# MCFS: Min-cut-based feature-selection

Carlos G. Vallejo, José A. Troyano, Fernando Enríquez, F. Javier Ortega, Fermín L. Cruz

*Department of Computer Languages and Systems, University of Seville, Avda. Reina Mercedes s/n. 41012 Seville, Spain*

A B S T R A C T

In this paper, MCFS (Min-Cut-based feature-selection) is presented, which is a feature-selection algorithm based on the representation of the features in a dataset by means of a directed graph. The main contribution of our work is to show the usefulness of a general graph-processing technique in the feature-selection problem for classification datasets. The vertices of the graphs used herein are the features together with two special-purpose vertices (one of which denotes high correlation to the feature class of the dataset, and the other denotes a low correlation to the feature class). The edges are functions of the correlations among the features and also between the features and the classes. A classic max-flow min-cut algorithm is applied to this graph. The cut returned by this algorithm provides the selected features. We have compared the results of our proposal with well-known feature-selection techniques. Our algorithm obtains results statistically similar to those achieved by the other techniques in terms of number of features selected, while additionally significantly improving the accuracy.

*Keywords:*
Machine-learning
Feature-selection
Nearest-neighbour
Correlations
Max-flow min-cut
Classification

## 1. Introduction

In supervised learning, a dataset can be described as a set of instances, each composed of a tuple of features $(f_1, f_2, \ldots, f_n)$ and a target variable. If the target variable is discrete (called herein class $c$), then we have an automatic classification problem, which consists of predicting the class of a new instance, given only its feature values. Classification techniques are based on the various combinations of the information supplied by the features and the classes in order to obtain the predictions. Even though it seems that having more features provides better accuracy in the classification phase, this is not always true. Firstly, a number of the features may be irrelevant, redundant, or sparse [1], and may even add noise. Moreover, the classifier may overfit the features, and a larger number of features always negatively affects the execution time, either in the calculus of the classifier in the case of eager classifiers, or in the generalization phase in the case of lazy classifiers. On the other hand, future data collection can benefit from the selection of the most relevant features. Finally, the presence of a lower number of features can make the proposed problem easier to understand.

For all these reasons, the selection of the most appropriate subset of features to be used in the generation and application of the classifier (*feature subset selection*) constitutes a crucial issue in machine-learning.

The feature-selection techniques can be divided into several types, whereby the two main groups are those of filter methods and wrapper methods [2]. The former takes into account only the information contained in the dataset in order to determine the features to be selected, while the latter uses an external classifier to evaluate the goodness of the set of selected features, and this goodness is that which guides the selection. This is the reason why wrapper techniques are, in general, slower under high dimensionality. There are studies that combine both approaches, and perform a filtering step before the application of the wrapper [3]. There is a third type of feature-selection technique (namely, embedded) which includes those machine-learning algorithms, such as decision trees, that, during the learning process, build models that only depend on a certain number of the features.

feature-selection techniques can be also classified as either forward techniques (beginning with an initially empty set of features, these proceed by adding features one by one according to certain criteria) or backward techniques (beginning with the set of all features, these proceed by eliminating features progressively).

In general, feature-selection techniques attempt to:

- Maintain the most *relevant* features: those that best predict the class, either individually or by interacting with other features.
- Eliminate the *redundant* features: those whose predictive power is already provided by other features.

All researchers agree that the feature-selection algorithms should choose the most relevant features. Although many studies emphasize the elimination of redundant features, others warn about the possible damage that this deletion may cause due to the exclusion of potentially relevant features [4,5].

The applications of feature-selection are numerous, and include, among many others [6,7], those related with genomic analysis [8], text mining [9], spam detection [10], image retrieval [11], image classification [12,13], and clustering [14].

Two highly regarded feature-selection techniques [15] are CFS (Correlation-based feature-selection) [16], due to the quality of the subset of selected features, and FCBF (Fast Correlation-Based Filter Solution) [17] thanks to its ability to work with datasets with many features and the small size of the subset of selected features. Both the FCBF and CFS methods remain referents of the state of the art for feature-selection, and they are still widely used in feature-selection comparatives [18–20].

The main idea that guides our work involves the application of a classic graph-based algorithm to the apparently distant field of feature-selection. This application has already been successfully performed with other machine-learning tasks such as instance selection [21] and clustering [22]. A conclusion can be drawn for other fields: if a way to express a problem can be found in terms of graphs, then min-cut can provide a good option towards solving the problem.

