

Table 19 reveals the average fitness and STD values of RBNRO-DE with its other peers, based on SVM, under identical implementation requirements. Notably, the proposed RBNRO-DE with SVM classifier demonstrates higher quality than other algorithms. By investigating Table 19, the results reveal that SVM-based RBNRO-DE produced the least values of fitness along with competitive STD in 21 out of 22 datasets, accounting for 95% of all datasets. Furthermore, all the used datasets are large-scale, which verifies that the proposed RBNRO-DE is capable of consistently executing on all datasets regardless of the size of the dataset. For the only dataset that BSFO won, the mean fitness value is very close to RBNRO-DE. None of the other

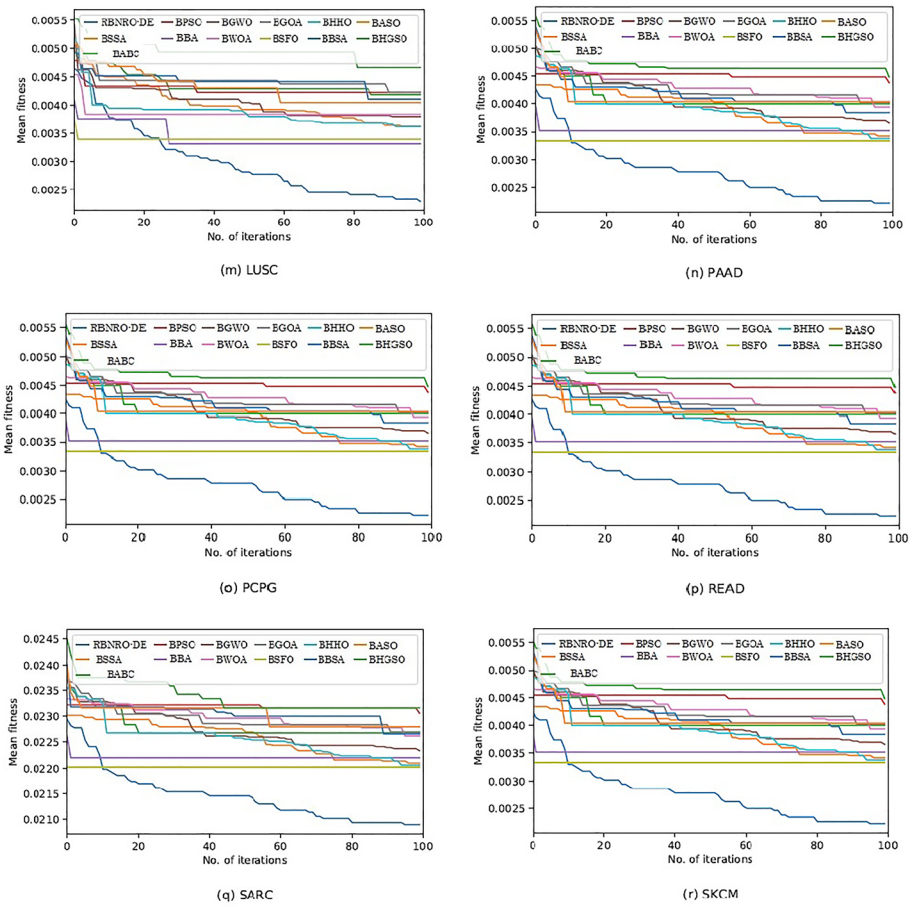

Fig. 6 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on *k*-NN classifier over all datasets (Cont.)

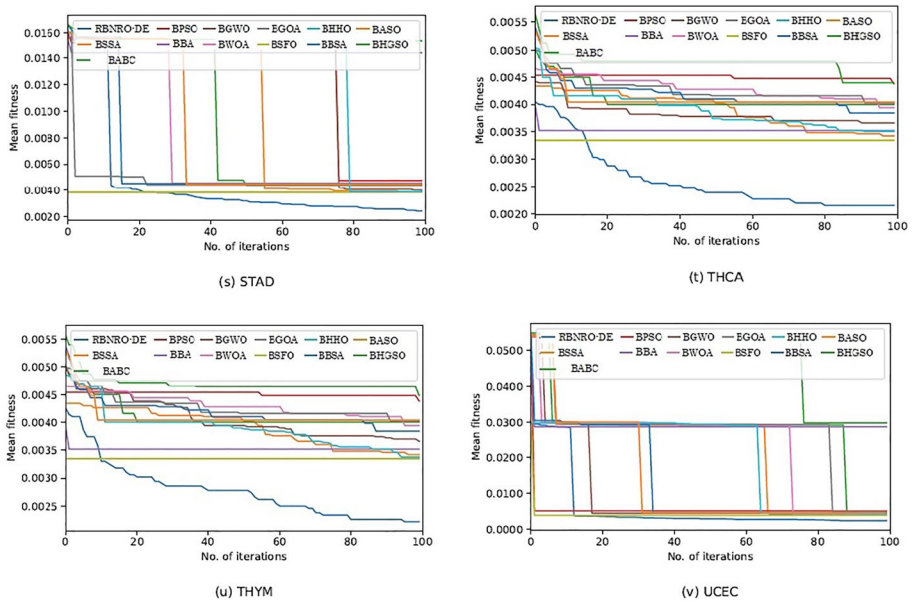

Fig. 7 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on *k*-NN classifier over all datasets (Cont.)

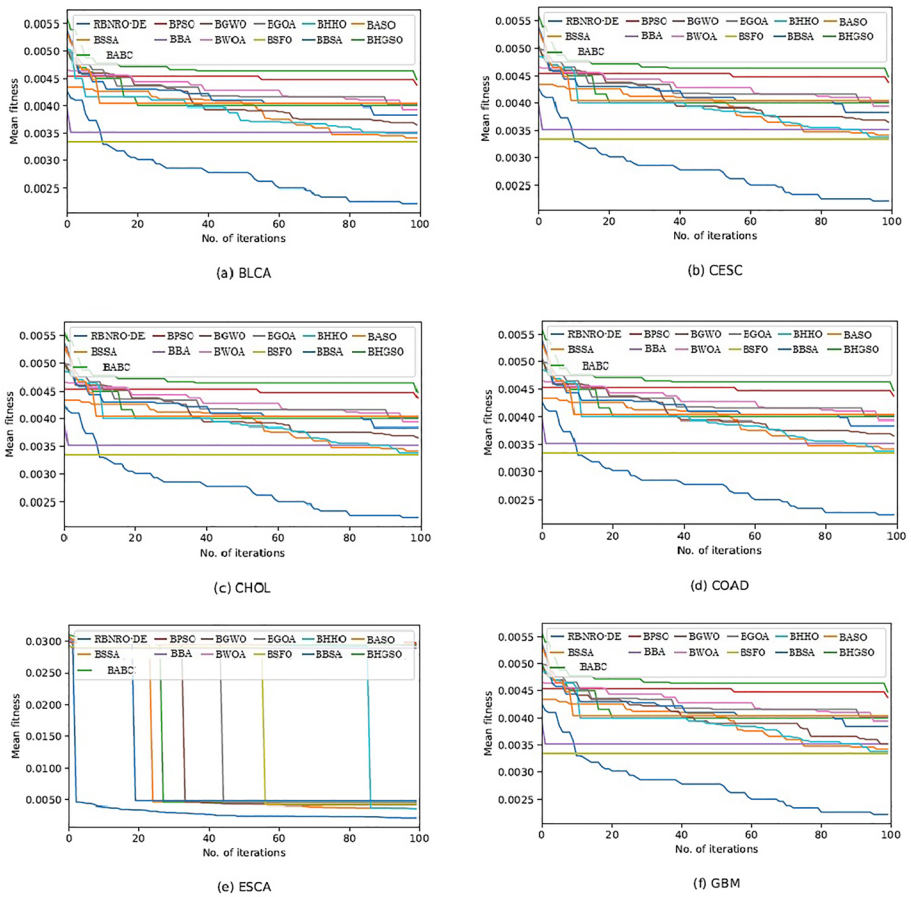

Fig. 8 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on SVM classifier over all datasets

algorithms compared to RBNRO-DE ranked first in the 22 datasets. Now, RBNRO-DE can be positively inferred to be promising, with a demonstrated ability to balance between exploitation and exploration in the search space on iterations and escape from local optima. While common algorithms may evolve, trapping it.

Based on the number of extracted features, the outcomes of the proposed RBNRO-DE and other counterparts for training the SVM classifier are revealed in Table 20. By investigating the scores, an attractive observation is made for the proposed RBNRO-DE based on SVM, which did better than other algorithms over 22 of the 22 datasets used in this paper. Furthermore, The excellence of the proposed RBNRO-DE with SVM in this context confirms its ability to identify the most significant regions of the search space and escape the search through regions of non-feasible spaces.

