Evolutionary Computation-based Feature Selection for Finding a Stable Set of Features in High-dimensional Data

Sadegh Salesi Mousaabadi

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

School of Science and Technology

Nottingham Trent University

September 2019

Abstract

Evolutionary Computation (EC) algorithms have proved to work well for feature selection because they are powerful search techniques and can produce multiple good solutions. However, they suffer from some limitations for real-world applications. Firstly, ECs require high computation time as they evaluate many solutions at each iteration. Secondly, a classifier is usually used as their fitness function which causes the selected subset to perform well only on the utilised classifier (e.g. classifier-bias). Lastly, ECs, as stochastic search methods, return a different final subset in different runs which poses a problem for finding a stable set of features (e.g. stability issue). To address computation time and classifier-bias limitations, this thesis proposes a new two-stage selection approach called filter/filter in which two filter feature selection algorithms are combined. In the first stage, a ranking algorithm forms a reduced dataset by selecting the most informative features from the original dataset. In the second stage, the reduced dataset is fed to a novel EC algorithm to select final feature subset. This new EC algorithm is a Tabu search hybridised with an Asexual Genetic Algorithm called TAGA. TAGA benefits from new search components and solution representation which can effectively reduce computation time. To select a classifier-unbiased final subset, a statistical criterion is used as the fitness function which evaluates the subset independent of any classifier. Experiments show that the proposed filter/filter requires an acceptable computation time and selects more classifier-unbiased features compared to the state-of-the-arts. To find a stable set of features, a novel Generalisation Power Index (GPI) is proposed to analyse the generalisation power of final subsets of an EC in several runs. Generalisation power refers to performance capability of a subset over wide range of classifiers. Compu-

tation results confirm that GPI is able to find a stable set of features which achieves near optimal accuracy when used to train various classifiers. To examine the suitability of the proposed methods for real-world applications, the filter/filter approach and GPI are integrated to select a stable set of features for METABRIC breast cancer subtype classification problem. Experimental results show that this integration not only can address the limitations of ECs for a real-world biomedical feature selection problem but it performs better than alternatives methods.

Acknowledgements

The completion of PhD studies is a special milestone achievement in one's academic life of higher education. This is now the time to express my thanks and appreciations toward the people who greatly influenced my research journey.

Primary thanks go to my director of studies Dr. Georgina Cosma because of her dedication in guiding and supporting me throughout the course of research. Dr Cosma has always allowed me a complete freedom to define and explore my own directions in research. I am so grateful and proud to be guided under her supervision as I have learnt a lot from her, not only research and academic skills but also teamwork and managing skills which will be beneficial to both my academic career and personal life.

I would like to thank Nottingham Trent University for awarding me a Vice Chancellor's scholarship to pursue my studies and for financially supporting me to attend conferences and summer schools to develop my academic knowledge.

I also would like to thank my supervisory team Prof. David Brown, Prof. Graham Pockley and Prof. Graham Ball for all their support and advice.

Special thanks must go to my parents for all their unconditional support and encouragement in my whole life. I also wish to thank my brother Mohammad, my sister Elahe, and my best friend Li Jia for supporting me during my PhD studies.

Declaration

The contents of this thesis are a result of my own work, and it contains nothing that is based on collaborative research. No part of the work contained in this thesis has been submitted for any degree or qualification at any other university. Parts of chapters 4 and 5 have been submitted to journals and they are currently under review. Parts of chapter 5 have been published in a conference proceeding [105].

Contents

List of Tables

List of Figures

List of Abbreviations

ACC	Accuracy
AGA	Asexual Genetic Algorithm
AI	Artificial Intelligence
ASU	Arizona State University
ATI	Average Tanimoto Index
BE	Backward Elimination
CART	Classification And Regression Trees
CGA	a Customised Genetic Algorithm for feature selection
CPU	Central Processing Unit
CSCoefficient	Compactness-Separation Coefficient
EC	Evolutionary Computation
GA	Genetic Algorithm
GPI	Generalisation Power Index
K**NN**	K Nearest Neighbour
LDA	Linear Discriminant Analysis
LOOCV	Leave-One-Out Cross-Validation
LS	Local Search

METABRIC	Molecular Taxonomy of Breast Cancer International Consortium
MI	Mutual Information
ML	Machine Learning
MLP	Multi-Layer Perceptron
mRMR	Minimum Redundancy-Maximum Relevance
NB	Naïve Bayes
NP-hard	Non-deterministic Polynomial-time hardness
PCA	Principal Component Analysis
PGA	Parallel Genetic Algorithm
QPFS	Quadratic Programming-based Feature Selection
RVM	Relevance Vector Machine
SFS	Sequential Forward Selection
SPECMI	Spectral relaxation Conditional Mutual Information
SVM	Support Vector Machine
TAGA	Tabu Asexual Genetic Algorithm
TGA	Tabu Genetic Algorithm
TL	Tabu List
TS	Tabu Search
UCI	University California Irvine

Chapter 1

Introduction

1.1 Overview

In machine learning and data mining, a dataset is usually described by a group of features and samples. Due to availability of sophisticated data collection tools, there are usually a large number of features to be taken into consideration when building a machine learning model, including many irrelevant and redundant features. Irrelevant and redundant features negatively influence the performance of machine learning models in terms of training time, which is mainly caused by the curse of dimensionality [47]. Therefore, to build up a reliable machine learning model which is able to process the data in an acceptable computation time, to improve learning accuracy, and to facilitate a better understanding of the learning model, a feature selection process is needed. Feature selection mainly focuses on selecting a subset of features which can efficiently describe the input data, whilst reducing the effects of redundant and irrelevant features and the impact these have when building machine learning models – hence using a subset of features but still provide good prediction results [51].

Feature selection is a non-deterministic polynomial-time hardness (NP-hard) problem with a large complex solution space [6]. Because, firstly, since the number of features in the optimal feature subset is not known in advance, the dimension of the search space cannot be reduced to feature subset with certain number of features. Secondly, since the features may have complementary or contradictory interactions with each other, the decision space is non-separable [50].

A variety of search techniques have been applied to feature selection including exhaustive search, greedy search, heuristic search, and random search [31, 77, 75]. However, due to the global search potential and heuristic guidelines, Evolutionary Computation (EC) techniques have recently received much attention from the feature selection community [128]. Many of EC methods select a small number of important features, produce higher accuracy, and generate small models that are efficient on unseen data. Consequently, EC techniques have now become important methods for handling high dimensional feature selection [132].

The paradigm of EC algorithms consist of stochastic search algorithms inspired by the process of Darwinian theory of natural selection [85, 32, 39]. The EC algorithms often start with a population of solutions. When applying EC for feature selection, each individual of the population represents a subset of features which is a potential solution to feature selection problem. The quality of the subsets are evaluated using the fitness criterion and then an iterative process is used to improve the solutions.

The motivation for applying ECs to the feature selection problems is that, unlike conventional feature selection methods that perform a local, greedy search in the space of candidate solutions and produce local optimal solutions, ECs are robust, adaptive search techniques, they can perform a global search

in the solution space, and they tend to better deal with attribute interactions than greedy methods [44, 33, 95, 43]. However, applying EC methods on high-dimensional data feature selection problems is still a challenge in the field of feature selection which is the primary motivation for this research project.

1.2 Problem definition

Feature selection approaches utilise a search technique to find the best feature subsets that optimise an evaluation criterion. Search techniques for feature selection can be separated in three categories [128]: exhaustive, heuristic, and EC methods. Exhaustive algorithms thoroughly search the entire subset space and hence, they become costly in terms of computation time for datasets which comprise a large number of features. Heuristic algorithms make locally optimal choices with the aim of finding a global optimum amongst local optima. However, heuristic methods lack of a global search strategy and consequently, they are usually trapped into local optima [119, 26]. Therefore, EC techniques are able to better solve feature selection problems because they benefit from a global search strategy and heuristic guidelines.

EC algorithms are powerful search techniques that do not need domain knowledge, do not make any assumptions about the search space, and can produce multiple good solutions. However, their application to real-world feature selection problems has been limited due to their high computation time and the stability issue.

To deal with large datasets, particularly high-dimensional data, EC algorithms require high computation time because they are iterative algorithms and in each iteration, they need to evaluate many subsets. In terms of the stability issue, EC algorithms are stochastic search algorithms and they reach different

solutions whenever they are run. This poses a problem to find the best subset of features. For more information on EC techniques for feature selection the readers are referred to [128, 132].

To resolve the computation time limitation of EC algorithms for feature selection, most of the existing EC-based large-scale feature selection approaches employ a two-stage approach called filter/wrapper [17]. In the first stage, a filter method, which statistically evaluates feature subsets in terms of intrinsic correlation between features in the subset, is utilised to find most discriminating features and to reduce the dimensionality of the feature space. In the second stage, a wrapper algorithm, in which the classification performance of a machine learning algorithm (e.g. classifier) is used to evaluate feature subsets, is employed to find the best candidate subset from the features identified in the first stage. Filter and wrapper feature selection methods are discussed further later in Section 2.1.

When employing a wrapper algorithm in the second stage, hybrid methods typically are biased toward the classifiers used [91]. A wrapper method has an embedded classifier and its objective is to find the best subset of features that achieves highest classification accuracy for a specific classifier[91]. The limitation of this is that the performance of the wrapper approaches depend on the classifier. In particular, choosing a different classifier will return a different subset of features, and this increases the complexity of finding the optimum subset of features from those returned by the algorithm over the various iterations. Therefore, when developing a feature selection algorithm, it is important to develop an algorithm for which the selected features can provide acceptable performance over a range of classifiers (herein, this is named as the classifier-bias issue). This is of significant importance particularly in cases where the best classifier for the data at hand is not known in advance and the

selected subset can be used to evaluate the performance of different classifiers without the need to repeat the feature selection process for each classifier.

Another limitation of EC algorithms for feature selection is that due to **the randomness in their nature [28], a different 'best feature subset' solution is returned every time they are run, and this is known as the stability issue [128]**. Stability issue can pose significant problems for the application when a specific set of features is sought after to construct prediction models. However, instability of EC algorithms has provided them with an advantage over other searching strategies. Because, unlike exhaustive and heuristic search strategies which are deterministic methods and provide a single final subset, EC techniques are able to produce **multiple high quality final subsets** in different runs which provide more options to search for an optimal or near optimal subset [128].

A solution to stability issue is to apply a further selection process on a set of subsets obtained from different runs of an EC algorithm to select the best subset. Existing solutions include typical aggregation [15] (e.g. intersection and union) and frequency-based methods [108]. However, these methods do not consider the performance of a classifier in their selection process and they can select a feature subset which when utilised to train a classifier can lead to poor classification accuracy. Classifier-based aggregation [16] is an alternative method, which uses the performance of one classifier to select the best subset of features. This approach may result in a biased subset with poor performance over various classifiers known as lack of generalisation power.

1.3 Aims of the research

This thesis focuses on developing solutions for addressing the limitations of EC algorithms for feature selection problems, specifically related to stability and computation time. Therefore, the main goals and the challenges which were identified during this research and need to be addressed are as follows:

- Computation time: EC algorithms are costly in terms of computation time as they need to assess a large number of solutions at each iteration. Therefore, this thesis investigates solutions to reducing the computation time needed by EC algorithms in finding optimal subsets of features from high-dimensional datasets. The selected features will be utilised for constructing machine learning models.

- Classifier-bias: EC-based algorithms are mostly embedded into a wrapper framework which uses the classification performance of a classifier (e.g. accuracy) to evaluate feature subsets. This can result in a final selected subset of features which is classifier-biased, meaning that the selected subset is only applicable for the specific classifier and may lead to a poor performance if applied to other classifiers – this is also known classifier-bias issue [91]. Hence, alternative evaluation metrics need to be considered as the fitness function for EC-based algorithms and new ones need to be proposed as necessary.

- New solution representations and search components: The dominant EC solution representation in the domain of feature selection is a binary representation which is followed by binary search operators. However, binary representation is not applicable to all types of evaluation metrics (e.g. information theory based metrics) and consequently new solution representations and search operators need to be developed.

- Stability issue: Stabilising EC algorithms involves removing their stochastic components which damages their random search nature. Therefore, this thesis rather concentrates on developing a method which is capable to find a stable set of features amongst all final subsets of an EC-based feature selection algorithm over several independent runs.

1.4 Objectives of the research

To address the limitations and challenges of EC techniques for feature selection discussed above, the following objectives are defined:

- Objective O1: Develop an EC-based feature selection algorithm which benefits from new solution representation and search components to reduce computation time taken by EC algorithms for finding optimal or near optimal subsets of features within high-dimensional datasets for building machine learning models.

- Objective O2: Develop a solution to address classifier-bias problem associated with EC algorithm embedded into wrapper frameworks for which the selected features are biased toward the utilised classifier. The proposed solution will select the features independent of the classification performance of any classifier and therefore, the selected feature will be able to provide acceptable performance over wide range of classifiers.

- Objective O3: Develop a solution to the stability issue [128] associated with the challenge of finding the best subset of features over several runs, when EC algorithms are adopted for feature selection tasks. A solution would be based on a Generalisation Power Index (GPI) which measures the performance of feature subsets in terms of generalisation power over

multiple classifiers.

- Objective 04: Evaluate the performance of the proposed algorithms on a real-world case study in particular the METABRIC breast cancer dataset. METABRIC dataset contains a large number of features and many samples, and the proposed algorithms are applied to METABRIC dataset in order to find the best biomarkers for detecting breast cancer subtypes.

1.5 Description of the work/contributions

The first contribution of this thesis (Objective O1) proposes a solution to the high computation time of EC algorithms as well as to the classifier-bias problem (Objective O2) of filter/wrapper methods. This proposed solution is a two-stage algorithm that combines a novel EC-based filter algorithm with another filter algorithm (e.g. Fisher score [82]) to create a filter/filter approach. The filter/filter approach here has a two-fold aim.

In the first stage, it reduces the size of the original dataset and as the result reduce the computation time required by EC algorithm to process the reduced dataset in the second stage. In the second stage, the EC algorithm benefits from a statistical evaluation metric (fitness function) which leads to select a final subset which is not classifier-biased and is able to provide high generalisation power over a range of classifiers. Next, the stages of the proposed approach are explained in detail.

In the first stage of the proposed filter/filter algorithm, the Fisher score feature selection algorithm [82], which is computationally cost effective, is applied to reduce the complexity of the datasets and to filter out the most promising features which are then fed to the next stage. In the second stage, a novel

EC-based feature selection algorithm called Tabu Asexual Genetic Algorithm (TAGA) is developed and applied to the reduced datasets in the previous stage. TAGA is an enhanced EC algorithm which needs considerably less computation time compared to its type and also utilises a statistical fitness function which obviates the limitation of wrapper/filter approaches in terms of classifier-bias. TAGA, is a string type long-term memory Tabu Search (TS) [48], with a new effective solution storing and restoring scheme, hybridised with an integer-coded Asexual Genetic Algorithm (AGA) [21] as the local search in order to provide new search directions for the algorithm. Moreover, a Sequential Forward Selection (SFS) procedure is added to AGA to enhance and accelerate the performance of the algorithm. AGA benefits from a new solution representation in order to steer the searching process more efficiently and works only based on a mutation operator and lacks a crossover operator.

The reasoning behind using AGA is the fact that a suitably designed mutation operator is sufficient to guide the searching process in order to find high quality solutions and therefore, removing crossover operator can increase the speed of classical GAs. Finally, the information theory-based minimum redundancy-maximum relevance (mRMR) [98] criterion is used as the fitness function of TAGA (rather than the output of a classifier which is used in filter/wrapper methods) to evaluate the subsets in terms of relevance and redundancy. In this way, not only the selected subsets are not classifier-biased (and hence the classifier bias issue is addressed) but also, the possibility of selecting correlated features in the same subset is reduced. Experiments were carried out on various high-dimensional datasets including image data, text data, and biological data. The quality of the selected subsets were evaluated using different classifiers. The experimental results demonstrate that the proposed algorithm outperforms the conventional and state-of-the-art feature selection algorithms

in most cases.

The second contribution is to address the stability limitation (Objective O3) of existing EC algorithms when applied to feature selection. This thesis proposes a novel generalisation power analysis approach based on a Generalisation Power Index (GPI) which measures the quality of a feature subset when applied to a range of classifiers taking into consideration classifiers optimal accuracy. The proposed approach finds the subset which is of the highest quality in terms of generalisation power (e.g. optimal subset) from a pool of many output feature subsets. These subsets were output by an EC algorithm in several runs and which are considered to be stable sets of features.

In feature selection, the quality of a subset can be measured through discrimination power and generalisation power. The discrimination power of a subset provides the classifier with an ability to approximate decision boundaries in the feature space and consequently, it results in optimal classification accuracy achievable by the classifier. The term generalisation power refers to the performance capability of a subset to achieve optimal accuracy when used to train various classifiers. According to these two terms, an optimal subset is able to provide enough discrimination power to any classifier in order to achieve their optimal classification performance. Therefore, this approach is able to analyse the output subsets of an EC algorithm in several independent runs in terms of their generalisation power on a range of classifiers and to select the subset with the highest generalisation power as the best subset. Computation results confirm that GPI has outperform alternative methods in finding a stable set of features which achieves optimal or near optimal accuracy when used to train various classifiers.

In order to test the applicability of the proposed methods in previous objectives for real-world problem (Objective O4), the TAGA embedded into a

filter/filter framework (objectives 1 and 2) and the GPI (objective 3) are combined to select a stable set of features for METABRIC breast cancer subtype classification problem. The results show that the combination of these two algorithms performs better than alternatives methods and is able to cover the limitations of ECs for a real-world biomedical feature selection problem.

1.6 Thesis outline

The thesis structure is illustrated in Fig 1.1 and an overview of the thesis structure is also provided below.

- Chapter 2 provides an overview of concepts in feature selection, discusses applications of EC algorithms for feature selection, identifies challenges encountered by EC algorithms for feature selection and discusses limitations of existing methods.

- Chapter 3 demonstrates the issues and limitations of EC-based algorithms for feature selection using a simple GA as an EC test case algorithm and a small dataset.

- Chapter 4 describes the first contribution of the thesis, a novel EC-based feature selection algorithm called Tabu Asexual Genetic Algorithm (TAGA). TAGA has been embedded into a new two-stage hybrid framework called filter/filter approach to deal with high computation time of EC methods as well as the classifier-bias limitation of existing filter/wrapper methods.

- Chapter 5 describes the second contribution of the thesis, a novel generalisation power analysis approach based on a Generalisation Power Index

(GPI) that measures the performance of feature subsets in terms of generalisation power over multiple classifiers.

- Chapter 6 describes the application of the proposed contributions, i.e. the TAGA algorithm and the generalisation power analysis approach developed in Chapters 4 and 5 respectively, applied to the METABRIC microarray dataset as a real-world case-study. The case-study is concerned with using the proposed approaches for finding the best subset of biomarkers for detecting breast cancer types.

- Chapter 7 summarises the steps which have been taken in this thesis to address the limitation of EC algorithms for feature selection, describes contributions and objectives of the thesis, and provides suggestions future work.

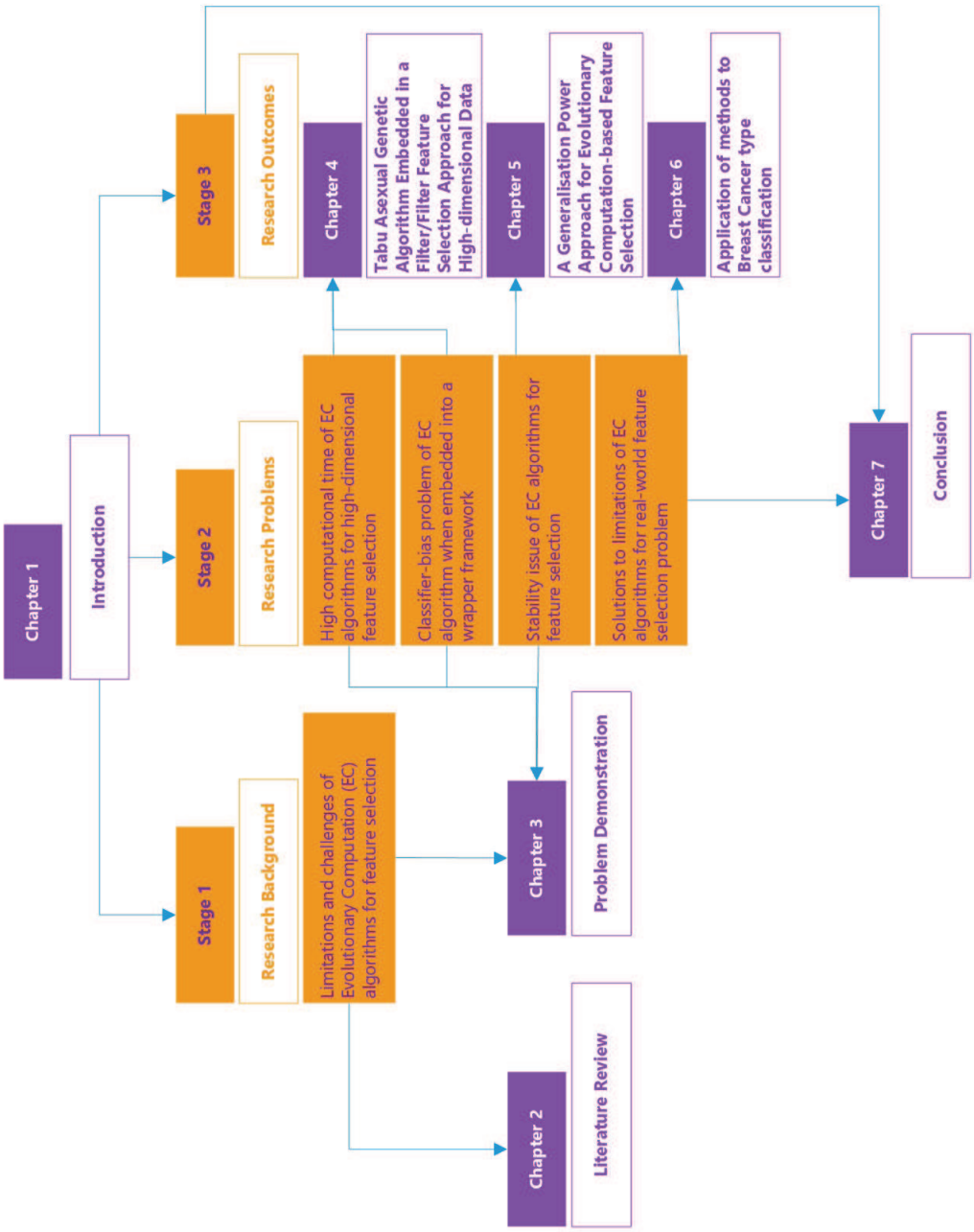

Figure 1.1: Thesis structure

Chapter 2

Literature Review

This chapter provides an overview of concepts in feature selection and discusses applications of EC algorithms for feature selection. Challenges encountered by EC algorithms for feature selection and the limitations of existing methods to deal with these challenges are also discussed in this chapter.

2.1 Introduction to feature selection

Real-world data has become high dimensional which, as a result, consists of a large number of features and samples. However, not all features are essential for constructing machine learning models (e.g. classifiers) since many of the features are redundant or even irrelevant. Redundant and irrelevant are two distinct notions; since one relevant feature may be redundant in the presence of another relevant feature with which it is strongly correlated [51]. The central premise when using a feature selection technique is that the data contains some features that are either redundant or irrelevant, and can thus be removed without incurring much loss of information [12]. Nevertheless, it is a challenging process for an algorithm to identify the best combinations of features from

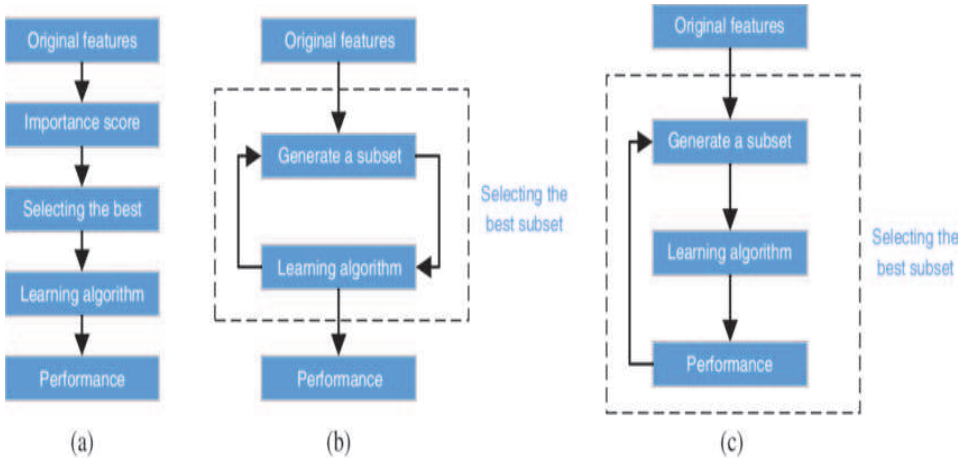

Figure 2.1: Three types of feature selection. (a) Filter (b) Wrapper (c) Embedded (adopted from [123])

high-dimensional datasets which contain a large number of features and a small sample size. Therefore, feature selection is the process for automatic selection of the most relevant features required for modelling and other machine learning tasks. Feature selection reduces the inputs required to construct machine learning models and this consequently reduces the complexity of the models in terms of time and computational processing power required by the model to learn the data [51]. Generally, feature selection techniques are used for four reasons: 1) simplification of models to make them easier to interpret by users [58], 2) shorter training times, 3) avoiding the curse of dimensionality, and 4) enhancing generalisation by reducing overfitting [12].

2.2 Feature selection methods

Traditional feature selection methods are commonly presented in three main categories based on how they combine the selection algorithm and the model building. These categories are: filter, wrapper, and embedded [17], and are illustrated in Figure 2.1.

In filter approaches, a statistical measure is applied to assign a score to each

feature, according to their correlation with other features and the target variable, and then the features with the highest scores are selected [104]. Because filter approaches are classifier-independent, the selected feature is not classifier-biased and provides high generalisation power over a wide range of classifiers. In addition, they have low computational complexity and they are easily applicable to high dimensional datasets. One important limitation of filter methods is that in highly correlated datasets, a redundant subset of features may be selected and consequently, when the selected subset is used for training a classifier it may prove not to be a good subset of features after all [51].

Wrapper approaches search through the space of all possible feature subsets and use the performance of a classifier to evaluate the usefulness of feature subsets and select the one that maximises the accuracy of the classifier [68]. Compared to filter approaches, the performance of wrapper approaches is often higher in terms of accuracy, but they require high computational efforts for high dimensional data as they may need to check all possible combination of feature subsets. Moreover, the selected subset is biased toward the particular classifier used and may show poor classification performance over other classifiers [24].

Embedded approaches [17] rank features during the training process of a classifier and thus simultaneously determine both the optimal features and the parameter tuning of the classifier to achieve higher accuracy. In fact, embedded methods learn which features best contribute to the accuracy of the model while the model is being constructed and therefore, the feature selection algorithm is integrated as part of the learning algorithm. Embedded strategies are computationally less expensive than wrapper approaches as they do not require running exhaustive search over all subsets and they mostly evaluate each feature individually based on the score calculated during tuning classifier

parameters. However, similar to wrapper methods, embedded methods are dependent on the performance of a classifier and thus they may still be computationally expensive for high dimensional data and the selected subset may be biased toward the particular classifier used.

Hybrid feature selection approaches have recently emerged and these are known to be more suitable for high dimensional problems, compared to the traditional approaches (filter, wrapper, and embedded) [17]. Hybrid methods combine the best properties of filters and wrappers [61] and benefit from sub-algorithms and therefore are considered more robust when compared to traditional approaches [17, 78].

A typical hybrid feature selection method, also called filter/wrapper, consists of two stages. In the first stage, a filter method is used in order to find most discriminating features and to reduce the dimensionality and complexity of the feature space. In the second stage, a wrapper algorithm is employed to find the best candidate subset from the features identified in first stage. Employing a wrapper algorithm in the second stage, typical hybrid methods inherit classifier-bias property of wrappers. In particular, choosing a different classifier will return a different subset of features, and this increases the complexity of finding the optimum subset of features. Therefore, when developing a feature selection algorithm, it is important to develop an algorithm for which the selected features can provide acceptable performance over wide range of classifiers.

2.3 Evolutionary computation algorithms for feature selection

2.3.1 Overview

EC techniques have recently received much attention from the feature selection community [128] as many EC methods select a small number of important features, produce higher accuracy, and generate small models that are efficient on unseen data. Consequently, EC techniques have now become important methods for handling high dimensional feature selection [132]. As shown in Figure 2.2, EC algorithms often start with an initial population of solutions. When EC algorithms are applied for feature selection, each individual of the population represents a subset of features which is a potential solution to feature selection problem. The quality of the subsets are evaluated using a fitness criterion and then an iterative process is used to improve the solutions.