The general idea of our algorithm, which is explained in detail in Section 3, can be summarized as follows: given a dataset, a directed graph is generated with a set of vertices corresponding to the features in the dataset, and with two additional vertices, $s$ and $t$, which represent a high and a low correlation with the target class, respectively. The edges in the graph link node $s$ to the feature nodes, which in turn are linked to node $t$. This graph can now be considered as a flow, where $s$ acts as the source node and $t$ as the sink node. The weights of the edges that connect vertices representing features are the correlations between these features. The edges that connect $s$ and $t$ with the former vertices become transformations of the correlations between features and the class. In order to compute these correlations, the definitions are used as given by [16] for the aforementioned CFS algorithm. Once the graph is built, the feature-selection problem is converted into a graph optimization problem, which is solved by using the max-flow min-cut algorithm [23]. As shown below, the vertices included in the resulting min-cut part related to node $s$ constitutes a good subset of the original features.

To the best of our knowledge, there has been no research that performs feature-selection using 'cuts' in the feature set. The only application of the min-cut algorithm related to the task of feature-selection task is discussed in [24], where the authors construct a graph of instances (not features) in order to obtain normalized cuts. These same authors then use the cuts to rank the features according to their capacity to obtain clusters of labelled and unlabelled instances. In contrast, we propose the construction of a graph of features for which the output of the max-flow min-cut algorithm yields the set of selected features. Neither method is comparable since [24] focuses on the selection of features by mixing labelled and unlabelled datasets with a semi-supervised approach.

The rest of the paper is organized as follows. In Section 2 the CFS algorithm is given, which allows us to find the necessary correlations for our technique and the max-flow min-cut algorithm is described with its diverse implementations. In Section 3 our algorithm is given in detail, while in Section 4 an explanation is given regarding the experimental work carried out, the parameter tuning, the comparisons between our technique and the aforementioned CFS and FCBF techniques, and also with the Nearest-Neighbour algorithm, and the statistical analysis of the results. Finally, in Section 5 the results obtained are recapitulated and future lines of research arising from this study are drawn up.

## 2. CFS and min-cut

### 2.1. CFS: Correlation feature-selection

The CFS algorithm [16] (Correlation feature-selection) is a forward-type algorithm as classified by [2], which is based on the correlations between the feature values themselves and between these values and the classes. This algorithm is based on the following hypothesis: *good feature subsets contain features highly correlated with the class, yet uncorrelated with each other.*

First of all, this algorithm discretizes the numeric features and then uses *symmetrical uncertainty* (a modification of a measure of information gain) to estimate the degree of association between discrete features.

### 2.2. Min-cut

#### 2.2.1. Max-flow min-cut problem

*Max-flow problem.* The problem of finding the "maximum flow" (max-flow) in a network combined with seeking the "minimum cut" (min-cut) [25], can be formulated as follows: Let $G = (V, E)$ be a directed graph, and let $s$ and $t$ be two vertices of $G$ with a special meaning, called *source* and *target*, respectively. Let us denote by $u \to v$ the directed edge from vertex $u$ to vertex $v$.

We define an $(s, t) -$ flow as the function $f : E \to \mathbb{R}^+ \cup \{0\}$ that satisfies the following restriction for every vertex $v$, except $s$ and $t$:

$$\sum_u f(u \to v) = \sum_w f(v \to w).$$

This means that the total flow reaching a vertex coincides with its total output flow (assuming, for the sake of simplicity, that $f(u \to v) = 0$ if there is no $u \to v$ edge).

The $f$ flow value is defined as the amount of fluid entering the network from the source node, and it is easily demonstrated that $|f|$ coincides with the flow that reaches $t$:

$$|f| = \sum_w f(s \to w) = \sum_u f(u \to t)$$

Suppose each edge $e$ has a capacity $c(e)$ which is a non-negative value. We impose the condition that $\forall e \in E, f(e) \leq c(e)$. The problem of finding the *maximum flow in the network* involves the calculation of an $(s, t) -$ flow with the above restrictions whose value is the largest possible.

The max-flow problem has many interesting applications, such as network connectivity [26], distributed programming [27], image processing [28] and opinion mining [29].

*Min-cut.* An $(s, t)-$cut is a partition of $V$ into two disjoint subsets $S$ and $T$ ($S \cup T = V$ and $S \cap T = \varnothing$), where $s \in S$ and $t \in T$.

The cost of the cut is the sum of the capacities of the vertices that cross from $S$ to $T$:

$$\|S, T\| = \sum_{v \in S} \sum_{w \in T} c(v \to w).$$

The problem of finding the *minimum cut* consists of finding an $(s, t) -$ cut, whose cost is the lowest possible.

