Neurocomputing 214 (2016) 866–880

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A new hybrid filter–wrapper feature selection method for clustering based on ranking

Saúl Solorio-Fernández *, J. Ariel Carrasco-Ochoa, José Fco. Martínez-Trinidad

Computer Sciences Department, National Institute of Astrophysics, Optics and Electronics, Luis Enrique Erro # 1, Santa María Tonantzintla, Puebla 72840, Mexico

ARTICLE INFO

Article history:
Received 9 February 2014
Received in revised form
9 July 2016
Accepted 11 July 2016
Communicated by Swagatam Das
Available online 25 July 2016

Keywords:
Feature selection for clustering
Feature ranking
Laplacian score
Weighted normalized Calinski–Harabasz index

ABSTRACT

Feature selection is a common task in areas such as Pattern Recognition, Data Mining, and Machine Learning since it can help to improve prediction quality, reduce computation time and build more understandable models. Although feature selection for supervised classification has been widely studied, feature selection in the absence of class labels, namely feature selection for clustering or unsupervised feature selection, has been less addressed. Most existing unsupervised feature selection approaches suffer from the called "Bias of Criterion Values to Dimension," which arises when feature subsets with different cardinality are evaluated by an internal evaluation clustering criterion. In this paper, we introduce a new hybrid filter–wrapper method for clustering, which combines the spectral feature selection framework using the Laplacian Score ranking and a modified Calinski–Harabasz index. The proposed method in the filter stage sorts the features according to their relevance, while in the wrapper stage, through our modified Calinski–Harabasz index that takes into account the cardinality of the feature subsets under evaluation, evaluates the features considering them as a subset rather than individually by using two well-known selection strategies. Experiments on different datasets show that the proposed method alleviates the "Bias of Criterion Values to Dimension" and, identifies and selects more relevant features than those selected by other reported hybrid filter–wrapper feature selection methods for clustering. Additionally, we also contrast our results against other filter and wrapper methods of the state-of-the-art.

1. Introduction

Feature selection is an active research area in Pattern Recognition, Data Mining, and Machine Learning. The main idea of feature selection is to get a good subset of features for representing a dataset, so that those features providing little information are eliminated and therefore they are not taken into account for further analysis. Feature selection for supervised classification has been widely studied, and there is a considerable amount of literature that addresses this problem; works as [1–8] to name a few, have been reported in this research area. Excellent surveys and introduction about feature selection can be found in [9–13]. On the other hand, feature selection for clustering (unsupervised classification) is more challenging, since the class labels are unavailable to guide the search. This issue has been less addressed [14], even though in recent years some works have been published [15–19].

Feature selection for clustering consists of identifying a feature subset that allows building good quality clusters [20–22]. Furthermore, feature selection for clustering does not only reduce the size of the data and the run-time of learning algorithms, but also leads to more compact learning models and possibly with better generalization capability [23]. It is worth mentioning that some works, such as [11,24,25] claim that overfitting can be alleviated by performing feature selection in an unsupervised manner.

There are two main approaches for unsupervised feature selection problems: filter and wrapper [26]. The filter approach estimates the importance of a given feature subset relying solely on the inherent properties of data, such as variance, entropy, correlation, local preservation, among others, filter methods do not depend on any particular clustering algorithm, and generally they are fast and scalable [11]. These methods can be broadly classified into univariate and multivariate approaches [12,19]. The former are the most common, and they use some criteria to assess each feature sorting them into a list (ranking). This kind of methods can identify and remove irrelevant features, but they are unable to remove redundant features because these methods do not take into account possible dependencies between features. Some of the most

* Corresponding author.
E-mail addresses: sausolofer@ccc.inaoep.mx (S. Solorio-Fernández), ariel@ccc.inaoep.mx (J.A. Carrasco-Ochoa), fmartine@ccc.inaoep.mx (J.Fco. Martínez-Trinidad).

http://dx.doi.org/10.1016/j.neucom.2016.07.026
0925-2312/

important and effective univariate unsupervised filter methods of the state-of-the-art belong to one of the following categories: SVD-Entropy-based methods [25,27–29], Graph-based feature selection methods [30–33], and Similarity-Entropy-based methods [34–36]. Meanwhile, multivariate filter approaches [17,19,37–39] assume that dependent features should be discarded, being independent features, among each other, those with the highest relevance. This approach can handle both irrelevant and redundant features, which improves the accuracy of the results compared to univariate filter-based feature selection methods.

On the other hand, the wrapper approach [26,40–46] estimates the importance of a feature subset by evaluating its precision as the quality of the clustering result after applying a specific clustering algorithm. Wrapper methods are often characterized by the high quality of the selected feature subsets, but they have a high computational cost [11].

Another less explored approach for unsupervised feature selection comprises those methods [47–49] that combine filter and wrapper approaches (Hybrid). These methods attempt to have a reasonable compromise between efficiency (computational effort) and effectiveness (quality of the clusters built by using the selected features). However, most of the methods based on the wrapper approach and particularly the hybrid feature selection methods for clustering generally are biased towards feature subsets of low or high cardinality, which often leads to trivial results [26]. Another limitation of these methods is in the evaluation criteria used for determining the best feature subset. Since some evaluation measures like those based on a scatter separability criterion [26] produce bad results when the number of features is larger than the number of instances, or when two or more features are multiples one from each other [50].

In this paper, we propose a new hybrid filter–wrapper method for unsupervised feature selection combining spectral feature selection using the Laplacian Score ranking, and a modified Calinski–Harabasz index. Experiments over different standard real and synthetic datasets show that our proposed method has a reasonable compromise between quality and performance, providing a solution to the limitations mentioned above and outperforming other state-of-the-art hybrid feature selection methods. Additionally, as we also show in the experiments, our method selects relevant features, eliminating those that do not contribute improving the quality of the clusters.

The rest of this paper is organized as follows. Section 2 gives a brief review of previous works. Section 3 introduces the proposed method based on ranking and the Weighted Normalized Calinski Harabasz index. Section 4 shows experimental results and comparisons on real and synthetic datasets. Finally, Section 5 provides some conclusions and future work.

2. Related work

In the literature, some hybrid unsupervised feature selection methods have been proposed. Some of them are designed specifically for handling text data [51,52], while others address the feature selection for fault diagnostic [53] and bearing defects detention on rotary machine health assessment [54]. Likewise, in [55,56] hybrid feature extraction/selection techniques are proposed to identify relevant features in a transformed space, but not in the original one, which is out of the scope of this paper. On the other hand, there are other works such as those proposed by [57–59], which use a filter and wrapper approach for unsupervised feature selection. However, these works solve the problem from another different perspective; performing feature selection assuming that a set of clusters can be modeled as being a set of different classes, where they can apply traditional supervised feature selection methods on data. Since the proposed method in this paper follows a filter–wrapper approach, in this section, we briefly review some methods applicable to datasets not linked to a particular data type, and can be classified as hybrid filter–wrapper unsupervised feature selection methods.

One of the first hybrid feature selection methods for clustering was introduced by Dash and Liu [49], the main idea in this method is to order the features according to their importance, using a measure based on the entropy of the similarity of the data (filter stage). Then (wrapper stage), they use the *k*-means algorithm and a scatter separability criterion for evaluating feature subsets to get a feature selection. Meanwhile, in Li et al. [47], following the same idea of Dash and Liu [49], the authors propose a new hybrid method combining the exponential entropy index with the Fuzzy Feature Evaluation Index (FFEI) for evaluating feature subsets in the filter stage. Later in the wrapper stage, they use the fuzzy-cmeans algorithm and a scatter separability criterion to select what they called a "compact" feature subset. The two hybrid methods aforementioned have the disadvantage that when they are applied to data with a large number of instances, they become impractical, due to its high computational complexity. To reduce the number of objects, these methods propose to use random sampling of instances; however, for many real world problems, this may not be a good option, because, in random sampling, relevant information could be ignored, and the quality of the algorithms may change unpredictably and significantly [23].

In another work, Hruschka et al. [48] propose a hybrid approach that combines the *k*-means clustering algorithm and a Bayesian filter. This filter is based on a Bayesian network using Markov Blanket property for feature selection. However in this work, only datasets with less than 30 features were used due to its high computational cost; additionally, the authors only compare their results against to those obtained by the wrapper method proposed by Dy in [26].

All these works, try to take into account the benefits of filter and wrapper approaches combining them in a suitable manner; thus each proposed method uses its own selection strategies and evaluation measures. However, it makes difficult a fair comparison among the different methods reported in the literature, since in unsupervised classification there is no a standard evaluation measure of the clustering, mainly because class labels are not available [26,60].

As we can see, hybrid unsupervised feature selection methods have been little studied, and existing methods have limitations in either the evaluation criteria, the number of features able to process, or because they are specifically designed for certain type of data. Therefore, it is important to propose new hybrid unsupervised feature selection methods not linked to a particular data type such that they have a good balance between quality and performance. For this reason, in this paper, we propose a new hybrid feature selection method for clustering which is faster than previous methods based on feature ranking and, unlike the other hybrid methods, our method does not perform a random sampling of instances. Also, our modified Calinski–Harabasz index avoids the bias that arises when features subsets with different cardinality are evaluated in the same clusters.

3. Proposed method

In this section, we present our feature selection method for clustering, which follows a hybrid filter–wrapper approach and consists of two stages: (I) building a ranking of features, and (II) selection of a relevant feature subset. In the first stage (filter stage), features are sorted according to their relevance. In this stage, we aim to identify features that are consistent with the structure of the data. Features are sorted according to their relevance to narrow the search in the space of all possible feature subsets ($2^n - 1$ subsets for n features); this procedure allows starting with a good approximation for the second stage. In the second stage (wrapper stage), the idea is to evaluate the features

considering them as a subset rather than individually, by implementing two selection strategies. The following subsections explain in detail each stage of the proposed method.

3.1. Filter stage

In the literature, the spectral graph theory has been used to introduce some feature selection methods and clustering algorithms [24,30–32,61]. These methods evaluate features according to their agreement with the graph Laplacian matrix of data similarities.

Formally, given a $X = \{\boldsymbol{x}_i\}_{i=1}^{m}$ dataset consisting of m instances, where $\boldsymbol{x}_i \in \mathbb{R}^n$, and a similarity measure, a matrix of similarities $W_{m\times m}$, representing the similarity between all pairs of instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$; $i, j = 1, \ldots, m$, can be built. We can interpret W as a weighted graph G (graph of similarities), whose nodes represent the instances, and the edges connecting each pair of nodes i, j represent the similarity between each pair of instances as a weight w_{ij}. The number of edges depends on the type of the graph that we want to use, such as the k-nearest neighbor or the fully connected graph [62] (Laplacian Score generally uses the k-nearest neighbor graph). The Laplacian matrix L is defined as:

$$L = D - W \tag{1}$$

where D is a diagonal matrix such that $d_{ii} = \sum_{j=1}^{m} w_{ij}$.

Given a weighted graph G, the Laplacian matrix L is a linear operator [30,62] on a vector $\boldsymbol{f} \in \mathbb{R}^m$, as follows:

$$\boldsymbol{f}^T L \boldsymbol{f} = \frac{1}{2}\sum_{i\neq j} w_{ij}\left(f_i - f_j\right)^2 \tag{2}$$

Eq. (2) quantifies how much the vector $\boldsymbol{f}$ locally varies [61] in G. This fact motivates to use L to measure the consistency of a feature f regarding the structure of the graph G. A feature $\boldsymbol{f}$ is consistent with the structure of a graph if $\boldsymbol{f}$ takes similar values for instances that are near each other in the graph, and $\boldsymbol{f}$ takes dissimilar values for instances that are far from each other. Thus a consistent feature would be relevant to separate the classes [30,62]. The Laplacian Score [24], proposed by He et al. [31], assesses the significance of individual features taking into account its consistency with the structure of the similarity graph. If we denote $\boldsymbol{f}_r = \left(f_{r1}, f_{r2}, \ldots, f_{rm}\right)^T$ with $r = 1, \ldots, n$ as the r-th feature and its values for the m instances. Then the Laplacian Score for $\boldsymbol{f}_r$ is calculated as:

$$L_r = \frac{\tilde{\boldsymbol{f}}_r^T L \tilde{\boldsymbol{f}}_r}{\tilde{\boldsymbol{f}}_r^T D \tilde{\boldsymbol{f}}_r} \tag{3}$$

where L is the Laplacian matrix of the weighted graph G, D is the degree diagonal matrix, and $\tilde{\boldsymbol{f}}_r = \boldsymbol{f}_r - (\boldsymbol{f}_r^T D\mathbf{1}/\mathbf{1}^T D\mathbf{1})$ represents the deviation from the mean of all observations of the $\boldsymbol{f}_r$ vector.