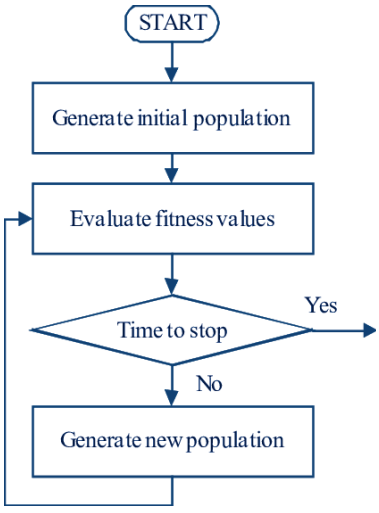

Figure 2.2: Flowchart of EC algorithms (adopted from [63])

2.3.2 Genetic algorithms

Genetic algorithms (GAs) are categorized as a part of EC which is an area of Artificial Intelligence (AI). GAs are inspired by Darwin theory of natural selection, and were first introduced and developed by Holland [54]. The GAs are population-based optimisation algorithms and each individual of the population is a solution for the problem. The variables of the solution are encoded into the chromosomes which are formed by a list of genes. A fitness function is also needed to measure the fitness of the encoded solutions. The GA starts with an initial population and the chromosomes compete against each other to survive. In each generation, some chromosomes are randomly selected, with a tendency towards fitter chromosomes, as the parents for reproduction and recombination to generate offspring (new solutions) which comprise the next generation. The mechanism by which the GAs generate offspring are: Crossover and Mutation. By the means of crossover operator, GAs typically search for advantageous patterns in the existing elite solutions to improve the quality of solutions even further. Furthermore, mutation operators are usually used to diversify the pool of solutions. In Figures 2.3 and 2.4 the searching process flowchart of a typical GA and its genetic operators are presented respectively.

Genetic Algorithm Solution Representations for Feature Selection

The dominant GA solution representation in the literature is binary string in which 1 shows the corresponding feature is selected and 0 means not selected [128]. Accordingly, binary search operators have been proposed in order to steer the search process. Many different approaches have been proposed in order to improve the performance of the GA in terms of solution representation and search operators. Li et al. [73] proposed a dynamic Adaboost learning

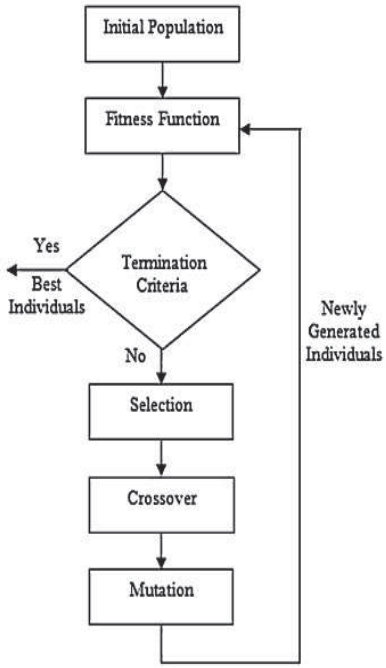

Figure 2.3: Flowchart of a typical GA. (adopted from [87])

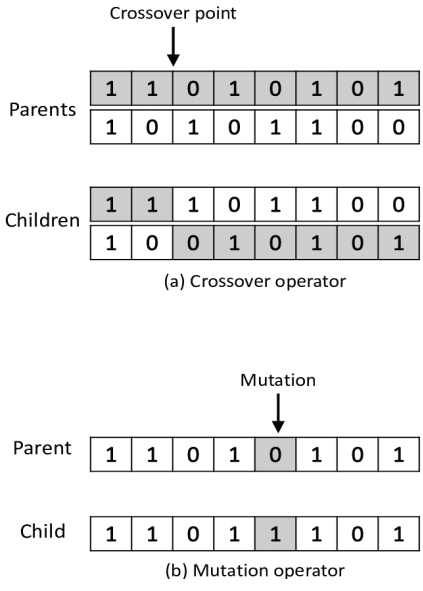

Figure 2.4: Two Genetic operators (a) Crossover operator, (b) Mutation operator (adopted from [4])

with feature selection based on parallel GA. The proposed algorithm uses a bio-encoding scheme in which the chromosomes are composed of two strings, one binary-encoded string to show selected features and the second one is a real-number encoding method which represents the weight of the features.
Genetic operators consist of two parts: operators for the binary-coded chromosome part and for the real-valued coded one, which are typical both binary and real-valued swap mutation and two-point crossover operators. Winkler et al. [126] developed a new representation for GA composed of one binary part for feature selection and a real-valued part for parameter optimisation which is able to discover optimal feature subset and optimal parameter values for a Support Vector Machine (SVM) classier using typical binary and real-valued genetic operators.
Souza et al. [112] developed a co-evolutionary Genetic Multilayer Perceptron (MLP) for feature selection which used three-level representation. The layers indicate feature selection, the neurons pruning, and the MLP architecture, respectively. Typical binary genetic operators are applied to guide the search process. It appears that the binary coded representation is the most common GA solution representation in the literature, followed by binary genetic operators. In binary representation, the selected features are determined by binary values and therefore subsets with different number of features are generated and compared. However, binary representation may not be applicable in cases when mRMR is used as GA fitness function because mRMR value is highly size-dependent and is comparable for the subsets with the same number of features. Therefore integer-coded solution representation seems more suitable for mRMR.
Integer-coded solution representations have also been studied for feature selection. Jeong et al. [59] suggested a new GA with an integer-coded solu-

tion representation which is able to further reduce the dimensionality of high dimensional data. In this representation, the length of each chromosome is equal to the number of desired features. In cases where the index of a feature appears more than once, a SFS operator is used to find alternative features. A similar approach was used by Ludwig et al. [79] where a GA with an integer-coded representation is combined with mRMR as fitness function. A potential limitation in [79] and [59] is that both algorithms work based on the crossover operator and suffer from the absence of a proper mutation operator which is important in generating diverse solutions. Because most of feature indexes are not present in the solutions, designing a swap mutation is difficult and consequently, the mutation is replaced with a SFS-like procedure to repairs faulty solutions. This SFS-like procedure however is only used to repair the solutions and therefore is unable to provide the search process with diverse solutions. Consequently, to address the limitation of the available integer-coded representation for GA, a novel representation that can reduce the dimensionality of the search space as well as a new genetic operators which are able to effectively steer the search process will be needed.

Asexual genetic algorithm

There is another version of GA in which the algorithm lacks of crossover operator and works only based on mutation operator which is called Asexual Genetic Algorithm (AGA) [22]. The AGA employs the survival of the fittest principle in an asexual reproduction scheme [7]. The development of this type of GA has been based on three assumptions [7]. Firstly, the success of any type of metaheuristic, from point-based to population-based, depends on the trade-off which it makes between intensification and diversification, with diversification aiming at exploring new regions, and intensification aiming at searching the

high quality regions already distinguished. Secondly, the mechanism of mutation by which the population is manipulated as well as the combinations of different mutation operators can highly affect the trade-off between intensification and diversification. Thirdly, biased-mutations with a tendency towards fitter solutions can play a twofold role in the sense of contributing to both intensification and diversification. In other words, using a biased-mutation not only can diversify the search process, but it also can guide the search towards high quality solutions. The application of AGA in different scientific fields has shown its advantages over the classical version of GA. Canto′ et al. [21] presented an AGA and applied the algorithm in finding the global maximum in functions composed of two variables and also parameter estimation in astronomy by minimisation of the chi-square. The results show that their algorithm needs less parameter tuning effort and is computationally less expensive than the standard version of GA and can converge to optimal or near optimal solutions in just a few generations. Chakroborty and Mandal [23] used a mutation-based GA to solve various types of the vehicle routing problem. The computation results reveal that their algorithm is fast and gives optimal or near optimal solutions with minimal computation effort. Amirghasemi and Zamani [7] developed a hybrid algorithm of AGA and TS for solving job shop scheduling problem called TGA. The effectiveness measurement of TGA indicates its coverage of the search space and its intensification on exploring high quality solutions. However, there are some limitations in their work. Firstly, they have used a short-term memory tabu list which length is randomly set in each iteration. One drawback of short-term memory tabu list is that the length of the tabu list may significantly affect the performance of the algorithm which necessities tuning the length of the tabu list. Nevertheless, there is still no effective method to properly determine the length of the tabu list [38]. Fur-

thermore, even if the length is properly tuned, the short-term memory tabu list may still trap into local optima. Secondly, their proposed tabu list storing strategy stores both the solutions and the moves. This storing strategy may require high computational effort to check whether a solution is in the tabu list. Thirdly, in the proposed algorithm, the new solutions are first generated and their fitness values are calculated through the fitness function and finally, their existence in the tabu list is check. However, a better strategy is to first check if the solutions are in the tabu list before calculating their fitness values which may significantly reduce the computational efforts. Lastly, the mutation operator is only applied on the best solution in the generation to explore its neighbourhood only. However, this results in exploring limited regions of the search space. Sometimes, mutating bad solutions will lead the search process into the regions with higher quality solutions [35].

As explained in subsection 2.3.2, integer-encoded solution representation is the suitable representation for GA when mRMR is the fitness function. Unlike other combinatorial problems with integer-encoded representation (e.g. scheduling and routing problems) for which the entire sequence of integers is the solution, part of the sequence is the solution for feature selection problems. In a crossover operator, information obtained from two parents are combined to generate new offspring. For feature selection, the parents are the subsets (part of the sequence) which contain small portion of entire features, few features in some cases. Therefore, recombination of the parents with limited feature diversity will most likely result in generating new solutions which are already discovered or are faulty containing the same features. Because these repetitive and faulty solutions will require high computational effort to be fixed or tabued (in cases if a tabu list is used), removing the crossover operator from GA may reduce the computation time of the entire algorithm without having

negative impact on the its performance.

Parallel GA for feature selection

GAs are computationally time consuming because they search through a large set of solutions to find optimal solutions. Parallel processing techniques can be used to improve the efficiency of GAs by exploiting the simultaneity of calculations performed in genetic algorithms [116]. Parallelising a GA for feature selection necessities parallelising other comparing algorithms in the same manner for a fair performance comparison. However, algorithm parallelisation is part of the parallel processing and its impact for feature selection needs to be investigated in the context of parallel processing. Because of the importance of algorithm parallelisation in terms efficiency, the application of Parallel GA (PGA) and its advantages and challenges are briefly explained in this section. The main idea of Parallel GA (PGA) is to split the entire population into several subpopulations and evolve all the subpopulations simultaneously on multiple processors [27]. A PGA actually uses various GAs running on separate processors to process one part of the population (subpopulation), with or without communication between the processors. PGAs divide the population into a few large subpopulations and genetic operators are carried out within the subpopulation on multiple processors. After several generations, individuals from different subpopulations will be exchanged and form the new subpopulations for further evolution. The following are some works which have applied PGA in the context of feature selection.

Chen et al. [27] developed a coarse-grained parallel genetic algorithm to simultaneously optimise the feature subset and parameters for SVM. The computation results demonstrates that the developed algorithm has been able to find optimal feature subset and parameters for SVM in significantly shorter

time when compared to a generic GA. Mokshin et al. [88] proposed a parallel genetic algorithm is to solve feature selection problem of production enterprise functioning. Their proposed PGA was implemented to search for the best number of parallel evolutionary paths. The effectiveness of the proposed approach was examined in comparison with results of typical feature selection algorithm including Fisher score and multiple determination coefficient. The computation results confirmed the superiority of the proposed algorithm over other algorithms in terms of prediction and speed. However, their computation experiments appear to have been performed unfairly as the other competing algorithms were not of state-of-the-art and most likely, a generic GA would also outperform them. Furthermore, the other algorithms were not parallelised in the same manner as their proposed PGA, and their computation results were obtained from a standard version.

Soufan et al. [111] developed an online filter/wrapper feature selection platform based on the parallel GA to reduce the computation time required for feature selection. The performance of the proposed platform was compared with other available feature selection tools both parallel and non-parallel ones (such as WEKA [52], and FST3 [109]). The computation results show superiority of their platform over other tools in terms of classification performance. However, in terms of computation time, the platform has not been able to compete against other tools (some of which were non-parallel) for some cases. PGAs can increase the diversity of population and significantly reduce computation time [118]. However, there exist some limitations in their applications which need to be taken into account. Algorithm parallelisation may need a complicated programming because it needs deep understanding of the algorithm procedures to find the parts that can actually be parallelised [14]. PGA may include redundant computation in which processors explore the same re-

gions of the search space. To avoid redundant computation, a communication process is needed for synchronising processors tasks. However, synchronisation is a sequential component which may impose significant limit on the amount of parallelisation [113]. Parallelisation may be less efficient when the iterations are less expensive in terms of processing time [106]. It may be because the time spent to manage redundant computation is much higher than the time if the algorithm is run sequentially.

2.3.3 Tabu search

Tabu Search (TS) was introduced by Fred Glover [48] as a general iterative metaheuristic that guides a local heuristic search procedure to explore the solution space for solving combinatorial optimization problems. Local Search (LS) algorithms take a potential solution to a problem and generate its immediate neighbours (i.e. the solutions that are similar to the original solution except for very few minor details) with the aim of finding an improved solution. However, LS methods have a tendency to become stuck in suboptimal regions or on plateaus where many solutions are equally fit.

A unique characteristic of TS is embodied in its exploitation of a memory, which records information about solutions and guides the moves from one solution to another. Therefore, the objective of TS is to prevent an embedded local search procedure from returning to recently visited areas (i.e. cycling) and escapes from local optima by using a Tabu List (TL) which incorporates attributes of explored solutions and therefore are forbidden to search in the future.

Two important components of TS are intensification strategies and diversification strategies [124]. Intensification strategies are based on modifying choice rules to encourage moves to the solutions previously found good in order to

search the attractive regions more thoroughly. Diversification strategies, on the other hand, encourage the search process to examine unvisited regions and to generate solutions that differ in various, significant ways from those seen before. Figure 2.5 present the procedure of a typical TS algorithm.

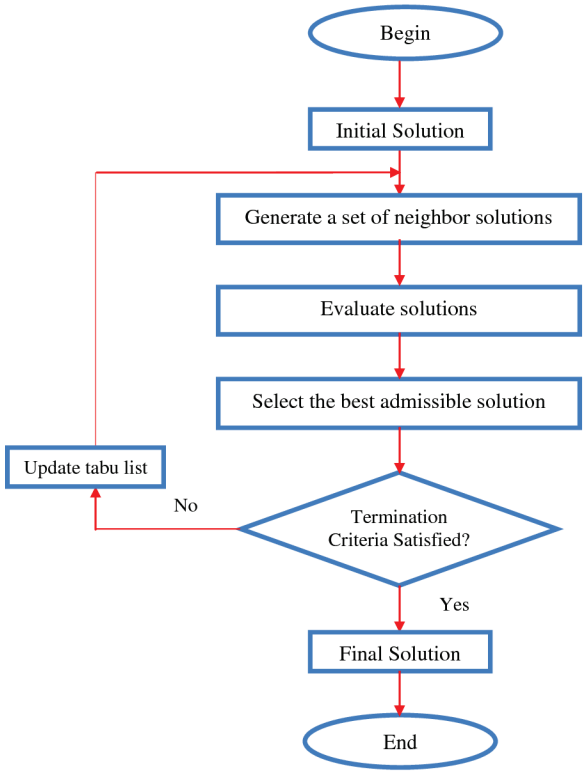

Figure 2.5: Flowchart of the Tabu Search procedure (adopted from [3])

There are several recent works which have applied TS in the context of feature selection. Huerta et al. [56] proposed a two stage gene selection approach for microarray datasets. At the first stage, several statistical filter methods are combined to select most informative genes and then, these genes are fed to the second stage.

In the second stage, a hybrid algorithm composed of Tabu Search, Genetic Algorithm and SVM is used to find the best feature subset. Wang et al. [122] developed a hybrid wrapper feature selection algorithm for gene expression data which incorporates imperialist competition algorithm to perform global

search, Tabu Search to conduct fine-tuned search, and Support Vector Machine as the classifier. Cui et al. [29] suggested an optimisation technique based on Tabu search and Compactness-Separation Coefficient(CS Coefficient) to perform dimensionality reduction and calculate optimal feature reduction number for image data. To verify the accuracy of classification,SVM and Relevance Vector Machine (RVM) classifiers were used.

Various types of tabu lists for feature selection

The memory structures also know as Tabu List (TL) used in TS can roughly be divided into three categories [48]: short-term, intermediate-term, long-term. Short-term and intermediate-term TS have been applied in different applications of feature selection [56, 124, 29]. Nevertheless, one drawback of using short-term and intermediate-term memory TL is to determine the number of maintained recent moves also known as the length of TL.

The length of the TL must be delicately tuned. However, existing theoretical research to determine the length of the TL is still insufficient in practice [38]. Even if the length of the TL is properly tuned, a TL with a finite size still cannot guarantee that the search procedure will not be trapped into local optima.

In fact, short-term and intermediate-term structures tend to be too local and spend most, if not all, of their time in a restricted portion of the search space. The negative consequence of this fact is that, although good solutions may be obtained, one may fail to explore the most interesting parts of the search space and thus end up with solutions that are still quite far from optimal solutions. Therefore, a long-term memory TS can relieve the problems caused by the short-term and intermediate-term memory.

The diversification of TS is usually based on some form of long-term memory

TL [46], such as a frequency memory, in which one records the entire solutions or moves generated during the search process. Long-term memory TSs are computationally expensive. It is mainly because in Long-term TS, unlike short-term and intermediate term in which TL is reset periodically, TL length is set on infinite and consequently becomes huge during the search process. Therefore, checking whether a move is in the TL is a time consuming task. To the best of our knowledge, application of long-term TS in the context of feature selection is limited.

Wang et al. [124] developed a hybrid feature selection approach using a long-term memory TS and probabilistic neural networks. In their algorithm, the TL length is set to infinite and the best solution in each iteration is added to TL. The results on various datasets show the superiority of their algorithm compared to previous works. However, their computation results show that the running time of the algorithm for small size datasets is expensive which makes the algorithm infeasible for high dimensional data. To cope with the complexity problem of long-term structures, one approach could be saving the solutions in the TL using an effective encoding scheme which accelerates the storing and restoring process. However, the proposed method by Wang et al. [124] lacks of any encoding scheme and the solutions are stored in their original binary representation.

2.4 Hybrid feature selection methods

In a hybrid method, two or more feature selection algorithms are sequentially combined which are usually of different conceptual origin [17]. Although in theory, combining two feature selection algorithms from the same type (e.g. filter/filter) is practical, the proposed approaches in the literature have mainly

focused on combination of filter methods with wrapper ones. Dowlatshahi et al. [36] proposed a three-stage filter/wrapper feature selection algorithm for microarray data.

In the first stage, multiple filter algorithms are used in order to find high ranked features according to their relevance. In the second stage, the features are ranked again and the ranking is used to weight the probability of selecting each feature. In the third stage, Competitive Swarm Optimization algorithm which uses the performance of K nearest neighbours (KNN) classifier as the fitness function is applied to find optimal subset from the weighted features.

Lu et al. [78] developed a feature selection algorithm hybridisation of mutual information maximisation as the filter stage and an adaptive GA as the wrapper algorithm which uses extreme learning machine classifier as the fitness function. Hancer [53] suggested a hybrid differential evolution approach which combines filter and wrapper approaches through an improved information theoretic and local search-KNN mechanism in a fuzzy framework in order to deal with both continuous and discrete data. Mafarja and Mirjalili [81] proposed a hybrid algorithm in which two incremental hill-climbing techniques as filter methods are hybridised with the Binary Ant Lion optimiser combined with KNN classifier as the wrapper method. Adair et al. [1] developed a hybrid filter/wrapper based on Mutual Information and Iterated Local Search called MRMR-ILS which uses KNN and SVM classifiers as fitness function to evaluate the performance of the proposed algorithm over three Brain Computer Interface datasets.

Filter/wrapper approaches accelerate the feature selection process and benefit from the advantages of both filter and wrapper methods [57]. However, wrapper-based algorithms suffer from lack of generalisation power and the selected subset is biased toward the classifier used (See section 2.6). The lim-

itation of wrapper approaches is that choice of feature set is dependent on the performance of the classifier. Using the wrapper approach, the selected feature subset may not be suitable when changing the classifier embedded in the wrapper approach, and therefore, the feature selection process will need to be repeated as it is in integral stage of the wrapper approach. The feature selection approaches should not only concentrate on classification performance, but also on finding stable and robust subsets [17].

2.5 Methods for combining subsets of features

This section provides an overview of the existing methods in the literature which combine several feature subsets into a single subset.

2.5.1 Aggregation methods for combining subsets of features

Aggregation methods are one type of ensemble methods for feature selection that are able to combine the output subsets of one or multiple feature selection algorithms [15]. Typical methods to combine subsets of features are Union and Intersection [15]. Intersection selects the features which appear in all subsets, whereas Union combines the unique features in a set of subsets. Both methods have been applied in the context of feature selection.

Viegas et al. [121] proposed a Genetic Programming approach for feature selection of high dimensional data in which several metrics are independently employed to measure the quality of subsets and at the end, Intersection and Union methods combine the subset of features obtained using each of those metrics as fitness function. Hsu et al. [55] developed a hybrid feature selection in which a filter algorithm is independently combined with an EC-based wrap-

per. In this approach, several filter algorithms are independently employed in the hybrid framework and at the end, Intersection is used to combine outputs. Tsai and Hsiao [117] combined multiple feature selection methods for stock prediction using Union, Intersection, and multi-intersection approaches. In this approach, three well-known feature selection methods including Principal Component Analysis (PCA), GA and Classification And Regression Trees (CART) are executed independently and the outputs are combined.

Although Intersection might seem a logical approach (if a feature appears in all subsets, it must be highly relevant), it can lead to very restrictive sets of features (an empty set in the worst case) which may result in removing most informative features. On the other hand, Union selects a subset with a large number of features. This approach produces better results than Intersection [5] however; the selected subset may still contain noisy and redundant features. Another important issue associated with both the Intersection and Union methods is that they do not consider classification performance of the features as a subset member [15]. The simplest approach to this would be to randomly choose a subset as the baseline and then add to the baseline the features which would improve classification performance [16]. However, this approach requires high computational cost for large data. Moreover, the subset selected by this approach is biased to the classifier used and lacks of generalisation over other classifiers.

2.5.2 Frequency-based methods for combining subset of features

In a set of feature subsets, the frequency of a feature indicates total number of occurrence of the feature in the subsets [108]. The frequency of a feature across

subsets can be an indicator of the relevance of a feature, such that features with higher frequency are considered to be more relevant than others given that they have frequently appeared in the subsets. In a set of subsets obtained from the outputs of an EC in different runs, it can be inferred that the features with higher frequency have been good quality features and that is why the EC algorithm has consistently selected them.

Few works in the literature have applied frequency-based approach to combine feature subsets. Bonilla-Huerta et al. [18] developed a two stage feature selection algorithm for microarray data in which statistical methods are combined a hybrid EC-based composed of GA and TS. At the end, a frequency analysis is applied on final subsets in different runs to generate a subset with most frequent features. Yousefpour et al. [130] proposed a frequency-based integration approach for sentiment analysis problem which integrates the subsets obtained through a hybrid algorithm composed of a filter and an EC-based wrapper in several runs. Pan [94] proposed a frequency-based approach for feature selection in which several ranking algorithms are employed to initially rank the feature and based on the rankings, a frequency list is created. The frequency list is then used to select most frequent features as the final subset. However, the main problem with frequency-based approaches is that these approaches do not consider the classification performance of the features in conjunction with other features in a subset. Hence, a subset of features that consists of highly frequent features might not result in good classification performance after all.

2.6 Generalisation power analysis for feature selection

Generalisation power enables the selected subset to be trainable over wide range of classifiers [91] without having to repeat the feature selection process to find the best subset for each classifier. Despite the importance of this topic, to the best of our knowledge, only one paper has investigated the application of generalisation power in the context of feature selection. Naghibi et al. [91] proposed an approach to analyse the generalisation power of the subsets in which multiple classifiers are used to obtain the classification performance of the subsets and optimal subset of one classifier is used to train other classifiers. However, this approach only examines and cannot measure the generalisation power of subsets. There is a need for methods which can measure the generalisation power of subsets over multiple classifiers. Good generalisation power will facilitate the selection of an optimal or near optimal subset which can achieve optimal or near optimal accuracy when used to train any classifier.

2.7 Mutual information for evaluating feature relevancy

The features of a dataset can be considered to fall into one of three different categories: strongly relevant features, weakly relevant features and irrelevant features [131]. While the strongly relevant features must be included in the optimal subset, the weakly relevant features are not always necessary but may become necessary for an optimal subset at certain conditions. To determine the relevance properties of the feature space, the Mutual Information (MI) concept was first introduced in [19]. In probability theory and information theory [107],

the mutual information of two random variables is a measure of the mutual dependence between the two variables. More specifically, it quantifies the amount of information obtained about one random variable through observing the other random variable. Given two random variables X and Y, their mutual information $I(X;Y)$ is defined in terms of their probability density functions $p(x)$, $p(y)$ and $p(x,y)$ for $x \in X$ and $y \in Y$:

$$I(X;Y) = \int \int \frac{p(x,y)log(p(x,y))}{p(x)p(y)} dxdy \tag{2.1}$$

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} \frac{p(x,y)log(p(x,y))}{p(x)p(y)} \tag{2.2}$$

where 2.1 and 2.2 are for continuous and discrete data, respectively. In the case of discrete x and y, it is easy to calculate $I(x,y)$. However, when at least one of the variables is continuous, it becomes difficult to compute their mutual information. To overcome this problem, a data discretisation method needs to be incorporated in the process. A density estimation method such as Parzen window [97] (e.g. kernel density estimation) is one of the commonly used alternatives to estimate mutual information. Parzen window is a non-parametric way for estimating the probability density function of a random variable.

2.7.1 Mutual information estimation

Margolin et al. [83] proposed an mutual information estimator using Gaussian Parzen window and copula-transformation method which is employed for mutual information estimation in this thesis. Parzen method requires two important definitions: window (kernel function) and window width (bandwidth).

Let R be a hypercube centred at z where the length of the edge of the hypercube is denoted by h, called bandwidth. Hence, the volume V is defined as $V = h^2$ for a 2-dimensional square, and $V = h^3$ for a 3-dimensional cube and so on for n-dimensional space. The kernel function characterises the local probability density function around each observation. There is a variety of kernel functions but for the sake of speed, a computationally efficient Gaussian kernel function [11] is used. Given a set of two-dimensional samples, $\vec{z}_i \equiv \{x_i, y_i\}, i = 1, ..., n$. Let $G(.)$ denote the kernel function with bandwidth h so that:

$$G(\frac{\vec{z} - \vec{z}_i}{h}) = \begin{cases} 1 & \frac{|\vec{z}-\vec{z}_i|}{h} \leq \frac{1}{2}, \\ 0 & otherwise. \end{cases} \tag{2.3}$$

Further, k, the total number observations falling within the region R is expressed as:

$$k = \sum_{i=1}^{n} G(\frac{\vec{z} - \vec{z}_i}{h}) \tag{2.4}$$

Then the kernel density approximation of the probability density function of $\vec{z}$ is calculated as follows:

$$p(\vec{z}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h^2} G(\frac{\vec{z} - \vec{z}_i}{h}) = \frac{k}{Vn} \tag{2.5}$$

With $p(x)$ and $p(y)$ being the marginal of $p(\vec{z})$, the MI is:

$$I(X, Y) = \frac{1}{n} \sum_{i=1}^{n} log \frac{p(x_i, y_i)}{p(x_i)p(y_i)} \tag{2.6}$$

The MI is re-parameterisation invariant [83] therefore, the copula-transformation (i.e., rank-order) [60] is employed to transform x and y before MI estimation task. The range of the transformed variables is between 0 and 1, and their marginal probability distributions are uniform. This decreases the influence

of arbitrary transformations involved in data pre-processing and removes the need to consider position-dependent kernel widths, h, which might be preferable for non-uniformly distributed data [83].

2.7.2 Minimum-Redundancy Maximum-Relevance (mRMR)

Peng et al. [98] proposed information theory based relevance and redundancy criteria to determine the characteristics of a feature subset. In particular they defined the relevance of a feature subset S as:

$$Rel = \frac{1}{\mid S \mid} \sum_{x_i \in S} I(x_i; C) \tag{2.7}$$

Where $\mid S \mid$ denotes the number of features in the subset S and $I(x_i; C)$ is mutual information between target class C and the ith variable in feature subset S. When the features are selected such that the relevance Rel is maximised, it is possible to have a high dependency (i.e., redundancy) amongst the selected features. Given two highly dependent features, removing one of them from the set S would not change the class-discriminative power. Hence, the redundancy of a feature subset S is defined as:

$$Red = \frac{1}{\mid S \mid^2} \sum_{x_i, x_j \in S} I(x_i; x_j) \tag{2.8}$$

Where $I(x_i; x_j)$ indicates the mutual information between ith and jth feature in subset S. The purpose of feature selection therefore, is to find a feature subset S with N features that jointly have the largest dependency on the target class C and have the minimal redundancy amongst themselves [98]. The mRMR score of a feature set is obtained by maximising the condition in Eq. 2.7 and minimising the condition in Eq. 2.8. Optimisation of both conditions simultaneously requires combining them into a single criterion function. The

simplest combination is the difference of Eq. 2.7 and 2.8 [98]. Consequently, this leads to a bi-criteria objective which is defined as follows:

$$mRMR = max(Rel - Red) \tag{2.9}$$

2.7.3 mRMR for feature selection

Feature selection can be defined as the process of selecting the most relevant features from an initial feature set [10] which can approximately be solved by mRMR approach [98].