Table 21 displays the average precision values of the proposed RBNRO-DE algorithm with SVM and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean precision values for 3 datasets. Alternatively, BSFO achieved identical results as the proposed RBNRO-DE in 19 datasets, while BABC, BGWO, BWOA, BBSA, and BASO performed similarly in 18 datasets. BSSA, BPSO, BGOA, and BHHO ranked fourth by achieving identical

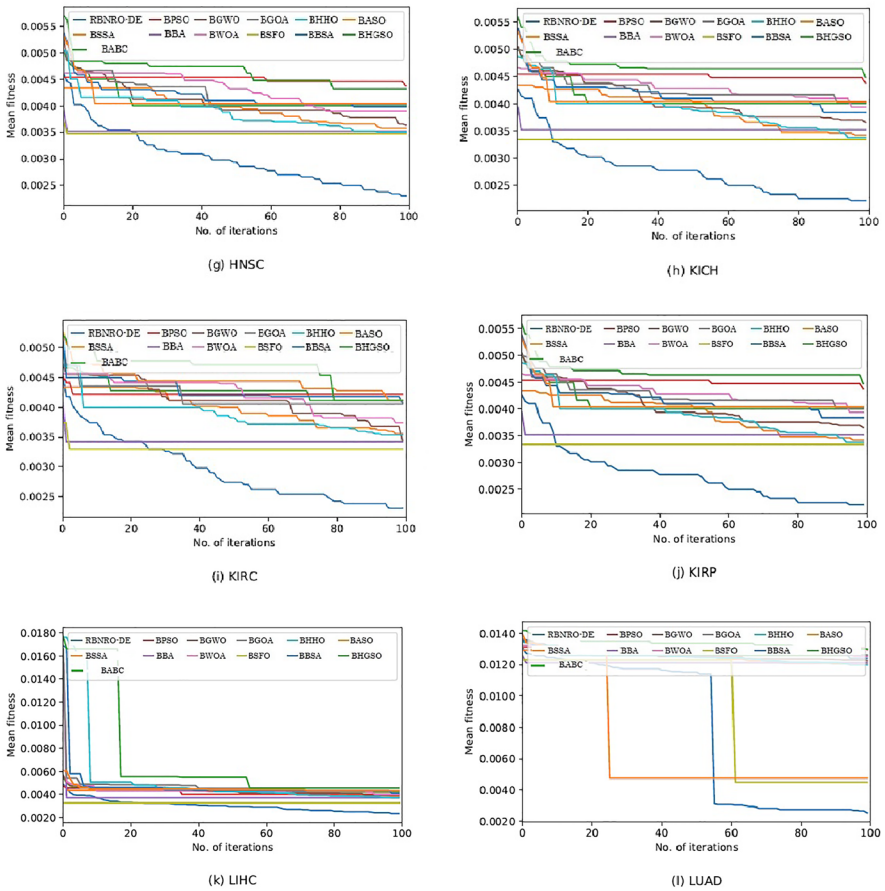

Fig. 9 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on SVM classifier over all datasets (Cont.)

results as the proposed RBNRO-DE in 17 datasets. Finally, BBA yielded identical results as the proposed RBNRO-DE in 12 datasets, ranking it last among all methods.

Table 22 displays the average recall values of the proposed RBNRO-DE algorithm with SVM and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean recall values for 3 datasets. Alternatively, BSSA, BABC, BGWO, BWOA, BGOA, BSFO, and BBSA achieved identical results as the proposed RBNRO-DE in 19 datasets, while BPSO and BHHO performed similarly in 18 datasets. BHGSO ranked fourth by achieving identical results as the proposed RBNRO-DE in 17 datasets. Finally, BASO and BBA yielded identical results as the proposed RBNRO-DE in 16 datasets, ranking it last among all methods.

Table 23 displays the average F1-score values of the proposed RBNRO-DE algorithm with SVM and its counterparts. Out of 22 datasets, the proposed RBNRO-DE performed better than other methods in terms of mean F1-score values for 5 datasets. Alternatively, BWOA and BSFO achieved identical results as the proposed RBNRO-DE in 17 datasets, while BSSA, BABC, BPSO, BGWO, BGOA, and BBSA performed similarly in 16 datasets. BHHO ranked fourth by achieving identical results as the proposed RBNRO-DE in 15 datasets. Finally, BBA yielded identical results as the proposed RBNRO-DE in 12 datasets, ranking it last among all methods.

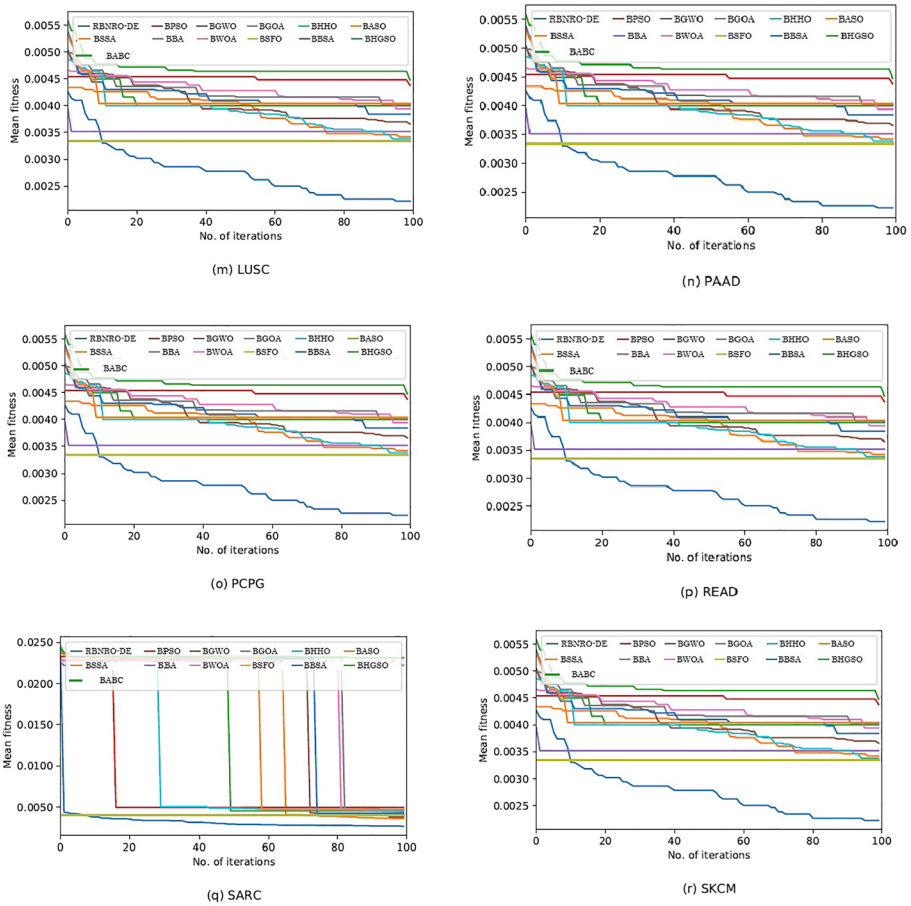

Fig. 10 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on SVM classifier over all datasets (Cont.)

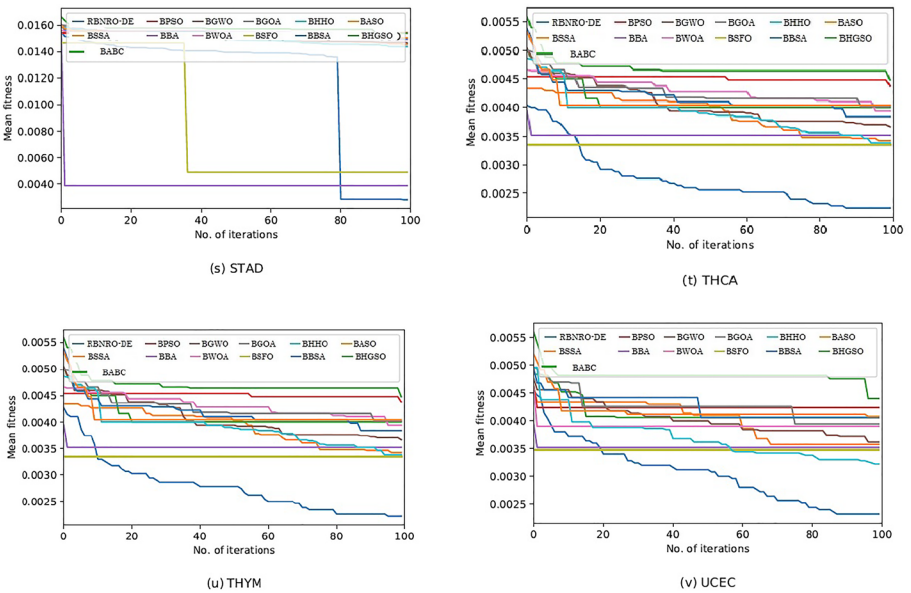

Fig. 11 Convergence performance of the proposed RBNRO-DE vs. the comparative algorithms based on SVM classifier over all datasets (Cont.)