In the Laplacian Score, the local structure of the data space is more important than the global structure [31]. In order to model the local structure, this method constructs a k-nearest neighbor graph, where k is the neighborhood degree for each instance in the graph (see [31] for details). This value must be specified a priori by the user, and it models the local neighborhood relations between instances.

According to the Laplacian Score, a good feature should have a small value for L_r [31]. Thus, the features can be arranged in a list according to their relevance. Those features at the top of the list are those with smaller values for L_r, and they will be considered as the most important. Therefore, in this stage for each $\boldsymbol{f}_r$, we will use the Laplacian Score F_r to build a list that contains ordered features from the most to the least relevant regarding data structure preservation. This ordering will be used to create feature subsets that will be evaluated in the second stage of the proposed method.

3.2. Wrapper stage

The goal of this stage is selecting the best feature subset of size n' (with $n' \leq n$), according to some quality function. Since we have an ordered list of features (feature ranking) previously generated in the first stage; in this step, we will evaluate feature subsets containing the most relevant features (according to the ranking). Alternatively, we will eliminate from the whole set of features those least relevant (backward elimination), limiting in this form the search space.

In our feature selection method, to evaluate the quality of the clusters formed by each feature subset, we propose to modify the Calinski–Harabasz index [63] to avoid the bias that occurs when feature subsets of different cardinality are evaluated in the same clusters [26]. Below we describe how this modification to the Calinski–Harabasz index is performed.

To measure the quality of the clusters generated by any clustering algorithm (in our case we use the k-means algorithm), we will take into account the within-cluster scatter and between-cluster separation [64]. Specifically, as we mentioned above, we will use the Calinski–Harabasz (CH) index [63], which is a measure to assess the separation among clusters and the compactness that exists within them [65]. The advantage of this index is that it can be used with any clustering algorithm, and unlike other evaluation indices such as the scatter separability criterion,[1] the CH index does not have the problem known as Small Sample Size Problem [24] that occurs when the number of features is larger than the number of instances, or when two or more features are identical or multiples one from each other [50].

Let S_w be the average of the within-class scatter matrix, and S_b be the between-class scatter matrix [26] defined as follows:

$$S_w = \frac{1}{c}\sum_{j=1}^{c} \Sigma_j$$

$$S_b = \frac{1}{c}\sum_{j=1}^{c} \left(\mu_j - M_0\right)\left(\mu_j - M_0\right)^T$$

$$M_0 = \frac{1}{c}\sum_{j=1}^{c} \mu_j$$

where Σ_j is the variance–covariance matrix of the j-th cluster, c is the number of clusters, and μ_j is the mean vector of the j-th cluster. S_w measures how much disperse are the data from the mean of the cluster to which they belong (average covariance of each cluster), meanwhile S_b measures how disperse are the means of the clusters with respect to the total mean (i.e. the covariance of the dataset whose members are the means of each cluster). The Calinski–Harabasz index, proposed by Calinski and Harabasz in [63], evaluates the quality of the clusters as:

$$CH = \frac{\mathrm{tr}(S_b)}{\mathrm{tr}(S_w)} \times \frac{m-c}{c-1} \tag{4}$$

where $\mathrm{tr}(\cdot)$ represents the trace[2] of the scattering matrices, and m is the number of instances in the dataset. Compact and separated clusters are expected to have small and large values for $\mathrm{tr}(S_w)$ and $\mathrm{tr}(S_b)$ respectively, therefore, the best clustering is the one having the largest ratio between $\mathrm{tr}(S_b)$ and $\mathrm{tr}(S_w)$.

3.2.1. Weighted normalized Calinski–Harabasz index

A problem with most of the feature subsets evaluation clustering criteria (including the CH index) is the bias that occurs when feature subsets with different cardinality are evaluated over the same

[1] The problem with this index is that the inverse matrix can become singular.

[2] In linear algebra, the trace of an n-by-n square matrix A is defined to be the sum of the elements on the main diagonal.

clusters. The value computed by these criteria increases or decreases monotonically regarding the number of features [26]. This behavior is not desired, since a biased criterion gives better values for subsets containing a single feature, or conversely with all the features, leading to trivial solutions. In particular, the Calinski–Harabasz index is biased toward low-cardinality, i.e., it tends to decrease as the number of features increases (see Experiment I in Section 4), therefore, in its current form, CH would select in most cases only one feature, giving a trivial solution. This behavior is because generally, the separation into clusters ($\mathrm{tr}(S_w)$) tends to increase faster than the separation among clusters ($\mathrm{tr}(S_b)$), due to that when the dimensionality increases, the distance among instances increases too [64]. To address this problem, we propose to modify the CH index taking into account the following points:

- Since the cardinality of the feature subsets always increases or decreases opposite to the CH index value; we should consider the cardinality of candidate feature subsets in the assessment. To do this, we propose to multiply the CH index value by the cardinality of the feature subset under evaluation. For illustrating this fact, consider a forward search strategy. As features are added, the value of CH index tends to decrease, but if it is multiplied by the number of features being assessed, the value of the index is normalized, i.e. the bias produced by the decreasing tendency is countered by the number of features which increases when a feature is added.
- Since in the Laplacian Score small values are better than large; we propose that the CH index takes into account not only the ordering computed in the first stage but also the relevance value of the features according to the Laplacian Score, creating a synergy between this last and the Calinski–Harabasz index. Therefore, we propose to multiply the CH index by the inverse value of the Laplacian score associated with the last r-th feature added or eliminated in the feature subset.

The result of the previous modifications is a new index that we have called Weighted Normalized Calinski–Harabasz Index, from now on referred as WNCH, which is defined as follows:

$$WNCH(S_0) = \frac{\mathrm{tr}(S_b^{(X_{S_0})})}{\mathrm{tr}(S_w^{(X_{S_0})})} \times \frac{m-c}{c-1} \times |S_0| \times \frac{1}{L_r} \tag{5}$$

where X_{S_0} is the dataset described exclusively by the candidate feature subset S_0, $(\mathrm{tr}(\cdot))/(\mathrm{tr}(\cdot))$ represents the ratio of the traces of the inter-class and intra-class scatter matrices respectively, m is the number of instances, c is the number of clusters, and L_r is the Laplacian Score value for the r-th feature added to or eliminated from the feature subset S_0. This modification counteracts the bias toward low-cardinality and produces that the Calinski–Harabasz index assesses in a more fair way the candidate feature subsets, by taking into account the relevance of the features. Therefore, the best feature subset will be the one with the highest WNCH value for the index defined in Eq. (5).

To find the best feature subset using our WNCH index, we propose to use two well-known searching strategies: simple ranking and backward elimination, creating two versions of our method as described below.

1. *Laplacian Score-WNCH-Simple Ranking* (*LS-WNCH-SR*): This version of the proposed method builds n feature subsets (as many subsets as features). Given an ordered list of features according to the Laplacian score, the first set only contains the top-ranked feature, the second set contains the two top-ranked features; the third set includes the three top-ranked features, and so on (see Algorithm 1). Each feature subset is evaluated through the proposed index WNCH. Finally, the result will be the feature subset that maximizes the WNCH index.
2. *Laplacian Score-WNCH-Backward Elimination* (*LS-WNCH-BE*): This version of the proposed method starts evaluating the subsets containing all the n features through the WNCH index. Then, a backward elimination search begins to assess feature subsets with $n-1$ cardinality; in this way, a feature is removed at a time from the subset of features, the feature to remove is taken from the least relevant to the most relevant according to Laplacian Score ranking. Thus every time a feature is removed from the set of features, the new subset is evaluated through the WNCH index, and if its WNCH index value is better than the previous subset, the new subset is updated as the best subset found so far. The user can specify how many subsets to evaluate for each cardinality (p parameter), this is useful to reduce the search space in those problems with a broad number of features. This method in contrast to the first one takes into account those features that alone are not relevant according to the Laplacian Score, but these features combined with others could produce a good subset of features.

Algorithm 1. Pseudo-code for LS-WNCH-SR method.

Input: X Dataset with m instances and n features
k_{LS} number of neighbors for the graph construction
c number of clusters
Output: S_{best} the best feature subset.
// Initializing variables
$indRank \leftarrow \varnothing$
$\gamma_{best} \leftarrow -\infty$
$S_0 \leftarrow \varnothing$
Calculate the Laplacian Score L_r for each feature $\boldsymbol{f}_r$ with $r=1$ to n.
Sort the L_r values in descending order, and assign the indexes of the features associated to each L_r into the *indRank* array.
for $i=1$ to n **do**
    $S_0 \leftarrow S_0 \cup indRank[i]$
    Run a clustering algorithm over X_{S_0}
    $\gamma \leftarrow WNCH(S_0)$ // Clusters evaluation
    if $\gamma > \gamma_{best}$ **then**
        $\gamma_{best} \leftarrow \gamma$
        $S_{best} \leftarrow S_0$
    end if
end for
return S_{best}

The pseudo-code of this version of the proposed method is shown in the Algorithm 2.

Algorithm 2. Pseudo-code for LS-WNCH-BE method.

Input: X Dataset with m instances and n features
k_{LS} number of neighbors for the graph construction
c number of clusters
p the exploration degree
Output: S_{best} the best feature subset.
// Initializing variables
$indRank \leftarrow \varnothing$
Calculate the Laplacian Score L_r for each feature $\boldsymbol{f}_r$ with $r=1$ to n.
Sort the L_r values in descending order, and assign the indexes of the features associated to each L_r into the *indRank* array.
if $|indRank|==1$ **then**
    return *indRank*
else
    $flag \leftarrow false$

```
8:        S_0 ← indRank
9:        Run a clustering algorithm over X_{S_0}
10:       γ_best ← WNCH(S_0) // Clusters evaluation
11:       counter ← 0
12:       for i=n down to 1 do
13:         S_0 ← indRank
14:         Remove the i-th feature of S_0
15:         Run a clustering algorithm over X_{S_0}
16:         γ ← WNCH(S_0)
17:         if γ > γ_best then
18:                 γ_best ← γ
19:                 S_best ← S_0
20:                 flag ← true
21:       end if
22:       counter ← counter + 1
23:       if counter ≥ p then
24:             exit for loop
25:       end if
26:     end for
27:     if flag ← true then
28:       Run Algorithm 2 with X_{S_best} //Recursion
29:     else
30:       S_best ← indRank
31:       returnS_best
32:     end if
33: end if
```

3.3. Computational complexity of the proposed method

In this section, we analyze the time complexity of both versions of the proposed method.

Algorithm 1 has two parts (filter and wrapper stages). First, we analyze the filter stage.

Actions carried out in steps 1–3 are considered to consume time $T1 = O(1)$. In step 4 we need $O(m^2n)$ operations to build W, D, and L, therefore, we need $T4 = O(m^2n)$ time to calculate scores for n features. The step 5 ranks n features needing $O(nlog(n))$ (using merge sort) operations which will consume $T5 = O(nlog(n))$ time.

For the wrapper stage (steps 6–14). The loop in the step 6 (for condition) is performed $(n+1)$ times,[3] taking a time of $T6 = O(n+1)$. Steps 7–12 are repeated n times, so, action performed in step 7 takes a time $T7 = O(n)$. For the step 8, we run the k-means clustering algorithm n times needing $O(tcmn^2)$ operations, where t is the number of iterations of k-means (in our case at most $t=1000$), so it will consume $T8 = O(tcmn^2)$ time. In the step 9, for the WNCH index we need to perform $O(cm^2n)$ operations, taking a time of $T9 = O(cm^2n)$. The steps 10–12 are executed in $T10 = O(n)$ time.