The capability of mRMR to select most informative features has widely been reported in many applications. Ju and He [62] developed a predictor called GlutPred for glutarylation sites prediction using mRMR and the incremental feature selection algorithm. Tint and Mikami [115] used mRMR to reduce redundant and irrelevant data for multicollinearity problem within causal factor analysis and prediction and compared the results with other two methods namely the maximum relevance (MaxRel) and the minimum redundancy (MinRed).

Chen et al. [25] proposed a prediction method to identify metabolic pathways of compounds. In their method, mRMR and incremental feature selection are employed to extract key features and the effectiveness of their method is proven in comparison with the random forest, Dagging and a method that integrates chemical-chemical interactions and chemical-chemical similarities. Ma et al. [80] presented an accurate method to predict RNA-binding proteins from amino acid sequences. In this method, they used mRMR combined with incremental feature selection to reduce the dimension of the features space and to improve the performance of the random forest classifier.

Fan et al. [40] designed a real-time static voltage stability assessment system

for large-scale power systems based on mRMR to explore the invisible association between operation variables and the voltage stability margin in bus systems. Liu et al. [76] developed a hybrid algorithm composed of mRMR and a fast classifier extreme learning machine for diagnosing erythemato-squamous diseases. In this algorithm, mRMR is employed as a feature selection tool for dimensionality reduction in order to further improve the diagnostic accuracy of the extreme learning machine classifier.

Bouzgou and Gueymard [20] proposed a framework for forecasting solar irradiance time series dataset. The proposed method is composed of two steps. In the first step, mRMR is used as the dimensionality reduction method to enhance the quality of the original time series dataset for the second step which is based on extreme learning machine classifier to forecast the outcome of the solar series representation. However, the application of the mRMR as the fitness function of the EC-based algorithms for feature selection has not been widely studied in literature and, to the best of our knowledge, is limited to one paper in which Ludwig et al. [79] developed a GA-based mRMR algorithm to predict air flow. This is mainly because computing MI between all pairs of the features in high-dimensional datasets is impractical when the number of features is very large and the Central Processing Unit (CPU) time required becomes prohibitive.

2.8 Conclusion

This chapter presents a comprehensive overview of feature selection methods and EC techniques for feature selection and revealed that the main challenges with EC feature selection are as follows.

Firstly, EC techniques suffer from the problem of being computationally ex-

pensive since they usually involve a large number of evaluations and each evaluation in a wrapper approach usually takes a relatively long time, especially when the number of instances is large. Secondly, EC techniques mostly are embedded in a wrapper approach and this causes the selected subset to be biased toward the utilised classifier.

Therefore, different classifiers will return different final subsets which makes finding a best subset of features core complex. Finally, EC techniques are random search techniques and they have random factors in their search process. Consequently, they select a different final subset every time they are run. This can pose a problem known as stability issue and requires a further subset selection process for real-world applications.

The next chapter, Chapter 3, demonstrates the issues and limitations of EC-based algorithms for feature selection using a simple GA as an EC test case algorithm and a small dataset.

Chapter 3

Problem Demonstration of Evolutionary Computation-based Algorithms for Feature Selection

3.1 Introduction

There are two important issues associated with EC-based feature selection algorithms [128]. Firstly, these algorithms require a high computation time since they usually involve a large number of evaluations [128, 28]. The second issue is that upon each run (or execution), EC-based algorithms return different feature subsets as the best solution, and finding the best solution out of all solutions returned can be a challenge. This is known as the stability issue [128, 28]. In addition to those two main issues, there is a limitation associated with evaluation metrics of EC algorithm for feature selection. EC algorithms are mostly embedded in a wrapper framework in which classification performance

of a specific classifier is employed to evaluate feature subsets and it causes the selected subset to be biased toward the specific classifier used and may results in poor performance over other classifiers [24]. This chapter demonstrates the issues and limitations when a simple EC algorithm is applied for feature selection in a sample dataset.

3.2 Methodology

This section, describes the methodology adopted to obtain computation results for demonstrating the issues and limitations when a simple EC algorithm is applied for feature selection in a sample dataset.

3.2.1 GA as a test case EC algorithm

In order to demonstrate the issues of EC algorithms for feature selection, the GA algorithm [54] is chosen as a test case EC algorithm. The reason for choosing GAs as opposed to other EC algorithms, is because GAs have been widely applied to feature selection problems [128]. The reason for choosing GAs as opposed to other EC algorithms, is because GAs have been widely applied to feature selection problems [128]. **However, experiments in this chapter can be easily performed for other EC algorithms.** The adopted GA is a standard version of GA (see subsection 2.3.2) with a fixed length binary representation, typical two-point crossover, and Bit Flip mutation operators. For subset evaluations, the GA is embedded in a wrapper feature selection framework for which the fitness function is the classification performance of a classifier (i.e. SVM).

3.2.2 Classifiers and validation approaches

For this experiment, three conventional machine learning classifiers are used, namely SVM, KNN, and Naïve Bayes (NB). The hyper parameters of the classifiers are experimentally tuned as follows: the number of K for the KNN classifier is set to 5, the kernel function for SVM classifier is set to linear function and data distribution for NB classifier is set to normal distribution. A 10-fold cross-validation is adopted for evaluating the performance of the algorithm in which the data samples are divided into roughly 10 equal folds and in each of 10 validation processes, one fold is taken as testing set and the remaining nine folds are used to train the learning algorithm. At the end of the k-fold validation process, a mean accuracy value is obtained for each validation set of each fold, and hence the ten values are averaged to provide overall classification performance.

3.2.3 The sample dataset

Heart disease dataset [2], a relatively small dataset, is chosen as sample dataset for this experiment which is publicly available on University California Irvine (UCI) machine learning repository [8]. Properties of this dataset are as follows: the dataset contains 44 features and 267 samples where the samples are divided into 2 classes.

3.2.4 Experimental setup

To analyse the problems of EC algorithms for feature selection task, a set of subsets of features obtained from multiple runs of an EC algorithm is needed. Therefore, the GA algorithm (see subsection 3.2.1) is independently run 20 times and at the end of each run the best subset is saved in a pool for analy-

sis. The parameters of the GA are experimentally set to the following values: population size is set to 50 individuals, mutation rate to 0.02, crossover rate to 0.8, and each run is repeated for 50 iterations to evolve the initial population.

3.2.5 Stability measure

The Average Tanimoto Index (ATI) approach proposed by Kalousis et al. [64] is used to measure the stability value of set of subsets in this chapter. Given $S = \{S_1; S_2; ...; S_\omega\}$ to be a system of ω feature subsets obtained from ω independent runs of a feature selection algorithm, similarity measure S_S between two subsets S_i and S_j is calculated using:

$$S_S(S_i, S_j) = \frac{\mid S_i \cap S_j \mid}{\mid S_i \cup S_j \mid} \tag{3.1}$$

where $\mid S_i \cap S_j \mid$ presents the number of common features in both subsets and $\mid S_i \cup S_j \mid$ stands for total number of all features in both subsets. To calculate ATI value for system S, the similarity index (Equation 3.1) is computed over all subset pairs and then is averaged using the following equation:

$$ATI(S) = \frac{2}{\omega(\omega - 1)} \sum_{i=1}^{\omega-1} \sum_{j=i+1}^{\omega} S_S(S_i, S_j) \tag{3.2}$$

The ATI value is in the range of [0,1] where 0 means high instability and 1 indicates a stable algorithm. The closer the ATI value to 1, the more stable the feature selection algorithm. Clearly, for deterministic algorithm, including exhaustive search methods and greedy search algorithms, the ATI value is equal to 1.

3.2.6 Classifier-bias analysis approach

To analyse the computation result in terms of classifier-bias of the selected subset, the approach proposed by Naghibi et al. [91] is used in which the optimum subset (the subset which achieves the highest possible accuracy for the classifier) of a classifier is used to train the other classifiers and the results are compared to the optimal classifiers' accuracies. If the obtained accuracies over other classifiers using the subset are not close enough to the classifiers' optimal accuracy, it can be concluded that the subset is biased toward a specific classifier and performs poorly over other classifiers.

3.3 Computation results and discussion

The GA is independently run 20 times over the Heart dataset [2], **with initial conditions set in 3.2.4**, using each of classifier separately as the fitness function and the results are presented in Table 3.1. In Table 3.1, the first column shows a combination of GA with a classifier as the fitness function denoted by GA + the name of the classifier, eg., GA+SVM. The other columns present computed values where ATI shows measured stability, T_total and T_Avg (in seconds) stand for total computation time for 20 runs and the average computation time over 20 runs respectively, Acc_Avg is the average accuracy of 20 final subsets obtained during 20 runs, and Acc_max and Acc_min presents the maximum and minimum accuracies of the subsets.

3.3.1 Stability analysis

The EC-based algorithms are stochastic in nature and have random factors in their searching process and this makes the algorithms unstable in terms of returned solutions in different runs. To measure the stability of GA (as a test

case EC algorithm), the stability value for a set of subsets composed of 20 final subsets obtained from different runs of the GA is calculate using Equation 3.2. The result are shown in Table 3.1. Taking a look at the ATI values calculated for each of the algorithms, this is clear that the GA has performed highly unstably for this small dataset with ATI value of 0.3. The instability of the GA for high-dimensional data can become worse as the number of features increases, and the probability of selecting totally different subsets of features in different runs will increase – which as a result will lead to poor stability performance.

Table 3.1: EC-based algorithms stability and high computation time problem demonstration results

Algorithm	ATI	T_total	T_Avg	Acc_Avg (%)	Acc_max (%)	Acc_min (%)
GA+SVM	0.35	1858.5	92.92	79.0	81.7	76.4
GA+NB	0.34	1720.3	86.02	66.8	70.4	62.2
GA+KNN	0.32	1199.7	59.98	74.3	80.2	70.8

3.3.2 Computation time analysis

For a wrapper approach feature selection algorithm, the required computation time highly depends on the classifier used as fitness function because each classifier uses different strategy to classify the data. For this experiment, three classifiers are used to: firstly demonstrate how different classifiers affect the computation time required by EC algorithms for feature selection; and secondly to analyse the computation time required by the GA as an EC test case algorithm embedded in a wrapper framework.

As can be seen in Table 3.1, amongst these three classifiers, the SVM classifier has required the highest computation time, and KNN has required the lowest. KNN is the simplest classifier amongst those three as it classifies a sample by considering the majority vote of its neighbours' classes, and SVM is the most

sophisticated one as uses a mathematical model to map the data into a higher dimension and uses hyper-planes to separate different classes. Therefore, these results imply that more sophisticated classifiers may lead to a higher computation time when used as the fitness function of a EC-based feature selection wrapper algorithm.

The average computation time over 20 runs has also been reported in Table 3.1 which shows computation time per run. As can be seen, KNN classifier has required the least computation time of 60 seconds however, this computation time for such a small dataset implies that an EC-based wrapper feature selection algorithm may become prohibitive for high-dimensional data because the computation time exponentially increases as the size of the data becomes higher.

3.3.3 Classifier-bias analysis

EC-based algorithms embedded into a wrapper framework provide a final selected subset which may be classifier-biased and consequently the selected subset is only trainable on a specific classifier employed for subset evaluation during the feature selection process. If the selected subset is used to train the other classifiers, it may lead to a poor performance since the features were selected based on the performance of a different classifier (which was used to initially build the wrapper framework).

To analyse classifier-bias problem, the approach proposed by Naghibi et al. [91] explained in subsection 3.2.6 is used. For this analysis, the classifiers' optimal accuracies over Heart dataset [2] are needed. Therefore the highest obtained accuracy for each classifier is considered as optimal accuracy for the classifier (Column Acc_max in Table 3.1). For each classifier, its optimal accuracy is used to train the other two classifiers and the results are shown in Table 3.2.

Table 3.2: Results for demonstrating EC-based algorithm Classifier-bias issue

Classifier		SVM (%)	NB (%)	KNN (%)
Subset	SVM_Opt	**81.7**	63.9 (9.3)	75 (6.5)
	NB_Opt	76.3 (6.6)	**70.4**	73.8 (8.0)
	KNN_Opt	75.7 (7.3)	65.9 (6.4)	**80.2**

In Table 3.2, each cell value presents the classification accuracy obtained when the optimal subset of one classifier is used to train the other classifiers, the values in bold show the optimal accuracy of the classifiers, and the values parenthesis present the error rate percentage compared to the classifier optimal accuracy. For example, value 63.9 (9.3%) indicates that when the optimal subset of the SVM classifier was used to train the NB classifier, a classification accuracy of 63.9% has been obtained and the percentage error rate of 9.3% has been obtained compared to NB optimal accuracy (70.4%).

The computation results show that the percentage error rates for such small dataset have been relatively high and consequently the subsets are biased toward the specific classifier and therefore, if the algorithm is used for selecting the features of a high-dimensional data, the error rate may increase even further.

3.4 Conclusion

EC algorithms are powerful search algorithms for combinatorial optimisation problems because of they benefit from a global search strategy and a heuristics search guidelines. However, the application of these algorithms for feature selection has been limited mainly because of issues associated with their stochastic nature.

Firstly, EC algorithms are random search algorithms meaning that they have random factors involved during their search procedure and this leads them

to select different subsets of features whenever they are run. This problem which is known as stability issue, and it is a significant problem particularly for applications that a specific set of features is sought after to implement machine learning models. Furthermore, having different final subsets raises the problem of finding the optimal subset from the various subsets. Secondly, EC-based algorithms require a high computation time for feature selection because they are iterative searching methods, meaning that they iteratively discover the searching space to evolve an initial population. Finally, embedding EC-based algorithms into a wrapper framework may lead to the selection of final subsets which are biased toward the classifier used for subset evaluation. A classifier-biased subset may only perform well on one specific classifier in terms of classification performance but shows a poor performance if used to train the other classifiers.

To clearly demonstrate the issues associated with the application of EC algorithm for feature selection to readers, this chapter employed a simple GA algorithm as the test case EC algorithm to solve Heart dataset [2], which is a relatively a small sample dataset and performed series of analyses to observe the issues. The analyses were performed in term of stability issue, computation time, and classifier-bias.

The computation results confirmed that the employed GA required high computation time, performed relatively unstable, and has selected classifier-biased subsets for this small subsets. This suggests that the algorithm may be impractical in terms of computation time or may perform weakly in terms of stability and classifier-bias for datasets with larger size and particularly high-dimensional-data. The experiments in this chapter can be performed for other EC algorithms and similar results can expected as stability and computation time are common issues amongst EC algorithms [128] and embedding EC al-

gorithms into a wrapper framework will result in the classifier-bias issue [91]. The following chapter, Chapter 4 proposes a solution to deal with the high computation time of EC algorithms for feature selection as well as the classifier-bias limitation of existing filter/wrapper methods. A novel EC-based feature selection algorithm is developed and embedded into a new two-stage hybrid framework called filter/filter approach.

Chapter 4

TAGA: Tabu Asexual Genetic Algorithm Embedded in a Filter/Filter Feature Selection Approach for High-dimensional Data

4.1 Introduction

Feature selection is the process of selecting an optimal subset of features required for building, maintaining or improving the performance of machine learning models. Recently, hybrid filter/wrapper feature selection methods have shown promising results for high-dimensional data. However, the selected feature subset by a filter/wrapper method is only optimal for the particular classifier used (classifier-biased), and may show poor generalisation performance over other classifiers. A subset which is not classifier-biased is trainable

over different classifiers without having to repeat the feature selection process. To address the classifier-bias problem of typical filter/wrapper methods, this chapter proposes **a novel EC-based filter feature selection algorithm** which is sequentially hybridised with Fisher score filter algorithm (the result of the Fisher score algorithm in the first stage is fed to the EC-based algorithm in the second stage for further processing) in a new hybrid framework called filter/filter (see Figure 4.1). The proposed algorithm is based on a long-term memory Tabu Search combined with a mutation-based Genetic Algorithm (TAGA). TAGA benefits from a new integer-coded solution representation, a novel mutation operator, and a new Tabu List encoding scheme and uses a maximum relevance minimum redundancy information theory-based criterion as fitness function. Experiments were carried out on various high-dimensional datasets including image data, text data, and biological data. The goodness of the selected subsets were evaluated using different classifiers to develop a goal-independent evaluation. The experimental results demonstrate that the proposed algorithm outperforms other feature selection algorithms in most datasets.

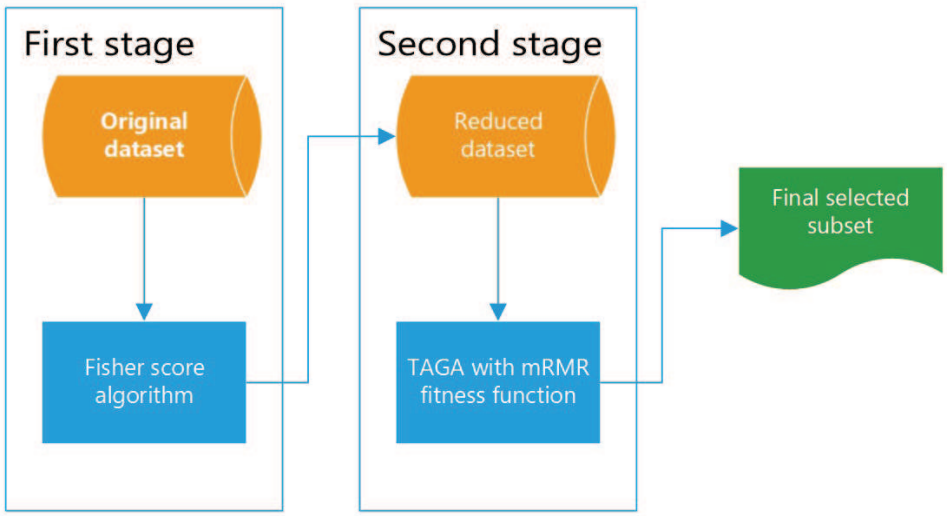

Figure 4.1: Diagram of TAGA embedded into filter/filter framework

4.2 TAGA components

This section presents the proposed Tabu Asexual Genetic Algorithm (TAGA) and its components.

4.2.1 Solution representation

As explained in subsection 2.3.2, mRMR is a highly size-dependent value and must be compared for the subset with the same number of features. In this chapter, an integer-coded representation is proposed which enables TAGA to produce and compare subsets with the same size as follows. Given the dataset D with N features and subset cardinality P ($1 \leq P < N$), each feature is assigned a unique ID from 1 to N. To produce solutions, random vectors composed of all N IDs are generated and P first features of each vector are selected as the selected subset. Figure 4.2 presents one possible solution for a dataset composed of 10 features to select 4 features, where the numbers in grey correspond to IDs of selected features. Then, the selected subset is input into the fitness function.

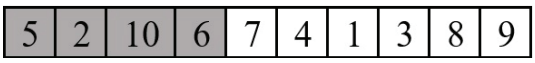

Figure 4.2: Solution representation used in TAGA

4.2.2 Proposed heuristic mutation operator

Mutation alters one or more gene values in a chromosome from its initial state. Mutation operators are used to maintain diversity in the population and help the population to escape from poor local optima. Hence, a GA can obtain better solutions using a mutation operator. A mutation method for combinatorial problems is the swap mutation in which two randomly selected genes of the solution are swapped [9]. Diversification in the feature selection

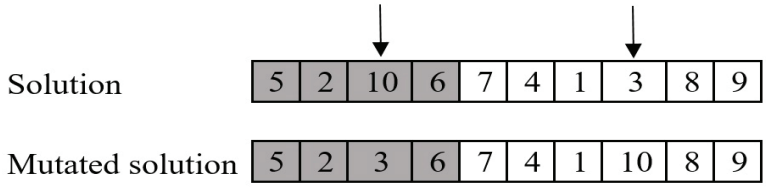

Figure 4.3: Representation of the proposed heuristic mutation operator used in TAGA

means to explore the regions of subsets which have not been discovered yet. Therefore, a swap mutation for the proposed solution representation is designed in such way that a feature from unselected features of the solution is swapped with one from selected features. Figure 4.3 indicates this procedure for the solution example in Figure 4.2. Two approaches are considered for swapping features in the candidate solution. In the first approach, one feature from the selected part and another feature from the unselected part of the solution are randomly chosen and the two features are swapped. The second approach is to choose the feature with the lowest mutual information value between the feature and the target from the selected part of the solution, and swap it with the feature with the highest mutual information value between the feature and target from the unselected part. This help to the mutual information between the target and features to be utilised for swapping. This mutation operator has been explained in Algorithm 1.

It should be mentioned that experiments were carried out with a two-point crossover operator integrated within TAGA in which both points were randomly selected from selected part of candidate solutions. However, due to poor performance of the crossover, utilisation of a crossover operator was ignored and only mutation was used.

Algorithm 1: Pseudocode of the proposed heuristic mutation operator

```
1  begin
2      With equal probability, randomly choose one of the swap criteria
         (Random or mutual information-based)
3      if mutual information-based criterion is chosen then
4          Find the feature with lowest mutual information with the target from
             selected features
5          Find the feature with highest mutual information with the target
             from unselected features
6          Swap the features
7      end
8      if random criterion is chosen then
9          Randomly select a feature from selected features
10         Randomly select a feature from unselected features
11         Swap the features
12     end
13 end
```

4.2.3 Tabu list design

The TL represents the memory structure by which TS prevents the search procedure from possible cycling and trapping into local optima. The TL consists of a list of previous solutions that must be avoided or the list of forbidden moves. There are three main considerations which should be taken into account while designing a TL, and these are: the length, the data storing strategy, and the encoding scheme.

List length

The length of TL specifies the maximum number of moves that can be stored in the TL, which is basically determined by memory type of the TS. A long-term memory TS is proposed, and thus the length of the TL is set to indefinite and therefore, all the moves are stored.

Data structure strategy

In TS, new solutions are generated by applying a move mechanism on the current solutions. A move can be defined as replacing the nodes of the current solution with other nodes in its neighbour solutions (e.g. the solutions which are the similar to the original solution with minor changes) or swapping the nodes of the current solution with each other. In order to prevent the algorithm from returning to previously visited solutions, these moves need to be recorded in the TL in an efficient way. The most commonly used TL data structure for combinatorial optimisation problem is to store a partial range of the new solution [100]. For this reason, when a node or set of adjacent nodes s of the current solution S are swapped or replaced with another node or set of nodes, the set s is stored in the TL. Unlike combinatorial problems, such as scheduling and vehicle routing, for which the order of the variables (nodes) in the sequence significantly influences the fitness of the solution, the order of features is not important for feature selection problems and thus different combinations of the same feature IDs are still the same subsets and achieve the same fitness value when passed to the fitness function. Therefore, storing a partial range of solutions in the TL is not a very effective approach for the proposed solution representation. For this reason, complete visited solutions, in this case the selected feature IDs (grey part of Figure 4.2), are stored.

The encoding scheme

A new long term-memory TS with a new encoding scheme is proposed, to overcome the existing limitations of the long-term memory TSs. The proposed TL encoding scheme is implemented as follows. The selected feature IDs are separated from the original solution and are sorted in ascending order. In feature selection problems, the order of the features is not important and

therefore, sorting feature IDs in the subsets facilitates identifying tabus when the TL becomes very large. Next, the ordered subset is transformed into string format in such a way that the feature IDs are placed one after another. For instance, the encoded tabu solution for the example in Figure 4.2 is '25610' (note that the quotation mark indicates that the solution is represented in string format).

Transforming subsets into string format is beneficial as it decreases the dimension of the TL. Given that the subset cardinality is m and TL length is l, storing the subsets as the vectors of feature IDs requires $m \times l$ checks to determine whether a subset is tabu. However, the checks will decrease to l for the proposed string encoding scheme as each encoded tabu solution is a single set of characters. The performance of the proposed TL is analysed in subsection 4.4.1.

4.2.4 Framework of TAGA for feature selection

Algorithm 3 provides the pseudocode of the proposed two stage filter/filter approach composed of Fisher score algorithm in the first stage and TAGA in the second stage. In the first stage of the algorithm (line 1), the Fisher score feature ranking algorithm is applied to dataset D and N_f elite features are selected to form the reduced dataset. The main idea of Fisher score is to construct a subset of features such that in the data space spanned by the features in the subset, the distances between samples from different classes are as large as possible, while the distances between samples from the same class are as small as possible [49]. In particular, when m features are selected, the original data matrix $X \in R^{(d \times n)}$ will be represented by $Z \in R^{(m \times n)}$ Then, the Fisher score is computed as follows:

$$f(Z) = \frac{tr(S_b)}{tr(S_t)} \tag{4.1}$$

where $tr(.)$ denotes the trace of a matrix; S_b is the between-class scatter matrix; and S_t is the within-class scatter matrix, which is defined as:

$$S_b = \sum_{k=1}^{c} n_k(\mu_k - \mu)(\mu_k - \mu)^T \tag{4.2}$$

$$S_t = \sum_{i=1}^{n} (z_i - \mu)(z_i - \mu)^T \tag{4.3}$$

where μ_k and n_k are the mean and sample number of the kth class, respectively, in the reduced data space and $\mu = \sum_{k=1}^{c} n_k \mu_k$ is the overall mean vector of the reduced data. The number of candidate subsets is combination of $\binom{m}{d}$ so the optimal feature subset selection problem can be solved by combination optimisation, but this is highly challenging and computationally prohibitive for high dimensional data [49]. To reduce the difficulty, a heuristic strategy is often used to calculate a score for each feature independently using some criterion [49]. Specifically, let μ_k^j and σ_k^j be the mean and deviation of samples from the kth class, corresponding to the jth feature. Let μ^j and σ^j denote the mean and deviation of the entire samples in the dataset corresponding to the jth feature. Then, the Fisher score of the jth feature is calculated as follows:

$$f(j) = \frac{\sum_{k=1}^{c} n_k(\mu_k^j - \mu^j)^2}{\sum_{k=1}^{c} n_k(\sigma^j)^2} \tag{4.4}$$

where $(\sigma^j)^2 = \sum_{k=1}^{c} n_k(\sigma_k^j)^2$. After obtaining the the Fisher score for all features, m first features with highest scores are selected to construct the final reduced feature subset. This procedure is shown in Algorithm 2.

Fisher score is one of the most widely used supervised feature selection methods

Algorithm 2: Pseudocode of Fisher score algorithm

```
1 begin
2     for all features do
3         Calculate mean and deviation of the samples from each class
4         Calculate mean and deviation of all samples
5         Calculate Fisher score using eq. 4.4
6     end
7     Arrange features in descending order based on their Fisher score
8     Select m first features
9 end
```

[49] and its performance and robustness to data containing noisy features for different applications have widely been discussed in the literature [82, 93, 127]. However, it selects each feature independently according to their scores under the Fisher criterion [49] regardless of interaction between the features. This leads to select a suboptimal subset of features which may perform weakly for some datasets. Therefore, for some specific datasets, the Fisher score algorithm can be replaced with other ranking algorithms for a better performance.

In the second stage, the reduced dataset from the first stage is fed to TAGA algorithm. Let D be an $m \times n$ case-by-dimension dataset where m is the total number of records and n is the total number of features. Let N_f be the number of features passed through the first stage to the second stage, C be the range of the subset cardinalities (size) from 1 to c to be explored, Pop_{size} be the population size, and μ_r be the mutation rate. Also, Let $MI_{xy} = \emptyset$ be a $1 \times n$ matrix and $MI_{xx} = \emptyset$ be an $n \times n$ matrix containing mutual information between the features and the target and pair of features, respectively. As TAGA is a random search algorithm and explore limited regions of the search space, it is very possible that some pairs of mutual information between the feature never are used during the search process. Therefore, the mutual information matrices are initialised with empty values and wherever needed these values

are calculated and stored in the matrices. This will help to reduce computation time specially for high dimensional data feature selection rather than calculating all mutual information values in advance.

Before feeding the reduced dataset to the second stage, the initial population is generated (line 2). The initial population, Pop, are in fact random vectors composed of integer numbers from one to N_f (total number of features in the reduced dataset). In the second stage (lines 3-5), a range of subset cardinalities (number of selected features to be explored) from 1 feature to c features is explored to find the best subset for each cardinality. For this, the selected part for individuals (the grey part in Figure 4.2) of the population is set to c, the size of subset cardinality (e.g. the number of features to be selected). Then, the number of individuals to be mutated is calculated as follows:

$$N_{mut} = (\mu_r \times Pop_{size} \times N_{sel})/2 \tag{4.5}$$

In the next step, the N_{sel} first features of each individual are specified to be the selected features (line 6). Line 7 calculates the fitness of the individuals based on their selected part using Equation (2.9). In lines 8 and 9, the necessary updates for the MI_{xy} and MI_{xx} matrices and the TL are performed. Then, the proposed mutation operator (Algorithm 1) is executed N_{mut} times to generate new solutions. The new solutions which are not in the TL are evaluated using the fitness function and replaced with least-fit individual in population. The MI_{xy} and MI_{xx} matrices and the TL are also updated whenever necessary (lines 11-26). This process continues until the stop criterion is reached. At the end, the best subset for the cardinality is saved for further evaluations through the classifiers (lines 27-29).