Table 24 Results extracted by Wilcoxon's rank-sum test of the proposed RBNRO-DE vs. the comparative algorithms based on *k*-NN classifier

RBNRO-DE–*k*-NN vs.	R^+	R^-	P-value	Winner
BSSA	253.0	0.0	0	RBNRO-DE
BABC	253.0	0.0	0	RBNRO-DE
BPSO	253.0	0.0	0	RBNRO-DE
BBA	253.0	0.0	0	RBNRO-DE
BGWO	253.0	0.0	0	RBNRO-DE
BWOA	253.0	0.0	0	RBNRO-DE
BGOA	253.0	0.0	0	RBNRO-DE
BSFO	253.0	0.0	0	RBNRO-DE
BHHO	253.0	0.0	0	RBNRO-DE
BBSA	253.0	0.0	0	RBNRO-DE
BASO	253.0	0.0	0	RBNRO-DE
BHGSO	253.0	0.0	0	RBNRO-DE

Table 25 Results extracted by Wilcoxon's rank-sum test of the proposed RBNRO-DE vs. the comparative algorithms based on the SVM classifier

RBNRO-DE–*k*-NN vs.	R^+	R^-	P-value	Winner
BSSA	253.0	0.0	0	RBNRO-DE
BABC	253.0	0.0	0	RBNRO-DE
BPSO	253.0	0.0	0	RBNRO-DE
BBA	253.0	0.0	0	RBNRO-DE
BGWO	253.0	0.0	0	RBNRO-DE
BWOA	253.0	0.0	0	RBNRO-DE
BGOA	253.0	0.0	0	RBNRO-DE
BSFO	253.0	0.0	0	RBNRO-DE
BHHO	253.0	0.0	0	RBNRO-DE
BBSA	253.0	0.0	0	RBNRO-DE
BASO	253.0	0.0	0	RBNRO-DE
BHGSO	253.0	0.0	0	RBNRO-DE

Convergence analysis

Figures 4, 5, 6, 7, 8, 9, 10, 11 show convergence performance of the proposed RBNRO-DE with *k*-NN and SVM classifiers in comparison with its counterparts, which are all implemented under identical conditions of iterations number and population size. From Figs. 4, 5, 6, 7, 8, 9, 10, 11, it is obvious that the proposed RBNRO-DE with *k*-NN and SVM classifiers achieved optimal convergence performance on all datasets. Hence, the convergence behavior of the RBNRO-DE with *k*-NN and SVM classifiers proves its ability to achieve the optimum results in time while striking an effective balance between exploration and exploitation.

Wilcoxon's rank-sum test

The effectiveness of the proposed RBNRO-DE is recognized by executing the Wilcoxon test as a pair-wise test to evaluate whether there is a statistically significant deviation between the fitness values achieved via the proposed approach and its peers [72]. According to the results shown in Tables 24 and 25, it is evident that the proposed

RBNRO-DE with *K*-NN and SVM classifiers exceeds all other algorithms in all datasets. Therefore, all P-values that are listed in Tables 24 and 25 are less than 0.05 (5% significance level) that demonstrate robust evidence against the null hypothesis and can show that the achieved results by the proposed method are statistically better and not happened by chance.

Computational complexity of the RBNRO-DE and other state-of-the-art meta-heuristic algorithms

Time computational complexity of the RBNRO-DE algorithm

To define the computational complexity of the proposed RBNRO-DE algorithm, we can analyze each of its five fundamental stages individually. These stages include feature filtration, population initialization, position improvement and adjustment, fitness function estimation, and DE technique. The comprehensive computational complexity of the proposed RBNRO-DE algorithm can then be summarized in big-O notation as $O_{time}(RBNRO-DE)$, and can be calculated in big-O notation through the following equations:

$$\begin{aligned} O_{time}(RBNRO-DE) = {} & O_{time}(\text{Features filtration}) + O_{time}(\text{Population initialization}) + \\ & O_{time}(\text{Position improvement and adjustment}) + \\ & O_{time}(\text{Fitness function estimation}) + O_{time}(\text{DE technique}). \end{aligned} \tag{31}$$

Let N is the size of the population, G_{max} means the maximum generations' number, and D denotes the dimension size of problem. The following can be acquired as follows:

$O_{time}(\text{Features filtration}) = O_{time}(D)$.
$O_{time}(\text{Population initialization}) = O_{time}(N)$.
$O_{time}(\text{Position improvement and adjustment}) = O_{time}(G_{max} \times N \times D)$.
$O_{time}(\text{Fitness function estimation}) = O_{time}(G_{max} \times N)$.
$O_{time}(\text{DE technique}) = O_{time}(N \times D)$. Therefore,

$$\begin{aligned} O_{time}(RBNRO-DE) = {} & O_{time}(D) + O_{time}(N) + O_{time}(G_{max} \times N \times D) + \\ & O_{time}(G_{max} \times N) + O_{time}(N \times D) = O_{time}(G_{max} \times N \times D). \end{aligned}$$

Space computational complexity of the RBNRO-DE algorithm

The amount of memory or storage space needed for an algorithm to solve a problem as the size of the input increases is referred to as space computational complexity. It is often stated as the amount of additional memory that the algorithm requires in addition to the input. It consists of combining the following two main components:

1. Input values space: It is the memory space needed to save the input data needed for the algorithm to operate. As exhibited in Algorithm 2 that provides the pseudocode of the proposed RBNRO-DE algorithm, there are nine input variables, which are: N, G_{max}, D, P_{Fi}, P_{β}, LB, UB, C_R, and W_M. Each variable represents just numerical values, so each uses 4 bytes of memory space. Therefore, these nine input variables' total memory space complexity is 36 bytes (9×4 bytes $= 36$ bytes). The input values space complexity is of constant space.

Table 26 The proposed RBNRO-DE and its peers based on the computational complexity

Algorithm	Time computational complexity	Space computational complexity
RBNRO-DE	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BSSA	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BABC	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BPSO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BBA	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BGWO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BWOA	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BGOA	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BSFO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BHHO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BBSA	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BASO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$
BHGSO	$O_{time}(G_{max} \times N \times D)$	$O_{space}(N \times D)$

2. Auxiliary space: It indicates the additional space that the algorithm uses, apart from the input. It comprises the memory needed for the algorithm's internal variables, data structures, and other parts. A certain amount of additional memory is used by the RBNRO-DE algorithm, regardless of the input size. This involves the following variables:

   - The positions' vector $X_{initial}$, whose size is $(N \times D)$ proportional to the initial population of N positions with dimension size D, and each position takes 4 bytes of memory space, so the memory space complexity taken by $X_{initial}$ is $(4 \times N \times D)$ bytes. Its space complexity is linear since the memory requires linear increases with value $(N \times D)$.
   - The variables σ_1, σ_2, g, P_{ne}^{s}, P_{ne}^{e}, Ne_i, Pa_i, Pc_i, $fit(X_i^{Fi})$, $fit(X_i^{Ion})$, $fit(X_i^{Fu})$, $fit(u_i)$, $fit(X_i)$, σ_μ, σ_ν, $fit(X_{opt})$. Each of these 16 variables represents just numerical values, so each one takes 4 bytes of memory space. Therefore, these eleven variables' total memory space complexity is 64 bytes (16×4 bytes $= 64$ bytes). Its space complexity is of constant space.
   - The positions' vectors X_i^{Fi}, X_i^{Ion}, X_i^{Fu}, X_i, X_r, X_j, X_{r1}^{Fi}, X_{r1}^{Ion}, X_{r2}^{Fi}, X_{r2}^{Ion}, X_{best}^{Fi}, X_{best}^{Ion}, X_{worst}^{Fi}, X_{best}, $Levy(\beta)$, μ, ν, X_i^{adjust}, X_i^{bin}, u_i, υ_i, X_{r_1}, X_{r_2}, X_{r_3}, X_{opt}. The size of each of these 25 positions' vectors is D proportional to the dimension size of the obtained positions, and each position takes 4 bytes of memory space. Therefore, the total memory space complexity for these eleven positions' vectors is $(100 \times D)$ bytes ($25 \times 4 \times D$ bytes). Its space complexity is linear since the memory requires linear increases with value D.

   Consequently, the total memory space complexity for all mentioned-above auxiliary variables is: $(4 \times N \times D) + 64 + (100 \times D)$ bytes.