Therefore, the total running time for this algorithm is:

$$T(n) = O(1) + O(m^2n) + O(n\ \log(n)) + O(n+1) + O(n) + O(tcmn^2) + O(cm^2n) + O(n)$$

We can assume that the highest-order terms in any given function dominate its rate of growth and thus defines its run-time order. So, we can conclude that $T(n) = O(tcmn^2 + cm^2n) = O(c(tmn^2 + m^2n))$.

Carrying out a similar analysis for the LS-WNCH-BE algorithm, we obtain a complexity of $O(m^2n^2 + cp(tmn^2 + m^2n))$ for the worst case.

[3] Note that an extra step is required to terminate the for loop, hence it makes $n+1$ and not n executions.

4. Experiments

To show the performance of both versions (LS-WNCH-SR and LS-WNCH-BE) of the proposed method, we made three types of experiments.

First, as in [26,43,47,49], we generated synthetic datasets composed of a mixture of multivariate Gaussian, these datasets were generated using the random functions *mvnrnd()* and *rand()* of Matlab.[4] The relevant features were generated following a normal distribution, while irrelevant features were generated following a uniform distribution. For details about the parameter values used to generate these datasets see Table 1. Also, in this experiment we included synthetic datasets for clustering [66] available at URL http://personalpages.manchester.ac.uk/mbs/julia.handl/generators.html (see Table 2), for these datasets the correct cluster assignments are known a priori. For the second experiment, we used real datasets taken from the UCI repository [67]. Finally, to show the scalability of the proposed method, a third experiment was done over biomedical microarray and text document datasets, which are larger in terms of the number of features and/or instances. Biomedical microarray datasets were taken from URL http://eps.upo.es/bigs/datasets.html; Meanwhile, text document datasets were taken from ASU feature selection repository available at URL http://featureselection.asu.edu/datasets.php. The details of UCI and large datasets are shown in Table 3.

4.1. Comparisons

We have compared the proposed method against the hybrid unsupervised feature selection methods proposed by Dash and Liu [49] and Li et al. [47]. The comparison against these methods was made because they followed a similar strategy to our method, i.e. they are based on feature ranking, and they are not linked to a particular type of data.

Additionally, we have contrasted our results against two unsupervised filter methods for feature selection of the state-of-the-art: SVD-Entropy and randomized feature selection algorithm for k-means (FS-Kmeans). SVD-Entropy [29] is an univariate filter method that measures the feature relevance based on the entropy of the data according to the singular values of the data matrix. To conduct the experiments fairly, for SVD-Entropy we used Simple Ranking (SR) as feature subset search strategy. The randomized feature selection algorithm for k-means [68] is an algorithm designed specifically for k-means, and according to the authors this algorithm gives a theoretical guarantee on the quality of the clusters that are produced after reducing the dimensionality. This algorithm selects features using probabilities that are computed via the right singular vectors of the matrix containing the data. We take the so-called normalized leverage scores as the measure to characterize the importance of a feature respect to k-means objective.

Finally, in order to compare our results against a method belonging to the wrapper approach, we also included the Simplified Silhouette-Sequential Forward Selection (SS-SFS) [41] method; which selects a feature subset that provides the best quality according to the simplified silhouette criterion using k-means as clustering algorithm for data partitioning.

4.2. Evaluation measures

Currently, in the literature, there are neither standard measures for evaluating clustering, nor standard measures for evaluating unsupervised feature selection methods; however, internal and external measures are often used for clustering evaluation [69]. External measures evaluate a clustering solution based on how much it

[4] The MathWorks Inc. URL http://www.mathworks.com

Table 1
Synthetic datasets used in our experiments.

Dataset	No. of instances	No. of features	No. of clusters	Mean	Covariance	Relevant features
S_1 [47]	1000	11	3	$\mu_1 = (0, 0, 0, 0, 0, 0)\, \mu_2 = (0, 2, 3, 4, 5, 3)\, \mu_3 = (5, 6, 7, 8, 1, 0)$	$\Sigma_1 \ldots \Sigma_3 = \mathbf{I}$	6–11
S_2 [26]	500	20	5	The true means μ were sampled from a uniform distribution on $[-5, 5]$.	The elements of the diagonal covariance matrices σ were sampled from a uniform distribution on $[0.7, 1.5]$	16–20
S_3 [43]	1000	10	4	$\mu_1 = (0, 3)\, \mu_2 = (1, 9)\, \mu_3 = (6, 4)\, \mu_4 = (7, 10)$	$\Sigma_1 \ldots \Sigma_4 = \mathbf{I}$	9,10
S_4 [26]	500	20	5	The true means μ were sampled from a uniform distribution on $[-10, 10]$	The elements of the diagonal covariance matrices σ were sampled from a uniform distribution on $[0.7, 1.5]$	6–20
S_5 [47]	1000	22	6	$\mu_1 = (0, 0, 0, 0, 0, 0, 0)\, \mu_2 = (0, 2, 3, 4, 5, 3, -2)\, \mu_3 = (5, 6, 7, 8, 9, 10, 2)\, \mu_4 = (2, 7, 8, 9, 10, 2, 0)\, \mu_5 = (-4, -6, 7, 2, 1, 3, 6)\, \mu_6 = (-2, -4, -5, 6, 8, 0, 7)$	$\Sigma_1 \ldots \Sigma_6 = \mathbf{I}$	16–22

Table 2
Characteristics of the Gaussian and Ellipsoid synthetic datasets [66] used in our experiments.

No.	Dataset	No. of instances	No. of features	No. of clusters
1	Gaussian10d4cno0	1288	10	4
2	Gaussian10d4cno1	958	10	4
3	Gaussian10d4cno2	838	10	4
4	Gaussian10d4cno3	1318	10	4
5	Gaussian10d4cno4	933	10	4
6	Gaussian10d4cno5	1139	10	4
7	Gaussian10d4cno6	977	10	4
8	Gaussian10d4cno7	1482	10	4
9	Gaussian10d4cno8	1482	10	4
10	Gaussian10d4cno9	1183	10	4
11	Ellipsoid.50d4c.1	1064	50	4
12	Ellipsoid.50d4c.2	984	50	4
13	Ellipsoid.50d4c.3	1371	50	4
14	Ellipsoid.50d4c.4	1098	50	4
15	Ellipsoid.50d4c.5	351	50	4

Table 3
Details of the real datasets used in our experiments.

	Datasets	No. of instances	No. of features	No. of classes	Relevant features
UCI datasets	Iris	150	4	3	3,4
	ionosphere	351	33	2	–
	pima	768	8	2	–
	wine	178	13	3	–
	monks-3	432	6	2	–
	wdbc	568	30	2	–
	sonar	208	60	2	–
	parkinsons	194	22	2	–
	vehicle silhouettes	845	18	4	–
	pendigits	7493	16	10	–
	spambase	4600	57	2	–
	segmentation	2100	19	7	–
	optdigits	3822	62	10	–
	waveform (noise)	5000	40	3	–
	musk (V2)	6597	166	2	–
Large datasets	Lymphoma	45	4026	2	–
	Embryonal tumors	60	7130	2	–
	Leukemia	38	7130	2	–
	Basehock	1993	4862	2	–
	PCMac	1943	3289	2	–
	Relathe	1427	4322	2	–

resembles a set of classes, commonly known as ground-truth or "expert classification", which has been manually tagged by human experts. Examples of external validation indices are: Accuracy (ACC) [31], Jaccard index [70], Rand index [71], Fowlkes-Mallows (FM) [72] and Normalized Mutual Information (NMI) [73]. On the other hand, internal validation measures [74] evaluate a clustering solution based on the degree of fitting among the clusters formed and the data itself. These validation measures do not use external knowledge. Examples of internal validation indices are: Dunn index [75], Davies–Boulding index [76], Silhouette index [77] and Calinski–Harabasz index [63].

In this paper, the indices used for validation were Jaccard (external) and global silhouette (internal). We used these indices because they are some of the most widely used for validation in the unsupervised classification context. The Jaccard index measures the similarity between clustering results and the information on the previously known classes as:

$$Jaccard = \frac{n_{11}}{n_{11} + n_{01} + n_{10}} \tag{6}$$

here n_{11} is the number of pairs of instances that are classified together in the same cluster, according to both "expert" classification and clustering algorithm. n_{10} is the number of pairs of instances that are classified together in the same class by the "expert" classification, but they are not classified together in the same cluster by the clustering algorithm. n_{01} is the number of pairs that are classified together in the same cluster by the clustering algorithm, but they are not classified together in the same class by the "expert" classification. The Jaccard index reflects how much similar are the clustering algorithm assignments and the expected classification, and its score values range from 0 (no match) to 1 (perfect match).

Before introducing the global silhouette index, let $\boldsymbol{x}_i$ be an instance belonging to the cluster A; the average dissimilarity of $\boldsymbol{x}_i$ to all other instances in A is denoted by $a(\{\boldsymbol{x}_i\})$. Now let C be another cluster $C \neq A$. The average dissimilarity of $\boldsymbol{x}_i$ to all instances of C will be denoted as $d(\{\boldsymbol{x}_i, C)$. After computing $d(\{\boldsymbol{x}_i, C)$ for all the clusters $C \neq A$, the smallest one is selected, i.e. $b(\{\boldsymbol{x}_i) = \min\{d(\{\boldsymbol{x}_i, C), C \neq A\}$. This value represents the dissimilarity of $\boldsymbol{x}_i$ to its closest neighbor cluster. The silhouette $s(\{\boldsymbol{x}_i)$ is given by:

$$s(\{\boldsymbol{x}_i) = \frac{b(\boldsymbol{x}_i) - a(\boldsymbol{x}_i)}{\max\{a(\boldsymbol{x}_i), b(\boldsymbol{x}_i)\}} \tag{7}$$

The average (global silhouette) of $s(\boldsymbol{i}_i)$ over $i = 1, 2, \ldots, m$, where m is the number of instances, can be used as a criterion to assess the quality of the clusters [77]. The global silhouette takes values in the range $[-1, 1]$ and it is maximized in the optimal clustering.

Additionally, we also report the retention rate and the run-time required by each feature selection method. The retention rate was calculated by multiplying the number of selected features by 100 and dividing by the total number of features.

4.3. Experiment settings

For our experiments, the datasets iris, wine, wdbc, pima, segmentation, parkinsons and spambase were standardized before feature selection and clustering. That is, each dimension was normalized to obtain mean 0 and standard deviation 1, this standardization was done because these datasets (except iris) have ranges of values with different scales for certain features, which affects the outcome of feature selection and clustering algorithms. For all the datasets, class labels were removed; thus they were not taken into account for feature selection and clustering.

In the proposed method (both LS-WNCH-SR and LS-WNCH-BE), the only parameter that can vary is k_{LS} (number of neighbors considered for building the graph) for the Laplacian Score, which could affect the selected feature subsets. According to [31] k_{LS} must be equal to or greater than 5; thus, in our experiments we use $k_{LS} = 5$. For all datasets (Tables 1–3), we set the number of clusters corresponding to the number of classes/clusters reported in the ground-truth. For the exploration degree (p parameter) in LS-WNCH-BE method, values from 1 to 5 were tested, and we use the value that produced the best results ($p=3$).

In both experiments I and II, since we do not know the relevant features for UCI (except for iris), Gaussian and Ellipsoid Synthetic datasets (Tables 2 and 3), we have validated the feature selection methods applying 10-fold cross-validation like in [26]. Performing feature selection over the training set, and then, performing clustering using the k-means algorithm (running it three times with different initial points) over the reduced test set. The clustering results of the three runs of k-means were validated through the average of external (Jaccard) and internal (global silhouette) validation indices. As a final result, we report mean and standard deviation values of the validation indices, run-times, and retention rates computed over the 10 independent runs for each dataset.

In order to perform the comparison of the results obtained by the proposed method against the results produced by state-of-the-art methods on the datasets mentioned above, we performed a two-sided Wilcoxon rank sum test [78] using a confidence level of 95%. In Tables 5, 6, 8, and 9 the column of the proposed method that obtained on average the best results appears in "**bold**". By applying the Wilcoxon test, we detect if the other methods get a result which has a statistically significant difference against the best method. Symbols "+" and "−" indicate statistically significant better or worse behavior respectively.