The population that evolved in the search process of the previous cardinality

is not discarded, but it is preserved for the next cardinality exploration. The next cardinality contains one more feature than the previous cardinality. Before the algorithm starts searching for the next subset cardinality, a SFS is applied on the selected part (see Figure 4.2.1) of each individual in evolved population to find the next suitable feature (line 30). This procedure helps the algorithm to obtain high quality solutions for searching the next cardinality which accelerates the convergence to an optimal solution (possibly the global one). In search process of the SFS, the features cannot be replaced with higher quality features once they are selected. However, in the proposed TAGA, none of the features are guaranteed to remain in the final subset because they might be replaced with higher quality features during the search process.

4.3 Experimental design

4.3.1 Benchmark methods

Commonly, the goodness of the feature subsets is evaluated through the performance of one specific classifier known as goal-dependent evaluation [91]. The goal-dependant evaluation, however, cannot evaluate if the subsets are classifier-biased. The aim of this experiment is to develop a goal-independent evaluation proposed by Naghibi et al. [91], which is to compare the performance of the feature selection algorithms over several datasets using multiple classifiers. Therefore, the goodness of the selected subsets are evaluated using 5 classifiers, namely SVM, KNN, CART, NB, and Linear Discriminant Analysis (LDA). The performance of the proposed TAGA is compared with the following greedy search and mMRM-based algorithms:

- Sequential forward selection (SFS): starts from an empty set and se-

Algorithm 3: Pseudocode of TAGA

input : D labelled dataset, N_f Number of filter stage features, C Cardinality range, Pop_{size} Population size, μ_r Mutation rate, $MI_{xy} = \emptyset$ Mutual information Matrix between features and target, $MI_{xx} = \emptyset$ Mutual information matrix between pairs of features

output: The best feature sets for cardinalities

```
Apply filter algorithm Select N_f first features
Generate Pop_size random vectors including 1 to N_f as initial population
for each cardinality in C do
    Set N_sel to size of cardinality number of features to be selected Set
      N_mut=(μ_r × Pop_size × N_sel) ÷ 2 number of genes to be mutated
    for each individual in population do
        Set the first N_sel features as the selected features
        Calculate the fitness of each individual based on selected features
          using mRMR Equation 2.9
        Update MI_xy and MI_xx matrices
        Update Tabu List
    end
    while stop criterion do
        for N_mut do
            Randomly select a solution from population with a tendency to
              fitter individuals
            Mutate the solution and generate two new solutions Algorithm 1
            for each new solution do
                if the new solution is not Tabu listed then
                    Calculate the fitness using mRMR Equation 2.9 Update
                      MI_xy and MI_xx matrices
                    Update Tabu List
                    Add the solution into new solution pool
                else
                    Dispose the solution
                end
            end
        end
        Replace least-fit individuals in the population with fitter solutions in
          new solution pool
    end
    Sort individuals in population in descending order based on their fitness
    Find the best individual
    Save selected features part of the best individual
    Apply SFS on population to find the next suitable feature for each
      individual
end
```

Table 4.1: Description of the datasets used in the experiments

Dataset Name	Colon Cancer	GLI_85	NCI9	SMK_CAN_187	TOX_171
Abrreviation	CLN	GLI	NCI	SMK	TOX
Type	Biological	Biological	Biological	Biological	Biological
# Features	2000	22283	9712	19993	5748
# Instances	62	85	60	187	171
# Classes	2	2	9	2	4
Dataset Name	Lymphoma	DBWorld e-mails	Dexter	Orlraws10P	Pixraw10P
Abrreviation	LYM	DBE	DEX	ORP	PIW
Type	Biological	Text	Text	Image	Image
# Features	4026	4702	20000	10304	10000
# Instances	96	64	300	100	100
# Classes	9	2	2	10	10

Table 4.2: Parameters settings of the EC algorithms

Parameter	TAGA	CGA
Population size	100	300
μ_r	0.03	-
μ_c	-	0.8
Stop Criterion	$500\times P$	$500\times P$
Range of cardinalities C	[1 50]	[1 50]
Tabu List length	Infinite	-

μ_c and μ_r are crossover and mutation rates respectively, and P is the size of the cardinality.

quentially adds the feature that maximises the objective function when combined with the other features in the set that have already been selected.

- Backward Elimination (BE): unlike SFS, BE starts from the full set of features and sequentially removes one feature so that the remaining features in the set maximise the objective function.

- ReliefF algorithm [69]: is the multi-class version of the original Relief algorithm [66] which scores the features based on the identification of feature value differences between nearest neighbour instance pairs. The feature scores are ranked and the top scoring features are selected.

- Fisher score algorithm: score the features such that the distances be-

tween samples from different classes become as large as possible, while the distances between samples from the same class become as small as possible [49].

- mRMR-mid [98]: is an optimal first-order incremental feature selection algorithm. It is a two-stage feature selection algorithm combining mRMR and other more sophisticated feature selection algorithms (e.g. wrappers).

- Quadratic Programming-based Feature Selection (QPFS)[103]: is based on optimising a quadratic function which is reformulated in a lower-dimensional space using the Nyström approximation method.

- Spectral relaxation Conditional Mutual Information (SPECCMI) [92]: is a global MI-based feature selection algorithm in which the quadratic optimisation problem is formulated based on the conditional mutual information and information theoretic relevancy and redundancy and it is solved via spectral relaxation.

- An integer-encoded version of GA customised for feature selection (denoted as CGA in the text for comparison purpose) [79]: search for the best subset of features within a range of subset cardinalities using the mRMR criterion. Unlike most of the GAs in the context of feature selection which use a binary solution representation, CGA has an integer-encoded solution representation.

4.3.2 Datasets and parameter settings

Table 4.1 shows the properties of the 10 datasets used in the experiments. All the datasets are available on the Arizona State University (ASU) feature

selection repository [71], except the Dexter [70] and DBWorld e-mails [41] datasets which are available on the UCI machine learning repository [8]. The datasets have been widely used in previous feature selection studies and include image data, text data, and biological data. One of the most important issues associated with EC-based feature selection algorithms for large-scale problems [128] is that these algorithms require a high computational cost since they usually involve a large number of evaluations. To resolve the computational cost issue, a popular approach is to employ a filtering stage to select elite features as the inputs of the EC algorithms [128]. Therefore, the Fisher score algorithm, which is computationally cost effective, is applied to reduce the number of input features.

The Fisher score algorithm is set to select top 100 features for all the datasets except for DBE and DEX for which 500 top features are selected (first step of TAGA, line 3 of Algorithm 3) and then, new reduced datasets are generated using those elite features. For a fair comparison, Fisher score was embedded in the filter/filter approach in the same way as other competing algorithms in this chapter. To assess how the results of feature selection algorithms will generalise to an independent dataset, the leave-one-out cross-validation (LOOCV) was adopted. LOOCV is suitable for assessing model performance in small sample size datasets when taking into consideration model bias and estimation variance [67, 89].

The common approach for finding the optimal subset cardinality when mRMR is used as metric criterion, which is to search for the best subsets over a range of cardinalities from 1 to a user-defined value [34, 79, 98, 91] is followed herein. The subset cardinalities ranged from 1 to 50, and this range was obtained experimentally.

Amongst the algorithms used in the experiments, TAGA and CGA are EC-

based algorithms and their parameter settings are summarised in Table 4.2. The other algorithms are deterministic, and besides a predefined cardinality number (i.e. number of desired features) they do not require any other parameter settings. The stop criterion for TAGA and CGA is defined when specific numbers of function evaluation are counted for which the algorithms perform an equal number of function evaluations. As the algorithms search for the best subsets within a range of cardinalities, the stop criteria depends on the size of cardinality with a constant weight which is experimentally set to 500 ($500\times P$). Since TAGA and CGA perform stochastic decision, it is possible to obtain different feature subsets over independent runs with different classification performances. Therefore, the algorithms are independently run 30 times for each dataset. In each run, the range of subset cardinalities is explored and the best subset found for each subset cardinality is saved and sent to 5 classifiers to gain prediction accuracy. According to these accuracies, the best subset for each classifier is obtained. At the end, the average of the accuracies of the best subsets over 30 runs is calculated for each classifier. All the experiments are carried in MATLAB 2017 on a Lenovo Thinkpad P50 laptop with Intel Core i7, 2.6 Ghz processor and 64 GB of RAM.

4.4 Results and discussion

4.4.1 Results of TAGA components

To examine the contribution of the proposed components, two experiments are performed. In the first experiment, the effectiveness of the proposed TL (see Section 4.2.3) is analysed. In the second experiment, the performance of the proposed heuristic mutation operator (see Section 4.2.2) is examined to understand its effect on the search process. For this, two variations of

TAGA are further defined, i.e., TAGA$_{NoTabu}$ and TAGA$_{NoHeuristic}$. The former variation of TAGA omits the TL, but uses the proposed heuristic mutation operator. The latter variation of TAGA uses the TL but the proposed heuristic mutation operator is replaced with a simple swap mutation operator.

Effectiveness of the tabu list

What makes the proposed TL different from other TLs in the literature is its string encoding scheme to store the tabu solutions. Therefore, the effectiveness of the proposed TL must be learnt from two aspects: effectiveness of the proposed string encoding scheme (see Section 4.2.3) in identifying already visited solutions, and the effect of the proposed TL on the search process as a part of TAGA. For this reason, two sub-experiments are designed as follows. In the first sub-experiment, Algorithm 4 is used to compare the performance of TL using proposed encoding scheme against when the solutions are simply stored as the vectors of feature IDs in terms of the number of correctly tabued solutions, running time, and the occupied memory space (see Table 4.3). In this algorithm, an initial solution is generated using the proposed solution representation and then, a specific number of its neighbour solutions is generated. Next, each solution is checked to determine whether the solution has been previously visited. The pseudocode of this procedure is outlined in Algorithm 4.

Suppose a dataset with 100 features, the initial solution is generated in such a way that 20 feature IDs are randomly selected out of 100 and then 20000 neighbour solutions are generated using different local search methods. Clearly, the same neighbour solutions are used for both methods. The results are presented in Table 4.3, where the second column stands for the number of unique solutions (#UnqSol) generated, the third column shows the number of

Algorithm 4: Pseudocode of Tabu List performance analysis

```
1  begin
2      Generate initial solution S_0
3      Generate n neighbours of S_0
4      Set TL to ∅
5      for all neighbour solutions do
6          Arrange the IDs in ascending order
7          if the solution is not in TL then
8              Add ordered IDs to TL
9          else
10             Ignore the solution
11         end
12     end
13     Count the number of previously visited solutions
14     Calculate the time
15 end
```

Table 4.3: Results of the Tabu List performance analysis

Method	#UnqSol	#RepSol	Time (s)	Space (MB)
Vector of IDs	16048	3952	31.5	2.6
Encoding scheme	16048	3952	3.2	3.0

solutions previously visited (#RepSol), the fourth column indicates running time in seconds, and the last column is the memory space occupied by TLs in megabytes. As shown in Table 4.3, the proposed encoding scheme has correctly identified all tabu solutions, has occupied comparable memory space, and has performed the job almost 10 times faster.

In the second sub-experiment, the effect of the TL is analysed when it is used as a component of TAGA. For doing this, both TAGA and its TAGA$_{NoTabu}$ variation are implemented over 10 datasets to obtain the classification accuracy for five classifiers. Table 4.4 presents the results when the algorithms are run 30 times and the best accuracies found over subset cardinality range from 1 to 50 are averaged for each classifier. The Wilcoxon post-hoc pairwise analysis is then applied on the average accuracies (last column of Table 4.4) to find out whether the results are significantly different. Table 4.5 presents the adjusted

ρ-values for the Wilcoxon test. As can be seen in Table 4.5, TAGA shows superiority over TAGA$_{NoTabu}$ algorithm. As the only difference between TAGA and TAGA$_{NoTabu}$ is the TL element, it can be concluded that the TL has effectively guided the search process into undiscovered areas which has led to higher performance.

Table 4.4: Performance comparison of various versions of TAGA over reduced datasets

Classifier	SVM (%)	LDA (%)	NB (%)	KNN (%)	CART (%)	Average (%)
			CLN Dataset			
TAGA	97.4±1.3 (13.9±1.0)	90.3±0.3 (8.9±4.7)	90.3±0.0 (3.9±2.1)	94.03±0.8 (10.6±8.1)	90.1±1.6 (9.3±6.4)	**92.43**
TAGA_{NoTabu}	94.0±1.9 (13.3±2.1)	90.0±0.7 (8.2±5.5)	90.3±0.0 (5.3±1.1)	93.9±1.3 (5.6±4.6)	87.3±1.8 (9.8±6.3)	91.1
$\text{TAGA}_{NoHeuristic}$	89.2±1.1 (12.0±5.8)	90.2±0.5 (8.9±3.9)	90.3±0.0 (6.8±2.9)	93.4±1.4 (6.1±3.3)	86.8±1.8 (7.4±5.7)	89.97
			GLI Dataset			
TAGA	100.0±0.0 (10.6±0.9)	98.8±0.0 (10.3±0.9)	98.2±0.8 (10.1±1.6)	98.8±0.0 (14.8±4.3)	92.8±0.8 (14.4±11.9)	**97.73**
TAGA_{NoTabu}	99.3±1.0 (10.6±2.5)	96.5±0.6 (6.3±3.2)	97.4±0.9 (7.2±3.2)	93.2±0.7 (9.3±4.0)	92.1±1.0 (5.9±5.7)	95.69
$\text{TAGA}_{NoHeuristic}$	95.8±2.0 (6.9±1.9)	96.8±1.2 (9.0±2.3)	96.9±0.6 (7.9±2.7)	95.8±0.6 (19.1±5.6)	91.5±1.6 (15.6±10.3)	95.36
			NCI Dataset			
TAGA	84.2±1.4 (34.1±2.2)	78.1±0.5 (35.8±5.6)	83.5±0.6 (37.9±3.7)	79.5±0.4 (40.5±3.5)	60.1±1.3 (22.3±13.1)	**77.08**
TAGA_{NoTabu}	80.3±2.6 (35.5±6.5)	75.8±2.0 (28.5±7.3)	83.0±1.1 (42.2±7.4)	78.2±1.5 (43.1±3.5)	59.5±2.7 (26.1±16.3)	75.37
$\text{TAGA}_{NoHeuristic}$	78.5±2.8 (31.7±10.0)	76.3±1.5 (33.2±5.5)	81.8±0.5 (38.2±7.6)	77.5±1.2 (42.7±6.6)	57.5±2.9 (22.9±14.5)	74.33
			SMK Dataset			
TAGA	81.9±2.0 (19.2±4.9)	79.7±0.4 (15.1±5.0)	79.6±0.7 (9.8±5.2)	75.2±0.5 (13.1±4.4)	77.8±1.1 (16.4±4.0)	**78.88**
TAGA_{NoTabu}	80.2±0.8 (13.8±5.1)	79.5±0.7 (12.8±5.1)	78.4±0.4 (12.1±8.0)	74.8±1.1 (14.7±4.0)	72.7±1.1 (12.5±5.1)	77.12
$\text{TAGA}_{NoHeuristic}$	79.6±0.7 (12.2±3.3)	78.9±0.6 (12.6±3.1)	79.0±0.6 (8.6±6.2)	74.3±0.9 (14.3±1.8)	72.7±0.5 (13.4±5.7)	76.89
			TOX Dataset			
TAGA	82.4±0.7 (30.7±3.0)	81.6±1.0 (27.1±1.5)	74.9±0.8 (21.5±2.8)	77.7±0.8 (23.6±6.5)	68.4±0.9 (19.2±8.8)	**77.00**
TAGA_{NoTabu}	83.0±1.4 (27.2±5.8)	81.2±1.2 (27.1±2.2)	73.6±1.4 (22.4±8.5)	72.7±0.6 (27.5±10.8)	63.0±1.8 (19.8±7.9)	74.7
$\text{TAGA}_{NoHeuristic}$	81.6±1.1 (25.0±5.1)	80.1±1.0 (27.5±0.8)	72.9±0.9 (18.6±5.9)	70.8±1.0 (19.9±7.1)	63.9±2.2 (15.5±8.0)	73.85
			LYM Dataset			
TAGA	96.7±0.4 (33.5±1.5)	96.9±0.0 (40.5±0.6)	94.8±0.0 (30.6±1.8)	94.3±0.5 (20.3±3.7)	84.4±0.4 (43±2.3)	**93.42**
TAGA_{NoTabu}	96.9±0.0 (31.4±0.8)	93.0±1.0 (17.4±3.3)	91.9±0.4 (29.1±6.2)	93.8±0.0 (37.1±4.6)	81.3±2.1 (29.6±11.4)	91.35
$\text{TAGA}_{NoHeuristic}$	96.8±0.3 (31.1±1.2)	92.1±0.7 (18.2±3.8)	91.7±0.9 (30.2±4.6)	93.4±0.5 (31.7±4.3)	82.2±2.2 (15.2±13.4)	91.23
			DBE Dataset			
TAGA	90.6±0.0 (10.3±5.0)	90.3±1.0 (10.3±5.0)	89.1±0.0 (19.9±8.1)	90.6±0.4 (8.2±1.1)	90.3±0.7 (6.6±1.4)	**90.18**
TAGA_{NoTabu}	89.5±0.8 (8.5±1.8)	88.3±0.8 (10.2±5.4)	88.8±1.2 (7.0±1.6)	90.0±0.8 (6.7±1.8)	88.0±1.1 (9.6±4.4)	88.92
$\text{TAGA}_{NoHeuristic}$	88.8±1.6 (14.0±2.7)	88.4±1.5 (13.6±2.3)	88.1±1.3 (14.3±2.5)	88.8±2.2 (12.8±2.5)	87.8±1.8 (15.0±3.4)	88.38
			DEX Dataset			
TAGA	93.3±0.3 (46.7±1.5)	84.5±0.2 (39.0±2.6)	91.4±0.2 (49.3±1.6)	89.2±0.3 (37.0±6.4)	86.3±0.5 (32.7±5.7)	**88.94**
TAGA_{NoTabu}	93.2±0.4 (44.2±3.0)	83.7±0.2 (33.9±3.8)	90.8±0.3 (39.8±3.3)	88.2±0.4 (30.2±4.8)	86.0±0.4 (32.3±6.0)	88.38
$\text{TAGA}_{NoHeuristic}$	93.1±0.2 (47.1±1.8)	83.3±0.2 (33.1±1.7)	90.8±0.2 (42.3±1.8)	88.2±0.2 (31.2±5.0)	86.0±0.4 (26.7±9.1)	88.28
			ORP Dataset			
TAGA	97.4±0.5 (15.7±2.3)	92.3±1.2 (13.7±2.3)	94.7±0.5 (18.2±1.7)	99.8±0.4 (13.4±4.7)	86.3±2.1 (14.0±8.3)	**94.1**
TAGA_{NoTabu}	98.9±0.3 (13.5±1.7)	91.9±1.1 (12.2±3.4)	95.0±0.5 (18.4±1.9)	98.9±0.3 (13.8±1.0)	84.5±1.9 (13.0±8.0)	93.84
$\text{TAGA}_{NoHeuristic}$	97.2±0.4 (13.6±3.3)	91.7±1.1 (12.2±2.4)	95.0±0.0 (16.8±1.3)	99.0±0.0 (15.8±3.9)	83.7±1.8 (6.6±5.8)	93.32
			PIW Dataset			
TAGA	97.6±0.5 (6.1±1.7)	96.8±0.4 (8.0±5.4)	95.8±0.4 (15.3±6.3)	97.0±0.0 (8.1±0.7)	98.6±0.5 (9.8±4.8)	**97.16**
TAGA_{NoTabu}	96.9±0.3 (10.4±4.1)	97.0±0.5 (11.4±5.6)	96.2±0.6 (5.4±2.4)	97.8±0.4 (15.5±2.5)	97.2±1.0 (8.3±5.3)	97.02
$\text{TAGA}_{NoHeuristic}$	97.2±0.4 (5.7±3.0)	97.0±0.5 (10.2±6.6)	95.9±0.3 (14.7±5.9)	97.0±0.0 (7.9±2.0)	97.7±1.1 (10.7±6.4)	96.96

For each classifier and algorithm, the first value is the classification accuracy, the values in parenthesis show the number of selected features, and the sign ± indicates standard deviation. The last column presents the average of the classification accuracies for each algorithm.

Effectiveness of heuristic mutation

In order to evaluate the performance of the proposed heuristic mutation, it needs to be compared with other mutation operators. The proposed mutation operator uses an integer-encoded representation. However, most of the existing mutation operators in the literature are following a binary representation which cannot be compared with the proposed mutation in this chapter. Therefore, the proposed heuristic mutation operator is compared with a simple swap mutation operator in which two feature IDs are selected and swapped from any part of the solution without taking into consideration if the chosen feature IDs are part of the final subset.

A similar set of experiments are performed for $\text{TAGA}_{NoHeuristic}$ algorithm, and as in the previous experiment, the results are compared with TAGA. The accuracy results of five classifiers and the Wilcoxon post-hoc pairwise analysis are available in Tables 4.4 and 4.5 respectively. Observing the tables, it can be seen that TAGA has outperformed $\text{TAGA}_{NoHeuristic}$. In the proposed solution representation, the final subset is composed of few features of the entire features. Consequently, when a simple mutation operator is applied, it is highly possible for the two swapped features to have been chosen from unselected part of the solution representation in Figure 4.2. Therefore the outcome will be a repetitive final subset which is already in the TL.

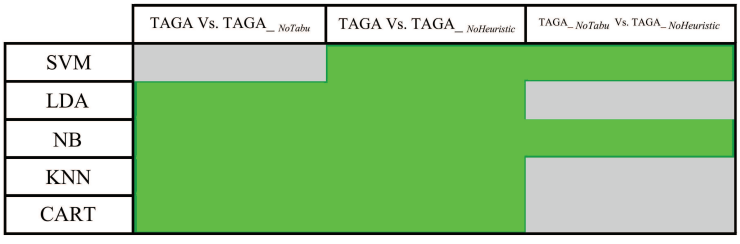

Figure 4.4: Analysing the effectiveness of the proposed components. Results of the Wilcoxon tests for each classifier. The green boxes indicate a significant difference.

Table 4.5: Mutation operators performance analysis adjusted ρ-value for Wilcoxon post-hoc pairwise comparison.

Algorithms	TAGA	TAGA_{NoTabu}	$\text{TAGA}_{NoHeuristic}$
TAGA	-	0.005	0.005
TAGA_{NoTabu}	-	-	0.005

Similarly, the same test was performed for each classifier and Figure 4.4 depicts the results. The existence of significant differences has been highlighted using the green box. In Fig. 4.4, when TAGA is compared with $\text{TAGA}_{NoHeuristic}$, TAGA has outperformed $\text{TAGA}_{NoHeuristic}$ for all classifiers. It is because the proposed mutation operator has properly designed for the feature selection problem and it is able to correctly drive the search process toward good quality solutions. Comparing TAGA with TAGA_{NoTabu}, TAGA performance has been significantly different from TAGA_{NoTabu} for most of classifiers which indicates that the designed TL has successfully avoided TAGA to explore already visited regions. Considering Table 4.5 and Figure 4.4, the results demonstrate the effectiveness of the proposed TAGA components.

4.4.2 Comparison of TAGA with greedy search algorithms

Table 4.6 shows the results obtained for 10 datasets and 5 classifiers using TAGA compared to four greedy search algorithms, namely: SFS, BE, Fisher score, and ReliefF to find the best subset over subset cardinality range from 1 to 50 features. To detect a statistically meaningful significant difference amongst the algorithms, the Friedman test is applied on the average accuracies (last column of Table 4.6).

Next, Wilcoxon post-hoc Pairwise Algorithm Comparison Analysis is carried out to determine which pairs of algorithms had significantly different perfor-

Table 4.6: Comparison of TAGA with greedy search algorithms over reduced datasets

Classifier	SVM (%)	LDA (%)	NB (%)	KNN (%)	CART (%)	Average (%)
			CLN Dataset			
TAGA	97.4±1.3 (13.9±1.0)	90.3±0.3 (8.9±4.7)	90.3±0.0 (3.9±2.1)	94.03±0.8 (10.6±8.1)	90.1±1.6 (9.3±6.4)	**92.43**
SFS	93.5 (7.0)	90.3 (10.0)	90.3 (7.0)	91.9 (2.0)	90.3 (6.0)	91.26
BE	90.3 (8.0)	90.3 (11.0)	90.3 (8.0)	93.5 (8.0)	87.1 (2.0)	90.32
Fisher	82.3 (4.0)	79.0 (4.0)	85.5 (37.0)	85.5 (24.0)	82.3 (37.0)	82.92
ReliefF	87.1 (20.0)	88.7 (14.0)	88.7 (49.0)	96.8 (45.0)	90.3 (18.0)	90.32
			GLI Dataset			
TAGA	100.0±0.0 (10.6±0.9)	98.8±0.0 (10.3±0.9)	98.2±0.8 (10.1±1.6)	98.8±0.0 (14.8±4.3)	92.8±0.8 (14.4±11.9)	**97.73**
SFS	98.8 (11.0)	98.8 (8.0)	98.8 (10.0)	98.8 (16.0)	90.6 (11.0)	97.16
BE	100.0 (9.0)	97.6 (6.0)	97.6 (9.0)	95.3 (24.0)	91.8 (10.0)	96.46
Fisher	96.5 (11.0)	94.1 (5.0)	92.9 (2.0)	89.4 (10.0)	85.9 (1.0)	91.67
ReliefF	95.3 (45.0)	90.6 (6.0)	94.1 (9.0)	92.9 (35.0)	89.4 (18.0)	92.46
			NCI Dataset			
TAGA	84.2±1.4 (34.1±2.2)	78.1±0.5 (35.8±5.6)	83.5±0.6 (37.9±3.7)	79.5±0.4 (40.5±3.5)	60.1±1.3 (22.3±13.1)	**77.08**
SFS	80.0 (31.0)	75.0 (46.0)	81.7 (49.0)	76.7 (40.0)	58.3 (40.0)	74.0
BE	73.0 (34.0)	75.0 (33.0)	81.7 (45.0)	75.0 (41.0)	58.3 (8.0)	72.6
Fisher	70.0 (27.0)	66.7 (41.0)	71.7 (25.0)	73.3 (22.0)	58.3 (6.0)	68.0
ReliefF	55.0 (23.0)	53.3 (32.0)	63.3 (44.0)	53.3 (36.0)	48.3 (23.0)	54.64
			SMK Dataset			
TAGA	81.9±2.0 (19.2±4.9)	79.7±0.4 (15.1±5.0)	79.6±0.7 (9.8±5.2)	75.2±0.5 (13.1±4.4)	77.8±1.1 (16.4±4.0)	**78.88**
SFS	79.1 (9.0)	80.2 (9.0)	78.6 (24.0)	75.9 (11.0)	71.7 (12.0)	77.1
BE	79.1 (37.0)	78.6 (16.0)	79.7 (38.0)	75.9 (13.0)	77.0 (17.0)	78.07
Fisher	80.2 (11.0)	79.1 (11.0)	78.6 (14.0)	73.3 (11.0)	72.7 (9.0)	76.78
ReliefF	77.5 (36.0)	72.7 (40.0)	77.0 (45.0)	70.1 (26.0)	62.6 (9.0)	71.98
			TOX Dataset			
TAGA	82.4±0.7 (30.7±3.0)	81.6±1.0 (27.1±1.5)	74.9±0.8 (21.5±2.8)	77.7±0.8 (23.6±6.5)	68.4±0.9 (19.2±8.8)	77.00
SFS	78.9 (20.0)	72.5 (22.0)	75.2 (17.0)	71.3 (19.0)	64.3 (7.0)	72.44
BE	78.4 (39.0)	73.1 (11.0)	73.7 (17.0)	72.5 (32.0)	64.9 (27.0)	72.51
Fisher	83.6 (35.0)	64.3 (5.0)	66.16 (5.0)	67.8 (33.0)	64.3 (15.0)	69.22
ReliefF	98.8 (45.0)	70.8 (50.0)	81.3 (48.0)	93.6 (36.0)	65.5 (48.0)	**82.0**
			LYM Dataset			
TAGA	96.7±0.4 (33.5±1.5)	96.9±0.0 (40.5±0.6)	94.8±0.0 (30.6±1.8)	94.3±0.5 (20.3±3.7)	84.4±0.4 (43±2.3)	**93.42**
SFS	94.8 (31.0)	94.8 (15.0)	93.8 (19.0)	93.8 (43.0)	82.3 (38.0)	91.9
BE	95.8 (27.0)	94.8 (31.0)	91.7 (28.0)	92.7 (31.0)	79.2 (21.0)	90.83
Fisher	91.7 (48.0)	84.4 (45.0)	83.3 (39.0)	88.5 (46.0)	82.3 (50.0)	86.04
ReliefF	94.8 (36.0)	95.8 (44.0)	95.8 (42.0)	95.8 (49.0)	79.2 (20.0)	92.28
			DBE Dataset			
TAGA	90.6±0.0 (10.3±5.0)	90.3±1.0 (10.3±5.0)	89.1±0.0 (19.9±8.1)	90.6±0.4 (8.2±1.1)	90.3±0.7 (6.6±1.4)	**90.18**
SFS	87.5 (5.0)	89.1 (28.0)	89.1 (5.0)	90.6 (10.0)	89.1 (7.0)	89.08
BE	89.1 (7.0)	89.1 (32.0)	89.1 (5.0)	89.1 (7.0)	87.5 (5.0)	88.78
Fisher	68.8 (40.0)	76.6 (44.0)	73.4 (39.0)	54.7 (1.0)	68.8 (39.0)	68.46
ReliefF	92.2 (17.0)	90.6 (12.0)	90.6 (6.0)	89.1 (6.0)	85.9 (1.0)	89.68
			DEX Dataset			
TAGA	93.3±0.3 (46.7±1.5)	84.5±0.2 (39.0±2.6)	91.4±0.2 (49.3±1.6)	89.2±0.3 (37.0±6.4)	86.3±0.5 (32.7±5.7)	**88.94**
SFS	93.0 (49.0)	83.3 (33.0)	90.7 (42.0)	88.3 (28.0)	86.3 (33.0)	88.32
BE	92.7 (43.0)	83.3 (3.0)	91.0 (48.0)	89.3 (49.0)	87.3 (32.0)	88.72
Fisher	83.7 (47.0)	75.0 (47.0)	72.0 (44.0)	76.0 (42.0)	77.3 (47.0)	76.8
ReliefF	87.0 (13.0)	84.3 (48.0)	73.3 (2.0)	85.3 (16.0)	84.7 (16.0)	82.92
			ORP Dataset			
TAGA	97.4±0.5 (15.7±2.3)	92.3±1.2 (13.7±2.3)	94.7±0.5 (18.2±1.7)	99.8±0.4 (13.4±4.7)	86.3±2.1 (14.0±8.3)	**94.1**
SFS	97.0 (14.0)	92.0 (12.0)	94.0 (16.0)	99.0 (14.0)	85.0 (15.0)	93.4
BE	97.0 (13.0)	93.0 (13.0)	95.0 (16.0)	100.0 (45.0)	83.0 (22.0)	93.6
Fisher	94.0 (45.0)	91.0 (42.0)	87.0 (49.0)	98.0 (49.0)	78.0 (35.0)	89.6
ReliefF	81.0 (50.0)	48.0 (11.0)	66.0 (40.0)	73.0 (5.0)	69.0 (5.0)	67.4
			PIW Dataset			
TAGA	97.6±0.5 (6.1±1.7)	96.8±0.4 (8.0±5.4)	95.8±0.4 (15.3±6.3)	97.0±0.0 (8.1±0.7)	98.6±0.5 (9.8±4.8)	**97.16**
SFS	97.0 (6.0)	95.0 (8.0)	95 (8.0)	97.0 (8.0)	98.0 (14.0)	96.4
BE	98.0 (6.0)	96.0 (47.0)	96.0 (21.0)	97.0 (6.0)	97.0 (19.0)	96.8
Fisher	97.0 (27.0)	97.0 (23.0)	97.0 (28.0)	97.0 (28.0)	97.0 (23.0)	97.0
ReliefF	96.0 (50.0)	85.0 (36.0)	93.0 (45.0)	94.0 (45.0)	91.0 (33.0)	91.8

For each classifier and algorithm, the first value is the classification accuracy, the values in parenthesis show the number of selected features, and the sign ± indicates standard deviation. The last column presents the average of the classification accuracies for each algorithm.

mance. Figure 4.5 depicts the results of this statistical test analysis. The adjusted ρ-values for Wilcoxon analysis have also been reported in Figure 4.5.