Finally, the total memory space computational complexity for the proposed RBNRO-DE algorithm can be calculated as follows:

Table 27 Mean classification accuracy values of the proposed RBNRO-DE with various recent algorithms based on *k*-NN

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Mean	**1.0000**	0.9891	0.9996	0.9915	0.9926
	STD	**0.0000**	0.0029	0.0021	0.0051	0.0056
CESC	Mean	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	**0.9938**	0.9912	0.9914	0.9912	0.9912
	STD	0.0041	**0.0000**	0.0016	**0.0000**	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	**0.9983**	0.9917	0.9972	0.9923	0.9928
	STD	0.0033	**0.0000**	0.0039	0.0021	0.0028
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUAD	Mean	**0.9913**	**0.9913**	**0.9913**	**0.9913**	**0.9913**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	Mean	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SKCM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	Mean	**1.0000**	0.9900	**1.0000**	0.9937	0.9956
	STD	**0.0000**	0.0033	**0.0000**	0.0055	0.0054
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	**0.9992**	0.9675	0.9942	0.9858	0.9817
	STD	**0.0045**	0.0115	0.0106	0.0124	0.0111
Ranking	W|T|L	**4|18|0**	0|17|5	0|18|4	0|17|5	0|17|5

Table 28 Mean fitness values of the proposed RBNRO-DE with various recent algorithms based on *k*-NN

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Mean	**0.0029**	0.0149	0.0048	0.0124	0.0114
	STD	**0.0002**	0.0027	0.0020	0.0048	0.0052
CESC	Mean	**0.0185**	0.0201	0.0198	0.0198	0.0198
	STD	0.0001	0.0003	0.0002	0.0002	**0.0001**
CHOL	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
COAD	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
ESCA	Mean	**0.0279**	0.0295	0.0292	0.0293	0.0292
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
GBM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0039
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
HNSC	Mean	**0.0089**	0.0131	0.0125	0.0127	0.0128
	STD	0.0038	0.0003	0.0015	**0.0002**	**0.0002**
KICH	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
KIRC	Mean	**0.0047**	0.0123	0.0070	0.0116	0.0109
	STD	0.0030	**0.0002**	0.0036	0.0018	0.0025
KIRP	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
LIHC	Mean	**0.0025**	0.0041	0.0038	0.0039	0.0038
	STD	0.0002	0.0003	**0.0001**	0.0002	0.0002
LUAD	Mean	**0.0112**	0.0127	0.0125	0.0125	0.0125
	STD	0.0002	0.0003	**0.0001**	0.0002	0.0002
LUSC	Mean	**0.0026**	0.0040	0.0039	0.0039	0.0039
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
PAAD	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	**0.0001**
PCPG	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
READ	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
SARC	Mean	**0.0212**	0.0228	0.0225	0.0225	0.0224
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
SKCM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0039
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
STAD	Mean	**0.0029**	0.0140	0.0043	0.0104	0.0086
	STD	0.0003	0.0031	**0.0002**	0.0052	0.0052
THCA	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
THYM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
UCEC	Mean	**0.0038**	0.0368	0.0103	0.0182	0.0225
	STD	**0.0044**	0.0110	0.0103	0.0120	0.0108
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22

Table 29 The number of extracted features by the proposed RBNRO-DE and various recent algorithms for training the *k*-NN

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Features number	**147**	206	223	200	204
	STD	**012.2759**	015.0320	015.4840	019.2569	021.9808
CESC	Features number	**125**	208	189	193	192
	STD	006.5072	014.0640	007.6887	008.9966	**005.9963**
CHOL	Features number	**125**	208	189	193	189
	STD	**006.5072**	014.0640	007.6887	007.9956	007.9905
COAD	Features number	**125**	208	189	193	191
	STD	**006.5072**	014.0640	007.6887	007.9956	007.6408
ESCA	Features number	**125**	208	189	194	190
	STD	**006.5072**	014.0640	007.6887	008.2476	008.2742
GBM	Features number	**125**	208	190	193	194
	STD	006.5201	013.6888	**005.8161**	007.9956	007.9244
HNSC	Features number	**139**	216	202	199	202
	STD	014.6040	015.6185	008.7405	011.5670	**008.7092**
KICH	Features number	**125**	208	189	193	191
	STD	**006.5072**	014.0640	007.6887	007.9956	008.7204
KIRC	Features number	**153**	204	212	198	192
	STD	027.5450	**011.1497**	016.6688	014.8582	016.8513
KIRP	Features number	**125**	208	189	193	192
	STD	**006.5072**	014.0640	007.6887	007.9956	009.1633
LIHC	Features number	**125**	206	192	193	191
	STD	007.9387	013.7602	**006.5599**	008.3483	011.0355
LUAD	Features number	**130**	202	195	195	193
	STD	008.1948	012.7743	**005.6474**	009.5550	010.0203
LUSC	Features number	**131**	202	196	196	195
	STD	007.3182	014.0493	**004.9912**	009.9052	008.9886
PAAD	Features number	**125**	208	189	193	191
	STD	**006.5072**	014.0640	007.6887	007.9956	007.4303
PCPG	Features number	**125**	208	189	193	192
	STD	**006.5072**	014.0640	007.6887	007.9956	009.6669
READ	Features number	**125**	208	189	193	190
	STD	**006.5072**	014.0640	007.6887	007.9956	008.8018
SARC	Features number	**125**	208	189	193	188
	STD	**006.5072**	014.0640	007.6887	008.1132	010.0555
SKCM	Features number	**125**	208	189	193	193
	STD	**006.5072**	014.0640	007.6887	007.9956	009.5313
STAD	Features number	**144**	203	217	211	210
	STD	013.0348	018.8214	**011.6924**	023.9357	022.4698
THCA	Features number	**125**	208	189	193	191
	STD	**006.8977**	013.9975	008.1367	007.9956	007.5925
THYM	Features number	**125**	208	189	193	191
	STD	**006.5072**	014.0640	007.6887	007.9956	009.0934
UCEC	Features number	**151**	232	226	211	216
	STD	016.5202	026.7279	**014.6514**	017.8582	024.8912
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22

Table 30 Mean classification accuracy values of the proposed RBNRO-DE with various recent algorithms based on SVM

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CESC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
CHOL	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
COAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
ESCA	Mean	**0.9915**	0.9752	0.9795	0.9752	0.9769
	STD	0.0121	**0.0046**	0.0103	**0.0046**	0.0077
GBM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
HNSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KICH	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
KIRP	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LIHC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUAD	Mean	**0.9922**	0.9913	0.9913	0.9913	0.9913
	STD	0.0026	**0.0000**	**0.0000**	**0.0000**	**0.0000**
LUSC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PAAD	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
PCPG	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
READ	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
SARC	Mean	**0.9975**	0.9811	0.9962	0.9843	0.9855
	STD	0.0075	**0.0000**	0.0064	0.0070	0.0080
SKCM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
STAD	Mean	**0.9900**	0.9889	0.9889	0.9889	0.9889
	STD	0.0033	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THCA	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
THYM	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
UCEC	Mean	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Ranking	W|T|L	**4|18|0**	0|18|4	0|18|4	0|18|4	0|18|4

$$\text{Space complexity(RBNRO-DE)} = \text{Input values space} + \text{Auxiliary space} = 36 + \big((4 \times N \times D) + 64 + (100 \times D)\big) \text{ bytes.}$$

Note that there are constant bytes, which will not be considered. For that, the total RBNRO-DE space computational complexity can be expressed in big-O notation as $O_{space}(RBNRO - DE)$, and can be computed in big-O notation after removing all constants as follows:

$$O_{space}(RBNRO - DE) = O_{space}(\text{Input values space}) + O_{space}(\text{Auxiliary space}) = O_{space}(1) + \big(O_{space}(N \times D) + O_{space}(1) + O_{space}(D)\big) = O_{space}(N \times D).$$

Comparison results between the RBNRO-DE and other state-of-the-art meta-heuristic algorithms based on the computational complexity

Creating a comprehensive comparison of the time complexity and space complexity of multiple meta-heuristic optimization algorithms can be challenging because these complexities can vary depending on the specific implementation, problem size, and other factors. Additionally, detailed time and space complexity analyses may not be available for all of the mentioned algorithms, and they may have different characteristics when applied to different problems. However, we try to provide a simplified comparison of these algorithms in terms of their general characteristics with respect to time and space complexity, as illustrated in Table 26.

Comparison results of the proposed RBNRO-DE versus various recent algorithms from the published literature

As previously clarified, no meta-heuristic algorithm has ever been applied to RNA-Seq gene expression data. Therefore, the RBNRO-DE algorithm is considered the first meta-heuristic algorithm to be proposed for solving GS problems of RNA-Seq gene expression data. This subsection presents the empirical results of comparisons based on the average classification accuracy values, fitness values, and selected features values in tackling the GS issue between the proposed RBNRO-DE and other recent meta-heuristic optimization techniques from the published literature, including Binary meerkat optimization algorithm (BMOA) [73], Binary Brown-bear Optimization (BBBO) algorithm [74], Binary Aquila Optimization (BAO) algorithm [75], and Binary African Vultures Optimization (BAVO) algorithm [76].

Comparisons based on the suggested k-NN classifier

The accuracy values of the proposed RBNRO-DE optimizer and other recent optimizers based on the *k*-NN classifier were compared in Table 27 under the same implementation conditions. Based on the empirical results, RBNRO-DE yielded the best results in four datasets. In the remaining 18 datasets, RBNRO-DE with *k*-NN and other competitive recent algorithms produced identical results.