Note that the minimum cut is the most economical way of cutting off all the flow that comes from $s$ and reaches $t$. It should be borne in mind, for further sections, that from this perspective the min-cut represents the solution of an optimization problem.

With the above premises, the Max-Flow Min-Cut Theorem [23] states that the value of the maximum-flow is equal to the cost of the minimum-cut.

*2.2.2. Algorithms that calculate the min-cut*

There are two main families of algorithms for the calculation of the max-flow/min-cut: the augmenting path methods, based on the Ford–Fulkerson style [23]; and the push-relabel Goldberg–Tarjan style methods [30]. There is also a third type of algorithm, which includes the calculation of the min-cut by approximation methods, such as the Monte-Carlo method [31].

For an exhaustive survey of the existing methods, together with their respective complexity analysis, the survey presented in [32] provides a good starting point, and in [33], a comprehensive comparison of many existing algorithms can be found, all of which are of polynomial order.

## 3. Our proposal: the MCFS algorithm

Choosing the best subset of features requires the exploration of all possible subsets, which is a task of cost $2^n$, where $n$ is the number of features in the dataset. Therefore, this becomes an NP problem. If a way can be found to transform this problem into an optimization problem that min-cut can handle, it will have become a polynomial order problem although, of course, the solution encountered may not be optimal since this can be only found by exhaustive search.

Our proposed algorithm, MCFS, is based on the ideas outlined in previous sections: creating a graph with nodes representing the features and arcs representing the correlations between them, and reducing the problem of finding the best subset of features to a network maximum flow calculation problem.

Now let us look at our algorithm in more detail:

1. We start from a labelled dataset composed of instances with $n$ features $(f_1, f_2, \ldots, f_n)$ and a class $c$.
2. From the above dataset, a directed graph is built that consists of:
   (a) $n$ vertices that correspond to the features in the dataset (we call these $v_1, v_2, \ldots, v_n$);
   (b) two more vertices, $s$ and $t$, that correspond to the sink and target, respectively;
   (c) A directed arc $e_{ij}$ between every pair of 'feature' vertices $v_i \rightarrow v_j$;
   (d) A directed arc between $s$ and each of the $v_i$, denoted $e_{si}$ and a directed arc between each of the $v_i$ and $t$, denoted $e_{it}$.
3. The following correlations are calculated:
   (a) correlations between the class and each feature $f_i$: $c_i$;
   (b) correlations between each pair of features $f_i$ and $f_j$: $c_{ij}$ (obviously, $c_{ij} = c_{ji}$).
4. Based on the correlations previously calculated, the following values to the edges can be assigned:
   (a) Since those feature pairs that strongly correlate with each other should be penalized (the information provided by one is already provided by the other), the inverse values of the correlations between each pair of features $(1 - c_{ij})$ are assigned to their connecting edges $e_{ij}$ $\forall i, j = 1 \ldots n, i \neq j$.
   (b) For the edges that connect the $s$ and $t$ nodes with the remaining nodes, the correlation value between each feature and the class is considered. The weight of an edge connecting nodes $s$ and $v_i$ $(e_{si})$ is directly proportional to $c_i$, which is the correlation value between that feature $f_i$ and the class. In the case of an edge connecting $v_i$ and $t$ $(e_{it})$, the weight is proportional to $1 - c_i$. In this way, the features with a high correlation with the class are favoured, since the selected features are those that remain in the subset where the node $s$ is included. As can be observed in Section 4.2, during experimentation it was convenient to apply transformation functions to the values of the edges from $s$ to the intermediate vertices $(e_{si})$ and from these nodes to $t$ $(e_{it})$. The transformation function that is applied to the initial arcs (those coming out of $s$) is called $f_s$. The transformation function applied to the final arcs (those pointing to $t$) is called $f_t$. Fig. 1 shows this structure.
5. Once the graph has been generated, its $(s, t)-$cut is calculated (to this end, we have used the hi_pr algorithm [34] implementation, which is a push-relabel type algorithm).
6. Finally, the interpretation of the obtained result is: If a vertex $v_i$ is in $S$, then that means feature $f_i$ is selected.

These steps are shown in Algorithm 1. Let us define $e(u, v, w)$ as the weighted edge pointing from node $u$ to node $v$ with weight $w$. $G = (V, E)$ is the weighted, directed graph formed by a set of nodes, $V$, and a set of edges, $E$. Let us define *corr* as the matrix of correlations between attributes: it is an $m \times m$ matrix whose elements, $corr_{ij}$, represent the correlation between attributes $f_i$ and $f_j$ or vice versa (it is a symmetrical matrix). Finally, let us define the vector *corrC*, where $corrC_i$ represents the correlation between the class and the attribute $f_i$. Both the matrix and the vector are obtained by CFS; let us name the function that performs such computation as matCFS.