The smaller the ρ-value, the stronger the evidence against the null hypothesis. The proposed algorithm shows meaningful superiority over greedy search algorithms. The reason resides in the fact that greedy algorithms make locally optimal choices with the aim of finding a global optimum amongst local optima. This weakens the performance of greedy search algorithms and in many problems, they are usually unable to produce an optimal solution. The main disadvantage of SFS is that in each iteration of the algorithm, the usefulness of a single feature is examined in the limited context of the previously selected features only.

Consequently, while selecting the final subset, the interaction between limited numbers of features is considered. Contrary to SFS and BE, TAGA can evaluate the contribution of a given feature in the context of all other features. However, BE overemphasises on feature interactions which may lead to a suboptimal solution [91]. Given that Relief and Fisher score are greedy search methods, the score of each feature is computed independently and therefore, the algorithms fail to consider the interaction between the features in a subset and as a consequence, fail to remove redundant features [49]. TAGA however, is a global search algorithm which directly searches for global optima.

The embedded SFS provides TAGA with local optimal solutions which facilitate the search process however, the features in the subset provided by SFS are not guaranteed to be in the final subset as they continuously are replaced with other feature for better solutions. In addition, TAGA evaluates the subsets and not the features individually, and each subset is composed of a portion of all features which avoids the algorithms from overemphasising feature interactions.

The same set of experiments were performed for obtaining the accuracy results of each classifier and the results with the corresponding adjusted ρ-values are

reported in Figure 4.6. As shown in Figure 4.6, for most of the classifiers, the proposed TAGA has either outperformed greedy algorithms (the points higher than zero line) or has performed comparably (the points near zero line), and this is because greedy algorithms look for global optimal solutions amongst immediate available local optimal solutions. Nevertheless, they are usually unable to reach a global optimum and they are stuck into a local optimum. It also can be seen in Figure 4.6 that the Relief algorithm has performed better than TAGA for most of the classifiers when both algorithms were applied on TOX dataset. Relief is a ranking algorithm, and the results presented in Table 4.6 were obtained when applying Relief without applying Fisher score filtering. The Fisher score algorithm measures the quality of the features independently without considering the interaction between the features in the subsets. Therefore, using Fisher score to filter out elite features for some datasets may result in suboptimal subsets with poor classification performance. Hence, Fisher score can be replaced with other ranking algorithms for better results. Since the Relief algorithm has achieved better results when applied to the TOX dataset, it can be concluded that Relief can filter out higher quality features for this specific dataset. Consequently, if Fisher score is replaced by Relief in the first stage of TAGA, better results for TOX dataset can be obtained.

4.4.3 Comparison of TAGA with other feature selection algorithms

Table 4.7 shows the results obtained for ten datasets and five classifiers using TAGA in comparison with four other algorithms, mRMR-mid [98], QFPS [103], SPECCMI [92], and CGA [79].

Table 4.7 at a glance shows that overall performance of TAGA has not been

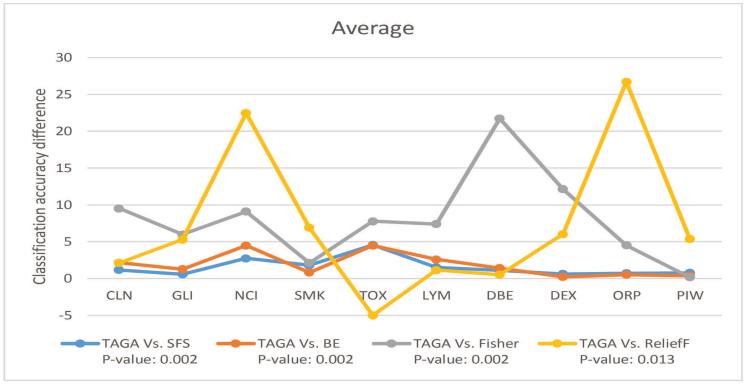

Figure 4.5: Comparing TAGA with Greedy search algorithms using the Friedman test and the Wilcoxon post-hoc analysis applied on the average of the accuracies. The y-axis is the classification accuracy difference and x-axis indicates the names of the compared algorithms.

better than the other competing algorithms when NB is the classifier, but it has performed better when the LDA is employed as the classifier. When TAGA performance over the datasets is considered, it has been the best method over GLI, SMK, DBE, and ORP but not over NCI, LYM, DEX, and PIW. To statistically analyse the results, the same statistical tests as in section 4.4.2 were applied. The results of the average performance (i.e. accuracy) of the methods across the various classifiers, and the accuracy of each individual classifier are provided in Figure 4.7 and Figure 4.8, respectively. The adjusted ρ -values have also been reported in the figures and the existence of a significant difference is highlighted with green colour. Looking at Figure 4.8, TAGA has outperformed midmRMR, SPECCMI, and CGA algorithms for most of the classifiers. However, no significant difference is observed when TAGA is compared with QPFS. The classification performance of the final subset selected by a feature selection algorithm may significantly vary from one classifier to another. Consequently, comparing the feature selection algorithms for a specific classifier will result in a goal-dependent analysis (see Section 4.3). As the aim of this thesis is to analyse the subsets in terms of classifier-bias using a goal-independent approach [91], a more critical analysis is performed on the

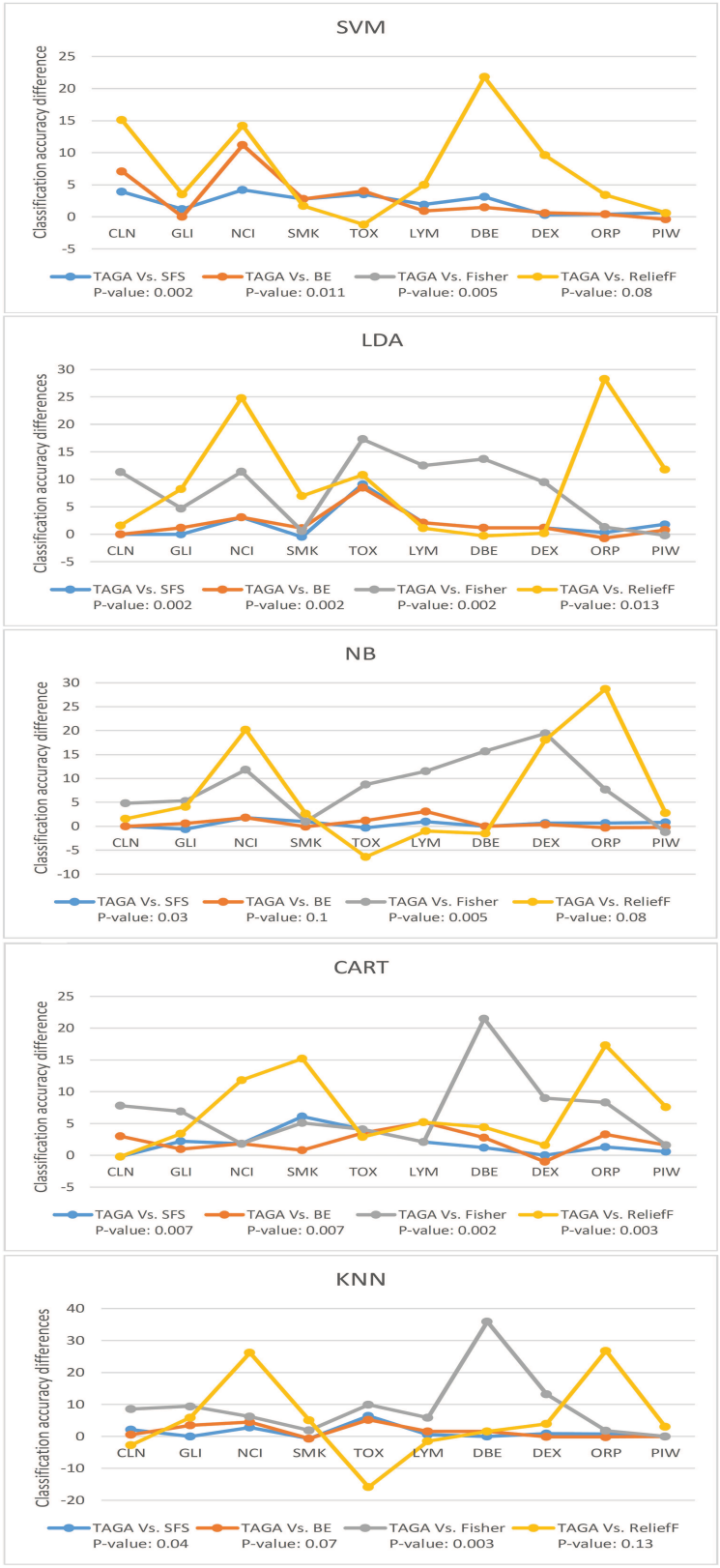

Figure 4.6: Comparing TAGA with Greedy algorithms. Results of the post-hoc tests for each classifier.

Table 4.7: Comparison of TAGA with other feature selection algorithms over reduced datasets

Classifier	SVM (%)	LDA (%)	NB (%)	KNN (%)	CART (%)	Average (%)
			CLN Dataset			
TAGA	97.4±1.3 (13.9±1.0)	90.3±0.3 (8.9±4.7)	90.3±0.0 (3.9±2.1)	94.03±0.8 (10.6±8.1)	90.1±1.6 (9.3±6.4)	**92.43**
mRMR-mid	90.3 (15.0)	88.7 (16.0)	90.3 (6.0)	98.4 (1.0)	90.3 (4.0)	91.61
QPFS	90.3 (2.0)	88.7 (1.0)	91.9 (7.0)	91.9 (3.0)	88.7 (1.0)	90.3
SPECCMI	90.3 (21.0)	90.3 (23.0)	90.3 (24.0)	93.5 (32.0)	87.1 (21.0)	90.3
CGA	91.5±2.6 (13.2±7.9)	90.2±0.5 (9.3±4.7)	90.8±0.8 (8.1±4.5)	95.2±1.3 (7.9±3.9)	87.4±2.0 (7.6±4.1)	91
			GLI Dataset			
TAGA	100.0±0.0 (10.6±0.9)	98.8±0.0 (10.3±0.9)	98.2±0.8 (10.1±1.6)	98.8±0.0 (14.8±4.3)	92.8±0.8 (14.4±11.9)	**97.73**
mRMR-mid	96.5 (25.0)	94.1 (13.0)	95.3 (19.0)	95.3 (5.0)	95.3 (7.0)	95.29
QPFS	94.1 (8.0)	95.3 (4.0)	96.5 (5.0)	94.1 (12.0)	92.9 (23.0)	94.6
SPECCMI	98.8 (22.0)	95.3 (7.0)	97.6 (16.0)	96.5 (16.0)	90.6 (18.0)	95.8
CGA	97.8±1.0 (11.7±4.1)	96.7±1.1 (6.5±2.8)	97.2±0.6 (13.6±6.1)	95.9±0.6 (16.2±7.0)	92.0±1.8 (6.2±5.9)	95.91
			NCI Dataset			
TAGA	84.2±1.4 (34.1±2.2)	78.1±0.5 (35.8±5.6)	83.5±0.6 (37.9±3.7)	79.5±0.4 (40.5±3.5)	60.1±1.3 (22.3±13.1)	77.08
mRMR-mid	66.7 (49.0)	65.0 (24.0)	65.0 (50.0)	65.0 (43.0)	58.3 (34.0)	64
QPFS	83.3 (26.0)	81.7 (14.0)	90.0 (17.0)	83.3 (39.0)	75.0 (3.0)	**82.7**
SPECCMI	85.0 (31.0)	78.3 (17.0)	83.3 (22.0)	80.0 (38.0)	68.3 (6.0)	79
CGA	83.0±2.0 (38.0±8.2)	75.7±2.6 (31.9±9.7)	79.7±2.0 (36.9±6.3)	78.8±2.4 (39.0±5.8)	59.5±3.5 (22.1±13.4)	75.33
			SMK Dataset			
TAGA	81.9±2.0 (19.2±4.9)	79.7±0.4 (15.1±5.0)	79.6±0.7 (9.8±5.2)	75.2±0.5 (13.1±4.4)	77.8±1.1 (16.4±4.0)	**78.88**
mRMR-mid	77.0 (30.0)	74.3 (27.0)	74.9 (16.0)	68.4 (7.0)	76.5 (11.0)	74.22
QPFS	79.7 (18.0)	78.6 (7.0)	82.4 (16.0)	74.3 (17.0)	71.7 (6.0)	77.3
SPECCMI	77.5 (22.0)	77.5 (21.0)	79.7 (17.0)	71.1 (13.0)	70.6 (2.0)	75.3
CGA	80.4±0.8 (16.3±6.0)	79.2±0.7 (12.9±4.7)	80.1±0.7 (18.0±5.1)	72.7±1.5 (8.8±6.3)	72.8±0.9 (12.8±3.5)	77.02
			TOX Dataset			
TAGA	82.4±0.7 (30.7±3.0)	81.6±1.0 (27.1±1.5)	74.9±0.8 (21.5±2.8)	77.7±0.8 (23.6±6.5)	68.4±0.9 (19.2±8.8)	**77.00**
mRMR-mid	81.3 (24.0)	78.9 (35.0)	74.3 (35.0)	72.5 (28.0)	67.8 (34.0)	74.97
QPFS	86.0 (29.0)	78.4 (14.0)	74.9 (13.0)	72.5 (15.0)	67.3 (34.0)	75.8
SPECCMI	78.4 (30.0)	77.8 (19.0)	75.4 (16.0)	77.2 (24.0)	63.7 (33.0)	74.5
CGA	82.2±1.4 (26.2±4.5)	79.8±1.0 (27.5±4.0)	73.0±1.0 (23.0±8.1)	74.3±2.6 (28.8±3.9)	64.0±2.1 (18.7±8.2)	74.64
			LYM Dataset			
TAGA	96.7±0.4 (33.5±1.5)	96.9±0.0 (40.5±0.6)	94.8±0.0 (30.6±1.8)	94.3±0.5 (20.3±3.7)	84.4±0.4 (43±2.3)	93.42
mRMR-mid	95.8 (39.0)	92.7 (26.0)	92.7 (37.0)	97.9 (22.0)	85.4 (26.0)	92.92
QPFS	99.0 (26.0)	97.9 (25.0)	94.8 (15.0)	97.9 (23.0)	85.4 (22.0)	95
SPECCMI	96.9 (28.0)	94.8 (17.0)	93.8 (19.0)	93.8 (36.0)	88.5 (28.0)	**93.5**
CGA	95.7±0.8 (33.7±6.9)	93.9±1.0 (22.8±8.8)	92.8±0.8 (23.8±4.9)	94.1±1.0 (30.3±7.0)	82.7±2.5 (22.5±6.9)	91.83
			DBE Dataset			
TAGA	90.6±0.0 (10.3±5.0)	90.3±1.0 (10.3±5.0)	89.1±0.0 (19.9±8.1)	90.6±0.4 (8.2±1.1)	90.3±0.7 (6.6±1.4)	**90.18**
mRMR-mid	81.3 (19.0)	85.9 (15.0)	82.8 (17.0)	89.1 (16.0)	84.4 (16.0)	84.69
QPFS	89.1 (16.0)	87.5 (10.0)	87.5 (5.0)	87.5 (13.0)	85.9 (10.0)	87.5
SPECCMI	85.9 (23.0)	89.1 (25.0)	92.2 (23.0)	87.5 (23.0)	84.4 (30.0)	87.8
CGA	88.8±1.9 (12.2±4.9)	88.4±1.3 (8.6±4.7)	93.1±1.5 (17.9±4.8)	90.6±1.3 (12.5±5.1)	87.3±1.6 (10.6±7.1)	89.66
			DEX Dataset			
TAGA	93.3±0.3 (46.7±1.5)	84.5±0.2 (39.0±2.6)	91.4±0.2 (49.3±1.6)	89.2±0.3 (37.0±6.4)	86.3±0.5 (32.7±5.7)	88.94
mRMR-mid	92.7 (49.0)	84.3 (42.0)	89.7 (49.0)	88.3 (43.0)	85.0 (40.0)	88
QPFS	93.3 (49.0)	83.0 (48.0)	87.0 (48.0)	87.3 (48.0)	84.3 (30.0)	87
SPECCMI	94.0 (49.0)	84.0 (34.0)	92.3 (49.0)	89.3 (47.0)	86.3 (36.0)	**89.2**
CGA	88.1±0.8 (42.2±7.1)	76.8±0.6 (30.2±15.0)	84.0±1.3 (32.2±10.1)	82.8±1.3 (37.2±9.5)	82.3±0.6 (24.4±9.5)	82.81
			ORP Dataset			
TAGA	97.4±0.5 (15.7±2.3)	92.3±1.2 (13.7±2.3)	94.7±0.5 (18.2±1.7)	99.8±0.4 (13.4±4.7)	86.3±2.1 (14.0±8.3)	**94.1**
mRMR-mid	95.0 (20.0)	86.0 (20.0)	90.0 (7.0)	97.0 (20.0)	84.0 (4.0)	90.4
QPFS	97.0 (12.0)	90.0 (12.0)	91.0 (17.0)	98.0 (9.0)	85.0 (14.0)	92.2
SPECCMI	88.0 (17.0)	77.0 (16.0)	87.0 (26.0)	94.0 (21.0)	81.0 (15.0)	85.4
CGA	97.5±0.7 (14.0±3.6)	91.4±1.6 (13.4±3.3)	94.3±1.3 (14.3±3.5)	98.5±1.0 (13.6±3.6)	84.2±2.5 (5.6±5.3)	93.18
			PIW Dataset			
TAGA	97.6±0.5 (6.1±1.7)	96.8±0.4 (8.0±5.4)	95.8±0.4 (15.3±6.3)	97.0±0.0 (8.1±0.7)	98.6±0.5 (9.8±4.8)	97.16
mRMR-mid	98.0 (5.0)	99.0 (5.0)	96.0 (5.0)	96.0 (7.0)	99.0 (10.0)	97.6
QPFS	98.0 (3.0)	98.0 (20.0)	95.0 (3.0)	97.0 (20.0)	98.0 (3.0)	97.2
SPECCMI	95.0 (15.0)	97.0 (17.0)	95.0 (19.0)	95.0 (11.0)	96.0 (29.0)	95.6
CGA	97.7±0.7 (6.8±4.1)	98.4±0.7 (11.3±6.2)	96.5±0.7 (10.4±4.6)	97.2±0.4 (10.2±5.1)	98.4±0.7 (5.9±4.5)	**97.64**

For each classifier and algorithm, the first value is the classification accuracy, the values in parenthesis show the number of selected features, and the sign ± indicates standard deviation. The last column presents the average of the classification accuracies for each algorithm.

average accuracies of all classifiers.

As can be seen from Table 4.7, TAGA has outperformed other other algorithms in terms of average accuracy (last column of Table 4.7) for most of datasets

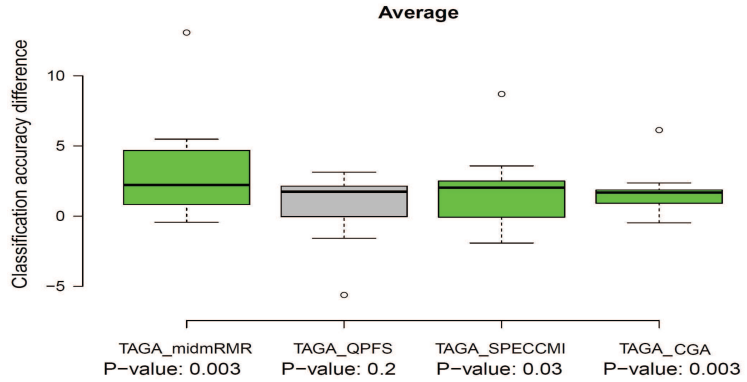

Figure 4.7: Comparing TAGA with other feature selection algorithms using the Friedman test and the Wilcoxon post-hoc analysis applied on the average of the accuracies. The y-axis is the classification accuracy difference and x-axis indicates the names of the compared algorithms.

and for the other datasets identical results have been obtained. However, the statistical analysis tests revealed that the performance of TAGA has been significantly different from all other feature selection methods except QPFS. The superiority of TAGA over mRMR-mid was predictable as the searching strategy of mRMR-mid algorithm is greedy in nature and similar to other greedy search algorithms discussed in subsection 4.4.2, it searches only amongst local optima. QPFS and SPECCMI are both quadratic programming-based algorithms and use the Nyström approximation method.

Interestingly, the statistical results revealed that TAGA has only outperformed SPECCMI when compared to quadratic programming-based algorithms, possibly due to the strategy of the two algorithms in employing Nyström approximation. In QPFS, a two level approximation is proposed to cast the quadratic programming problem into a lower dimensional subspace. This two level approximation provides acceptable approximation for small size datasets however, it might not yield a precise enough approximation for large datasets [91]. As opposed to QPFS, in SPECMI, only one level of approximation with a fixed sampling rate is applied. This strategy leads to a better approximation when high redundancy exists in the dataset [92]. Before applying the

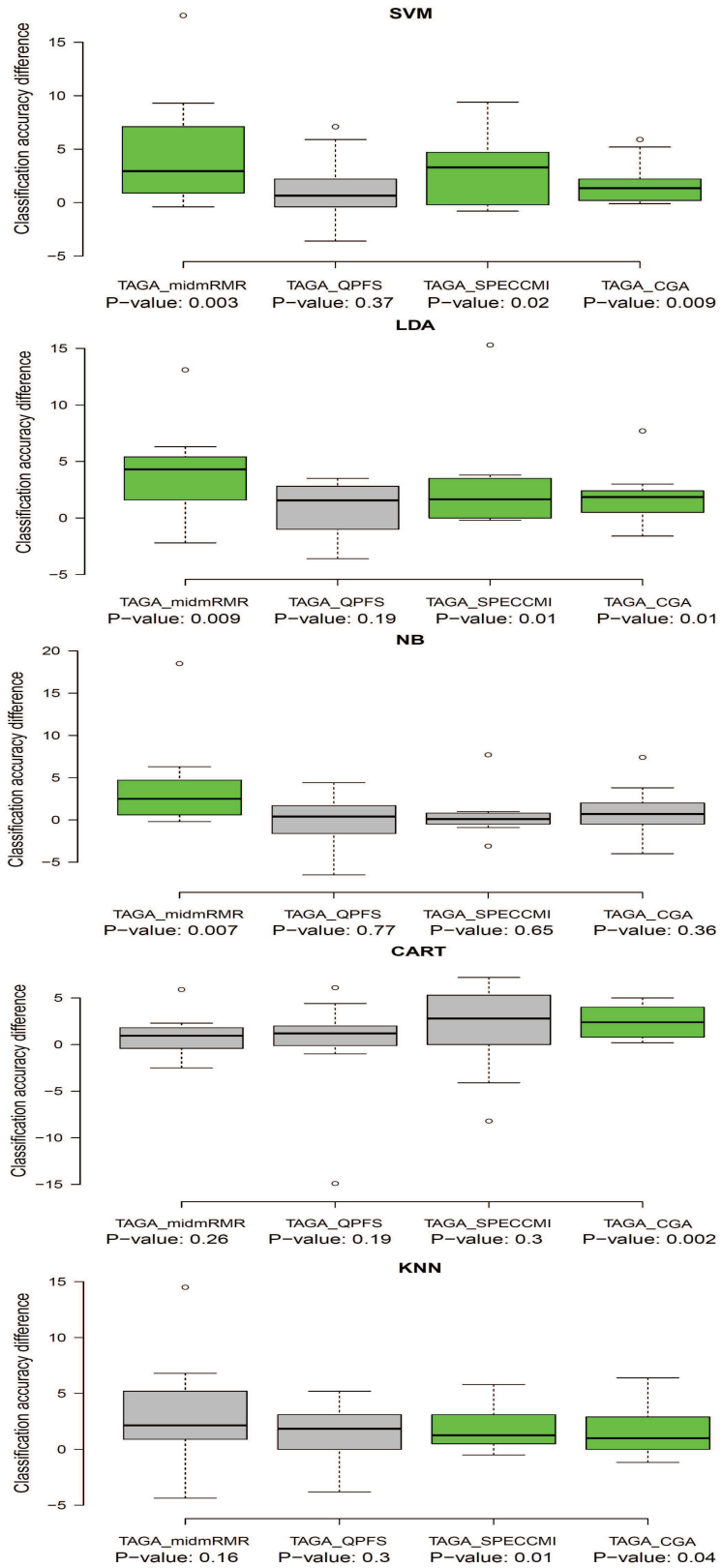

Figure 4.8: Comparing TAGA with other feature selection algorithms. Results of the post-hoc tests for each classifier.

feature selection algorithms, a filtering step is applied to the datasets. This filtering step reduces the size of the datasets and results in less redundancy in the datasets. Therefore, it can be concluded that the filtering step has enhanced the performance of the QFPS and has worsened the performance of SPECMI. Although the statistical analysis does not show any significant difference between the results of TAGA and QPFS, TAGA has performed better than QPFS, in terms of average accuracies, for many datasets and for the rest, comparable results have been obtained except for the NCI dataset for which QPFS has highly outperformed TAGA. QPFS approximates mutual information values using Nyström method which are less accurate than the values estimated by TAGA's mutual information estimator and therefore, the features selected by QPFS are of lower quality than the ones selected by TAGA. NCI is a highly correlated and redundant dataset and for such a dataset, features with lower quality may in fact be better for classification [91]. This might be the reason that is why the results of QPFS is considerably better than TAGA for NCI dataset. However, when NCI is removed from the statistical analysis, TAGA shows significant difference over QPFS with the p-value of 0.03.