For the large datasets, a training set and an independent test set are available; therefore we performed the feature selection on the training set, and then we validated the performance of the feature selection methods over the test set reduced to the selected features.

All the methods used in the experiments were implemented in Matlab® R2015b, using a computer with Intel Core i7-5820k 3.30 GHz processor with 32 GB DDR4 RAM, running 64-bit Windows 10 Pro.

4.4. Experiment I

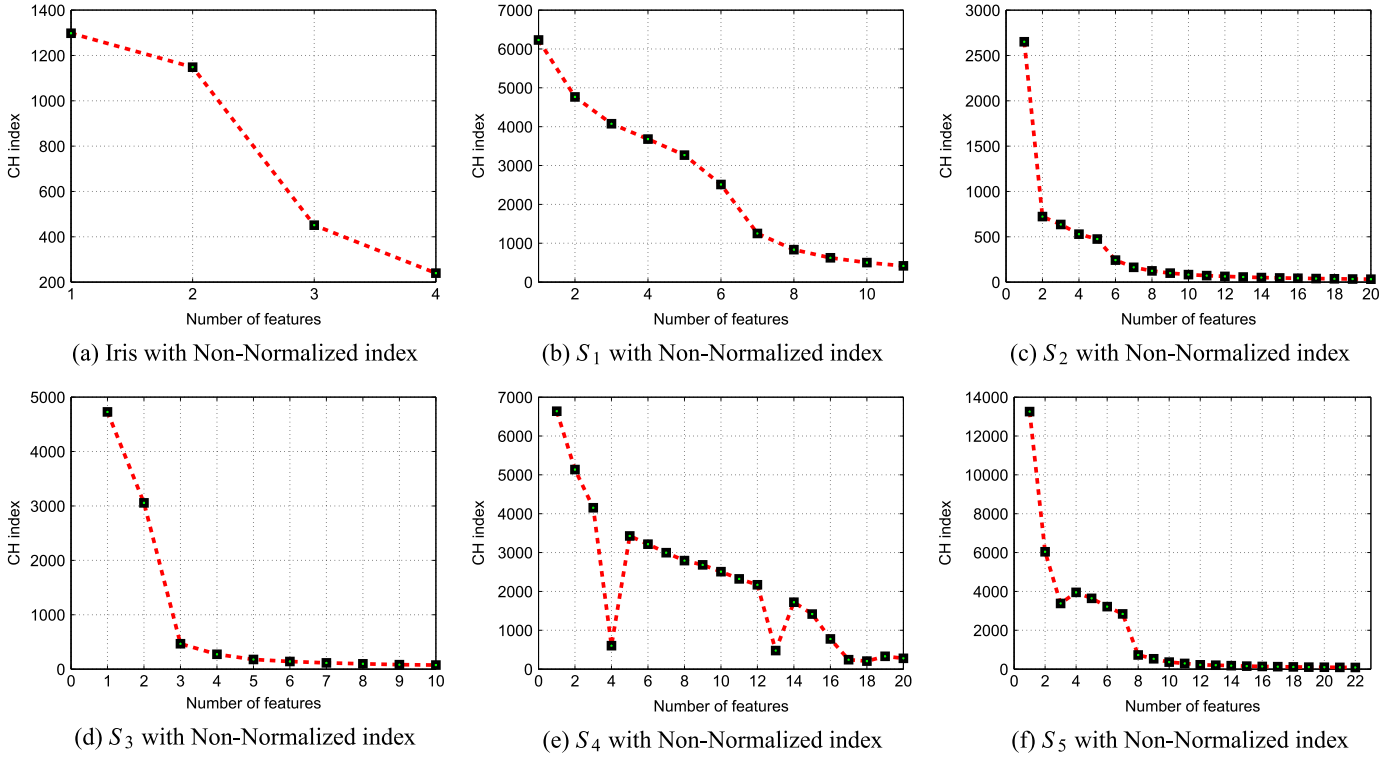
This experiment was divided into two parts, in the first part, the objective is to check whether the proposed methods can rank and select the features considered as relevant. In order to show the impact of the proposed modifications to the Calinski–Harabasz index, some experiments were performed with the iris and the S_1–S_5 synthetic datasets. These datasets were ranked by the Laplacian Score and evaluated through the Weighted Normalized Calinski–Harabasz and non-Normalized Calinski–Harabasz indices. The results of this experiment are shown in Figs. 1 and 2, where we can observe that generally the weighted normalized index grows when relevant features are added, and once all the relevant features have been added, the index tends to decrease. While the non-normalized index, usually only chooses one feature, and it tends to decrease in almost all cases.

In the iris dataset, it is well known that just one class is linearly separable from the other two, and it is also known that the features F3 (petal length) and F4 (petal width) are the most relevant. In Fig. 2a, we can see that our method effectively chooses F3 and F4 as the first two relevant features. Likewise, in Fig. 2b, we can observe that, for the synthetic dataset S_1 the index grows when the relevant features are added, and decreases for irrelevant ones, selecting only the first five relevant features. Similar cases are shown in Fig. 2 for S_2, S_3, S_4 and S_5 datasets.

Table 4 shows the final feature subset chosen by the hybrid feature selection methods on the datasets used in this experiment. In this table, we can see that the proposed method (both versions) ranks the relevant features through the Laplacian Score correctly at the beginning of the list, then, the features are selected subsequently with the WNCH index. For example, in Fig. 3, we can see the scatter plots of the synthetic datasets S_1 and S_3 using the first features selected by the hybrid feature selection methods according to their respective ranking. In Fig. 3a, for S_1, our method (both versions) selects the features F6, F7 and F8 (features with Gaussian distribution), which draw well-defined clusters. While the other hybrid methods (see Figs. 3b and c) in most cases include several irrelevant features and the clusters are not well defined. Figs. 3d, e, and f show a similar case for S_3.

For the second part of this experiment, we have evaluated the feature selection methods over the synthetic datasets shown in Table 2, as we said before, in these datasets the relevant features are unknown in advance, therefore, we applied the evaluation strategy described in Section 4.3.

As it can be seen in Table 5, the average value of the Jaccard index for these fifteen synthetic datasets shows that LS-WNCH-BE performs better than the other feature selection methods (including not applying feature selection), achieving significantly better performance in the overall Wilcoxon test evaluation than SVD-Entropy and SS-SFS methods. Meanwhile, LS-WNCH-SR is the second best method on average for these datasets, being significantly better than SVD-Entropy and SS-SFS methods as shown in Table 7.



(a) Iris with Non-Normalized index (b) S_1 with Non-Normalized index (c) S_2 with Non-Normalized index

(d) S_3 with Non-Normalized index (e) S_4 with Non-Normalized index (f) S_5 with Non-Normalized index

Fig. 1. Calinski Harabasz index (CH) without adjustment for datasets iris, S_1, S_2, S_3, S_4 and S_5.

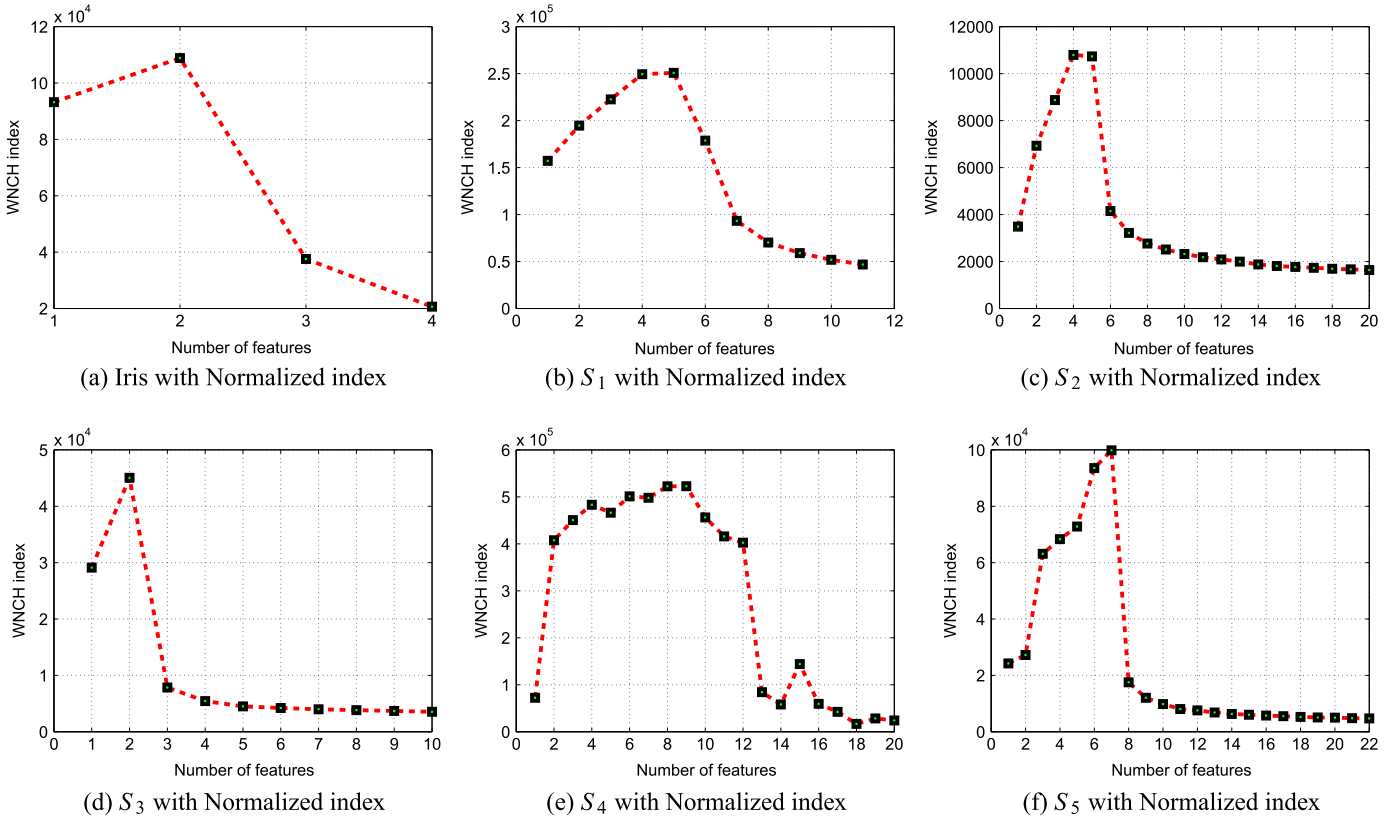


Fig. 2. Weighted Normalized Calinski–Harabasz index (WNCH) for datasets iris, S_1, S_2, S_3, S_4 and S_5.

On the other hand, Table 6 shows the results for the global silhouette index. In this table, we can see that the best method on average was SS-SFS, followed by SVD-Entropy, both being significantly better in the overall Wilcoxon statistical test than our methods as can be seen in Table 7. This result is because SS-SFS and SVD-Entropy achieve a significant reduction of data dimensionality (by selecting only 7.3% and 14.8% of the original features, respectively) as we can see in Fig. 4a. The global silhouette index gives better values to feature subsets with very low cardinality (even with a single feature), because, as it is well known, this index is strongly biased to reward feature subsets of low cardinality [42]. However, it is important to highlight that when too few features are selected, relevant information is lost, and the clustering quality is affected according to the external validation measures, as we can see in the results shown in Tables 5 and 7 for Jaccard index; where SS-SFS and SVD-Entropy are the worst methods. In contrast, LS-WNCH-SR and LS-WNCH-BE methods got a retention rate of 90.0%, and 91.1%, respectively (see Fig. 4a), obtaining on average better clustering quality using the Jaccard index, and also good values for the global silhouette.

Finally, Fig. 4b shows that the proposed methods are the second fastest after the filter ones, outperforming on average the other hybrid feature selection methods.