Table 28 compares the performance of the proposed RBNRO-DE algorithm with other algorithms based on *k*-NN using the same implementation requirements. The results indicate that the proposed algorithm outperforms its competitors in producing

Table 31 Mean fitness values of the proposed RBNRO-DE with various recent algorithms based on SVM

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0039
	STD	**0.0001**	0.0003	0.0002	0.0002	**0.0001**
CESC	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0039
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
CHOL	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
COAD	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	**0.0001**
ESCA	Mean	**0.0113**	0.0287	0.0242	0.0284	0.0267
	STD	0.0118	**0.0046**	0.0099	**0.0046**	0.0074
GBM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
HNSC	Mean	**0.0025**	0.0041	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
KICH	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
KIRC	Mean	**0.0025**	0.0041	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
KIRP	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
LIHC	Mean	**0.0026**	0.0046	0.0040	0.0040	0.0040
	STD	**0.0001**	0.0003	**0.0001**	0.0002	0.0002
LUAD	Mean	**0.0103**	0.0128	0.0124	0.0125	0.0124
	STD	0.0025	0.0003	**0.0001**	0.0002	0.0002
LUSC	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
PAAD	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
PCPG	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
READ	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
SARC	Mean	**0.0067**	0.0228	0.0068	0.0196	0.0183
	STD	0.0073	**0.0003**	0.0062	0.0066	0.0077
SKCM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
STAD	Mean	**0.0125**	0.0152	0.0148	0.0149	0.0147
	STD	0.0031	0.0003	**0.0001**	0.0002	0.0002
THCA	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
THYM	Mean	**0.0025**	0.0042	0.0038	0.0039	0.0038
	STD	**0.0001**	0.0003	0.0002	0.0002	0.0002
UCEC	Mean	**0.0025**	0.0042	0.0039	0.0039	0.0038
	STD	**0.0001**	0.0002	**0.0001**	0.0002	0.0002
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22

Table 32 The number of extracted features by the proposed RBNRO-DE and various recent algorithms for training the SVM

Benchmark	Metric	RBNRO-DE	BMOA	BBBO	BAO	BAVO
BLCA	Features number	**125**	208	189	193	193
	STD	**006.5286**	014.0640	007.9983	008.0170	006.6821
CESC	Features number	**125**	208	189	193	193
	STD	**006.5072**	014.0640	007.6887	007.9956	008.0890
CHOL	Features number	**125**	208	189	193	192
	STD	**006.5072**	014.0640	007.6887	007.9956	010.8417
COAD	Features number	**125**	208	189	193	192
	STD	**006.5072**	014.0640	007.6887	007.9956	007.4521
ESCA	Features number	**140**	208	196	194	193
	STD	025.9610	013.8923	016.6035	**008.1260**	017.7150
GBM	Features number	**124**	208	188	193	192
	STD	**006.9048**	013.6888	007.3182	007.9822	007.9989
HNSC	Features number	**125**	206	190	194	192
	STD	**005.3109**	015.9243	006.5255	008.1131	008.2482
KICH	Features number	**125**	208	189	193	189
	STD	**006.5072**	014.0640	007.6887	007.9956	008.1326
KIRC	Features number	**127**	205	192	193	192
	STD	006.4339	014.0017	**006.0144**	008.7689	007.8977
KIRP	Features number	**125**	208	189	193	189
	STD	**006.5072**	014.0640	007.6887	007.9956	008.6346
LIHC	Features number	**130**	231	201	201	202
	STD	007.2622	013.0229	**006.2286**	011.3090	011.3774
LUAD	Features number	**127**	208	189	193	189
	STD	008.4892	014.0640	**007.3964**	009.0367	008.2643
LUSC	Features number	**125**	208	189	193	192
	STD	**006.5072**	014.0640	007.6887	007.9956	009.5594
PAAD	Features number	**125**	208	189	193	190
	STD	**006.5072**	014.0640	007.6887	007.9956	009.7116
PCPG	Features number	**125**	208	189	193	189
	STD	**006.5072**	014.0640	007.6887	007.9956	009.0618
READ	Features number	**125**	208	189	193	190
	STD	**006.5072**	014.0640	007.6887	007.9956	007.7618
SARC	Features number	**146**	208	216	201	197
	STD	015.8213	**014.289**2	016.4592	021.0692	017.7620
SKCM	Features number	**125**	208	189	193	190
	STD	**006.5072**	014.0640	007.6887	007.9956	009.0873
STAD	Features number	**129**	208	190	193	187
	STD	010.9780	013.4048	**006.9580**	009.5803	009.4139
THCA	Features number	**126**	208	189	193	190
	STD	**006.0646**	014.0640	007.6887	007.9956	008.5437
THYM	Features number	**125**	208	189	193	190
	STD	**006.5072**	014.0640	007.6887	007.9956	009.1520
UCEC	Features number	**127**	209	194	193	188
	STD	**005.9258**	011.5318	007.4748	007.7749	008.0639
Ranking	W|T|L	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22

Table 33 Classification accuracy values of the proposed RBNRO-DE with *k*-NN and different filter and embedded methods

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**1.0000**	0.9884	0.9884	0.9767	0.9535	0.9884	0.9884	0.9884	0.9884
CESC	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**	**0.9839**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	0.9848	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9697
ESCA	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**	**0.9744**
GBM	**1.0000**	**1.0000**	0.9688	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**0.9938**	0.9823	0.9912	0.9735	0.9823	0.9823	0.9902	0.9912	0.9823
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**0.9983**	0.9917	0.9917	0.9923	0.9928	0.9917	0.9917	0.9917	0.9917
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	**1.0000**	**1.0000**	0.9765	0.9882	**1.0000**	**1.0000**	**1.0000**	0.9882
LUAD	**0.9913**	**0.9913**	**0.9913**	0.9652	0.9565	**0.9913**	**0.9913**	**0.9913**	0.9652
LUSC	**1.0000**	**1.0000**	**1.0000**	0.9910	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9910
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**	**0.9811**
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**1.0000**	0.9889	0.9889	0.9778	**1.0000**	0.9889	0.9778	**1.0000**	0.9889
THCA	**1.0000**	**1.0000**	**1.0000**	0.9821	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9911
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**0.9992**	0.9500	0.9500	0.9500	0.9750	0.9750	0.9750	0.9750	0.9750
Ranking (W|T|L)	**4|18|0**	0|17|5	0|15|7	0|13|9	0|16|6	0|17|5	0|17|5	0|18|4	0|12|10

higher-quality fitness values with lower standard deviation across all datasets used in the experiments. It is worth noting that all these datasets are large-scale, which demonstrates the proposed algorithm's ability to perform consistently regardless of the dataset size. Additionally, the RBNRO-DE algorithm has shown remarkable performance in balancing exploitation and exploration to avoid getting trapped in local optima. Overall, these results suggest that the proposed RBNRO-DE algorithm is promising and has the potential to evolve beyond the other recent algorithms.

Table 29 shows the number of extracted features using the suggested RBNRO-DE and other recent optimization algorithms for training the *k*-NN classifier. The proposed RBNRO-DE exceeded the other recent algorithms in all datasets regarding the number of the selected features. Also, the RBNRO-DE's ability to determine the most instructive features is attributable to the capability to explore the feasible regions while maintaining enhanced classification accuracy.

Comparisons based on the suggested SVM classifier

The mean accuracy results of the suggested RBNRO-DE optimizer and other recent optimization methods regarding the SVM classifier were shown in Table 30 under identical implementation conditions. The proposed RBNRO-DE produced the most promising results in four datasets based on the results. In the remaining 18 datasets,

Table 34 Fitness values of the proposed RBNRO-DE with *k*-NN and different filter and embedded methods

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**0.0029**	0.0215	0.0215	0.0330	0.0560	0.0215	0.0215	0.0215	0.0215
CESC	**0.0185**	0.0260	0.0260	0.0260	0.0260	0.0260	0.0260	0.0260	0.0260
CHOL	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
COAD	**0.0025**	0.0100	0.0100	0.0250	0.0100	0.0100	0.0100	0.0100	0.0400
ESCA	**0.0279**	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354
GBM	**0.0025**	0.0100	0.0100	0.0409	0.0100	0.0100	0.0100	0.0100	0.0100
HNSC	**0.0089**	0.0275	0.0188	0.0363	0.0275	0.0275	0.0100	0.0188	0.0275
KICH	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
KIRC	**0.0047**	0.0182	0.0182	0.0100	0.0100	0.0182	0.0182	0.0182	0.0182
KIRP	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
LIHC	**0.0025**	0.0100	0.0100	0.0333	0.0216	0.0100	0.0100	0.0100	0.0216
LUAD	**0.0112**	0.0186	0.0186	0.0444	0.0530	0.0186	0.0186	0.0186	0.0444
LUSC	**0.0026**	0.0100	0.0100	0.0189	0.0100	0.0100	0.0100	0.0100	0.0189
PAAD	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
PCPG	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
READ	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
SARC	**0.0212**	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287
SKCM	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
STAD	**0.0029**	0.0210	0.0210	0.0320	0.0100	0.0210	0.0320	0.0100	0.0210
THCA	**0.0025**	0.0100	0.0100	0.0277	0.0100	0.0100	0.0100	0.0100	0.0188
THYM	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
UCEC	**0.0038**	0.0595	0.0595	0.0595	0.0348	0.0348	0.0100	0.0100	0.0348
Ranking (W|T|L)	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

the proposed RBNRO-DE with SVM and other recent competitive algorithms yielded equivalent results.