TAGA has performed better than CGA, in terms of average accuracies (see Table 4.7). The statistical analysis has also revealed the superiority of TAGA over CGA (see Figure 4.7). CGA is an EC algorithm similar to TAGA. TAGA and CGA both use an integer-encoded solution representation and search for the best subsets in a range of subset cardinalities. However, with TAGA each solution contains all the features of the dataset and only part of the solution is considered as the final subset (Figure 4.2), whereas CGA uses an integer-coded solution representation in which the solutions contain the selected features only. Based on this solution representation, designing a proper mutation operator is a non-trivial task. Therefore, CGA works only on a designed crossover

operator and in cases that one feature appears more than once in the solutions, a SFS-like mutation operator repairs the solutions. To compensate the absence of an effective mutation operator in generating diverse solutions, the algorithm needs a large amount of individuals. To compare fairly the two algorithms, the stop criterion is set when a specific number of function evaluation is counted. This stop criterion makes CGA stop working at early generations as a large amount of individuals are evaluated at each generation and therefore, the initial population cannot be completely evolved.

4.4.4 Classifier-bias analysis

It is important to examine the subset selected by TAGA to understand if they are not classifier-biased and consequently, can provide high generalisation power over a range of classifiers. As can be seen in Table 4.7, the number of features in the subset with which the classifiers reach their highest performance (optimum subset) varies noticeably. However, it has been observed that the optimum subset of any classifier usually achieves optimal or near optimal accuracy when used to train other classifiers [91]. According to this observation, Naghibi et al. [91] proposed an approach to analyse the classifier-bias of a feature selection algorithm in which the optimum subset of a classifier is used to train the other classifiers and the results are compared to the optimal classifiers' accuracies.

The closer the obtained accuracies are to the classifiers' optimal accuracies, the less classifier-biased the subset and the higher the generalisation power. A prior knowledge of optimal accuracies of the classifiers over datasets would be useful for this analysis. However, this prior knowledge is not available. To solve the problem, the highest accuracy obtained for each classifier during the feature selection process is considered as the classifier's optimal accuracy and

the corresponding subset is considered as the optimal subset.

TAGA's classifier-bias analysis is shown in Table 4.8 for 4 classifiers and 4 datasets. The KNN optimal subset is used to train the other classifiers. As can be seen, optimal or near optimal accuracy has been obtained by most of classifiers using the best features subset of KNN provided by TAGA.

Table 4.8: TAGA classifier-bias analysis (%)

Classifier		SVM	LDA	NB	CART
GLI	KNN optimum subset	96.47	96.44	94.12	91.76
	Classifier Optimum Acc.	100	98.8	98.82	94.12
DEX	KNN optimum subset	92.33	84.7	90.33	86.67
	Classifier Optimum Acc.	94	84.7	91.67	87.67
PIW	KNN optimum subset	97	97	95	99
	Classifier Optimum Acc.	98	97	97	99
ORL	KNN optimum subset	97	92	95	82
	Classifier Optimum Acc.	98	93	96	89

4.4.5 Running time analysis

In the last experiment, the running time of TAGA is analysed. Table 4.9 shows the comparison of TAGA against SFS, BE, and CGA in terms of running time needed to search for the best subset over subset cardinality range from 1 to 50. The cell values provide the running time of the algorithms in seconds. The last column shows the average running times over the 10 datasets.

As shown in Table 4.9, TAGA has been the second fastest algorithm when the average running time is considered. TAGA has outperformed the other algorithms when applied to the SMK, TOX, DBE, and DEX datasets. SFS has outperformed TAGA for the datasets which have a small sample size including CLN, GLI, NCI, LYM, ORP, and PIW.

However, for datasets with a larger sample size such as SMK, TOX or the datasets for which more feature are extracted through the first filtering stage (DBE and DEX) TAGA has shown better performance. This could be due to

Table 4.9: TAGA running time analysis in seconds

	Dataset										
Algorithm	**CLN**	**GLI**	**NCI**	**SMK**	**TOX**	**LYM**	**DBE**	**DEX**	**ORP**	**PIW**	**Avg.**
SFS	34	48	33	140	123	59	242	323	62	63	112.7
TAGA	123	122	127	139	132	117	154	202	129	148	139.3
CGA	158	181	168	183	174	161	229	286	199	183	192.2
BE	183	223	162	649	587	294	1208	1486	288	301	538.1

the reason that TAGA calculates the mutual information between the features wherever necessary but the SFS need to calculated mutual information values whenever a new feature is added to the pool. Therefore, SFS needs a bigger portion of mutual information values between the features for those datasets which requires higher computations.

It should be mentioned that the Relief, Fisher, mRMR-mid, QPFS, and SPEC-CMI algorithms are eliminated from this comparison. Although Relief and the Fisher algorithms are fast, they have shown poor performance compared against TAGA (see Table 4.6). Regarding mRMR-mid, QPFS, and SPECCMI, these algorithms have partly been coded into another programming languages which makes a fair comparison impossible. Nevertheless, the current running time analysis still provides a good sense to readers of how fast TAGA can perform.

4.5 Conclusion

This chapter proposed a novel EC-based feature selection algorithm called TAGA which is embedded into a new two-stage hybrid framework called filter/filter approach. The filter/filter approach was designed to address classifier-bias limitations of existing filter/wrapper methods. In the first-stage, Fisher score was used to select the most informative features which were used as input into the second stage.

In the second stage, TAGA, a mutation-based GA hybridised with a long-term memory TL and guided by a SFS procedure, was applied in order to select the final subset of features. TAGA benefits from a novel integer-coded representation and mutation operator. In addition, a new TL encoding scheme was proposed in order to make the solution storing and restoring processes computationally more effective.

Exhaustive experiments were performed using five classifiers and ten datasets selected from wide range of applications to evaluate the performance of TAGA. The proposed TAGA was compared with greedy search and other algorithms found in the literature. All the other algorithms used in the comparison were also embedded in a filter/filter framework. The computation results confirmed that TAGA outperformed other feature selection algorithms. The filter/filter approach with the embedded TAGA feature selection algorithm can be adopted when developing predictive models for biomedical or other tasks.

The next chapter, Chapter 5 discusses Generalisation Power in detail and proposes a novel approach to measure the performance of feature subsets in terms of generalisation power over multiple classifiers.

Chapter 5

A Generalisation Power Approach for Evolutionary Computation-based Feature Selection

5.1 Introduction

Feature selection is the process of selecting an optimal subset of features required for maintaining or improving accuracy of data mining models. EC algorithms, with their efficient global search capabilities, are good approaches to feature selection. However, the main limitation of EC algorithms for feature selection is that due to their stochastic nature, different 'best feature subset' solutions are returned every time they are run.

Existing solutions to the stability issue include typical aggregation (e.g. intersection and union) and frequency-based methods, but because these methods do not consider the performance of a classifier in their selection process, the

methods can select a feature subset which when utilised to train a classifier can lead to poor classification accuracy.

Classifier-based aggregation is an alternative method, which uses the performance of one classifier to select the best subset of features, and this approach may result in a biased subset with poor performance over various classifiers known as lack of generalisation power. A subset with high generalisation power is able to achieve optimal or near optimal accuracy over multiple classifiers.

To address limitations of existing methods, this chapter proposes a novel approach called generalisation power analysis that measures the performance of feature subsets in terms of generalisation power over multiple classifiers.

5.2 Proposed generalisation power analysis approach

Generalisation power refers to the classification performance capability of a feature subset over wide range of classifiers. From a generalisation perspective, an optimal subset is able to achieve optimal accuracy if applied over any classifier. Nevertheless, such an optimal subset does not exist in practice but there are near-optimal subsets which are able to closely follow optimal accuracies when used to train multiple classifiers.

In order to discover those near-optimal subsets, a method is needed which can measure the value of generalisation power for subsets. Consequently, this chapter proposes a Generalisation Power Index (GPI) in which the generalisation power of subset s over set of classifiers C can be defined as follows:

$$GPI_s^C = \frac{\sum_{c \in C}(Opt_c^D - Acc_s^c)}{\mid C \mid} \tag{5.1}$$

Where, $\mid C \mid$ is the number of classifiers in C, Opt_c^D stands for the optimal accuracy of the classifier c over the dataset D, Acc_s^c represents the accuracy of the classifier c when trained using feature subset s.

To apply GPI on the output solutions of an EC algorithm, algorithm 5 is proposed.

Algorithm 5: Pseudocode of Generalisation Power Analysis Algorithm

```
1  begin
2      for n times do
3          Run the EC feature selection algorithm
4          Save the best subsets in subset pool S
5      end
6      Remove repetitive subsets from S
7      for each classifier in C do
8          for each subset in S do
9              Train the classifier using the subset and save the accuracy
10         end
11     end
12     for each classifier do
13         Consider the highest obtained accuracy as optimal accuracy
14     end
15     for each subset in S do
16         Calculated GPI using Eq. 5.1
17     end
18     Choose the subset with lowest GPI value as the best subset
19 end
```

In algorithm 5, the EC feature selection algorithm is run n times (the typical value for n in the literature is 30 times) and in each run the best subsets in terms of EC fitness function are saved in subset pool S. It should be noticed that in a single run of an EC algorithms, it is likely that more than one best subset (optimal or near optimal subsets) is generated therefore, all of the best subsets are saved in the pool and the repetitive subsets are removed later (lines 1-6). Then, all the remaining subsets in S are used to train the classifiers in C and the corresponding accuracies are obtained using a suitable validation

method (lines 7-11). To calculate GPI, the optimal accuracies of the classifiers over datasets are needed.

A prior knowledge of classifiers optimal accuracy over a dataset can be a great help for calculating GPI. However, obtaining optimal accuracies is an arduous task if not impossible. Thus, to tackle this problem, one approach is to consider the highest obtained accuracy for each classifier over each dataset as optimal accuracy of the classifier over the dataset.

In the next step, the GPI value is calculated for all subsets in the pool. Finally, the subsets are sorted according to their GPI value and the subset with the lowest GPI value is considered to be the subset which provides highest generalisation power over the classier used.

5.3 Experimental design

5.3.1 The EC algorithm adopted for the experiments and its parameter settings

In order to examine the performance of the proposed approach, a set of subsets obtained in multiple independent runs of an EC feature selection algorithm is needed. For this purpose, a hybrid EC-based feature selection algorithm was implemented which is composed of two stages: 1) filtering stage 2) EC-based selection stage.

In the filtering stage, the Fisher score ranking feature selection algorithm [82], which is computationally cost effective, is applied to reduce the complexity of the dataset and to filter out the most promising features. The Fisher score algorithm is set to select the top 100 features of the datasets. The selected features are then used to form reduced datasets which are then fed to the

second stage.

In the second stage, the reduced dataset from previous stage is fed to a standard GA with a fixed length binary representation, typical two-point crossover, and Bit Flip mutation operators which selects the final subset out of 100 features. To evaluate the fitness of the subsets, Equation 5.2 is used. It is a two criteria fitness function considering both the number of selected features and the classification performance as proposed in [105].

$$f = ((1 - \alpha) * \frac{ACC}{100}) - (\alpha * \frac{n}{N}) \tag{5.2}$$

where, α is the weighting parameter, n is the number of features in the subset, N presents total number of features in the dataset, and ACC stands for the KNN average classification accuracy over a 10-fold cross-validation test. Both ACC and n are normalised to their highest possible values. The parameter α is set to a small value of 0.01. This small value gives priority to the subsets with highest accuracy but in cases that two subsets with different number of features have the same accuracy, the subset with fewer features is selected as the best subset. The crossover and mutation rates are set to typical values of 0.7 and 0.01, respectively. The stop criteria is set when 100 generations are counted and the GA is independently run 20 times.

At the end of each run, the best subset is saved for one further selection process using the proposed GPI and benchmark methods. It should be noticed that in a single run of an EC algorithms, it is likely that more than one best subset is generated therefore; all of the best subsets are extracted and saved.

Table 5.1: Description of datasets used in experiments

Dataset Name	Abbrev.	Type	# Features	# Instances	# Classes
GLI_85	GLI	Biological	22283	85	2
TOX_171	TOX	Biological	5748	171	4
SMK_CAN_187	SMK	Biological	19993	187	2
GLA-BRA-180	GLA	Biological	49151	180	4
CLL-SUB-111	CLL	Biological	11340	111	3
COIL20	COI	Image	1024	1440	20
Yale 64x64	YALE	Image	1024	165	15
RELATHE	REL	Text	4322	1427	2
BASEHOCK	BAS	Text	4862	1993	2
PCMAC	PCM	Text	3289	1943	2
Arcene	ARC	Mass Spectrometry	10000	200	2

5.3.2 Datasets and classifiers

Table 5.1 shows the properties of the 11 datasets used in the experiments. All the datasets are available on the ASU feature selection repository [72]. The datasets are of most up-to-date datasets available publicly and cover wide range of applications including image data, text data, and biological data.

During the experiments, the subsets are evaluated using 5 classifiers, namely SVM, KNN, CART, NB, LDA. The number of K in KNN classifier is set to 5 for all classifiers and the hyper parameters for other classifiers, such as: kernel function for SVM, data distribution type for NB, discrimination type for LDA, are experimentally set for each dataset. To guarantee valid results for making a reliable predictions, the k-fold cross-validation (the value for is set to k=10 as there are enough samples) was adopted.

5.3.3 Benchmark methods for combining subsets of features

Assuming $S = \{s_1, s_2, ..., s_n\}$ a set of n final subsets, the proposed generalisation power analysis approach is compared with the following methods:

- Union: this method combines unique features in the set to generate the

best subset using the following formula.

$$S_{best} = \{s_1 \cup s_2 \cup, ..., \cup s_n\} = \bigcup_{i=1}^{n} s_i \tag{5.3}$$

It must be mentioned that intersection were removed from computational experiments because it led to an empty set for most of the cases.

- Classifier-based aggregation: As proposed in [16] the first subset in set S is selected as $s_{baseline}$ baseline subset and its accuracy over a desired classifier is calculated. The features in $s_{baseline}$ will always become part of the final selection s_{final}. For selecting other features, the unique features in S - $s_{baseline}$ will become part of the s_{final} only if they improve s_{final} accuracy. For this experiment, four classifiers including SVM, LDA, NB, and KNN are used to develop several classifier-based aggregation methods.

- Frequency-based measure: frequency-based approaches consider how consistently a feature has appeared in a set of subsets. Somol and Novovicova [108] defined $C(f)$ the consistency of feature f in a set of subsets of features S as:

$$C(f) = \frac{F_f - F_{min}}{F_{max} - F_{min}} \tag{5.4}$$

where, F_f is the frequency of feature f, and F_{min} and F_{max} present minimum and maximum frequencies in S, respectively. In order to measure the consistency of subsets in this chapter, the consistency of subset $s \in S$ is defined as the average of consistencies over all features in s:

$$C(s) = \frac{1}{\mid s \mid} \sum_{f \in s} C(f) \tag{5.5}$$

5.4 Results and discussion

5.4.1 Comparing the performance of GPI with benchmark algorithms

This section describes the experiments carried out to compare the performance of the GPI compared to alternative methods used in the literature in terms of classification accuracy of the selected subset. These methods are Union, four classifier-based methods [16] including SVM-based, LDA-based, NB-based, KNN-based (see subsection 5.3.3), and Frequency-based [108].

For each dataset, the set of subsets obtained in several runs of the GA (explained in subsection 5.3.1) was input into the GPI and other benchmark methods to find a single best subset. The best subsets obtained from each method are then evaluated using classifiers.

Table 5.2 shows the classification accuracy results obtained for 4 classifiers over 11 datasets using GPI selected subset compared to the subsets selected by other six methods. In Table 5.2, the results of datasets are separated by lines and for example, the value of 94.12 in GLI dataset part corresponds to Union(79) and SVM, meaning that the final subset selected by Union for GLI dataset contains 79 features and when the subset was used to train SVM classifier, 94.12% accuracy was achieved.

To detect a statistically meaningful significant difference amongst the algorithms, the Friedman test [45] is applied on the results of each classifier (columns two to five in Table 5.2).

Next, Wilcoxon signed-rank post-hoc pairwise comparison analysis is carried out to determine which pairs of algorithms have had significantly different performance. The results are presented in Figure 5.1 which consists of four sub-figures each of which depicts the results of statistical test analysis for one

classifier in boxplots. Each boxplot compares GPI with another method in which classification accuracy differences between GPI and other algorithms across 11 datasets are plotted. The adjusted ρ-values for Wilcoxon analysis have also been reported in Figure 5.1. The smaller the ρ–value, the stronger the evidence against the null hypothesis. The existence of a significant at level of $\rho = 0.05$ has been spotted with green boxes.

As can be seen, the proposed GPI shows superiority over Union for all classifier. The reason resides in the selection strategy of Union algorithm in which the features with at least one appearance in the set of subsets are selected. This strategy leads to a subset with large number of features (as obvious in Table 5.2) which, as a results, contains noisy and irrelevant features. Regarding classifier-based methods, as expected, when GPI is compared with classifier-based methods for their corresponding classifier (for example when the final subset of SVM-based method is used to train SVM classifier) GPI has not been able to outperform classifier-based methods (Figure 5.1). The reason is that the classifier-based method directly search for the features which maximise the classification performance of a desired classifier when added to the baseline. However, the selected subset is biased toward the classifier used and may not perform well when used to train other classifiers.

For frequency-based method, GPI has outperformed frequency-based method for SVM, LDA, and NB classifiers but has not been able to perform better over KNN classifier. This is mainly because in the initial features selection process through GA (see subsection 5.3.1), KNN classifier was used as the fitness function and therefore, the most frequent features are more likely to provide good classification performance over the same classifier. However, these results shows a subset that contains highly frequent features does not necessarily result in a good classification performance over different classifier

(lack of generalisation power).

The goodness of a feature subset is usually evaluated based on its classification performance over a single classifier. This type of evaluation, known as goal-dependent, is clearly unable to analyse the generalisation power of the subset over several classifiers. In order to analyse the subsets from a generalisation power perspective, Naghibi et al. [91] proposed a goal-independent method in which the goodness of a subset is measured based on its average classification performance over multiple classifiers. In this experiment, a goal-independent analysis is performed to understand which method provides a final subset with highest generalisation power. Therefore, the Friedman test following with Wilcoxon signed-ranked post-hoc analysis is applied on the last column in Table 5.2 which provides average accuracies for four classifiers. Figure 5.2 depicts the results for all six methods with reported ρ-values. As expected, GPI has shown superior performance over other algorithms when significant level is set at $\rho = 0.05$.

In the next experiment, a new classifier, which has never been involved in computation results for both GPI and classifier-based methods, is trained using the final subsets of seven methods. This experiment follows a two-fold aim.

Firstly, GPI employs classification performance of several classifiers to select the best subset and consequently, it may seem unfair to compare it with other methods in terms of generalisation power only. Secondly, it has been observed that optimal and near optimal feature subset of any classifier, usually achieves optimal or near-optimal classification performance in conjunction with other classifiers [91].

As GPI tries to find the subset which performs well over multiple classifiers, this experiment investigates that whether the subset selected by GPI, as a near-optimal subset for the classifiers involved in GPI, will provide acceptable

Table 5.2: Comparison of GPI with other algorithms over four classifiers

Classifier	SVM (%)	LDA (%)	NB (%)	*K*NN (%)	Average (%)
			GLI Datasets		
Union (79)	94.12	**95.29**	**94.12**	91.76	93.82
GPI (17)	**97.65**	**95.29**	**94.12**	**97.65**	**96.18**
SVM_based (16)	**97.65**	91.76	90.59	**97.65**	94.41
LDA_based (16)	95.29	92.94	90.59	96.47	93.82
NB_based(16)	96.47	92.94	91.76	**97.65**	94.70
*K*NN_based (15)	96.47	91.76	90.59	**97.65**	94.12
Freq_based (21)	95.29	94.12	90.59	96.47	94.12
			SMK Datasets		
Union (100)	74.33	64.71	70.05	66.31	68.85
GPI (48)	**79.14**	**79.68**	**77.01**	78.07	**78.48**
SVM_based (55)	**79.14**	71.12	75.4	75.4	75.27
LDA_based (58)	71.66	76.47	73.26	71.12	73.13
NB_based (56)	77.54	72.19	76.47	73.8	75
KNN_based (51)	70.59	70.05	72.19	80.21	73.26
Freq_based (51)	78.07	78.07	73.26	**80.75**	77.54
			TOX Datasets		
Union (99)	76.02	67.25	60.82	69.01	68.28
GPI (38)	83.04	82.46	64.33	**83.63**	**78.37**
SVM_based (51)	**89.47**	85.38	63.74	69.01	76.9
LDA_based (52)	81.87	**88.89**	63.74	67.25	75.44
NB_based (47)	79.53	78.36	**71.35**	77.78	76.76
*K*NN_based (40)	75.44	78.36	65.5	81.29	75.15
Freq_based (29)	77.19	82.46	68.42	77.78	76.46
			BAS Datasets		
Union (93)	94.03	91.72	93.23	90.22	92.3
GPI (59)	94.53	93.53	**94.38**	**94.53**	**94.24**
SVM_based (53)	94.48	94.23	93.53	87.86	92.53
LDA_based (64)	**94.58**	93.03	94.03	93.73	93.84
NB_based (60)	94.43	93.33	94.33	89.31	92.85
*K*NN_based (51)	94.48	94.23	93.43	86.25	92.1
Freq_based (43)	94.48	**94.48**	93.08	86.1	92.04
			COI Datasets		
Union (98)	**95.42**	89.58	65.62	94.65	86.32
GPI (46)	95.07	89.93	**78.06**	96.6	**89.92**
SVM_based (59)	95.21	89.37	74.58	95.9	88.77
LDA_based (62)	95.14	**90.62**	74.17	95.56	88.87
NB_based (57)	95	84.72	**78.06**	95.83	88.4
*K*NN_based (52)	94.65	84.93	77.15	96.81	88.39
Freq_based (47)	94.58	89.79	74.37	**97.01**	88.94
			ARC Datasets		
Union (89)	67.5	65	72.5	62.5	66.88
GPI (24)	67.5	65.5	**75**	**75**	**70.75**
SVM_based (21)	**69**	**66**	69	61.67	66.42
LDA_based (25)	68.5	65	72.5	68	68.5
NB_based (24)	68.5	65	72.5	63.57	67.39
*K*NN_based (18)	68	**66**	69	61.11	66.03
Freq_based (11)	68	65.5	69	64	66.63
			REL Datasets		
Union (96)	83.04	80.59	79.96	76.8	80.1
GPI (51)	84.02	**82.69**	**81.64**	80.31	**82.18**
SVM_based (56)	**84.09**	81.5	81.15	79.54	81.57
LDA_based (62)	82.55	**82.69**	80.66	78.49	81.1
NB_based (61)	**84.09**	81.22	81.5	78.84	81.41
*K*NN_based (51)	82.2	79.12	79.33	**80.38**	80.26
Freq_based (39)	80.87	77.65	78.77	79.82	79.28
			YALE Datasets		
Union (100)	67.88	69.09	62.42	66.06	66.36
GPI (53)	72.73	**72.73**	**64.85**	69.7	**70**
SVM_based (50)	**73.33**	67.27	61.82	72.12	68.64
LDA_based (55)	69.09	71.52	60.61	69.09	67.58
NB_based (52)	70.3	69.09	64.24	69.09	68.18
*K*NN_based (47)	70.91	66.06	60.61	**73.94**	67.88
Freq_based (49)	64.85	63.03	58.18	72.73	64.7
			GLA Datasets		
Union (96)	45.56	63.33	62.78	73.33	61.25
GPI (37)	**68.89**	73.89	**65**	76.67	**71.11**
SVM_based (41)	62.22	66.11	61.11	75.56	66.25
LDA_based (49)	62.78	**77.22**	62.78	75	69.45
NB_based (41)	52.22	67.78	63.33	77.78	65.28
*K*NN_based (38)	57.22	67.78	61.11	**78.33**	66.11
Freq_based (29)	57.78	66.11	61.67	**78.33**	65.97
			CLL Datasets		
Union (98)	**55.86**	45.05	54.05	63.06	54.51
GPI (38)	**55.86**	67.57	**67.57**	76.58	**66.9**
SVM_based (27)	56.76	63.96	56.76	68.47	61.49
LDA_based (32)	51.35	**72.07**	57.66	60.36	60.36
NB_based (33)	54.05	62.16	66.67	68.47	62.84
*K*NN_based (26)	**55.86**	64.86	57.66	73.87	63.06
Freq_based (19)	52.25	63.06	58.56	**77.48**	62.84
			PCM Datasets		
Union (87)	90.17	88.16	84.3	88.11	87.69
GPI (63)	**90.27**	**88.78**	89.5	89.24	**89.45**
SVM_based (42)	90.22	81.78	88.73	89.66	87.6
LDA_based (55)	90.22	87.03	89.6	88.68	88.88
NB_based (44)	90.07	83.69	**89.76**	89.19	88.18
*K*NN_based (38)	89.86	81.68	88.63	89.66	87.46
Freq_based (41)	90.02	80.34	88.42	**90.17**	87.24

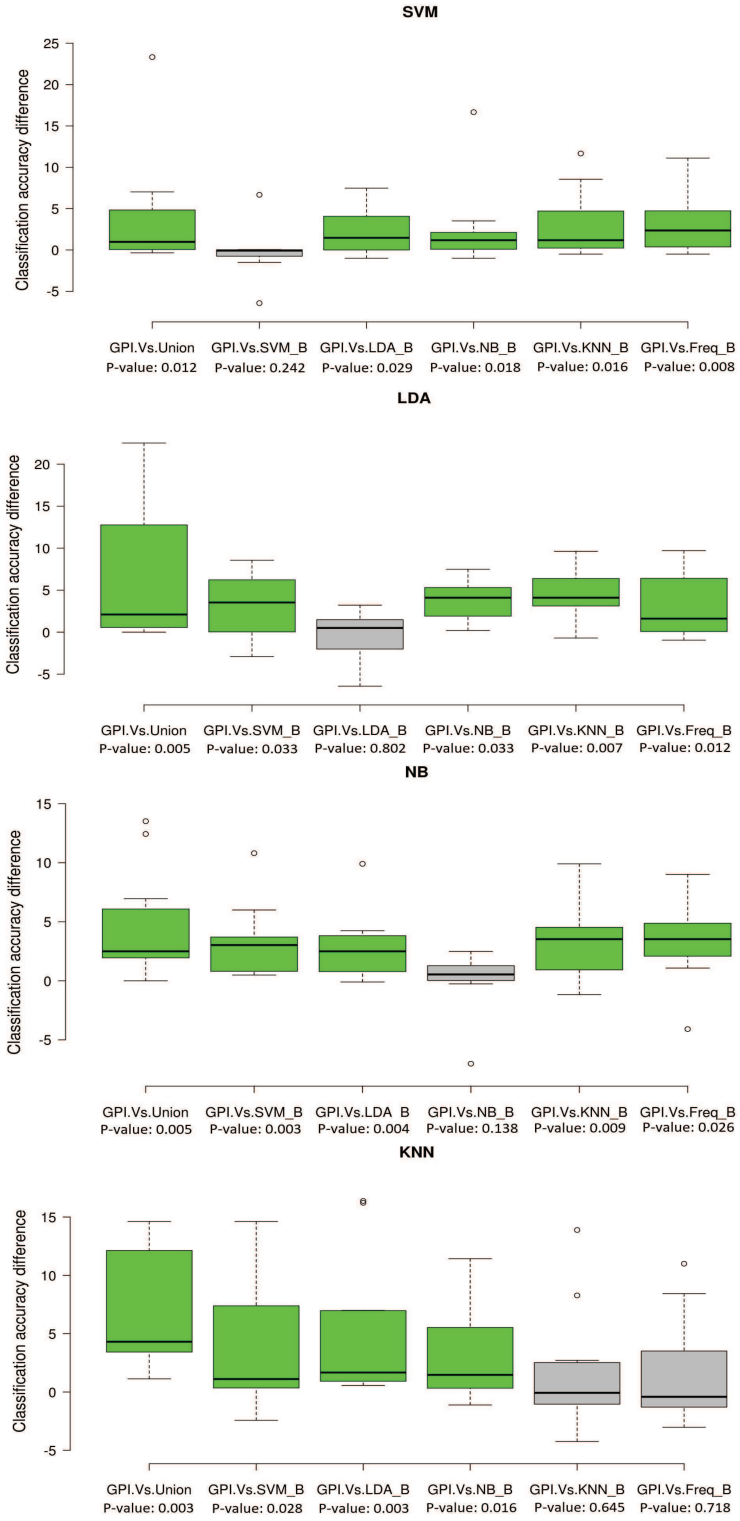

Figure 5.1: Comparing GPI with other algorithms using the Friedman test and the Wilcoxon post-hoc analysis for each classifier. The y-axis is the classification accuracy difference and the x-axis indicates the names of the compared algorithms.