In Fig. 2, an example of this method applied to a small graph is given with only four features for the sake of simplicity. On the left, there is a graph in which the weights of the edges represent the corresponding capacities. On the right, the graph is represented with the same vertices but, in this case, the weights of the edges are now the maximum flow that can pass through each arc. The sinuous line represents the min-cut.

As discussed above, the min-cut represents the most economical way to cut the flow from $s$ to $t$, so the edges with the lowest weights are more susceptible to being cut, in contrast to those with a high weight. In order to reach our goal of separating the best features (those that will stay 'closer' to node $s$) from the worst ('closer' to node $t$), we assign the weights in our graph to favour three types of edges by granting them higher weights:

- Those indicating the selection of features that strongly correlate with the class: high-weight edges connected to $s$ (the weight assigned between $s$ and $v_i$ is directly $c_i$ and this favours the selection of those $v_i \in S$ with high $c_i$, i.e., with low $1 - c_i$).
- Those indicating not to select features that do not strongly correlate to the class: high-weight edges connected to $t$ (the weight assigned between $t$ and $v_i$ is $1 - c_i$ and this favours the discarding of those $v_i \in T$ with low $c_i$, i.e., with high $1 - c_i$).
- Those indicating the selection of feature pairs that do not strongly correlate with each other (the information that they provide is complementary and not already provided by the other): low values of $c_{ij}$ (the weight assigned between $v_i$ and $v_j$ is $1 - c_{ij}$ and this favours keeping together those $v_i$ and $v_j$ with low $c_{ij}$, i.e., with high $1 - c_{ij}$).

Therefore, we are tackling an optimization problem. Let us define $c_{vt}$ as the weights of the edges that go from the intermediate nodes to $t$, $c_{sv}$ as the weights that go from $s$ to the intermediate nodes, and $c_{uv}$ as one minus the weights of the edges that go from one intermediate vertex to another. Since $S$ and $T$ are the two

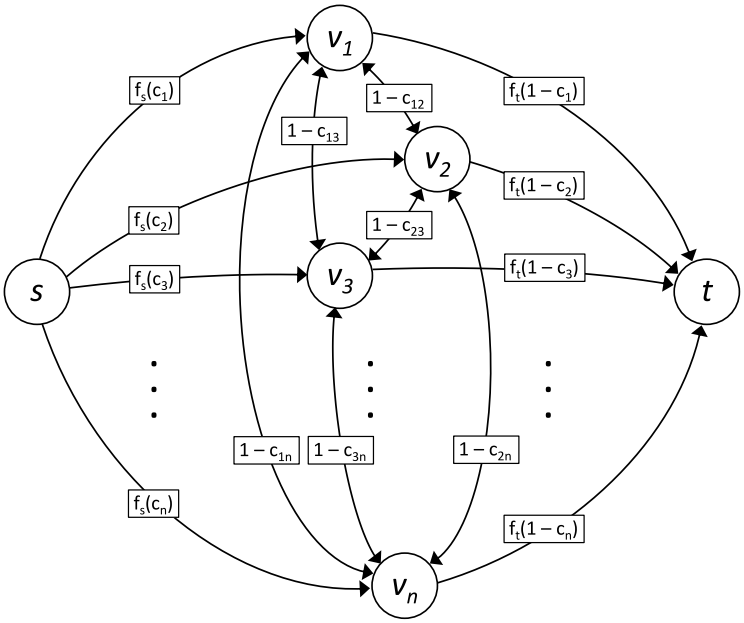


**Fig. 1.** Generated graph structure. For simplicity, the edges between each pair of intermediate vertices are represented by a two-way edge. In fact, there are two edges: one in each direction with the same weight.