Table 31 shows the fitness values of the suggested RBNRO-DE and other recent optimization algorithms regarding the SVM classifier. The outcomes show that the proposed technique exceeds its peers by producing the smallest fitness values with lower standard deviation across all benchmarks employed in the experimentations. It is worth noting that all these datasets are large-scale, demonstrating the suggested algorithm's capability to perform consistently regardless of the size of the dataset. Also, the proposed RBNRO-DE has shown promising performance in balancing exploitation and exploration to avoid getting trapped in local optima.

Table 32 displays the number of extracted features chosen by the suggested RBNRO-DE and other recent optimization algorithms for training the SVM classifier. The proposed RBNRO-DE exceeded the other recent algorithms in all datasets regarding the number of selected features. Also, the RBNRO-DE's ability to determine the most instructive features is attributable to the capability to explore the feasible regions while maintaining enhanced classification accuracy.

Table 35 The number of extracted features by the proposed RBNRO-DE with different filter and embedded methods for training the *k*-NN

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**147**	500	412	300	300	206	200	310	500
CESC	**125**	500	439	300	300	209	203	220	500
CHOL	**125**	500	152	290	280	204	158	180	200
COAD	**125**	500	287	310	190	202	201	270	200
ESCA	**125**	500	339	300	300	206	234	250	200
GBM	**125**	500	254	209	190	199	255	190	500
HNSC	**139**	500	382	310	300	202	180	420	500
KICH	125	500	**106**	250	160	196	231	190	160
KIRC	**153**	500	470	300	300	206	199	390	200
KIRP	**125**	500	339	290	280	202	199	170	200
LIHC	**125**	500	410	130	200	218	192	420	200
LUAD	**130**	500	423	240	300	196	231	430	200
LUSC	**131**	500	293	300	300	218	194	350	300
PAAD	**125**	500	360	300	300	205	221	250	500
PCPG	**125**	500	327	500	500	199	199	200	500
READ	**125**	500	324	500	500	203	235	190	300
SARC	**125**	500	431	500	500	200	200	200	500
SKCM	**125**	500	242	500	500	181	192	180	500
STAD	**144**	500	428	500	500	217	216	340	200
THCA	**125**	500	311	500	300	199	194	152	300
THYM	**125**	500	369	300	300	203	205	210	500
UCEC	**151**	500	422	500	500	219	206	280	300
Ranking (W|T|L)	**21|0|1**	0|0|22	1|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Comparison results of the proposed RBNRO-DE versus different filter and embedded methods

This subsection presents the experimental results of the proposed RBNRO-DE and various filter and embedded methods.

Comparisons based on the suggested k-NN classifier

Table 33 shows the results of the proposed RBNRO-DE compared with other filter and embedded methods based on the *k*-NN classifier regarding the classification accuracy values that are fairly evaluated under the same implementation conditions. The empirical results show that the proposed RBNRO-DE is ranked first by achieving the best results in 4 out of 22 datasets. Lasso regularization is ranked second by yielding identical results in 18 datasets as the proposed RBNRO-DE. It should also be noted that variance threshold, linear regression, and ridge regularization methods ranked third by yielding identical results in 17 datasets as the proposed RBNRO-DE.

The average fitness values of the proposed RBNRO-DE algorithm based on *k*-NN and various filter and embedded methods are shown in Table 34. From the results presented in Table 34, it can be observed that RBNRO-DE with *k*-NN produces the least fitness values for all datasets. Additionally, it is noteworthy that the proposed RBNRO-DE can perform consistently on all datasets, irrespective of their size, as all the datasets used in this study are large-scale.

Table 36 Classification accuracy values of the proposed RBNRO-DE with SVM and different filter and embedded methods

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**1.0000**	**1.0000**	**1.0000**	0.9884	0.9767	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CESC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
CHOL	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
COAD	**1.0000**	**1.0000**	0.9848	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9697
ESCA	**0.9915**	0.9744	0.9744	0.9744	0.9744	0.9744	0.9744	0.9744	0.9744
GBM	**1.0000**	**1.0000**	0.9688	0.9688	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
HNSC	**1.0000**	**1.0000**	**1.0000**	0.9735	0.9912	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KICH	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
KIRP	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LIHC	**1.0000**	0.9765	0.9765	0.9765	0.9765	0.9765	0.9765	0.9765	0.9882
LUAD	**0.9922**	0.9913	0.9913	0.9565	0.9652	0.9913	0.9913	0.9913	0.9652
LUSC	**1.0000**	**1.0000**	**1.0000**	0.9910	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9910
PAAD	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PCPG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
READ	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
SARC	**0.9962**	0.9811	0.9811	0.9811	0.9811	0.9811	0.9811	0.9811	0.9811
SKCM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
STAD	**0.9900**	0.9889	0.9889	0.9778	0.9889	0.9889	0.9889	0.9889	0.9889
THCA	**1.0000**	**1.0000**	**1.0000**	0.9821	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9911
THYM	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
UCEC	**1.0000**	**1.0000**	**1.0000**	0.9500	0.9500	**1.0000**	**1.0000**	**1.0000**	0.9500
Ranking (W|T|L)	**5|17|0**	0|17|5	0|15|7	0|11|11	0|14|8	0|17|5	0|17|5	0|17|5	0|13|9

Table 35 shows the number of selected features using the proposed RBNRO-DE with the *k*-NN classifier and different filter and embedded methods. The proposed RBNRO-DE surpassed the other algorithms in 21 out of 22 datasets regarding the number of extracted features. Correlation ranked second by achieving the best results in one dataset.

Comparisons based on the suggested SVM classifier

Table 36 shows the results of the proposed RBNRO-DE compared with other filter and embedded methods based on the SVM classifier regarding the classification accuracy values that are fairly evaluated under the same implementation conditions. The empirical results show that the proposed RBNRO-DE is ranked first by achieving the best results in 4 out of 22 datasets. Lasso regularization is ranked second by yielding identical results in 18 datasets as the proposed RBNRO-DE. It should also be noted that variance threshold, linear regression, and ridge regularization methods ranked third by yielding identical results in 17 datasets as the proposed RBNRO-DE.

The average fitness values of the proposed RBNRO-DE algorithm based on SVM and various filter and embedded methods are shown in Table 37. From the results presented in Table 37, it can be observed that RBNRO-DE with SVM produces the least fitness values for all datasets.

Table 37 Fitness values of the proposed RBNRO-DE with *k*-NN and different filter and embedded methods

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**0.0025**	0.0100	0.0100	0.0215	0.0330	0.0100	0.0100	0.0100	0.0100
CESC	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
CHOL	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
COAD	**0.0025**	0.0100	0.0100	0.0250	0.0100	0.0100	0.0100	0.0100	0.0400
ESCA	**0.0113**	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354	0.0354
GBM	**0.0025**	0.0100	0.0100	0.0409	0.0409	0.0100	0.0100	0.0100	0.0100
HNSC	**0.0025**	0.0100	0.0100	0.0363	0.0188	0.0100	0.0100	0.0100	0.0100
KICH	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
KIRC	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
KIRP	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
LIHC	**0.0026**	0.0333	0.0333	0.0333	0.0333	0.0333	0.0333	0.0333	0.0216
LUAD	**0.0103**	0.0186	0.0186	0.0530	0.0444	0.0186	0.0186	0.0186	0.0444
LUSC	**0.0025**	0.0100	0.0100	0.0189	0.0100	0.0100	0.0100	0.0100	0.0189
PAAD	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
PCPG	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
READ	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
SARC	**0.0067**	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287	0.0287
SKCM	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
STAD	**0.0029**	0.0210	0.0210	0.0320	0.0210	0.0210	0.0210	0.0210	0.0210
THCA	**0.0025**	0.0100	0.0100	0.0277	0.0100	0.0100	0.0100	0.0100	0.0188
THYM	**0.0025**	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
UCEC	**0.0025**	0.0100	0.0100	0.0595	0.0595	0.0100	0.0100	0.0100	0.0595
Ranking (W|T|L)	**22|0|0**	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

Table 38 shows the number of selected features using the proposed RBNRO-DE with the SVM classifier and different filter and embedded methods. The proposed RBNRO-DE surpassed the other algorithms in 21 out of 22 datasets regarding the number of selected features. Correlation ranked second by achieving the best results in one dataset.