4.5. Experiment II (UCI datasets)

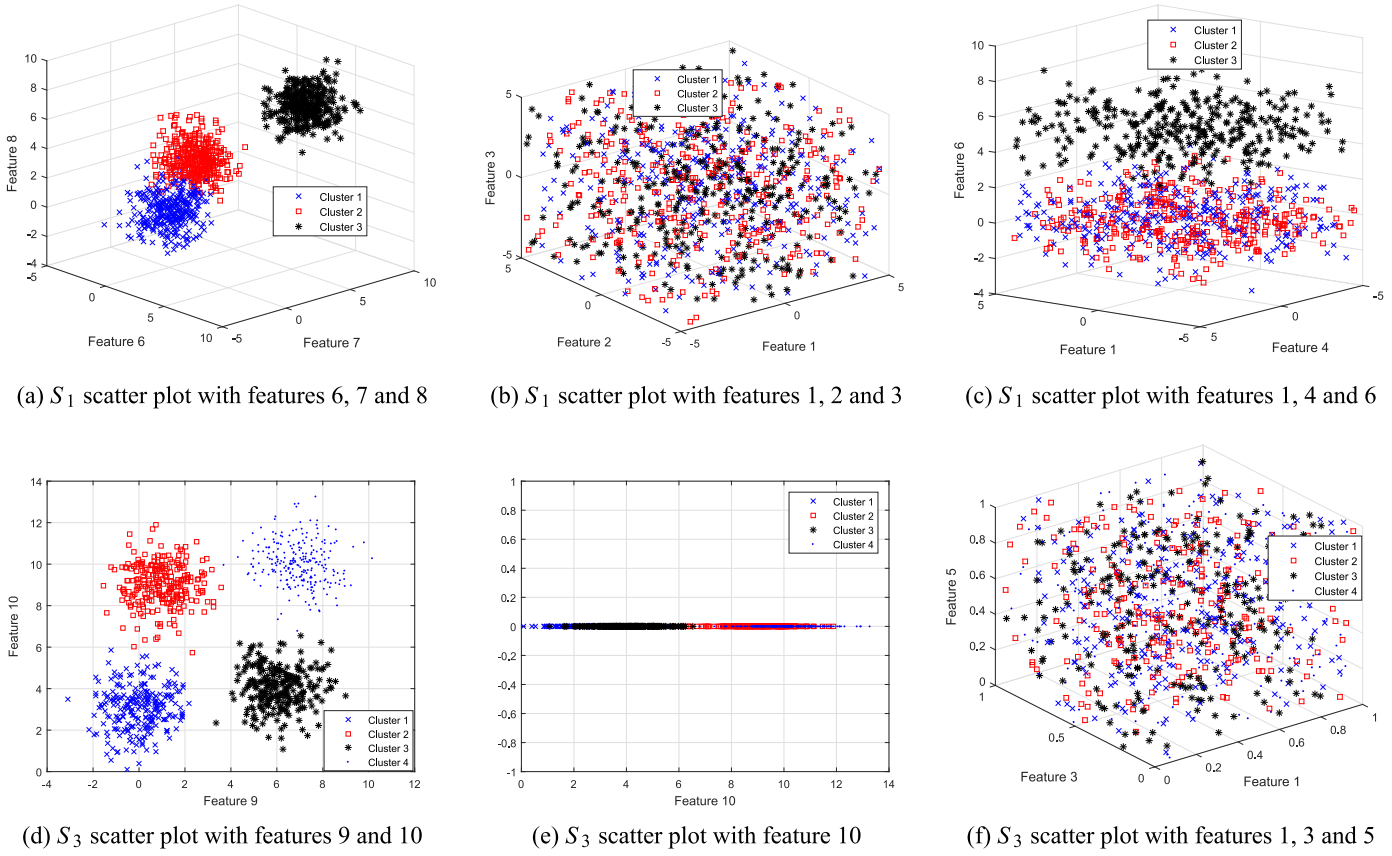
A second experiment was conducted for testing our method with supervised datasets from the UCI repository. For this experiment, in Table 8, the average value of the Jaccard index for these fifteen real datasets shows that LS-WNCH-BE performs better than the other feature selection methods. Moreover, compared to the original dataset (i.e., the dataset with all the features), the average value of the Jaccard index using LS-WNCH-BE method outperforms the Jaccard index value obtained by using the original dataset. According to Table 12, when we apply the overall Wilcoxon statistical test it remains tied with other methods.

On the other hand, in Table 9, we can see that the best method was SS-SFS, while LS-WNCH-SR was the second best for the average value of the global silhouette, being significantly better in the overall statistical test than the other remaining feature selection methods as Table 12 shows. Meanwhile, the LS-WNCH-BE got similar results to Li et al. and Dash and Liu hybrids methods respectively.

Table 10 shows the retention rate of each feature selection method. We can see that, among the hybrid methods, the method with the lowest retention rate was LS-WNCH-SR; it retained on average 53.92% of the features, while LS-WNCH-BE obtained a

Table 4
Results of hybrid features selection methods over iris and synthetics datasets.

Datasets	**LS-WNCH-SR**	**LS-WCHN-BE**	Dash and Liu	Li et al.
Iris	{3,4}	{3,4}	{3,4}	{3,4}
S_1	{6,7,8,9,10}	{6,7,8,9,10}	{1,2,3,4,6,7,8,9,10,11}	{1,4,6,7,8,9,10,11}
S_2	{16,17,18,20}	{16,17,18,19,20}	{1,4,5,6,7,8,9,10,11,13,16,17,18,19,20}	{1,4,5,6,8,10,12,16,17,18,19,20}
S_3	{9,10}	{9,10}	{10}	{1,3,5,8,9,10}
S_4	{7,8,9,11,12,13,14,15,19}	{7,8,9,10,11,12,13,14,15,18,19,20}	{7,8,9,11,12,13,14,15,18,19}	{1–20}
S_5	{16,17,18,19,20,21,22}	{16,17,18,19,20,21,22}	{1,2,10,11,12,15,16,17,18,19,20,21,22}	{1–22}



(a) S_1 scatter plot with features 6, 7 and 8 (b) S_1 scatter plot with features 1, 2 and 3 (c) S_1 scatter plot with features 1, 4 and 6

(d) S_3 scatter plot with features 9 and 10 (e) S_3 scatter plot with feature 10 (f) S_3 scatter plot with features 1, 3 and 5

Fig. 3. Scatter plots of the synthetics datasets S_1 and S_3 with the features selected by LS-WNCH-SR\BE (a, d), Dash and Liu (b, e), and Li et al. (c, f) methods respectively according to Table 4.

retention rate of 74.73% on average. On the other hand, the filter methods FS-Kmeans and SVD-Entropy got an average retention rate of 22.10% and 18.6% respectively. The SS-SFS wrapper method had the lowest rate of retention (6.49%) since most of the time it selects only one feature. The proposed method (both versions) does not choose trivial solutions, and it has a reasonable compromise between retention rate and clustering quality in terms of internal validation indices (global silhouette), while in terms of external validation indices (Jaccard), is the best one.

Table 11 shows the run-time of feature selection methods. In this table, we can observe that among the hybrid methods, LS-WNCH-SR was the fastest on average, followed by LS-WNCH-BE. The hybrid methods proposed by Dash and Liu and Li et al. required much more time in average than our proposed method. Considering all the methods, ours is the third fastest on average just after the filter methods in contrast to the other hybrid and wrapper methods that needed more runtime. Therefore, our method is faster and produces better clustering results than the other hybrid methods.

Notice that the wrapper method SS-SFS could not get a result

Table 5
Jaccard index results on 15 Gaussian and Ellipsoid synthetic datasets.

Dataset	**Hybrid**				**Filter**		**Wrapper**	
	LS-WNCH-SR	**LS-WNCH-BE**	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS	Original
Gaussian10d4cno0	0.766	**0.775**	0.747	0.774	0.347 –	0.736	0.627 –	0.764
Gaussian10d4cno1	0.967	**0.953**	0.841 –	0.923	0.531 –	0.910	0.461 –	0.956
Gaussian10d4cno2	0.821	**0.888**	0.840	0.830 –	0.403 –	0.811 –	0.446 –	0.835
Gaussian10d4cno3	0.499 –	**0.815**	0.832	0.427 –	0.426 –	0.632 –	0.427 –	0.797
Gaussian10d4cno4	0.815	**0.902**	0.712 –	0.905	0.604 –	0.827 –	0.571 –	0.846
Gaussian10d4cno5	0.801	**0.852**	0.551 –	0.827	0.428 –	0.803	0.594 –	0.844
Gaussian10d4cno6	0.749	**0.841**	0.579 –	0.525 –	0.567 –	0.753 –	0.399 –	0.853
Gaussian10d4cno7	0.968	**0.933**	0.765 –	0.979 +	0.605 –	0.912	0.343 –	0.930
Gaussian10d4cno8	0.887	**0.877**	0.699 –	0.885	0.441 –	0.828	0.480 –	0.788 –
Gaussian10d4cno9	0.741	**0.744**	0.706	0.576 –	0.360 –	0.678	0.465 –	0.728
Ellipsoid.50d4c.1	0.511	**0.515**	0.494	0.572	0.525 +	0.544	0.474	0.510
Ellipsoid.50d4c.2	0.479	**0.475**	0.462	0.484	0.474	0.498	0.457 –	0.478
Ellipsoid.50d4c.3	0.618 –	**0.688**	0.587 –	0.641	0.438 –	0.638	0.396 –	0.654
Ellipsoid.50d4c.4	0.442	**0.442**	0.435	0.443	0.359 –	0.428	0.341 –	0.463 +
Ellipsoid.50d4c.5	0.519	**0.505**	0.478	0.522	0.498	0.501	0.381 –	0.531
Average	**0.706**	**0.747**	**0.649**	**0.688**	**0.467**	**0.700**	**0.457**	**0.732**

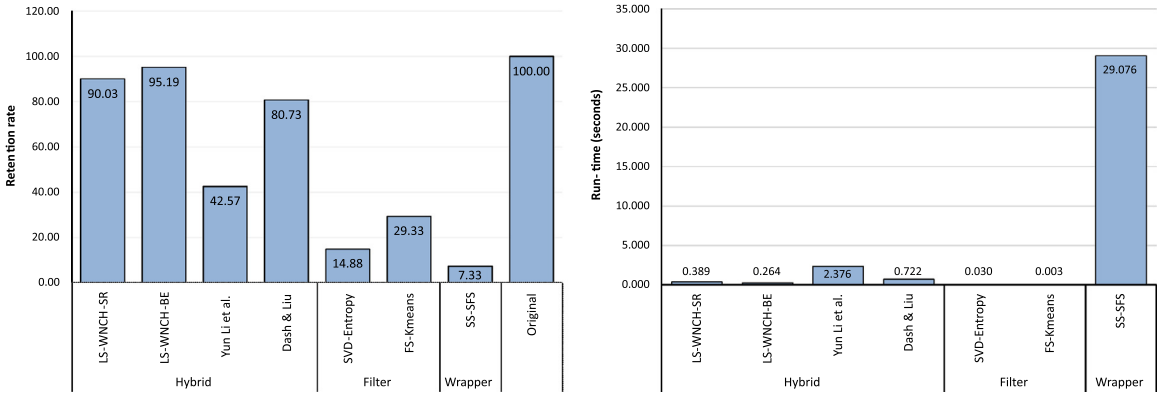
Table 6
Global silhouette results on 15 Gaussian and Ellipsoid synthetic datasets.