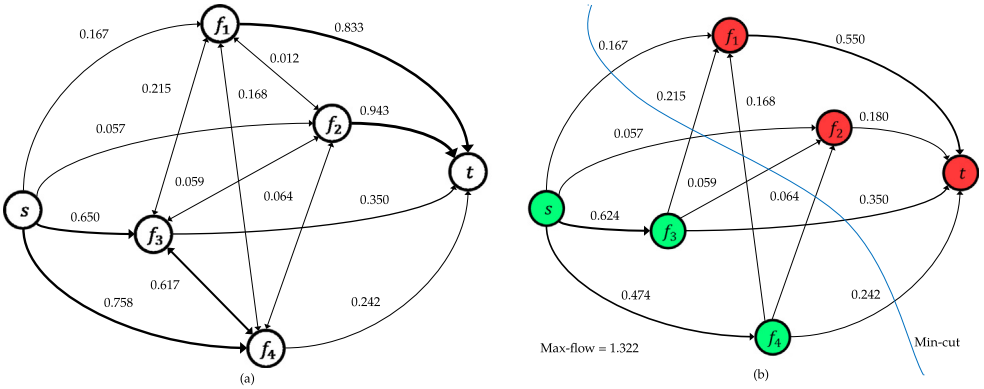


**Fig. 2.** On the left, the initial graph for an example of a dataset with four features. The edges between features are actually two edges, one in each direction. The graph on the right is the result of applying the max-flow min-cut algorithm. The min-cut is shown by the sinuous line. Features $f_3$ and $f_4$ are on the side of the source node and are therefore the features selected.

feature subsets sought with the selected and discarded features respectively, Eq. (1) has to be minimized, which coincides with the minimization of the min-cut calculation.

$$\sum_{v \in S} c_{vt} + \sum_{u \in T} c_{su} + \sum_{u \in S, v \in T} c_{uv} \tag{1}$$

As mentioned above, certain functions are incorporated to transform the weights of the edges that go from node $s$ to the intermediate nodes, and from these nodes to node $t$. These functions are intended to modulate the influence of $c_{sv}$ and $c_{vt}$ with $c_{uv}$. The values of these weights were determined in the experimental phase and will be discussed later.

Following the classification outlined in [2] our technique is of the filter type. As for the forward/backward classification, it is not one type or the other: the features that are included in the selection are chosen in one step.

In the Three-Dimensional Framework that appears in the complete study [35], our algorithm would be located among the Filter-type algorithms, since it uses Dependency Measures, although it could not be integrated into any of the three search strategies proposed therein.

*Algorithm complexity.* The most complex part of the complete algorithm and, hence, that which defines its total complexity, is the min-cut calculation. As noted, we have used the implementation of [34], whose complexity is $O(n^2\sqrt{m})$ where $n = |V|$ and $m = |E|$. In our case, $m = n(n+1)$: $n$ edges from $s$ to $v_i$, another $n$ edges from $v_i$ to $t$ and $n(n-1)$ edges between $v_i$ and $v_j$, one in each direction. Thus the complexity of our algorithm is $O(n^2\sqrt{m}) = O(n^2\sqrt{n(n+1)}) = O(n^3)$, which corresponds to the values observed experimentally.

## 4. Experimentation

### 4.1. Databases and comparison criteria

In order to achieve a statistical result that could corroborate the validity of our algorithm, we experimented with 38 datasets: 35 were taken from the UCI repository [36], all of which had 10 or more features; those with fewer features were discarded since they are considered irrelevant for a study of feature reduction. We also used 3 datasets (Arcene, Gisette, and Madelon) that competed in NIPS 2003 [37]. Table 1 shows the number of instances and the number of features of the datasets used, together with

**Algorithm 1:** MCFS.

**Input**: $D$: Dataset, $it$: input threshold, $ot$: output threshold
**Output**: $S$: Subset with the chosen attributes

$\langle \boldsymbol{corr}, \boldsymbol{corrC} \rangle = \text{matCFS}(D)$
/* ***corr*** is the matrix of correlations between attributes. ***corrC*** is a vector containing the correlations between classes and attributes. */
// Now the graph $G = (V, E)$ is created.
$V = \{v_1, v_2, \ldots, v_m, s, t\}$
/* Vertex $v_i$ corresponds to attribute $f_i$, $i = 1 \ldots m$; $s$ and $t$ are the source and sink nodes, respectively. */
**for** $i = 1$ **to** $m$ **do**
  **if** $(corrC_i \geq it)$ **then**
    $E = E \cup \{e(s, v_i, (m-1) \cdot corrC_i)\}$
  **end if**
  **if** $((1 - corrC_i) \geq ot)$ **then**
    $E = E \cup \{e(v_i, t, (m-1) \cdot (1 - corrC_i))\}$
  **end if**
  **for** $j = i + 1$ **to** $m$ **do**
    **if** $(i \neq j)$ **then**
      $E = E \cup \{e(v_i, v_j, corr_{ij})\}$
      $E = E \cup \{e(v_j, v_i, corr_{ij})\}$
    **end if**
  **end for**
**end for**
$C_s = \text{minCut}(G)$
/* minCut is the algorithm that computes the min-cut; $C_s$ is the cut containing $s$. */
$S = \emptyset$
**for** $i = 1$ **to** $m$ **do**
  /* The attributes are added to the output subset. */
  **if** $(v_i \in C_s)$ **then**
    $S = S \cup \{f_i\}$
  **end if**
**end for**

the number of classes of each dataset. This work is focused on supervised classification, working with labelled data, but there are also ways to face the feature-selection problem using a semi-supervised approach, which uses both labelled and unlabelled data [38].