Discussion

Based on the empirical analysis, it can be demonstrated that the proposed RBNRO-DE with *k*-NN and SVM classifiers yielded more reliable results than other recent algorithms for handling the GS strategy on (rnaseqv2 illuminahiseq rnaseqv2 un edu Level 3 RSEM genes normalized) with more than 20,000 genes to pick the best informative genes and assessed them through 22 cancer datasets. Binary versions of the most common meta-heuristic algorithms have been compared with the proposed RBNRO-DE algorithm. In most of the 22 cancer datasets, the RBNRO-DE algorithm based on *k*-NN and SVM classifiers achieved optimal convergence and classification accuracy up to 100% integrated with a feature reduction size down to 98%, which is very evident when compared to its counterparts, according to Wilcoxon's rank-sum test (5% significance

Table 38 The number of extracted features by the proposed RBNRO-DE with different filter and embedded methods for training the SVM

Benchmark	RBNRO-DE	Variance Threshold	Correlation	Chi-square	ANOVA	Linear-Regression	Ridge Regularization	Lasso Regularization	Elastic Net
BLCA	**125**	500	412	300	300	206	200	310	500
CESC	**125**	500	439	300	300	209	203	220	500
CHOL	**125**	500	152	290	280	204	158	180	200
COAD	**125**	500	287	310	190	202	201	270	200
ESCA	**140**	500	339	300	300	206	234	250	200
GBM	**124**	500	254	209	190	199	255	190	500
HNSC	**125**	500	382	310	300	202	180	420	500
KICH	125	500	**106**	250	160	196	231	190	160
KIRC	**127**	500	470	300	300	206	199	390	200
KIRP	**125**	500	339	290	280	202	199	170	200
LIHC	**130**	500	410	130	200	218	192	420	200
LUAD	**127**	500	423	240	300	196	231	430	200
LUSC	**125**	500	293	300	300	218	194	350	300
PAAD	**125**	500	360	300	300	205	221	250	500
PCPG	**125**	500	327	500	500	199	199	200	500
READ	**125**	500	324	500	500	203	235	190	300
SARC	**146**	500	431	500	500	200	200	200	500
SKCM	**125**	500	242	500	500	181	192	180	500
STAD	**129**	500	428	500	500	217	216	340	200
THCA	**126**	500	311	500	300	199	194	152	300
THYM	**125**	500	369	300	300	203	205	210	500
UCEC	**127**	500	422	500	500	219	206	280	300
Ranking (W|T|L)	**21|0|1**	0|0|22	1|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22	0|0|22

level). Moreover, the RBNRO-DE optimizer showed a more significant exploration and exploitation behaviour than its peers, verified by subsequent underlying causes.

Firstly, a pre-processing phase uses the Relief algorithm to identify the relevant features by computing a weight for every feature to describe its relationship and then ignoring the irrelevant features with the lowest weights. The second phase includes applying the binary NRO algorithm combined with the DE technique to determine the most relevant and non-redundant features. When solving large-scale problems, the NRO algorithm is susceptible to the local optimal trap. To prevent this, the DE technique is included in the NRO algorithm.

Moreover, the suggested RBNRO-DE based on the *k*-NN and SVM classifiers emphasizes its behaviour to obtain the optimal solution on time, ensuring an effective equilibrium between exploration and exploitation capabilities. Eventually, due to the non-exact repeatability of the optimization outcomes, separate optimizer implementations can generate various subsets of attributes, which may confuse the user. Therefore, on different occasions or applications, RBNRO-DE or other optimizers implemented here can select multiple subsets of features.

Conclusion and future work

In this study, we applied the meta-heuristic RBNRO-DE algorithm for solving FS problems of RNA-Seq gene expression data for the first time and identifying possible biomarkers for various tumour types to improve the best solution. Results were satisfactory, demonstrating the algorithm's capabilities and effectiveness were significantly increased. *k*-NN and SVM, two well-known classifiers, were used to assess the usefulness of each subset of the chosen features. The performance of the proposed RBNRO-DE algorithm was compared to binary versions of 12 well-known meta-heuristic algorithms to validate it on various tumour types with multiple samples. The evaluation was conducted using a variety of metrics, such as the AVG_{Fit}, AVG_{Acc}, and AVG_{Feat} values. The suggested algorithm in this research, RBNRO-DE based on *k*-NN and SVM classifiers, performed better than the other algorithms for dealing with FS problems results. Future research could examine how the RBNRO-DE algorithm integrates with various optimization algorithms. To further explore the effectiveness of the RBNRO-DE algorithm for FS in supervised classification, other classifiers (such as DTs, artificial neural networks, etc.) could be used.

Acknowledgements
Not applicable.

Author contributions
Amr A. Abd El-Mageed: Conceptualization, methodology, software, formal analysis, investigation, data curation, validation, writing—original draft, writing—review and editing. Ahmed E. Elkhouli: investigation, visualization, data curation, validation, writing—original draft, writing—review and editing. Amr A. Abohany: resources, formal analysis, data curation, validation, writing—original draft, writing—review and editing. Mona Gafar: resources, validation, writing—review and editing. All authors read and approved the final manuscript.

Funding
Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

Availability of data and materials
For transparency and reproducibility, the developed software and the relevant Python code of this paper are publicly available and obtainable in [77].

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Received: 24 June 2023 Accepted: 14 March 2024
Published online: 03 April 2024

References

1. Wang Z, Gerstein M, Snyder M. Rna-seq: a revolutionary tool for transcriptomics. Nat Rev Genet. 2009;10(1):57–63.
2. Metzker ML. Sequencing technologies-the next generation. Nat Rev Genet. 2010;11(1):31–46.
3. Lyu B, Haque A. Deep learning based tumor type classification using gene expression data. In: Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics; 2018. p. 89–96.
4. Kim Y-W, Oh I-S. Classifier ensemble selection using hybrid genetic algorithms. Pattern Recogn Lett. 2008;29(6):796–802.
5. Li Y, Wang G, Chen H, Shi L, Qin L. An ant colony optimization based dimension reduction method for high-dimensional datasets. J Bionic Eng. 2013;10(2):231–41.