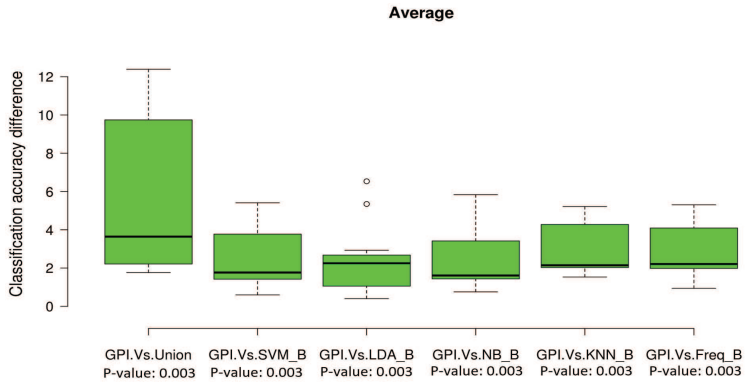

Figure 5.2: Comparing GPI with other algorithms using the Friedman test and the Wilcoxon post-hoc analysis applied on the average of the accuracies. The y-axis is the classification accuracy difference and the x-axis indicates the names of the compared algorithms.

performance if used to train other classifiers. For this, the CART classifier is used and the results presented in Figure 5.3 reveal that GPI has significantly performed better than the other methods when the subsets are used to train the CART classifier. This is mainly because the subset provided by GPI has a higher generalisation power over different classifiers.

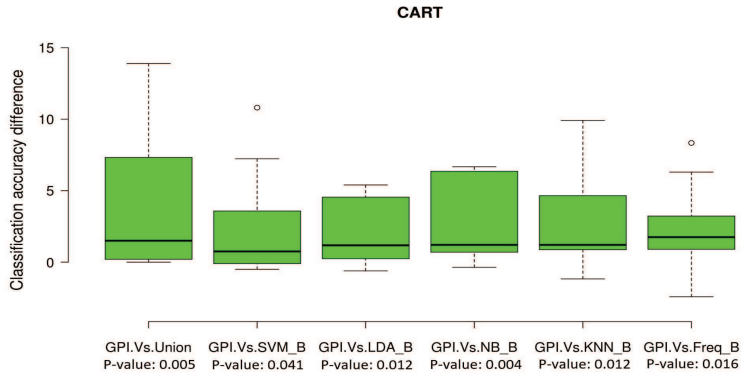

Figure 5.3: Comparing GPI with OTHER algorithms using the Friedman test and the Wilcoxon post-hoc analysis for CART classifier. The y-axis is the classification accuracy difference and the x-axis indicates the names of the compared algorithms.

Table 5.3: Comparing the classification performance of the subset selected by GPI with subsets selected by other algorithms over the CART classifier (%)

Methods	Union	GPI	SVM_based	LDA_based	NB_based	KNN_based	Freq_based
GLI	80.00	88.24	87.06	87.06	87.06	87.06	84.71
SMK	63.64	70.05	64.71	64.71	63.64	62.57	67.38
TOX	59.65	59.65	59.65	55.56	53.22	60.82	56.73
BAS	94.38	94.43	93.68	94.23	93.93	93.53	93.13
COI	91.32	91.67	91.87	91.39	90.90	89.03	90.00
ARC	66.00	67.50	68.00	66.50	66.00	66.00	67.00
REL	81.50	83.04	82.76	81.71	82.41	82.20	81.29
YALE	56.97	56.97	55.15	57.58	55.76	55.76	59.39
GLA	52.78	66.67	59.44	61.67	60.00	60.00	58.33
CLL	52.25	64.86	54.05	59.46	58.56	54.95	58.56
PCM	89.71	90.27	90.53	90.38	90.63	90.53	90.12

5.4.2 Running time analysis

In the last experiment, the running time of GPI is analysed. Table 5.4 shows the comparison of TAGA against classifier-based methods in terms of running time needed to select the best subset out of set of output subsets obtained from several runs of an EC algorithm. The cell values provide the running time of the algorithms in seconds.

For GLI, SMK, BAS, ARC, REL, and PCM datasets, the running times of all methods are roughly comparable. However, for TOX, COIL, YALE, GLA, and CLL, GPI has required high running time. This is mainly because GPI uses multiple classifiers to determine the best subset and therefore the running time of all involved classifiers affects the running time of GPI. The datasets for which GPI has required high running time are of multiclass datasets. As can be seen in Table 5.4, for those datasets, the running time of the SVM classifier increased significantly. SVMs are inherently two-class classifiers. The most common way to convert SVMs into multiclass classifiers is one-against-all method that is to build up SVM classifiers equal to the number of classes and

Table 5.4: GPI running time analysis in seconds

	Dataset										
Method	GLI	SMK	TOX	BAS	COIL	ARC	REL	YALE	GLA	CLL	PCM
GPI	6.3	8.4	111.7	19.7	257.9	6.4	22.9	133.1	72.2	68.5	21.3
SVM-based	4.6	4.4	412.2	17.7	543.4	5.7	25.2	308	215.1	678	24.3
LDA-based	3.3	3.4	3.9	4.5	6.1	3.7	4.7	3.8	2.9	7.9	5.5
NB-based	5	6.8	10.4	8.8	48.3	6.9	8.7	27.9	21.2	58.1	9.3
KNN-based	3.3	2.6	3.2	4.9	4.6	3.9	4.1	3	2.3	6.1	5.3

to choose the class with highest score. Consequently, OAA is a time consuming method which has reflected in running time of GPI.

It must be mentioned that in this chapter one filtering step has applied prior to running the EC algorithm to reduce the complexity of the datasets (see subsection 5.3.1). GPI, unlike classifier-based methods, select a subset from a set of subsets and does not produce a new subset. Classifier-based methods take one subset as the baseline and add other features to the baseline if the classification performance of a desired classifier is improved.

This combination strategy is similar to sequential forward selection and in cases that the datasets contain large number of features; it requires a high computational cost, which can be even higher than the time necessary for the feature selection process. With filtering step, the dimension of the datasets has been reduced to 100 features which has helped classifier-based method to keep their running times low.

5.5 Conclusion

This chapter proposes a novel approach toward addressing the stability issue when using EC feature selection algorithms. In the proposed approach a selection process based on the generalisation power analysis of the subsets is adopted to select the best feature subset out of many subsets obtained from the output of an EC algorithm when executed several times (given that a

different feature set may be selected whenever the algorithm is run).

The proposed approach works based on an index called Generalisation Power Index (GPI) that measures the generalisation power of the subsets over multiple classifiers. GPI measures the quality of a feature subset using GPI index (Eq. 5.1) when applied on wide range of classifiers taking into account the optimal accuracy of the classifiers over the dataset.

A simple GA, which optimises the accuracy of the KNN classifier over a 10-fold cross-validation, was developed and adopted to select the best feature subsets. To validate the performance of the proposed approach, GPI was applied on the set of subsets obtained from the GA in different runs for eleven datasets and the goodness of GPI outputs were evaluated using five classifiers.

The proposed GPI was compared with various aggregation methods and a frequency-based method. The computation results confirmed that GPI has outperformed other methods for most of the cases, and for other cases similar results have been obtained.

EC algorithms are powerful search techniques that do not need domain knowledge, do not make any assumptions about the search space, and can produce multiple good solutions. However, their application to real-world has been limited due to their stability issue. EC algorithms reach different solutions whenever they are run and this is a problem, particularly in the biomedical domains, when a specific set of features is sought after to construct prediction models.

The proposed approach can help EC algorithms to be more applicable to real-world problems. Moreover, a subset with high generalisation power can guarantee near optimal classification performance over various classifiers. This can obviate the choice of classifier in cases that the best classifier for the data at hand is not known in advance and the selected subset can be used to eval-

uate the performance of different classifiers without having to repeat a feature selection process for each classifier. This solution can be of importance to researchers with limited combinatorial optimisation and machine learning knowledge.

The next chapter, Chapter 6 discusses the application of the proposed TAGA and GPI methods (these we described in Chapters 4 and 5 respectively), to a real-world large-scale microarray dataset for the task of breast cancer type classification.

Chapter 6

Application of methods to Breast Cancer type classification

6.1 Introduction

In recent years, microarray-based gene expression profiling has provided a better understanding of breast cancer [102]. Breast cancer consists of a group of different diseases characterized by distinct molecular aberrations rather than a specific disease with different histological characteristics and clinical results [102]. Breast tumor analysis using microarray data has significantly improved the taxonomy of disease and the discovery of new biomarkers for clinical practice [120, 101, 65, 37]. In this case, the prediction of intrinsic subtypes of breast cancer has known as a valuable strategy for determining the diagnosis and prognosis of patients and their response to therapy [86]. Moreover, the high quality of the microarray gene expression dataset processed by the Molecular Taxonomy of Breast Cancer International Consortium (METABRIC) [30] offers a unique chance to discover biomarkers that best discriminate against intrinsic subtypes.

However, such a high-dimensional dataset makes computing expensive and can complicate a predictive model's interpretation. To address these issues, feature selection methods are applied on the METABRIC dataset to extract the most informative biological information (e.g. biomarkers) to reduce the expression dataset into the smallest possible subset of genes predictors. Mucaki et al. [90] used various feature selection and machine learning methods including SVM, BE, and mRMR to identify optimal subsets of genes that can accurately predict therapeutic response of patients.

Milioli [86] analysed the application of a ranking feature method based on newly propose evaluation metric called CM1 score [84] to identify novel biomarkers for subtype individuation that naturally appears from the METABRIC breast cancer data set. Yang et al. [129] designed a forward selection-like feature selector for molecular subtype classification in which the features are sequentially added to the subset and prognostic score of the subset is maximised as measure function and the algorithm stops adding new feature to the subset when overfitting occurs. Firoozbakht et al. [42] used a chi-square feature selection algorithm to select the most informative genes for developing a predictive model to predict breast cancer subtypes of METABRIC dataset. Selecting a subset of relevant features is crucial to the analysis of high-dimensional data. In fact, feature selection has three main advantages: it reduces computational costs, mitigates the possibility of overfitting due to high inter-variable correlations, and makes the model easier to interpret clinically [13].

In terms of selecting final subset, feature selection algorithms can be divided into two categories: deterministic algorithms and non-deterministic algorithms. In a deterministic algorithm including greedy search algorithms and score-based ranking methods, for a given particular input, the algorithm will always select the same final subset but in case of a non-deterministic algorithm (such

as EC algorithms), for the same input, the algorithm may produce different final subsets in different runs which is known as the stability issue (discussed in Chapter 3). However, for some domains of application, particularly biomedical data, a set of specific features is sought after to construct predictive models which ensures optimum results in terms of both predictive performance and stability (i.e. robustness to changes in parameters and input data). This could be a reason as to why EC techniques have not been widely applied in real-world datasets, particularly METABRIC dataset.

To deal with the stability issue of EC algorithms, a generalisation power approach was developed in chapter 5 in which a further selection process based on the generalisation power analysis of the subsets is adopted to select a best subset out of many subsets obtained from the output of an EC algorithm in several runs. The proposed approach works based on GPI that measures generalisation power of the subsets over multiple classifiers.

In fact, GPI measures how closely a subset has been able to follow optimal subsets of multiple classifiers in terms of classification performance. For applying GPI analysis to the METABRIC dataset, TAGA embedded in a Filter/filter hybrid framework, developed in chapter 4, is run several times and the final subsets are saved for further processing through the GPI approach.

TAGA, is a string type long-term memory TS hybridised with an integer-coded AGA [21] as the LS in order to provide new search directions for the algorithm. TAGA uses mRMR criterion [98] to evaluate the subset and therefore the selected subset is not classifier-biased.

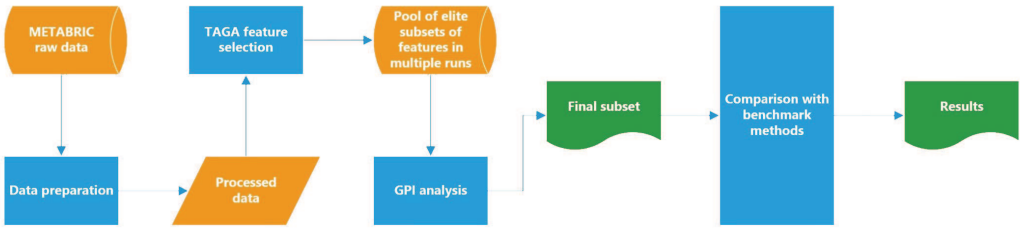

Figure 6.1: Experimental methodology flowchart

6.2 Experimental design

6.2.1 Experiment methodology

Figure 6.1 shows the flowchart of steps adopted to obtain experimental results. In the first step, the raw METABRIC data is extracted and appropriate data pre-processing methods are applied to prepare the data.

In the next step, TAGA is applied on the data to discover the most informative features for predicting cancer subtypes. TAGA is an EC algorithm it and produces a different solution in different runs, and for this reason TAGA was executed multiple times (i.e. multiple runs) and at the end of each run the best selected subset of features was saved into a pool of elite subsets. After that, a generalisation power analysis is performed on the elite subsets available in the pool to find the best subset.

Finally, the performance of final subset is compared with the other subsets obtained through other available benchmark feature selection methods in the literature namely: CM1 score [84], Chi-squared [74], mRMR [98], and BE-SVM [90]. These steps are elaborated in the following subsections.

Subtype	Basal	Her2	LumA	LumB	Normal	N/A
# Samples	331	239	715	490	199	6

Table 6.1: The number of samples corresponding to each subtype

6.2.2 Data preparation

Dataset description

The breast cancer microarray dataset integrated by Molecular Taxonomy of Breast Cancer International Consortium (METABRIC), known as METABRIC dataset, is used in this study which is hosted by the European Bioinformatics Institute (EBI) and stored in the European Genome-Phenome Archive (EGA)[1], under the EGAS00000000083 accession number. It consists of transcriptomic information processed on the Illumina HT-12 v3 platform (cDNA microarray profiling), as described in [125]. METABRIC divided the log2-normal gene expression values of primary tumors into two subsets: training (997 samples) and validation (989 sample) and each sample contains expression information of 48,803 probe IDs (features).

METABRIC dataset contains information on the long-term clinical and pathological outcomes of patients,, including the sample assignment into intrinsic subtypes. In the early 2000s, five molecular subtypes were proposed for breast cancer: luminal A, luminal B, HER2-enriched, normal-like and basal-like breast tumours [99, 110]. Later, Parker et al. [96] proposed a list of 50 genes for identifying subtypes of METABRIC samples known as PAM50 in conjunction with the Prediction Analysis for Microarrays (PAM) classification algorithm [114]. The numbers and the percentage of samples corresponding to each subtype are listed in Table 6.1 and Figure 6.2 respectively.

[1]http:// www.ebi.ac.uk/ega/

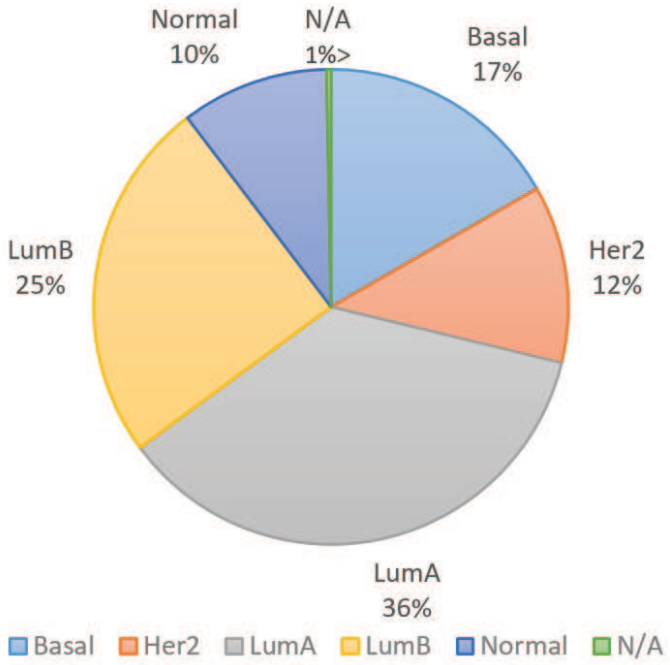

Figure 6.2: Percentage plot of subtypes in METABRIC dataset

Data pre-processing

There are 22 missing values in the data and there are 6 samples for which the subtype of cancer is not available. To pre-process the data, the samples with unavailable subtype are removed from the data and then the missing values are imputed using mean imputation method. Mean imputation is a method in which a certain variable's missing value is replaced by the mean of the available cases. After pre-processing the data, the number of samples is reduced to 1980 with 48803 probe values (e.g. variables).

Data normalization

z-score method is employed to normalise the data. Z-score is a data normalisation strategy that handle outlier issue in the data. The advantage of the transformation of the Z-score is that in a set of raw scores it takes into account both the mean value and the variability. Z-score indicates how much a value

deviates from the mean of using the following equation.

$$Z - score = \frac{value - \mu}{\sigma} \tag{6.1}$$

where, μ stands for the mean value of the feature, and σ presents the standard deviation. If a value is exactly the same as the mean value of the feature, it will be normalised to 0. If it is lower than the mean, it is normalised to a negative value, and if it is greater than the mean it is normalised to a positive value. The magnitude of these negative and positive values is determined by the original feature's standard deviation. If the unnormalised data has a high standard deviation, the normalised values are closer to 0.

6.2.3 Benchmark methods

CM1

The CM1 score is a supervised univariate method utilised for measuring the difference of samples' expression levels in two or more different classes [84]. For each breast cancer intrinsic subtype, CM1 ranks the features to select highly discriminative ones. Let X and Y be a partition of a set of samples into two classes, with X the class of interest and Y all other classes. A sample belongs either to class X or to class Y. For each probe i the CM1 score is calculated using Equation 6.2.

$$CM1_i(X, Y) = \frac{\bar{x}_i - \bar{y}_i}{1 + (max\{y_i\} - min\{y_i\})} \tag{6.2}$$

where, $\bar{x}_i$ is probe i mean expression value for the samples in class X, $\bar{y}_i$ shows probe i mean expression value for samples in class Y, $max\{y_i\}$ and $min\{y_i\}$ are probe i minimum and maximum expression values for samples in the class

Y.

Equation 6.2 can be interpreted as the normalised difference between the mean expression values in the class X and the class Y. The normalisation is proportional to the range of values in class Y. To find the most discriminative probes (e.g. features), as proposed by [86], for each breast cancer subtype, the CM1 score of the probes for each 5 subtypes is calculated using Equation 6.2 which results in 5 lists of CM1 scores. Then, for each subtype, the 10 most important probes (5 with the greatest positive CM1 score values and 5 with the smallest negative CM1 score values) are chosen. As one probe might be chosen for more than one subtype, only unique probes are selected as final subset of probes.

Chi-squared Method

Chi-squared is an efficient method for numerical data feature selection that automatically and adaptively discretises and selects numerical features [74]. Chi-squared is a univariate filter-based on the c2 statistic [74]. This method evaluates the godness of each feature by calculating the the chi-squared statistic value taking into account the classes. The higher the chi-squared statistic value is, the more relevant the feature will be. This method evaluates the relevance of each feature independently an therefore, it is usually fast in terms of computation time.

mRMR

Peng et al. [98] proposed a sequential forward selection algorithm in which information theory-based definitions of mRMR is maximised to select those features that have the highest relevance with the target class and the lowest redundancy. The mRMR equation is explianed as follows.

$$mRMR = \max_{j \in Q-S} \left[I(f_i, y) - \frac{1}{S} \sum_{s \in S} I(f_j, f_s) \right] \tag{6.3}$$

where f_j is the jth feature in the initial F-dimensional feature space, f_s is a variable that has already been selected in the feature subset S, s is an individual feature and Q contains all the features in the initial feature space, S contains the selected features and $Q-S$ contains those features that are not selected. The features are sequentially added to the features subset starting from an empty set and only those features are kept that maximise mRMR value of the entire subset. This process continues until the stop criterion is met.

Backward Elimination-SVM (BE-SVM)

Backward elimination feature selection is a greedy algorithm in which one feature of the set is left out in a reduced feature set and the classification is then assessed using SVM classifier; features that maintain or lower the miss-classification rate are kept in the subset and the rest are discarded. The procedure is repeated until the subset with the lowest miss-classification rate is selected as the optimal subset of features.

6.2.4 Learning algorithms and evaluation metric

During the experiments, the subsets are evaluated using five conventional machine learning classifiers, namely SVM, KNN, CART, NB, and LDA. For classifying the METABRIC dataset, the number of k for the KNN classifier was set to $K = 5$, the kernel function for SVM classifier was set to linear function, the discrimination type for LDA was set to diagonal co-variance, and data distribution for NB classifier was set to normal distribution. All the parameters

were chosen experimentally.

To guarantee valid results for making reliable predictions, 10-fold cross-validation was adopted in which the data samples are divided into roughly 10 equal folds and in each of 10 validation processes, one fold is taken as testing set and the other nine folds are used to train the learning algorithm. At the end of the k-fold validation process, a mean accuracy value was obtained for each validation set of each fold, and hence the ten values were averaged to provide overall classification performance.

6.3 Results and discussion

This section brings together the proposed methods, TAGA (chapter 4) and GPI (chapter 5), and describes the experiments carried out to determine the effectiveness of combining TAGA embedded into a filter/filter framework with GPI to select the most informative features from the METABRIC dataset for the breast cancer subtype classification task. The proposed approach is compared to other existing methods in the literature, namely CM1 [86], Chi-squared [42], mRMR [90], and BE-SVM [90].

For obtaining computation results, in the first stage of the TAGA, Fisher score algorithm is experimentally set to select the top 100 features of the METABRIC dataset and a reduced dataset is formed. Then, TAGA is experimentally set to run 30 times on the reduced dataset to select the best subset. At the end of each run, the best subset is transferred into a pool of elite subsets. After the 30th run when there are enough elite subsets in the pool, GPI analysis is applied on the subsets to choose the best final subset. The best subset is then evaluated using various classifiers and compared with the subsets obtained from alternative methods in terms of classification performance

using a 10-fold cross validation method. It must be mentioned that in order to have a fairer comparison, Fisher score was also applied on the dataset for other competing algorithms in the same way as TAGA and the subset cardinality range for all the algorithms is experimentally set to 50 features which allows the algorithm to select up to 50 features in their final subset.

Table 6.2 shows the 10-fold cross validation classification accuracy results obtained for 5 classifiers over the METABRIC dataset using combination of TAGA with GPI (denoted by TAGA-GPI in Table 6.2) selected subset compared to the subsets selected by other four methods. In Table 6.2, the values in parenthesis show the number of selected features for each algorithm, the cell values present the percentage of classification accuracy for a pair of classifier-algorithm, the last column presents the average of four classification accuracies for each algorithm, and the highlighted values show the best classification performances.

As can be seen in Table 6.2, TAGA-GPI has outperformed other methods for LDA, NB, and KNN and for CART classifier TAGA-GPI has also performed better than the other algorithms except Chi-squared method for which comparable accuracy has been obtained. TAGA-GPI has also had the best average accuracy over five classifiers (last column of Table 6.2) which indicates the selected features provide better generalisation power over wide range of classifiers.

However, for SVM classifier, BE-SVM algorithm has had the best performance. For SVM classifier, TAGA has not had the best performance and BE-SVM has outperformed it. This result is predictable because in BE-SVM, the BE algorithm directly searches for a subset of features which optimise the classification performance of the SVM classifier. Therefore, it is not surprising that the selected subset has had the best performance over SVM classifier. On the

other hand, TAGA-GPI tries to find a subset which has the highest generalisation power over a range of classifiers. Therefore, the selected subset through TAGA-GPI might not achieve an optimal accuracy over a specific classifier but it achieves near optimal accuracy for all classifiers. That is why the average accuracy for five classifiers (last column of Table 6.2) is higher for TAGA-GPI compared to BE-SVM.

Classifier	SVM (%)	LDA (%)	NB (%)	KNN (%)	CART (%)	Average (%)
BE-SVM (50)	**83.69**	74.11	74.87	77.46	71.28	76.28
CM1 (31)	80.55	74.77	76.19	75.94	71.28	75.75
mRMR (50)	76.85	65.55	68.24	73.66	64.84	69.83
Chi-Squared (50)	81.00	73.51	75.08	78.82	**72.59**	76.20
TAGA-GPI (18)	81.76	**78.88**	**79.03**	**78.98**	71.63	**78.10**

Table 6.2: Comparison of TAGA-GPI with other methods over METABRIC dataset. For each classifier and selection method, the values in parentheses is the number of selected features by each algorithm and the cell values are the classification accuracy. The last column reports the average of the classification accuracies for each algorithm.

6.4 Conclusion

In this chapter, the application of the developed TAGA embedded into a filter/filter framework and GPI approach described in chapters 4 and 5 respectively on the METABRIC cancer subtype classification problem was investigated. The METABRIC dataset contains expression profiles to identify breast cancer subgroups in an effort to help physicians provide better treatment recommendations to patients.

The feature selection methods which have been used used so far in the literature to select the most informative features for METABRIC dataset are deterministic greedy search algorithms [84, 74, 98, 90] which make locally optimal choices with the aim of finding a global optimum amongst local optima and

therefore, usually are stuck into local optimum. However, EC-based feature selection algorithms which have powerful global search capability and have not been considered as an option for METABRIC dataset in the literature thus far to the best of the author's knowledge. This is mainly because, for biomedical and healthcare classification purposes a specific subset of features is sought after to build up machine learning algorithms upon however, EC algorithms are random search algorithms meaning that they provide stochastic solutions rather than deterministic solutions.

To employ an EC algorithm in real-world application, in this chapter TAGA and GPI methods which were previously developed in chapters 4 and 5 are combined to select the best subset of features for METABRIC dataset. In the experiments described in this chapter, TAGA was firstly run independently multiple times and the final subsets in each run were further analysed using the GPI approach to select the best subset. Finally, the selected subset was compared against the subset selected by greedy feature selection methods which are currently being used for METABRIC dataset in terms of classification performance over various classifiers. experimental results on the METABRIC show that the proposed approach of combination of TAGA and GPI is promising for the biomedical feature selection task of finding the most stable subset of features for building future proof prediction models.

In Chapter 7, the solutions to address EC-based algorithms limitations for feature selection are summarised, the contribution and the objectives of the thesis are reviewed, and suggestions for future work are provided.

Chapter 7

Conclusions, Discussion and Future Work

7.1 Conclusions and discussion

This thesis proposes solutions to deal with the problem of EC-based feature selection algorithms for selecting a set of features for building machine learning models. For this purpose, four objectives were defined and for each of which a solution was proposed. Each of these objectives, achievements, and future work are described in the discussion that follows.

- Objective O1: Develop an EC-based feature selection algorithm which benefits from new solution representation and search components to reduce computation time taken by EC algorithms for finding optimal or near optimal subsets of features within high-dimensional datasets for building machine learning models.
- Objective O2: Develop a solution to address classifier-bias problem associated with EC algorithm embedded into wrapper frameworks, particularly filter/wrapper approaches, for which the selected features are

biased toward the utilised classifier. The proposed solution will select the features independent of the classification performance of any classifier and therefore, the selected feature will be able to provide acceptable performance over wide range of classifiers.

Objectives 1 and 2 are related and therefore the contribution for both objectives is discussed below. To resolve the computation time limitation of EC algorithms for feature selection (Objective 1) as well as the classifier-bias problem of filter/wrapper approaches (Objective 2), a novel hybrid feature selection framework called filter/filter approach was proposed (see Chapter 4). The filter/filter is a two-stage feature selection approach which combines two filter feature selection algorithms. In the first stage, Fisher score was used to reduce the search space by selecting the most informative features which were fed as input into the second stage. The first stage reduces the size of the original dataset and as a result reduces the computation time required by EC algorithm to process the reduced dataset in the second stage. Fisher score has shown robust and promising performance [82, 93, 127] for feature selection problems. However, Fisher score can be replaced with other ranking algorithm for a better performance on specific datasets if needed.

In the second stage, a new EC-based feature selection algorithm was developed which is called TAGA. TAGA is a mutation-based GA hybridised with a long-term memory TL and guided by a SFS procedure. TAGA benefits from new solution representation and search components which are able to significantly reduce the computation time. The solution representation for TAGA is an integer-encoded one which is composed of two parts for both selected features of the subset and unselected features which are used to explore new regions of the search space through a novel mutation operator (see Fig. 4.2). The proposed mutation swaps the features in the selected part of the solu-

tion with the features from unselected part taking into account the quality of features in terms of the mutual information between the features and the target. In addition, a new TL encoding scheme was proposed in order to make the solution storing and restoring processes computationally more effective. The new encoding scheme converts the solutions into string values, which need less computation time to be processed. To overcome classifier-bias issue, the mRMR evaluation metric was employed as the fitness function for TAGA which statistically evaluates the subsets and therefore, the selection process is independent of any classifier and the selected subsets are not biased toward a specific classifier.

Exhaustive experiments were carried out and the proposed TAGA was compared with greedy search and the state-of-the-art algorithms found in the literature. To have a fair performance comparison, the Fisher score was used to reduce the datasets for all comparing algorithms. The computation results confirmed that TAGA outperformed most of the algorithms in terms of classification performance. In terms of the computation time, TAGA performed relatively fast and required comparable computation time to the alternative algorithms.