The results of MCFS are compared with those of CFS [16], which is considered one of the methods that achieves the highest accuracy, and also with FCBF [17], which is among those methods that obtain a better reduction, in addition to being very fast especially with datasets with a large number of features. A description of CFS is given in Section 2.1. The FCBF method is based on the calculation of the symmetrical uncertainty with the class and the concepts of *Predominant correlation* and *Predominant feature*, which prevent the correlations between all pairs of features ($O(n^2)$ for $n$ features) from being calculated. In the end, three successive heuristics are employed to select the final features.

Two metrics are defined for our experiments: accuracy and size. The accuracy is measured as the percentage of the number of instances correctly classified out of the total number of instances for every dataset. Accuracy results are given by the 1NN classifier, without any feature-selection, as the accuracy baseline. The accuracy results of each selection method (MCFS, CFS and FCBF) are calculated by the application of 1NN, using the reduced datasets given by each method. All experiments are performed using stratified 10-fold cross validation.

The size is measured as the ratio (shown as a percentage) between the number of selected features by using the technique and the total number of features of the dataset. This means that a lower numerical value indicates that the set of features is also smaller, and consequently that the algorithm has a greater reduction capacity. For baseline experiments, when no selection is applied, the size is always 100.

### 4.2. Parameter adjustment

During implementation, the values of the correlations with the class, that is, the weights of the edges that link the source node with the intermediate vertices were normalized, such that the average was 0.5. As mentioned above, transformations were applied to the weights of the edges that go from the source node to intermediate nodes and from said intermediate nodes to the target. On the other hand, it was observed experimentally that better results were obtained when the capacities of the edges going from the source to the intermediate vertices (once normalized they add up to the same amount as those that link the intermediate nodes with the target) were balanced with respect to the capacities of the edges that link intermediate nodes: in a graph representing a dataset with $n$ features, there are $n$ edges from the source to intermediate nodes (and from these to the target) and $n(n-1)$ edges between the intermediate nodes. Hence, the weights of the non-intermediate edges are multiplied by $n-1$.

It was also observed experimentally that better results were obtained if the edges from the source to the intermediate nodes that showed low correlations with the class were directly removed, and also their equivalent edges between the intermediate nodes and the target. To this end, two thresholds called *input threshold* ($it$) and *output threshold* ($ot$) were set.

- At the source node, if the correlation of a feature (bear in mind that these were normalized) exceeded $it$, then the edge was incorporated multiplied by $n-1$, otherwise, the edge was not included.
- At the target node, if one minus the normalized correlation exceeded $ot$, then the edge was incorporated multiplied by $n-1$, otherwise, the edge was not included.

Let us consider this decision. In simple cases, such as Iris, which has only four features, this means something very obvious: the arcs that connect $s$ with the features that correlate with the class (in Iris, $f_3$ and $f_4$) are arcs with non-zero capacity, as are the arcs linking $f_1$ and $f_2$ with $t$. Clearly, the min-cut consists of $f_3$ and $f_4$. At this point, the result may seem trivial, but what happens when there are many features? Our algorithm discards a priori those features that do not correlate with the class, although the correlations between features may cause the final min-cut to obtain a set of features where some of the initially discarded features are included or possibly a number of those considered as candidates are discarded.

In order to adjust the threshold values, we carried out thousands of executions on all experimental datasets with various values of $it$ and $ot$, the same for all sets, so that the values obtained would not be specific to a particular set, but would remain as generic as possible. Finally, the values $it = 0.52$ and $ot = 0.49$ were reached.