Dataset	Hybrid				Filter		Wrapper	
	LS-WNCH-SR	LS-WNCH-BE	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS	Original
Gaussian10d4cno0	**0.479**	0.474	0.594+	0.490	0.528+	0.452	0.760+	0.445
Gaussian10d4cno1	**0.608**	0.599	0.587−	0.581	0.571−	0.579	0.751+	0.599
Gaussian10d4cno2	**0.446**	0.506+	0.488+	0.448	0.563+	0.462	0.714+	0.445
Gaussian10d4cno3	**0.680**	0.475−	0.721	0.770+	0.768+	0.400−	0.766+	0.393−
Gaussian10d4cno4	**0.525**	0.531	0.546	0.531	0.661+	0.495	0.751+	0.491
Gaussian10d4cno5	**0.493**	0.518+	0.506	0.481	0.760+	0.476	0.802+	0.476
Gaussian10d4cno6	**0.533**	0.477	0.659+	0.711+	0.699+	0.444−	0.789+	0.476
Gaussian10d4cno7	**0.628**	0.603	0.580−	0.636	0.699+	0.606	0.755+	0.602
Gaussian10d4cno8	**0.524**	0.515	0.628+	0.508	0.661+	0.524	0.752+	0.455−
Gaussian10d4cno9	**0.484**	0.463	0.607+	0.541	0.727+	0.409−	0.775+	0.416−
Ellipsoid.50d4c.1	**0.596**	0.578	0.673+	0.614	0.659+	0.601	0.771+	0.590
Ellipsoid.50d4c.2	**0.610**	0.608	0.743+	0.616	0.691+	0.620	0.787+	0.614
Ellipsoid.50d4c.3	**0.567**	0.585	0.545	0.587	0.635	0.600	0.844+	0.591
Ellipsoid.50d4c.4	**0.625**	0.648	0.636	0.652	0.633	0.629	0.827+	0.592
Ellipsoid.50d4c.5	**0.538**	0.568	0.537	0.553	0.664+	0.564	0.752+	0.555
Average	**0.555**	**0.543**	**0.603**	**0.581**	**0.661**	**0.524**	**0.773**	**0.516**

Table 7
Summary table for the Wilcoxon statistical test applied for synthetic datasets using the Jaccard and global silhouette indices.

Evaluation measure	(−,=,+)/**overall**[a]	**Li et al.**	**Dash and Liu**	**SVD-Entropy**	**FS-Kmeans**	**SS-SFS**	**Original**
Jaccard	**LS-WNCH-BE**	(7,8,0)	(3,10,2)	(12,2,1)/−	(4,11,0)	(14,1,0)/−	(1,13,1)
	LS-WNCH-SR	(6,8,2)	(1,13,1)	(12,3,0)/−	(1,13,1)	(13,2,0)/−	(1,13,1)/+
G. silhouette	**LS-WNCH-BE**	(1,7,7)/+	(2,11,2)	(1,2,12)/+	(6,9,0)	(0,0,15)/+	(5,10,0)
	LS-WNCH-SR	(2,6,7)	(0,13,2)	(1,2,12)/+	(3,12,0)	(0,0,15)/+	(3,12,0)

[a] The **overall** represents the Wilcoxon statistical significance test applied to the average of all datasets.



(a) Retention rate (b) Run-time

Fig. 4. Average of run-time and retention rate of feature selection methods over the Gaussian and Ellipsoid synthetic datasets [66].

for the musk V2 dataset after 48 h, for this reason, we marked its cell in the table with the symbol ⋄. In order to compute the average run-time for SS-SFS over musk V2 dataset, we used the time we waited before stopping SS-SFS, i.e., 48 h.

4.6. *Experiment III (larger datasets)*

Finally, in order to show the performance of the proposed method over large datasets, we present some results with Lymphoma, Embryonal tumors, Leukemia, Basehock, PCMac and Relathe datasets (with 4026, 7130, 7130, 4862, 3289, and 4322 features respectively). The first three datasets are gene expression profiling (biomedical data), and the remaining three belong to a document collection of 20 newsgroups, which are larger in terms of instances than the biomedical data. In this experiment, we compare LS-WNCH-SR and LS-WNCH-BE methods against the filter methods SVD-Entropy and FS-Kmeans which were the fastest in our previous experiment.

Fig. 5a shows that the results of LS-WNCH-BE and LS-WNCH-SR are better than those results reached by the filter methods regarding the quality of the selected features using the Jaccard index. Furthermore, in Fig. 5b, we can also observe that in terms of the global silhouette index the proposed method LS-WNCH-SR is the best one. In this experiment, we can observe that the lowest retention rates (Fig. 5c) were obtained by SVD-Entropy and FS-Kmeans methods. While LS-WNCH-SR and LS-WNCH-BE methods got a retention rate of 84.0 and 96.1 respectively, both obtaining better evaluations regarding clustering quality measures (Jaccard

Table 8
Jaccard index results on 15 supervised datasets from the UCI repository.

Dataset	**Hybrid**				**Filter**		**Wrapper**	
	LS-WNCH-SR	**LS-WNCH-BE**	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS	Original
iris	0.858	**0.858**	0.836	0.858	0.754−	0.565−	0.837−	0.566−
ionosphere	0.427−	**0.429**	0.425	0.433+	0.459+	0.414−	0.562+	0.432
pima	0.361−	**0.426**	0.437	0.421	0.415−	0.415	0.520+	0.430+
wine	0.810	**0.822**	0.639−	0.872+	0.464−	0.719−	0.358−	0.872+
monks-3	0.367+	**0.334**	0.365+	0.376+	0.363+	0.378+	0.332	0.371+
wdbc	0.722−	**0.743**	0.730−	0.735	0.714−	0.673	0.525−	0.739
sonar	0.336	**0.336**	0.353+	0.349+	0.352+	0.353+	0.420−	0.341
parkinsons	0.489+	**0.482**	0.489+	0.480	0.441−	0.481	0.566+	0.462−
vehicle silhouettes	0.226	**0.223**	0.226	0.223	0.218−	0.220	0.224	0.224
pendigits	0.158−	**0.448**	0.417−	0.167−	0.172−	0.425−	0.148−	0.442
spambase	0.519−	**0.540**	0.520−	0.522	0.516−	0.507−	0.522	0.581+
segmentation	0.231−	**0.367**	0.321−	0.331−	0.282−	0.394	0.141−	0.387
optdigits	0.105−	**0.494**	0.473	0.104−	0.152−	0.482	0.100	0.500
waveform (noise)	0.335	**0.335**	0.333−	0.301−	0.334−	0.329−	0.276	0.335
musk (V2)	0.447	**0.447**	0.444	0.451	0.553+	0.443	⋄	0.447
Average	**0.426**	**0.486**	**0.467**	**0.442**	**0.413**	**0.453**	**0.395**	**0.475**

Table 9
Global silhouette results on 15 supervised datasets from the UCI repository.

Dataset	**Hybrid**				**Filter**		**Wrapper**	
	LS-WNCH-SR	LS-WNCH-BE	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS	Original
iris	**0.828**	0.828	0.817	0.828	0.790−	0.649−	0.838	0.655−
ionosphere	**0.678**	0.552	0.531−	0.417−	0.625−	0.479−	1.000+	0.414−
pima	**0.793**	0.732−	0.321−	0.375−	0.729	0.367−	0.934+	0.300−
wine	**0.685**	0.602−	0.569	0.451−	0.631−	0.440−	0.785+	0.450−
monks-3	**0.861**	0.961+	0.351−	0.334−	0.515−	0.388−	1.000+	0.353−
wdbc	**0.763**	0.549−	0.554−	0.543−	0.779+	0.560−	0.850+	0.517−
sonar	**0.792**	0.332−	0.495−	0.370−	0.474−	0.420−	0.795	0.336−
parkinsons	**0.886**	0.764−	0.766−	0.724−	0.819−	0.679−	0.934+	0.640−
vehicle silhouettes	**0.716**	0.640	0.742	0.665	0.742+	0.638−	0.716	0.646
pendigits	**0.809**	0.441-	0.448−	0.754−	0.528−	0.440−	0.845+	0.447−
spambase	**0.952**	0.542-	0.766−	0.812	0.984+	0.743−	0.999+	0.458−
segmentation	**0.865**	0.523-	0.585−	0.639−	0.708−	0.405−	0.956+	0.417−
optdigits	**0.875**	0.296-	0.302−	0.668−	0.332−	0.293−	0.884	0.297−
waveform (noise)	**0.457**	0.309	0.381−	0.478	0.496+	0.461	0.490+	0.309
musk (V2)	**0.876**	0.447-	0.620−	0.473−	0.792−	0.491−	⋄	0.445−
Average	**0.789**	**0.567**	**0.550**	**0.568**	**0.662**	**0.496**	**0.859**	**0.445**

Table 10
Retention rate of each feature selection method on 15 supervised datasets from the UCI repository.

Dataset	**Hybrid**				**Filter**		**Wrapper**	
	LS-WNCH-SR	LS-WNCH-BE	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS	**Original**
iris	50.00	50.00	52.50	50.00	25.00	75.00	25.00	100.00
ionosphere	48.48	58.79	48.48	98.48	26.36	6.06	3.03	100.00
pima	12.50	25.00	96.25	72.50	16.25	25.00	12.50	100.00
wine	66.92	56.92	41.54	100.00	23.08	23.08	7.69	100.00
monks-3	16.67	18.33	98.33	100.00	33.33	33.33	16.67	100.00
wdbc	62.00	90.00	85.67	89.67	27.00	6.67	3.33	100.00
sonar	100.00	100.00	26.33	78.17	14.83	3.33	1.67	100.00
parkinsons	57.27	70.00	71.36	80.45	4.55	9.09	4.55	100.00
vehicle silhouettes	100.00	100.00	5.56	92.22	5.56	22.22	5.56	100.00
pendigits	6.25	98.13	90.00	6.25	18.75	62.50	6.25	100.00
spambase	44.39	77.54	57.72	40.53	13.68	3.51	1.75	100.00
segmentation	27.37	76.84	56.84	38.42	26.32	36.84	5.26	100.00
optdigits	16.94	99.52	86.13	1.94	11.29	16.13	1.61	100.00
waveform (noise)	100.00	100.00	67.50	16.50	22.50	7.50	2.50	100.00
musk (V2)	100.00	99.88	36.20	91.69	10.54	1.20	⋄	100.00
Average	**53.92**	**74.73**	**61.36**	**63.79**	**18.60**	**22.10**	**6.49**	**100.00**

Table 11
Run-time of feature selection methods (in seconds) on 15 supervised datasets from the UCI repository.

Dataset	**Hybrid**				**Filter**		**Wrapper**
	LS-WNCH-SR	LS-WNCH-BE	Li et al.	Dash and Liu	SVD-Entropy	FS-Kmeans	SS-SFS
iris	0.06767	0.07892	0.04475	0.04484	0.00370	0.00392	0.12102
ionosphere	0.21151	1.22124	0.59085	0.38252	0.02662	0.00188	7.88034
pima	0.10210	0.46204	0.52920	0.18964	0.00287	0.00147	0.69901
wine	0.08384	0.25853	0.09725	0.12942	0.00276	0.00209	1.07112
monks-3	0.03273	0.17648	0.14942	0.09007	0.00125	0.00130	0.23715
wdbc	0.20351	0.32689	1.13157	0.46577	0.02614	0.00255	8.47970
sonar	0.42396	0.08512	0.56819	0.63858	0.09742	0.00230	20.20708
parkinsons	0.13899	0.51764	0.18177	0.19390	0.01044	0.00149	2.52920
vehicle silhouettes	0.28960	0.13359	1.47528	0.44353	0.00935	0.00359	5.22079
pendigits	5.06631	5.04770	74.61057	10.17512	0.02776	0.01802	154.96224
spambase	5.17154	45.33177	103.26390	15.35292	0.38582	0.00285	487.84946
segmentation	0.84929	3.03526	7.63515	1.74844	0.01740	0.00879	23.06585
optdigits	19.42787	6.43280	93.72070	30.45886	0.48140	0.02175	1117.153494
waveform (noise)	4.49821	3.23403	91.59981	13.06310	0.25253	0.00427	435.870258
musk (V2)	20.43295	12.48323	646.84797	78.16288	6.50188	0.00682	⋄
Average	**3.800**	**5.255**	**68.163**	**10.103**	**0.523**	**0.006**	**23 191.023**

Table 12
Summary table for the Wilcoxon statistical test applied for UCI datasets using the Jaccard and global silhouette indices.