6. Tabakhi S, Moradi P, Akhlaghian F. An unsupervised feature selection algorithm based on ant colony optimization. Eng Appl Artif Intell. 2014;32:112–23.
7. Jafari P, Azuaje F. An assessment of recently published gene expression data analyses: reporting experimental design and statistical factors. BMC Med Inf Decis Making. 2006;6(1):1–8.
8. Gu Q, Li Z, Han J. Generalized fisher score for feature selection; 2012 arXiv preprint arXiv:1202.3725.
9. Mishra D, Sahu B. Feature selection for cancer classification: a signal-to-noise ratio approach. Int J Sci Eng Res. 2011;2(4):1–7.
10. Vergara JR, Estévez PA. A review of feature selection methods based on mutual information. Neural Comput Appl. 2014;24(1):175–86.
11. Shreem SS, Abdullah S, Nazri MZA, Alzaqebah M. Hybridizing relief, MRMR filters and ga wrapper approaches for gene selection. J Theor Appl Inf Technol. 2012;46(2):1034–9.
12. Abdel-Basset M, Sallam KM, Mohamed R, Elgendi I, Munasinghe K, Elkomy OM. An improved binary grey-wolf optimizer with simulated annealing for feature selection. IEEE Access. 2021;9:139792–822.
13. Tang J, Duan H, Lao S. Swarm intelligence algorithms for multiple unmanned aerial vehicles collaboration: a comprehensive review. Artif Intell Rev. 2023;56(5):4295–327. https://doi.org/10.1007/s10462-022-10281-7.
14. Wang D, Tan D, Liu L. Particle swarm optimization algorithm: an overview. Soft Comput. 2018;22:387–408.
15. Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Opt. 2007;39(3):459–71. https://doi.org/10.1007/s10898-007-9149-x.
16. Xue J, Shen B. A novel swarm intelligence optimization approach: sparrow search algorithm. Syst Sci Control Eng. 2020;8(1):22–34.
17. Emary E, Zawbaa HM, Hassanien AE. Binary grey wolf optimization approaches for feature selection. Neurocomputing. 2016;172:371–81.
18. Yang X-S. A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010). Springer; 2010. p. 65–74.
19. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95:51–67.
20. Hichem H, Elkamel M, Rafik M, Mesaaoud MT, Ouahiba C. A new binary grasshopper optimization algorithm for feature selection problem. J King Saud Univ Comput Inf Sci. 2022;34(2):316–28.
21. Shadravan S, Naji HR, Bardsiri VK. The sailfish optimizer: a novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng Appl ArtifIntell. 2019;80:20–34.
22. Meng X-B, Gao XZ, Lu L, Liu Y, Zhang H. A new bio-inspired optimisation algorithm: Bird swarm algorithm. J Exp Theor Artif Intell. 2016;28(4):673–87.
23. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H. Harris hawks optimization: algorithm and applications. Future Generat Comput Syst. 2019;97:849–72.
24. Holland JH. Genetic algorithms. Sci Am. 1992;267(1):66–73.
25. Storn R, Price K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Opt. 1997;11(4):341–59.
26. Khalid AM, Hosny KM, Mirjalili S. COVIDOA: a novel evolutionary optimization algorithm based on coronavirus disease replication lifecycle. Neural Comput Appl. 2022. https://doi.org/10.1007/s00521-022-07639-x.
27. Tang D, Dong S, Jiang Y, Li H, Huang Y. Itgo: invasive tumor growth optimization algorithm. Appl Soft Comput. 2015;36:670–98.
28. Simon D. Biogeography-based optimization. IEEE Trans Evol Comput. 2008;12(6):702–13.
29. Van Laarhoven PJ, Aarts EH. Simulated annealing. In: Simulated annealing: theory and applications. Springer; 1987. p. 7–15.
30. Rashedi E, Nezamabadi-Pour H, Saryazdi S. Gsa: a gravitational search algorithm. Inf Sci. 2009;179(13):2232–48.
31. Zhao W, Wang L, Zhang Z. Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowl Based Syst. 2019;163:283–304.
32. Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S. Henry gas solubility optimization: a novel physics-based algorithm. Future Generat Comput Syst. 2019;101:646–67.
33. Ma S, Huang J. Penalized feature selection and classification in bioinformatics. Brief Bioinform. 2008;9(5):392–403.
34. Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc Ser B Stat Methodol. 2005;67(2):301–20.
35. Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Mach Learn. 2002;46(1):389–422.
36. Díaz-Uriarte R, De Andres SA. Gene selection and classification of microarray data using random forest. BMC Bioinform. 2006;7(1):1–13.
37. Oh I-S, Lee J-S, Moon B-R. Hybrid genetic algorithms for feature selection. IEEE Trans Pattern Anal Mach Intell. 2004;26(11):1424–37.
38. Cadenas JM, Garrido MC, MartíNez R. Feature subset selection filter-wrapper based on low quality data. Expert Syst Appl. 2013;40(16):6241–52.
39. Sarafrazi S, Nezamabadi-Pour H. Facing the classification of binary problems with a GSA-SVM hybrid system. Math Comput Model. 2013;57(1–2):270–8.
40. Wei Z, Huang C, Wang X, Han T, Li Y. Nuclear reaction optimization: a novel and powerful physics-based algorithm for global optimization. IEEE Access. 2019;7:66084–109. https://doi.org/10.1109/ACCESS.2019.2918406.
41. Li Y, Kang K, Krahn JM, Croutwater N, Lee K, Umbach DM, Li L. A comprehensive genomic pan-cancer classification using the cancer genome atlas gene expression data. BMC Genomics. 2017;18(1):1–13.
42. Khalifa NEM, Taha MHN, Ali DE, Slowik A, Hassanien AE. Artificial intelligence technique for gene expression by tumor RNA-seq data: a novel optimized deep learning approach. IEEE Access. 2020;8:22874–83.
43. Dillies M-A, Rau A, Aubert J, Hennequet-Antier C, Jeanmougin M, Servant N, Keime C, Marot G, Castel D, Estelle J, et al. A comprehensive evaluation of normalization methods for illumina high-throughput RNA sequencing data analysis. Brief Bioinform. 2013;14(6):671–83.

44. Xiao Y, Wu J, Lin Z, Zhao X. A deep learning-based multi-model ensemble method for cancer prediction. Comput Methods Progr Biomed. 2018;153:1–9.
45. Liu M, Xu L, Yi J, Huang J. A feature gene selection method based on relieff and PSO. In: 2018 10th international conference on measuring technology and mechatronics automation (ICMTMA), IEEE; 2018. p. 298–301.
46. Danaee P, Ghaeini R, Hendrix DA. A deep learning approach for cancer detection and relevant gene identification. In: Pacific symposium on biocomputing. World Scientific; 2017. p. 219–29.
47. Kira K, Rendell LA, et al. The feature selection problem: traditional methods and a new algorithm. In: AAAI; 1992. p. 129–34:2.
48. Kononenko I. Estimating attributes: analysis and extensions of relief. In: European conference on machine learning. Springer; 1994. p. 171–82.
49. Fergusson JE. The history of the discovery of nuclear fission. Found Chem. 2011;13(2):145–66.
50. Wei Z, Huang C, Wang X, Han T, Li Y. Nuclear reaction optimization: a novel and powerful physics-based algorithm for global optimization. IEEE Access. 2019;7:66084–109.
51. Salimi H. Stochastic fractal search: a powerful metaheuristic algorithm. Knowl Based Syst. 2015;75:1–18.
52. Zhuoran Z, Changqiang H, Hanqiao H, Shangqin T, Kangsheng D. An optimization method: hummingbirds optimization algorithm. J Syst Eng Electr. 2018;29(2):386–404.
53. Alpaydin E. Introduction to machine learning. MIT press; 2020.
54. Cunningham P, Delany SJ. k-nearest neighbour classifiers-a tutorial. ACM Comput Surv (CSUR). 2021;54(6):1–25.
55. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory. 1967;13(1):21–7.
56. Thaher T, Heidari AA, Mafarja M, Dong JS, Mirjalili S. Binary harris hawks optimizer for high-dimensional, low sample size feature selection. In: Evolutionary machine learning techniques. Springer; 2020. p. 251–72.
57. Mafarja M, Mirjalili S. Whale optimization approaches for wrapper feature selection. Appl Soft Comput. 2018;62:441–53.
58. Tharwat A, Hassanien AE, Elnaghi BE. A ba-based algorithm for parameter optimization of support vector machine. Pattern Recogn Lett. 2017;93:13–22.
59. Schölkopf B, Smola AJ, Bach F, et al. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press; 2002.
60. Gupta R, Alam MA, Agarwal P. Modified support vector machine for detecting stress level using EEG signals. Comput Intell Neurosci. 2020;2020:1–14.
61. Li S. Global face pose detection based on an improved PSO-SVM method. In: Proceedings of the 2020 international conference on aviation safety and information technology; 2020. p. 549–53.
62. Mastromichalakis S, Chountasis S. An MR image classification scheme based on fourier moment analysis and linear support vector machine. J Inf Opt Sci. 2020;42:1–19.
63. Gopi AP, Jyothi RNS, Narayana VL, Sandeep KS. Classification of tweets data based on polarity using improved RBF kernel of SVM. Int J Inf Technol. 2020;15:1–16.
64. Faris H, Mafarja MM, Heidari AA, Aljarah I, Ala'M A-Z, Mirjalili S, Fujita H. An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowl Based Syst. 2018;154:43–67.
65. Abdel-Basset M, Ding W, El-Shahat D. A hybrid harris hawks optimization algorithm with simulated annealing for feature selection. Artif Intell Rev. 2020;54:1–45.
66. Sallam KM, Elsayed SM, Sarker RA, Essam DL, Improved united multi-operator algorithm for solving optimization problems. In: IEEE congress on evolutionary computation (CEC). IEEE. 2018;2018. p. 1–8.
67. Normalized-level3 RNA-seq gene expression dataset. https://gdac.broadinstitute.org/.
68. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM. Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw. 2017;114:163–91. https://doi.org/10.1016/j.advengsoft.2017.07.002.
69. Mirjalili S, Mirjalili SM, Yang X-S. Binary bat algorithm. Neural Comput Appl. 2014;25:663–81.
70. Mirjalili S, Lewis A. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm Evol Comput. 2013;9:1–14.
71. Ghosh KK, Ahmed S, Singh PK, Geem ZW, Sarkar R. Improved binary sailfish optimizer based on adaptive β-hill climbing for feature selection. IEEE Access. 2020;8:83548–60.
72. Derrac J, García S, Molina D, Herrera F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput. 2011;1(1):3–18.
73. Xian S, Feng X. Meerkat optimization algorithm: a new meta-heuristic optimization algorithm for solving constrained engineering problems. Expert Syst Appl. 2023;231: 120482. https://doi.org/10.1016/j.eswa.2023.120482.
74. Prakash T, Singh PP, Singh VP, Singh SN. A novel brown-bear optimization algorithm for solving economic dispatch problem. In: Advanced control & optimization paradigms for energy system operation and management. River Publishers; 2023. p. 137–64.
75. Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-Qaness MA, Gandomi AH. Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng. 2021;107250:157.
76. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S. African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems. Comput Ind Eng. 2021;158: 107408.
77. Python code for gene selection via relief binary nuclear reaction optimization algorithm based on differential evolution). https://github.com/D-Amr-Atef/Gene_Selection_RBNRO_Algorithm.git.

Publisher's Note