**Table 1**
Number of instances, features, and classes of the datasets used in the experimentation.

| Dataset | #instances | #features | #classes | Dataset | #instances | #features | #classes |
|---|---|---|---|---|---|---|---|
| Ads | 3 279 | 1 558 | 2 | Madelon | 2000 | 500 | 2 |
| Anneal.ORIG | 898 | 38 | 6 | Multifeatures | 2000 | 649 | 10 |
| Arcene | 100 | 10 000 | 2 | Mushroom | 8124 | 22 | 2 |
| Autos | 205 | 25 | 7 | Musk2 | 6598 | 166 | 2 |
| Colic | 368 | 22 | 2 | Optdigits | 5620 | 64 | 7 |
| Colic.ORIG | 368 | 27 | 2 | Page blocks | 5473 | 10 | 5 |
| Credit-g | 1 000 | 20 | 2 | Promoters | 106 | 57 | 2 |
| Cylinder-band | 540 | 39 | 2 | Schizo | 340 | 14 | 2 |
| Flags | 194 | 29 | 8 | Sick | 3772 | 29 | 2 |
| Gisette | 6 000 | 5 000 | 2 | Solar flare 1 | 323 | 12 | 2 |
| Heart (Cleveland) | 303 | 13 | 2 | Solar flare 2 | 1066 | 12 | 3 |
| Heart (Hungarian) | 294 | 13 | 2 | Sonar | 208 | 60 | 2 |
| Heart (Long Beach VA) | 200 | 13 | 2 | Spambase | 4601 | 57 | 2 |
| Heart (Statlog) | 270 | 13 | 2 | Splice | 3190 | 60 | 3 |
| Heart (Swiss) | 123 | 13 | 2 | Sponge | 76 | 45 | 3 |
| Hepatitis | 155 | 19 | 2 | Vehicle | 846 | 18 | 4 |
| Image segmentations | 2 310 | 19 | 7 | Vote | 435 | 16 | 2 |
| Ionosphere | 351 | 34 | 2 | Waveform | 5000 | 40 | 3 |
| Led creator + 17 | 10 000 | 24 | 10 | Wine | 178 | 13 | 3 |

### 4.3. Experimental results and statistical analysis

It is hard to balance the size reduction of the set of features and, while simultaneously achieving the best accuracy possible. Figs. 3, 4, and 5 graphically express this trade-off. A comparison of the accuracy and the size of the feature set selected by our algorithm against the rest of algorithms analysed is presented. Each point represents the result obtained for a given dataset.

The horizontal axis represents the difference, as a percentage, between the number of features selected by MCFS and those selected by the corresponding technique. As can be observed, MCFS is fairly balanced with CFS but obtains worse results than FCBF, which is an algorithm well-known for achieving large reduction rates. In Fig. 5 where MCFS is compared to BASELINE, all the points are naturally on the left of 0, since BASELINE has no reduction.

The vertical axis represents the difference, as a percentage, between the accuracy obtained by MCFS and that obtained by the corresponding technique. This means that if the point is above the horizontal axis, then MCFS obtains better accuracy than the other technique. As can be observed, our technique greatly surpasses FCBF and CFS, and remains more balanced with BASELINE in the number of points, although the positive points lie further from the axis than those in the negative part. The appearance of a point in the lower right-hand quadrant means that the other technique surpasses ours in accuracy and reduction rate; this only occurs, however, with 5 of the 38 datasets in the comparison with CFS (including those that fall on either of the two axes), which is the same as when our technique is compared with FCBF.

In the case of the BASELINE, obviously there is no appearance of any points in the right-hand quadrants. The remaining cases are in the other quadrants, where the points mean that our technique is better in accuracy, reduction, or in both measures. However, this simply gives a visual idea of the performance of our technique. A closer examination of the results of MCFS and the other algorithms is given in the following sections, resulting in a rigorous statistical analysis.

#### 4.3.1. Accuracy

Table 2 shows the accuracy values of the techniques tested herein.

In comparison to the nearest-neighbour technique, which has no reductions, MCFS obtains better accuracy results in 25 of the databases; the results are levelled in the other 3. This shows that MCFS has major editing capabilities.

**Table 2**
Accuracy of all techniques and MCFS. This table shows the accuracy achieved with each technique, expressed as the percentage of hits using the test set. Stratified 10-fold cross validation has been used.