Evaluation measure	**(-,=,+)/overall**[a]	**Li et al.**	**Dash and Liu**	**SVD-Entropy**	**FS-Kmeans**	**SS-SFS**	**Original**
Jaccard	**LS-WNCH-BE**	(6,6,3)	(4,7,4)	(11,0,4)	(6,7,2)	(6,3,4)	(2,9,4)
	LS-WNCH-SR	(2,7,6)	(2,6,7)	(7,1,7)	(5,5,5)	(6,1,5)	(2,7,6)
G. silhouette	**LS-WNCH-BE**	(4,6,5)	(5,4,6)	(2,1,12)	(7,5,3)	(0,0,14)/+	(8,6,1)
	LS-WNCH-SR	(12,3,0)/−	(11,4,0)/−	(9,1,5)/−	(14,1,0)/−	(0,4,10)	(13,2,0)/−

[a] The **overall** represents the Wilcoxon statistical significance test applied to the average of all datasets.

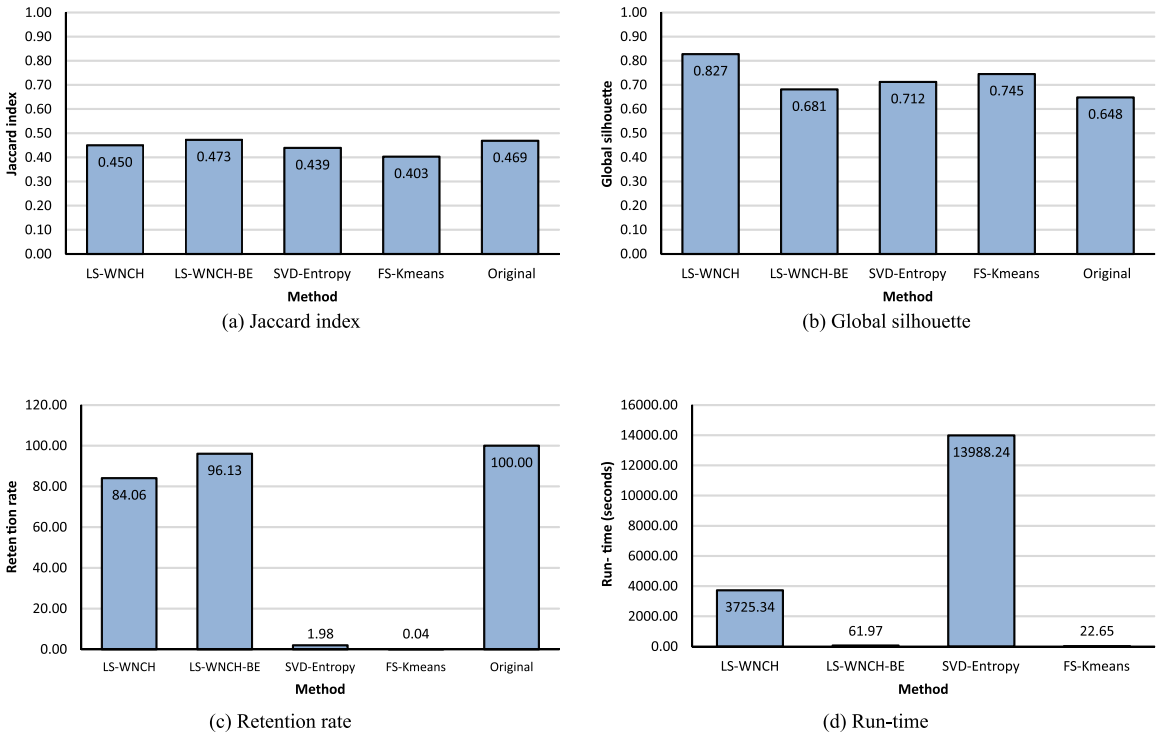


Fig. 5. Average of Jaccard index, global silhouette, retention rate and run-time of LS-WNCH-SR, LS-WNCH-BE, SVD-Entropy and FS-Kmeans methods over the large datasets.

and global silhouette indices).

Finally, we can see that in these larger datasets the proposed method (both versions) is faster than the filter method SVD-Entropy (see Fig. 5d), while LS-WNCH-BE is almost as fast as FS-Kmeans.

In summary, as we can see from Figs. 5a, b, and d, the proposed

method has a reasonable compromise between quality and runtime, regarding the Jaccard index LS-WNCH-BE obtained the best results, and regarding the global silhouette LS-WNCH-SR was the best one. On the other hand, the retention rates achieved by our methods indicate that there is not a great loss of information compared to the other filter methods, as it can be seen in Fig. 5c.

5. Conclusions and future work

In this paper, we proposed a new hybrid feature selection method for unsupervised classification based on the Laplacian Score ranking jointly with a modification of the Calinski–Harabasz index (CH).

From the experiments, we can conclude that the proposed normalization of the CH index, which is used for guiding the search of a good subset of features, allows identifying relevant feature subsets that help to reveal clusters with a high intra-class cohesion and high inter-class separability.

We can also conclude that the proposed method, according to the Jaccard index; selects good subsets of features that produce clusters with high quality compared to other hybrid methods based on ranking. In particular, the version LS-WNCH-BE of our method selects feature subsets that produce clusters with higher quality in terms of the Jaccard index (external evaluation index), in both synthetic and real datasets. For real datasets, when we evaluate the quality of the clusters with the global silhouette, an internal validation index, the best method is SS-SFS since this one is designed for optimizing this index; however the quality in terms of the Jaccard index is the worst on average for all datasets. On the other hand, it is important to note that the results show that the run-times of our method are better than other hybrid, wrapper, and filter methods, being a good option for problems with large datasets.

In the future, we will study how to improve the efficiency of the wrapper stage of the proposed method in order to make it more efficient for larger datasets.

Acknowledgments

The authors are grateful to the referees for their useful comments and suggestions which have led to an improvement of this paper.

References

[1] M. Dash, H. Liu, Feature selection for classification, Intell. Data Anal. 1 (1997) 131–156, URL ⟨http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.6038⟩.
[2] E.P. Xing, M.I. Jordan, R.M. Karp, Feature selection for high-dimensional genomic microarray data, in: Proceedings of the Eighteenth International Conference on Machine Learning, ICML'01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001, pp. 601–608. URL ⟨http://dl.acm.org/citation.cfm?id=645530.655823⟩.
[3] M. Dash, H. Liu, Hybrid search of feature subsets, in: Proceedings of the 5th Pacific Rim International Conference on Artificial Intelligence: Topics in Artificial Intelligence, PRICAI'98, Springer-Verlag, London, UK, 1998, pp. 238–249. URL ⟨http://dl.acm.org/citation.cfm?id=646965.712598⟩.
[4] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, J. Mach. Learn. Res. 5 (2004) 1205–1224, URL ⟨http://dl.acm.org/citation.cfm?id=1005332.1044700⟩.
[5] L. Zhang, G. Sun, J. Guo, Feature selection for pattern classification problems, in: The Fourth International Conference on Computer and Information Technology, 2004, {CIT}'04, IEEE, 2004, pp. 233–237. http://dx.doi.org/10.1109/CIT.2004.1357202.
[6] R. Pudimat, R. Backofen, E.G. Schukat-Talamazzini, Fast feature subset selection in biological sequence analysis, Int. J. Pattern Recognit. Artif. Intell. 23 (02) (2009) 191, http://dx.doi.org/10.1142/S0218001409007107, URL ⟨http://www.worldscinet.com/ijprai/23/2302/S0218001409007107.html⟩.
[7] J. Liang, S. Yang, Y. Wang, An optimal feature subset selection method based on distance discriminant and distribution overlapping, Int. J. Pattern Recognit. Artif. Intell. 23 (08) (2009) 1577, http://dx.doi.org/10.1142/S0218001409007715, URL ⟨http://www.worldscinet.com/ijprai/23/2308/S0218001409007715.html⟩.
[8] F.F. Gonzalez-Navarro, L.A. Belanche-Muñoz, Feature selection for microarray gene expression data using simulated annealing guided by the multivariate joint entropy, Comput. Sist. 18 (2), 275–293.
[9] H. Liu, H. Motoda, Computational Methods of Feature Selection, Chapman Hall, CRC (2008).
[10] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Trans. Knowl. Data Eng. 17 (4) (2005) 491–502.
[11] I. Guyon, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182, URL ⟨http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.3.8934⟩.
[12] S. Alelyani, J. Tang, H. Liu, Feature Selection for Clustering: A Review, 2013.
[13] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Electr. Eng. 40 (1) (2014) 16–28, http://dx.doi.org/10.1016/j.compeleceng.2013.11.024, URL ⟨http://www.sciencedirect.com/science/article/pii/S0045790613003066⟩.
[14] G. Ritter, Robust Cluster Analysis and Variable Selection, CRC Press, 2014.
[15] T. Liu, Y. Liang, W. Ni, A novel approach to feature selection for clustering, in: 2012 Fifth International Conference on Intelligent Computation Technology and Automation (ICICTA), IEEE, 2012, pp. 41–44.
[16] C. Boutsidis, M. Magdon-Ismail, Deterministic feature selection for k-means clustering, IEEE Trans. Inf. Theory 59 (9) (2013) 6099–6110, http://dx.doi.org/10.1109/TIT.2013.2255021 arXiv:1109.5664v4.
[17] S. Tabakhi, P. Moradi, F. Akhlaghian, An unsupervised feature selection algorithm based on ant colony optimization, Eng. Appl. Artif. Intell. 32 (2014) 112–123.
[18] J. Tang, H. Liu, An unsupervised feature selection framework for social media data, IEEE Trans. Knowl. Data Eng. 26 (12) (2014) 2914–2927.
[19] P. Moradi, M. Rostami, A graph theoretic approach for unsupervised feature selection, Eng. Appl. Artif. Intell. 44 (2015) 33–45.
[20] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999) 264–323, http://dx.doi.org/10.1145/331499.331504.
[21] Y. Kim, W.N. Street, F. Menczer, Feature selection in data mining, in: J. Wang (Ed.), Data Mining: Opportunities and Challenges, IGI Global, Hershey, PA, USA, 2003, pp. 80–105. URL ⟨http://dl.acm.org/citation.cfm?id=903826.903831⟩.
[22] N. Sondberg-Madsen, C. Thomsen, J.M. Pena, Unsupervised feature subset selection, in: Proceedings of the Workshop on Probabilistic Graphical Models for Classification, Citeseer, 2003, pp. 71–82.
[23] S.K. Pal, P. Mitra, Pattern Recognition Algorithms for Data Mining, 1st Edition, Chapman and Hall, CRC, 2004.
[24] S. Niijima, Y. Okuno, Laplacian linear discriminant analysis approach to unsupervised feature selection, IEEE/ACM Trans. Comput. Biol. Bioinform. (TCBB) 6 (4) (2009) 605–614.
[25] D. Devakumari, K. Thangavel, Unsupervised adaptive floating search feature selection based on contribution entropy, in: 2010 International Conference on Communication and Computational Intelligence (INCOCCI), IEEE, 2010, pp. 623–627.
[26] J.G. Dy, C.E. Brodley, Feature selection for unsupervised learning, J. Mach. Learn. Res. 5 (2004) 845–889, http://dx.doi.org/10.1016/j.patrec.2014.11.006.
[27] O. Alter, O. Alter, Singular value decomposition for genome-wide expression data processing and modeling, Proc. Natl. Acad. Sci. U. S. A. 97 (18) (2000) 10101–10106.
[28] R. Varshavsky, A. Gottlieb, M. Linial, D. Horn, Novel unsupervised feature filtering of biological data, Bioinformatics 22 (14) (2006) e507–e513, http://dx.doi.org/10.1093/bioinformatics/btl214, URL ⟨http://bioinformatics.oxfordjournals.org/content/22/14/e507.abstract⟩.
[29] M. Banerjee, N.R. Pal, Feature selection with SVD entropy: some modification and extension, Inf. Sci. 264 (2014) 118–134, http://dx.doi.org/10.1016/j.ins.2013.12.029.
[30] Z. Zhao, H. Liu, Spectral feature selection for supervised and unsupervised learning, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 1151–1157.
[31] X. He, D. Cai, P. Niyogi, Laplacian score for feature selection, in: Advances in Neural Information Processing Systems 18, vol. 186, 2005, pp. 507–514. URL ⟨http://books.nips.cc/papers/files/nips18/NIPS2005_0149.pdf⟩.
[32] R. Liu, N. Yang, X. Ding, L. Ma, An unsupervised feature selection algorithm Laplacian score combined with {distance-based} entropy measure, in: 2007 Workshop on Intelligent Information Technology Applications, vol. 3, 2009, 65-68 doi: http://dx.doi.org/10.1109/IITA.2009.390.
[33] P. Padungweang, C. Lursinsap, K. Sunat, Univariate filter technique for unsupervised feature selection using a new Laplacian score based local nearest neighbors, in: Asia-Pacific Conference on Information Processing, 2009, APCIP 2009, vol. 2, IEEE, 2009, pp. 196–200.
[34] M. Dash, H. Liu, J. Yao, Dimensionality reduction of unsupervised data, in: Proceedings of Ninth IEEE International Conference on Tools with Artificial Intelligence, IEEE, 1997, pp. 532–539. http://dx.doi.org/10.1109/TAI.1997.632300.
[35] M. Dash, H. Liu, Handling large unsupervised data via dimensionality reduction, in: 1999 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 1999. URL ⟨http://www.almaden.ibm.com/cs/dmkd/papers/dash.ps⟩.
[36] M. Dash, K. Choi, P. Scheuermann, H. Liu, Feature selection for clustering – a filter solution, in: 2002 IEEE International Conference on Data Mining, 2002, Proceedings, 2002, pp. 115–122 http://dx.doi.org/10.1109/ICDM.2002.1183893.
[37] P. Mitra, C.A. Murthy, S.K. Pal, Unsupervised feature selection using feature similarity, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 301–312.
[38] S. Tabakhi, A. Najafi, R. Ranjbar, P. Moradi, Gene selection for microarray data classification using a novel ant colony optimization, Neurocomputing 168 (2015) 1024–1036, http://dx.doi.org/10.1016/j.neucom.2015.05.022, URL