- Objective O3: Develop a solution to the stability issue [128] associated with the challenge of finding the best subset of features over several runs, when EC algorithms are adopted for feature selection tasks. A solution would be based on a Generalisation Power Index (GPI) which measures the performance of feature subsets in terms of generalisation power over multiple classifiers.

In order to stabilise an EC algorithm, its random factors need to be removed from the search process which is against stochastic nature of EC algorithm. To

address stability issue of EC algorithms for feature selection (Objective 3), this thesis propose a further subset selection to find the most stable set of features based on based on the generalisation power analysis. The proposed approach works based an index called GPI that measures the generalisation power of the subsets in a pool of subsets obtained from the output of an EC algorithm when executed several times over multiple classifiers (see Chapter 5). In fact, GPI measures the quality of a feature subset when applied on wide range of classifiers taking into account the optimal accuracy of the classifiers over the dataset. Therefore, GPI select the best subset which is able to achieve optimal or near optimal accuracy when applied over wide range of classifiers. The features of the best subset are considered a stable set of feature representing the final output of the EC algorithm. To validate the performance of the proposed approach, GPI was applied on the set of subsets obtained from a test case EC algorithm in different runs and the proposed GPI was compared with various alternative methods. The computation results confirmed that GPI outperformed other algorithms in finding a stable set of features which are able to provide acceptable classification performance over wide range of classifiers. The proposed GPI approach can help EC algorithms to be more applicable to real-world problems which applications is currently limited due to their stability issue.

- Objective 04: Evaluate the performance of the proposed algorithms on a real-world case study in particular the METABRIC breast cancer dataset. METABRIC dataset contains a large number of features and many samples, and the proposed algorithms are applied to METABRIC dataset in order to find the best biomarkers for detecting breast cancer subtypes.

To evaluate the performance of the proposed TAGA and GPI (Chapters 4 and 5) on a real-world case study (Objective 4), the algorithms were employed to select most informative biomarkers for METABRIC cancer subtype classification problem. For this purpose, TAGA and GPI were sequentially combined in such a way that TAGA was firstly run independently multiple times and the final subsets in each run were stored in a pool and then were analysed further using GPI approach to select the best subset. The subset selected by the combination of TAGA and GPI was compared against the subset selected by other feature selection methods which are currently being used for METABRIC dataset in terms of classification performance over various classifiers. The computation results proved that not only, the combination of TAGA embedded into a filter/filter framework and GPI was able to address the limitations of EC algorithms for a biomedical real-world problem but, it also performed better than alternatives methods in terms of finding a stable set of features which achieves optimal or near optimal accuracy when applied on various classifiers. Therefore, the proposed approaches are promising for the biomedical feature selection tasks of finding the most stable subset of features for building future proof prediction models.

7.2 Future work

In this thesis some limitations and challenges of EC algorithms for feature selection were addressed. However, during the research, other challenges were identified which can be the topic of future studies.

Firstly, hybrid feature selection algorithms employ an algorithm to filter out the most informative features. However, it was observed that the number of features filtered out affect classification performance and therefore in this

thesis a set of experiments were performed to empirically adjust the number of features to be filtered out. The question which arises here is:

- what methods or algorithms can be implemented to efficiently adjust the number of features to be filtered out in the filtering stage in order to identify the features which achieve optimum machine learning performance?

Secondly, another observation was that applying filtering algorithms in the filtering stage would boost the overall classification performance compared to when the feature selection process is applied on the original dataset. However, using a different filtering algorithm for a different dataset may lead to better results. Consequently, another question is:

- how to efficiently choose the best filtering algorithm for a specific dataset, and/or is there any way to develop a filtering algorithm which performs well when applied on any dataset?

Thirdly, TAGA benefits from a novel integer-encoded solution representation which allows statistical evaluation metrics (such as mRMR) to be applied as fitness functions for EC algorithms as opposed to when binary solution representation is employed. However, the integer-encoded representation requires the user to define the size of cardinality to be discovered in advance. In this thesis, the cardinality size was defined experimentally. Two questions arise here:

- what methods can be implemented to identify the optimum cardinality size of feature subsets from high-dimensional datasets which contain a large number of features?

- is it possible that an algorithm automatically finds the optimum cardinality size rather than it to be user or experimentally defined?

Fourthly, TAGA is an enhanced EC algorithm which addresses the limitations of EC algorithms for feature selection, particularly in terms of computation time. Furthermore, the designed TL for TAGA helps it to have high diversity amongst solutions. However, algorithm parallelisation can significantly improve the performance of an EC algorithm in terms of solution diversity and computation time. The question that arises here is:

- what is the best approach to parallelising TAGA and what is the impact of the parallelised TAGA on high-dimensional data which is of low or high sample size?

Finally, the effectiveness of the proposed approaches was confirmed on a large-scale breast cancer type classification problem. The proposed approaches are promising for various tasks which concern finding the most stable subset of features for building future proof prediction models. Further future work involves investigating the effectiveness of the proposed approaches for other tasks that require building machine learning models using a stable set of features including those for biomedical, engineering, clinical and other applications.

Bibliography

[1] Adair, J., Brownlee, A. E., and Ochoa, G. Mutual information iterated local search: A wrapper-filter hybrid for feature selection in brain computer interfaces. In *International Conference on the Applications of Evolutionary Computation* (2018), Springer, pp. 63–77.

[2] Aha, D., and Kibler, D. Instance-based prediction of heart-disease presence with the cleveland database. *University of California 3*, 1 (1988), 3–2.

[3] Ali, Z. A. Concentric tabu search algorithm for solving traveling salesman problem. Master's thesis, Eastern Mediterranean University (EMU)-Doğu Akdeniz Üniversitesi (DAÜ), 2016.

[4] Alsharoa, A., Celik, A., and Kamal, A. E. Energy-efficient 5g networks using joint energy harvesting and scheduling. *5G Networks: Fundamental Requirements, Enabling Technologies, and Operations Management* (2018), 427–452.

[5] Álvarez-Estévez, D., Sánchez-Maroño, N., Alonso-Betanzos, A., and Moret-Bonillo, V. Reducing dimensionality in a database of sleep eeg arousals. *Expert Systems with Applications 38*, 6 (2011), 7746–7754.

[6] Amaldi, E., and Kann, V. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science 209*, 1-2 (1998), 237–260.

[7] Amirghasemi, M., and Zamani, R. An effective asexual genetic algorithm for solving the job shop scheduling problem. *Computers & Industrial Engineering 83* (2015), 123–138.

[8] Asuncion, A., and Newman, D. Uci machine learning repository, 2007.

[9] Back, T., Fogel, D. B., Whitley, D., and Angeline, P. J. Mutation operators. *Evolutionary computation 1* (2000), 237–255.

[10] Battiti, R. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks 5*, 4 (1994), 537–550.

[11] Beirlant, J., Dudewicz, E. J., Györfi, L., and Van der Meulen, E. C. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences 6*, 1 (1997), 17–39.

[12] Bermingham, M. L., Pong-Wong, R., Spiliopoulou, A., Hayward, C., Rudan, I., Campbell, H., Wright, A. F., Wilson, J. F., Agakov, F., Navarro, P., et al. Application of high-dimensional feature selection: evaluation for genomic prediction in man. *Scientific reports 5* (2015).

[13] Bichindaritz, I., Englebert, C., Regua, A., and Kotula, L. Feature selection and case-based reasoning for survival analysis in bioinformatics. In *The Thirty-First International Flairs Conference* (2018).

[14] Blelloch, G. E., and Maggs, B. M. Parallel algorithms. *ACM Computing Surveys (CSUR) 28*, 1 (1996), 51–54.

[15] Bolón-Canedo, V., and Alonso-Betanzos, A. Ensembles for feature selection: A review and future trends. *Information Fusion 52* (2019), 1–12.

[16] Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. Distributed feature selection: An application to microarray data classification. *Applied soft computing 30* (2015), 136–150.

[17] Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A., Benítez, J. M., and Herrera, F. A review of microarray datasets and applied feature selection methods. *Information Sciences 282* (2014), 111–135.

[18] Bonilla-Huerta, E., Hernández-Montiel, A., Morales-Caporal, R., and Arjona-López, M. Hybrid framework using multiple-filters and an embedded approach for an efficient selection and classification of microarray data. *IEEE/ACM transactions on computational biology and bioinformatics 13*, 1 (2016), 12–26.

[19] Bonnlander, B. V., and Weigend, A. S. Selecting input variables using mutual information and nonparametric density estimation. In *Proceedings of the 1994 International Symposium on Artificial Neural Networks (ISANN94)* (1994), pp. 42–50.

[20] Bouzgou, H., and Gueymard, C. A. Minimum redundancy–maximum relevance with extreme learning machines for global solar radiation forecasting: Toward an optimized dimensionality reduction for solar time series. *Solar Energy 158* (2017), 595–609.

[21] Cantó, J., Curiel, S., and Martínez-Gómez, E. A simple algorithm for optimization and model fitting: Aga (asexual genetic algorithm). *Astronomy & Astrophysics 501*, 3 (2009), 1259–1268.

[22] Cantó, J., Curiel, S., and Martínez-Gómez, E. A simple algorithm for optimization and model fitting: Aga (asexual genetic algorithm). *Astronomy & Astrophysics 501*, 3 (2009), 1259–1268.

[23] Chakroborty, P., and Mandal, A. An asexual genetic algorithm for the general single vehicle routing problem. *Engineering Optimization 37*, 1 (2005), 1–27.

[24] Chandrashekar, G., and Sahin, F. A survey on feature selection methods. *Computers & Electrical Engineering 40*, 1 (2014), 16–28.

[25] Chen, L., Chu, C., and Feng, K. Predicting the types of metabolic pathway of compounds using molecular fragments and sequential minimal optimization. *Combinatorial Chemistry & High Throughput Screening 19*, 2 (2016), 136–143.

[26] Chen, L.-F., Su, C.-T., and Chen, K.-H. An improved particle swarm optimization for feature selection. *Intelligent Data Analysis 16*, 2 (2012), 167–182.

[27] Chen, Z., Lin, T., Tang, N., and Xia, X. A parallel genetic algorithm based feature selection and parameter optimization for support vector machine. *Scientific Programming 2016* (2016).

[28] Cosma, G., Brown, D., Archer, M., Khan, M., and Pockley, A. G. A survey on computational intelligence approaches for predictive modeling in prostate cancer. *Expert systems with applications 70* (2017), 1–19.

[29] Cui, Y., Wang, J., Liu, S. B., and Wang, L. G. Hyperspectral image feature reduction based on tabu search algorithm. *Journal of Information Hiding and Multimedia Signal Processing* (2015), 154–162.

[30] Curtis, C., Shah, S. P., Chin, S.-F., Turashvili, G., Rueda, O. M., Dunning, M. J., Speed, D., Lynch, A. G., Samarajiwa, S., Yuan, Y., et al. The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature 486*, 7403 (2012), 346.

[31] Dash, M., and Liu, H. Feature selection for classification. *Intelligent data analysis 1*, 3 (1997), 131–156.

[32] De Jong, K. A. *Evolutionary computation: a unified approach*. MIT press, 2006.

[33] Dhar, V., Chou, D., and Provost, F. Discovering interesting patterns for investment decision making with glowera genetic learner overlaid with entropy reduction. *Data Mining and Knowledge Discovery 4*, 4 (2000), 251–280.

[34] Ding, C., and Peng, H. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology 3*, 02 (2005), 185–205.

[35] Djellali, H., Djebbar, A., Zine, N. G., and Azizi, N. Hybrid artificial bees colony and particle swarm on feature selection. In *IFIP International Conference on Computational Intelligence and Its Applications* (2018), Springer, pp. 93–105.

[36] Dowlatshahi, M. B., Derhami, V., and Nezamabadi-pour, H. A novel three-stage filter-wrapper framework for mirna subset selection

in cancer classification. In *Informatics* (2018), vol. 5, Multidisciplinary Digital Publishing Institute, p. 13.

[37] Dowsett, M., Sestak, I., Lopez-Knowles, E., Sidhu, K., Dunbier, A. K., Cowens, J. W., Ferree, S., Storhoff, J., Schaper, C., and Cuzick, J. Comparison of pam50 risk of recurrence score with onco type dx and ihc4 for predicting risk of distant recurrence after endocrine therapy. *Journal of Clinical Oncology 31*, 22 (2013), 2783–2790.

[38] Dréo, J., Pétrowski, A., Siarry, P., and Taillard, E. *Metaheuristics for hard optimization: methods and case studies*. Springer Science & Business Media, 2006.

[39] Eiben, A. E., Smith, J. E., et al. *Introduction to evolutionary computing*, vol. 53. Springer, 2003.

[40] Fan, Y., Li, X., and Zhang, P. Real-time static voltage stability assessment in large-scale power systems based on maximum-relevance minimum-redundancy ensemble approach. *IEEE Access 5* (2017), 27281–27291.

[41] Filannino, M. Dbworld e-mail classification using a very small corpus. *The University of Manchester* (2011).

[42] Firoozbakht, F., Rezaeian, I., Ngom, A., Rueda, L., and Porter, L. A novel approach for finding informative genes in ten subtypes of breast cancer. In *2015 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)* (2015), IEEE, pp. 1–6.

[43] Freitas, A. A. Understanding the crucial role of attribute interaction in data mining. *Artificial Intelligence Review 16*, 3 (2001), 177–199.

[44] Freitas, A. A. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer Science & Business Media, 2002.

[45] Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association 32*, 200 (1937), 675–701.

[46] Gendreau, M. An introduction to tabu search. *Handbook of metaheuristics* (2003), 37–54.

[47] Gheyas, I. A., and Smith, L. S. Feature subset selection in large dimensionality domains. *Pattern recognition 43*, 1 (2010), 5–13.

[48] Glover, F. Tabu search: A tutorial. *Interfaces 20*, 4 (1990), 74–94.

[49] Gu, Q., Li, Z., and Han, J. Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725* (2012).

[50] Gu, S., Cheng, R., and Jin, Y. Feature selection for high-dimensional classification using a competitive swarm optimizer. *Soft Computing 22*, 3 (2018), 811–822.

[51] Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *Journal of machine learning research 3*, Mar (2003), 1157–1182.

[52] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: an update. *ACM SIGKDD explorations newsletter 11*, 1 (2009), 10–18.

[53] Hancer, E. Differential evolution for feature selection: a fuzzy wrapper–filter approach. *Soft Computing* (2018), 1–16.

[54] Holland, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[55] Hsu, H.-H., Hsieh, C.-W., and Lu, M.-D. Hybrid feature selection by combining filters and wrappers. *Expert Systems with Applications 38*, 7 (2011), 8144–8150.

[56] Huerta, E., Montiel, A., Caporale, R., and Lopez, M. Hybrid framework using multiple-filters and an embedded approach for an efficient and robust selection and classification of microarray data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1, 1–1.

[57] Hui, K. H., Ooi, C. S., Lim, M. H., Leong, M. S., and Al-Obaidi, S. M. An improved wrapper-based feature selection method for machinery fault diagnosis. *PloS one 12*, 12 (2017), e0189143.

[58] James, G., Witten, D., Hastie, T., and Tibshirani, R. *An introduction to statistical learning*, vol. 112. Springer, 2013.

[59] Jeong, Y.-S., Shin, K. S., and Jeong, M. K. An evolutionary algorithm with the partial sequential forward floating search mutation for large-scale feature selection problems. *Journal of The Operational research society 66*, 4 (2015), 529–538.

[60] Joe, H. *Multivariate models and multivariate dependence concepts*. CRC Press, 1997.

[61] Jović, A., Brkić, K., and Bogunović, N. A review of feature selection methods with applications. In *Information and Communica-*

tion Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on (2015), IEEE, pp. 1200–1205.

[62] Ju, Z., and He, J.-J. Prediction of lysine glutarylation sites by maximum relevance minimum redundancy feature selection. *Analytical biochemistry 550* (2018), 1–7.

[63] Kachitvichyanukul, V. Comparison of three evolutionary algorithms: Ga, pso, and de. *Industrial Engineering and Management Systems 11*, 3 (2012), 215–223.

[64] Kalousis, A., Prados, J., and Hilario, M. Stability of feature selection algorithms. In *Fifth IEEE International Conference on Data Mining (ICDM'05)* (2005), IEEE, pp. 8–pp.

[65] Kelly, C. M., Bernard, P. S., Krishnamurthy, S., Wang, B., Ebbert, M. T., Bastien, R. R., Boucher, K. M., Young, E., Iwamoto, T., and Pusztai, L. Agreement in risk prediction between the 21-gene recurrence score assay (oncotype dx®) and the pam50 breast cancer intrinsic classifier in early-stage estrogen receptor–positive breast cancer. *The oncologist 17*, 4 (2012), 492–498.

[66] Kira, K., and Rendell, L. A. A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning* (1992), pp. 249–256.

[67] Kohavi, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* (1995), vol. 14, International Joint Conferences on Artificial Intelligence Organization (IJCAI), pp. 1137–1145.

[68] Kohavi, R. Wrappers for performance enhancement and oblivious decision graphs. Tech. rep., CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, 1995.

[69] Kononenko, I., Šimec, E., and Robnik-Šikonja, M. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence 7*, 1 (1997), 39–55.

[70] Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research 5*, Apr (2004), 361–397.

[71] Li, J., Cheng, K., Wang, S., Morstatter, F., Robert, T., Tang, J., and Liu, H. Feature selection: A data perspective. *arXiv:1601.07996* (2016).

[72] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR) 50*, 6 (2018), 94.

[73] Li, R., Lu, J., Zhang, Y., and Zhao, T. Dynamic adaboost learning with feature selection based on parallel genetic algorithm for image annotation. *Knowledge-Based Systems 23*, 3 (2010), 195–201.

[74] Liu, H., and Setiono, R. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence* (1995), IEEE, pp. 388–391.

[75] Liu, H., and Yu, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering 17*, 4 (2005), 491–502.

[76] Liu, T., Hu, L., Ma, C., Wang, Z.-Y., and Chen, H.-L. A fast approach for detection of erythemato-squamous diseases based on extreme learning machine with maximum relevance minimum redundancy feature selection. *International Journal of Systems Science 46*, 5 (2015), 919–931.

[77] Liu, Y., Tang, F., and Zeng, Z. Feature selection based on dependency margin. *IEEE Transactions on Cybernetics 45*, 6 (2015), 1209–1221.

[78] Lu, H., Chen, J., Yan, K., Jin, Q., Xue, Y., and Gao, Z. A hybrid feature selection algorithm for gene expression data classification. *Neurocomputing 256* (2017), 56–62.

[79] Ludwig, O., Nunes, U., Araújo, R., Schnitman, L., and Lepikson, H. A. Applications of information theory, genetic algorithms, and neural models to predict oil flow. *Communications in Nonlinear Science and Numerical Simulation 14*, 7 (2009), 2870–2885.

[80] Ma, X., Guo, J., and Sun, X. Sequence-based prediction of rna-binding proteins using random forest with minimum redundancy maximum relevance feature selection. *BioMed research international 2015* (2015).

[81] Mafarja, M. M., and Mirjalili, S. Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection. *Soft Computing* (2018), 1–17.

[82] Malina, W. On an extended fisher criterion for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5 (1981), 611–614.

[83] Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., and Califano, A. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC bioinformatics 7*, 1 (2006), S7.

[84] Marsden, J., Budden, D., Craig, H., and Moscato, P. Language individuation and marker words: Shakespeare and his maxwell's demon. *PloS one 8*, 6 (2013), e66813.

[85] Michalewicz, Z., Fogel, D. B., and Bèack, T. *Evolutionary Computation. Vol. 2, Advanced Algorithms and Operators.* Taylor & Francis, 2000.

[86] Milioli, H. H., Vimieiro, R., Riveros, C., Tishchenko, I., Berretta, R., and Moscato, P. The discovery of novel biomarkers improves breast cancer intrinsic subtype prediction and reconciles the labels in the metabric data set. *PLoS One 10*, 7 (2015), e0129711.

[87] Mohananthini, N., and Yamuna, G. Comparison of multiple watermarking techniques using genetic algorithms. *Journal of Electrical Systems and Information Technology 3*, 1 (2016), 68–80.

[88] Mokshin, V., Saifudinov, I., Sharnin, L., Trusfus, M., and Tutubalin, P. A parallel genetic algorithm of feature selection for analysis of complex system. In (2018), pp. 2874–2883.

[89] Molinaro, A. M., Simon, R., and Pfeiffer, R. M. Prediction error estimation: a comparison of resampling methods. *Bioinformatics 21*, 15 (2005), 3301–3307.

[90] Mucaki, E. J., Baranova, K., Pham, H. Q., Rezaeian, I., Angelov, D., Ngom, A., Rueda, L., and Rogan, P. K. Pre-

dicting outcomes of hormone and chemotherapy in the molecular taxonomy of breast cancer international consortium (metabric) study by biochemically-inspired machine learning. *F1000Research 5* (2016).

[91] Naghibi, T., Hoffmann, S., and Pfister, B. A semidefinite programming based search strategy for feature selection with mutual information measure. *IEEE transactions on pattern analysis and machine intelligence 37*, 8 (2015), 1529–1541.

[92] Nguyen, X. V., Chan, J., Romano, S., and Bailey, J. Effective global approaches for mutual information based feature selection. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), ACM, pp. 512–521.

[93] Olyaee, S., Dashtban, Z., and Dashtban, M. H. Design and implementation of super-heterodyne nano-metrology circuits. *Frontiers of Optoelectronics 6*, 3 (2013), 318–326.

[94] Pan, Y. A proposed frequency-based feature selection method for cancer classification. *Masters Theses and Specialist Projects* (2017).

[95] Papagelis, A., and Kalles, D. Breeding decision trees using evolutionary techniques. In *ICML* (2001), vol. 1, pp. 393–400.

[96] Parker, J. S., Mullins, M., Cheang, M. C., Leung, S., Voduc, D., Vickery, T., Davies, S., Fauron, C., He, X., Hu, Z., et al. Supervised risk predictor of breast cancer based on intrinsic subtypes. *Journal of clinical oncology 27*, 8 (2009), 1160.

[97] Parzen, E. On estimation of a probability density function and mode. *The annals of mathematical statistics 33*, 3 (1962), 1065–1076.

[98] Peng, H., Long, F., and Ding, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence 27*, 8 (2005), 1226–1238.

[99] Perou, C. M., Sørlie, T., Eisen, M. B., Van De Rijn, M., Jeffrey, S. S., Rees, C. A., Pollack, J. R., Ross, D. T., Johnsen, H., Akslen, L. A., et al. Molecular portraits of human breast tumours. *nature 406*, 6797 (2000), 747.

[100] Piniganti, L. A survey of tabu search in combinatorial optimization.

[101] Prat, A., Ellis, M. J., and Perou, C. M. Practical implications of gene-expression-based assays for breast oncologists. *Nature reviews Clinical oncology 9*, 1 (2012), 48.

[102] Reis-Filho, J. S., and Pusztai, L. Gene expression profiling in breast cancer: classification, prognostication, and prediction. *The Lancet 378*, 9805 (2011), 1812–1823.

[103] Rodriguez-Lujan, I., Huerta, R., Elkan, C., and Cruz, C. S. Quadratic programming feature selection. *Journal of Machine Learning Research 11*, Apr (2010), 1491–1516.

[104] Ruiz, R., Riquelme, J. C., Aguilar-Ruiz, J. S., and García-Torres, M. Fast feature selection aimed at high-dimensional data via hybrid-sequential-ranked searches. *Expert Systems with Applications 39*, 12 (2012), 11094–11102.

[105] Salesi, S., and Cosma, G. A novel extended binary cuckoo search algorithm for feature selection. In *2017 2nd International Conference on*

Knowledge Engineering and Applications (ICKEA) (2017), IEEE, pp. 6–12.

[106] Shamir, J. Fundamental speed limitations on parallel processing. *Applied optics 26*, 9 (1987), 1567–1567.

[107] Shannon, C. E. A mathematical theory of communication. *Bell system technical journal 27*, 3 (1948), 379–423.

[108] Somol, P., and Novovicova, J. Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 11 (2010), 1921–1939.

[109] Somol, P., Vácha, P., Mikeš, S., Hora, J., Pudil, P., and Zid, P. Introduction to feature selection toolbox 3–the c++ library for subset search, data modeling and classification. *Research Report for Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic* (2010).

[110] Sørlie, T., Perou, C. M., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., Hastie, T., Eisen, M. B., Van De Rijn, M., Jeffrey, S. S., et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences 98*, 19 (2001), 10869–10874.

[111] Soufan, O., Kleftogiannis, D., Kalnis, P., and Bajic, V. B. Dwfs: a wrapper feature selection tool based on a parallel genetic algorithm. *PloS one 10*, 2 (2015).

[112] Souza, F., Matias, T., and Araújo, R. Co-evolutionary genetic multilayer perceptron for feature selection and model design. In *Emerg-*

ing Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on (2011), IEEE, pp. 1–7.

[113] Suleman, A. Parallel programming: When amdahls law is inapplicable? *Future chips, June* (2011).

[114] Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences 99*, 10 (2002), 6567–6572.

[115] Tint, Y., and Mikami, Y. A minimum redundancy maximum relevance-based approach for multivariate causality analysis. *International Journal of Advanced Computer Science and Applications 8*, 9 (2017), 13–20.

[116] Torlapati, J., and Clement, T. P. Using parallel genetic algorithms for estimating model parameters in complex reactive transport problems. *Processes 7*, 10 (2019), 640.

[117] Tsai, C.-F., and Hsiao, Y.-C. Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. *Decision Support Systems 50*, 1 (2010), 258–269.

[118] Umbarkar, A., and Joshi, M. Review of parallel genetic algorithm based on computing paradigm and diversity in search space. *ICTACT Journal on Soft Computing 3*, 4 (2013), 615–622.

[119] Unler, A., and Murat, A. A discrete particle swarm optimization method for feature selection in binary classification problems. *European Journal of Operational Research 206*, 3 (2010), 528–539.

[120] Van't Veer, L. J., Dai, H., Van De Vijver, M. J., He, Y. D., Hart, A. A., Mao, M., Peterse, H. L., Van Der Kooy, K., Marton, M. J., Witteveen, A. T., et al. Gene expression profiling predicts clinical outcome of breast cancer. *nature 415*, 6871 (2002), 530.

[121] Viegas, F., Rocha, L., Gonçalves, M., Mourão, F., Sá, G., Salles, T., Andrade, G., and Sandin, I. A genetic programming approach for feature selection in highly dimensional skewed data. *Neurocomputing 273* (2018), 554–569.

[122] Wang, S., Kong, W., Zeng, W., Hong, X., et al. Hybrid binary imperialist competition algorithm and tabu search approach for feature selection using gene expression data. *BioMed research international 2016* (2016).

[123] Wang, S., Zhang, Y., Zhan, T., Phillips, P., Zhang, Y.-D., Liu, G., Lu, S., and Wu, X. Pathological brain detection by artificial intelligence in magnetic resonance imaging scanning (invited review). *Progress in Electromagnetics Research 156* (2016), 105–133.

[124] Wang, Y., Li, L., Ni, J., and Huang, S. Feature selection using tabu search with long-term memories and probabilistic neural networks. *Pattern Recognition Letters 30*, 7 (2009), 661–670.

[125] Weigelt, B., Mackay, A., A'hern, R., Natrajan, R., Tan, D. S., Dowsett, M., Ashworth, A., and Reis-Filho, J. S. Breast cancer molecular profiling with single sample predictors: a retrospective analysis. *The lancet oncology 11*, 4 (2010), 339–349.

[126] Winkler, S. M., Affenzeller, M., Jacak, W., and Stekel, H. Identification of cancer diagnosis estimation models using evolutionary

algorithms: a case study for breast cancer, melanoma, and cancer in the respiratory system. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation* (2011), ACM, pp. 503–510.

[127] Xuan, J., Wang, Y., Dong, Y., Feng, Y., Wang, B., Khan, J., Bakay, M., Wang, Z., Pachman, L., Winokur, S., et al. Gene selection for multiclass prediction by weighted fisher criterion. *EURASIP Journal on Bioinformatics and Systems Biology 2007* (2007), 3–3.

[128] Xue, B., Zhang, M., Browne, W. N., and Yao, X. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation 20*, 4 (2016), 606–626.

[129] Yang, T.-H. O., Cheng, W.-Y., Zheng, T., Maurer, M. A., and Anastassiou, D. Breast cancer prognostic biomarker using attractor metagenes and the fgd3–susd3 metagene. *Cancer Epidemiology and Prevention Biomarkers 23*, 12 (2014), 2850–2856.

[130] Yousefpour, A., Ibrahim, R., and Hamed, H. N. A. Ordinal-based and frequency-based integration of feature selection methods for sentiment analysis. *Expert Systems with Applications 75* (2017), 80–93.

[131] Yu, L., and Liu, H. Efficient feature selection via analysis of relevance and redundancy. *Journal of machine learning research 5*, Oct (2004), 1205–1224.

[132] Zhang, M. Keynote talks: Evolutionary feature selection and dimensionality reduction. In *Intelligent and Evolutionary Systems (IES), 2017 21st Asia Pacific Symposium on* (2017), IEEE, pp. ix–xii.