⟨http://www.sciencedirect.com/science/article/pii/S0925231215006451⟩.
[39] X. Wang, X. Zhang, Z. Zeng, Q. Wu, J. Zhang, Neurocomputing unsupervised spectral feature selection with l 1 -norm graph, Neurocomputing 200 (2016) 47–54, http://dx.doi.org/10.1016/j.neucom.2016.03.017.
[40] M.H. Law, A.K. Jain, M. Figueiredo, Feature selection in mixture-based clustering, in: Advances in Neural Information Processing Systems, 2002, pp. 625–632.
[41] E.R. Hruschka, T.F. Covoes, Feature selection for cluster analysis: an approach based on the simplified silhouette criterion, in: International Conference on Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, vol. 1, IEEE, 2005, pp. 32–38.
[42] M. Breaban, H. Luchian, A unifying criterion for unsupervised clustering and feature selection, Pattern Recognit. 44 (4) (2011) 854–865.
[43] M.H.C. Law, M.A.T. Figueiredo, A.K. Jain, Simultaneous feature selection and clustering using mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 26 (9) (2004) 1154–1166.
[44] W. Sheng, X. Liu, M. Fairhurst, A niching memetic algorithm for simultaneous clustering and feature selection, IEEE Trans. Knowl. Data Eng. 20 (7) (2008) 868–879.
[45] D. Dutta, P. Dutta, J. Sil, Simultaneous continuous feature selection and K clustering by multi objective genetic algorithm, in: Advance Computing Conference (IACC), 2013 IEEE 3rd International, IEEE, 2013, pp. 937–942.
[46] D. Dutta, P. Dutta, J. Sil, Simultaneous feature selection and clustering with mixed features by multi objective genetic algorithm, Int. J. Hybrid. Intell. Syst. 11 (1) (2014) 41–54.
[47] Y. Li, B.-L. Lu, Z.-F. Wu, A hybrid method of unsupervised feature selection based on ranking, in: 18th International Conference on Pattern Recognition {(ICPR'06)}, Hong Kong, China, 2006, pp. 687–690. http://dx.doi.org/10.1109/ICPR.2006.84. URL ⟨http://dl.acm.org/citation.cfm?id=1172253⟩.
[48] E.R. Hruschka, E.R. Hruschka, T.F. Covoes, N.F.F. Ebecken, Feature selection for clustering problems: a hybrid algorithm that iterates between k-means and a Bayesian filter, in: Fifth International Conference on Hybrid Intelligent Systems, 2005, {HIS}'05, IEEE, 2005. http://dx.doi.org/10.1109/ICHIS.2005.42.
[49] M. Dash, M. Dash, H. Liu, H. Liu, Feature selection for clustering, in: Knowledge Discovery and Data Mining. Current Issues and New Applications, vol. 1805, Springer, 2000, pp. 110–121. http://dx.doi.org/10.1007/3-540-45571-X_13.
[50] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, John Wiley & Sons 2001, pp. 544–546.
[51] A. Jashki, M. Makki, E. Bagheri, A.A. Ghorbani, An iterative hybrid filter–wrapper approach to feature selection for document clustering, in: Proceedings of the 22nd Canadian Conference on Artificial Intelligence (AI'09) 2009, 2009.
[52] J. Hu, C. Xiong, J. Shu, X. Zhou, J. Zhu, An improved text clustering method based on hybrid model, Int. J. Mod. Educ. Comput. Sci. (IJMECS) 1 (1) (2009) 35.
[53] Y. Yang, Y. Liao, G. Meng, J. Lee, A hybrid feature selection scheme for unsupervised learning and its application in bearing fault diagnosis, Expert Syst. Appl. 38 (9) (2011) 11311–11320, URL ⟨http://dblp.uni-trier.de/db/journals/eswa/eswa38.html#YangLML11⟩.
[54] J. Yu, A hybrid feature selection scheme and self-organizing map model for machine health assessment, Appl. Soft Comput. 11 (5) (2011) 4041–4054.
[55] W. Fan, N. Bouguila, D. Ziou, Unsupervised hybrid feature extraction selection for high-dimensional non-gaussian data clustering with variational inference, IEEE Trans. Knowl. Data Eng. 25 (7) (2013) 1670–1685.
[56] Y.B. Kim, J. Gao, A new hybrid approach for unsupervised gene selection, in: 2006 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology, 2006, CIBCB'06, IEEE, 2006, pp. 1–8.
[57] Y. Luo, S. Xiong, Clustering ensemble for unsupervised feature selection, in: Fourth International Conference on Fuzzy Systems and Knowledge Discovery, vol. 1, {IEEE} Computer Society, Los Alamitos, {CA}, {USA}, 2009, pp. 445–448. http://dx.doi.org/10.1109/FSKD.2009.449.
[58] E.R. Hruschka, T.F. Covoes, J.E.R. Hruschka, N.F.F. Ebecken, Adapting supervised feature selection methods for clustering tasks, in: Methods for Clustering Tasks In Managing Worldwide Operations and Communications with Information Technology (IRMA 2007 Proceedings), Information Resources Management Association (IRMA) International Conference Vancouver 2007, Idea Group Publishing, Hershey, 2007, pp. 99–102. http://dx.doi.org/10.4018/978-1-59904-929-8.ch024.
[59] M. Dash, Y.-S. Ong, RELIEF-C: efficient feature selection for clustering over noisy data, in: 2011 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, 2011, pp. 869–872.
[60] L. Talavera, An evaluation of filter and wrapper methods for feature selection in categorical clustering, in: Advances in Intelligent Data Analysis VI, Springer, 2005, pp. 440–451.
[61] D. Garcia-Garcia, R. Santos-Rodriguez, Spectral clustering and feature selection for microarray data, in: International Conference on Machine Learning and Applications, 2009, {ICMLA}'09, IEEE, 2009, pp. 425–428. http://dx.doi.org/10.1109/ICMLA.2009.86.
[62] U. Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416, http://dx.doi.org/10.1007/s11222-007-9033-z, URL ⟨http://dl.acm.org/citation.cfm?id=1288832⟩.
[63] T. Calinski, J. Harabasz, A dendrite method for cluster analysis, Commun. Stat. – Theory Methods 3 (1) (1974) 1–27, http://dx.doi.org/10.1080/03610927408827101, URL ⟨http://www.tandfonline.com/doi/abs/10.1080/03610927408827101?journalCode=lsta19#preview⟩.
[64] M. Morita, R. Sabourin, F. Bortolozzi, C.Y. Suen, Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition, in: Seventh International Conference on Document Analysis and Recognition, 2003, Proceedings, IEEE, 2003, pp. 666–670. http://dx.doi.org/10.1109/ICDAR.2003.1227746.
[65] L. Vendramin, R.J.G.B. Campello, E.R. Hruschka, On the comparison of relative clustering validity criteria, in: SDM, 2009, pp. 733–744.
[66] J. Handl, J. Knowles, Cluster generators for large high-dimensional data sets with large numbers of clusters, 2005.
[67] M. Lichman, {UCI} machine learning repository, 2013. URL ⟨http://archive.ics.uci.edu/ml⟩.
[68] C. Boutsidis, M.W. Mahoney, P. Drineas, Unsupervised feature selection for the k-means clustering problem, Adv. Neural Inf. Process. Syst. 22 (2009) 153–161, URL ⟨http://papers2://publication/uuid/976677C3-A2F2-4F98-9CF7-CE550A69D36C⟩.
[69] M. Halkidi, Y. Batistakis, M. Vazirgiannis, On clustering validation techniques, J. Intell. Inf. Syst. 17 (2–3) (2001) 107–145, http://dx.doi.org/10.1023/A:1012801612483, URL ⟨http://dl.acm.org/citation.cfm?id=607585.607609⟩.
[70] P. Jaccard, The distribution of the flora in the alpine zone, New Phytol. 11 (2) (1912) 37–50.
[71] W.M. Rand, Objective criteria for the evaluation of clustering methods, J. Am. Stat. Assoc. 66 (336) (1971) 846–850.
[72] E.B. Fowlkes, C.L. Mallows, A method for comparing two hierarchical clusterings, J. Am. Stat. Assoc. 78 (383) (1983) 553–569.
[73] A. Strehl, J. Ghosh, Cluster ensembles-a knowledge reuse framework for combining partitionings, in: AAAI/IAAI, 2002, pp. 93–99.
[74] Y. Liu, Z. Li, H. Xiong, X. Gao, J. Wu, Understanding of internal clustering validation measures, in: 2010 IEEE 10th International Conference on Data Mining (ICDM), IEEE, 2010, pp. 911–916.
[75] J.C. DUNN, A fuzzy relative of the ISODATA process and its use in detection compact well-separated clusters, J. Cybern. 3 (1973) 32–57.
[76] D.L. Davies, D.W. Bouldin, A cluster separation measure, Pattern Anal. Mach. Intell., IEEE Trans. 2 (1979) 224–227.
[77] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, 9th Edition, Wiley-Interscience, 1990.
[78] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.

Saúl Solorio-Fernández received his M.Sc. degree in Computational Sciences from the National Institute of Astrophysics, Optics and Electronics (INAOE), Mexico, in 2010. He is currently pursuing his Ph.D. at the same academic institution. His research focuses on Unsupervised Features Selection, Clustering, Data Mining and Pattern Recognition.

José Fco. Martínez-Trinidad received his B.S. degree in Computer Science from Physics and Mathematics School of the Autonomous University of Puebla (BUAP), Mexico in 1995, his M.Sc. degree in Computer Science from the faculty of Computers Science of the Autonomous University of Puebla, Mexico in 1997 and his Ph. D. degree from the Center for Computing Research of the National Polytechnic Institute (CIC, IPN), Mexico in 2000. Professor Martinez-Trinidad edited/authored ten books and over one hundred and fifty journal and conference papers on subjects related to Pattern Recognition.

Jesús A. Carrasco-Ochoa received his Ph.D. degree in Computer Science from the Center for Computing Research of the National Polytechnic Institute (CIC-IPN), Mexico, in 2001. He works as a full-time researcher at the National Institute of Astrophysics, Optics and Electronics of Mexico. He has published more than 100 papers on topics related to Pattern Recognition and Data Mining, and co-edited seven books. His current research interests include Logical Combinatorial Pattern Recognition, Data Mining, Testor Theory, Feature and Prototype Selection, Text Analysis, Fast Nearest Neighbor Classifiers and Clustering.