| Database | 1NN | FCBF | CFS | MCFS |
|---|---|---|---|---|
| Ads | 95.91 | 96.07 | 96.74 | 96.80 |
| Anneal.ORIG | 95.43 | 92.09 | 93.10 | 93.43 |
| Arcene | 81.00 | 76.00 | 88.00 | 85.00 |
| Autos | 74.63 | 79.02 | 79.02 | 84.39 |
| Colic | 80.16 | 76.09 | 76.09 | 82.61 |
| Colic.ORIG | 82.61 | 83.15 | 83.15 | 85.33 |
| Credit-g | 72.40 | 66.40 | 66.40 | 69.00 |
| Cylinder-band | 76.85 | 60.93 | 72.22 | 79.81 |
| Flags | 57.73 | 21.65 | 21.65 | 60.31 |
| Gisette | 95.75 | 89.95 | 94.75 | 96.37 |
| Heart (Cleveland) | 75.25 | 76.57 | 76.57 | 78.55 |
| Heart (Hungarian) | 78.23 | 81.29 | 79.93 | 76.87 |
| Heart (Long Beach VA) | 71.00 | 63.00 | 63.00 | 63.00 |
| Heart (Statlog) | 74.44 | 78.15 | 76.67 | 76.67 |
| Heart (Swiss) | 91.87 | 89.43 | 89.43 | 89.43 |
| Hepatitis | 81.94 | 84.52 | 81.94 | 80.65 |
| Image segmentations | 97.01 | 96.49 | 97.32 | 97.36 |
| Ionosphere | 86.61 | 88.03 | 89.46 | 91.74 |
| Led creator + 17 | 51.29 | 56.88 | 63.55 | 64.77 |
| Madelon | 56.50 | 56.00 | 77.85 | 83.90 |
| Multifeatures | 98.00 | 97.95 | 98.50 | 97.25 |
| Mushroom | 100.00 | 99.02 | 99.02 | 100.00 |
| Musk2 | 95.85 | 86.54 | 94.04 | 96.10 |
| Optdigits | 98.72 | 97.26 | 98.72 | 98.45 |
| Page blocks | 95.91 | 93.86 | 95.52 | 94.77 |
| Promoters | 79.25 | 88.68 | 88.68 | 89.62 |
| Schizo | 60.59 | 73.82 | 73.82 | 100.00 |
| Sick | 96.29 | 96.13 | 96.13 | 96.37 |
| Solar flare 1 | 95.67 | 97.52 | 95.36 | 97.52 |
| Solar flare 2 | 99.04 | 99.44 | 99.44 | 99.44 |
| Sonar | 85.58 | 77.88 | 86.06 | 86.54 |
| Spambase | 90.83 | 91.44 | 91.28 | 90.11 |
| Splice | 75.05 | 81.72 | 81.72 | 87.05 |
| Sponge | 94.74 | 93.42 | 93.42 | 94.74 |
| Vehicle | 70.09 | 48.82 | 60.52 | 64.30 |
| Vote | 91.72 | 94.02 | 94.02 | 94.94 |
| Waveform | 73.82 | 71.76 | 79.20 | 79.20 |
| Wine | 96.07 | 96.07 | 96.63 | 96.07 |
| Average | 83.52 | 81.50 | 83.92 | 86.66 |

A statistical analysis has been performed of the results in order to objectively assess the MCFS algorithm. The results of all techniques have been compared using the Friedman test, which is the method required to compare the performance of one new technique versus others on different databases [39,40]. The Friedman test ($\chi_F^2 = 19.583$, $df = 3$, $p < 0.001$) shows

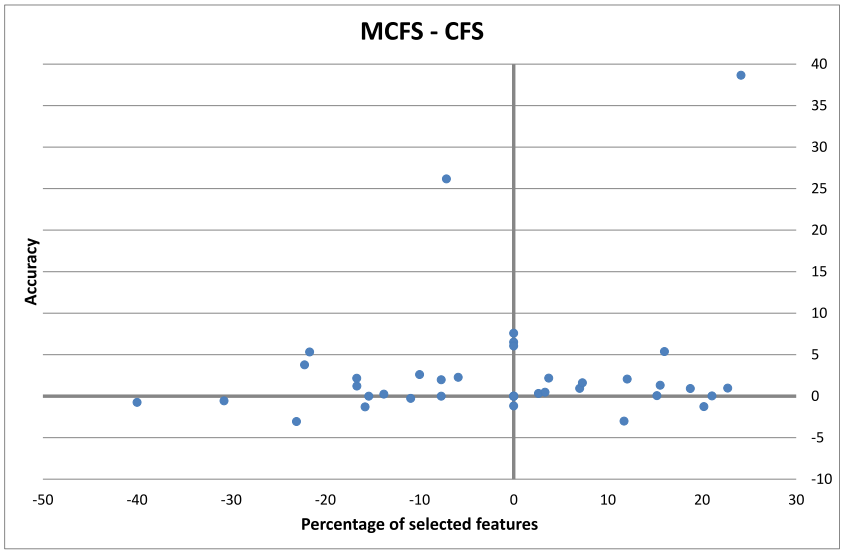


**Fig. 3.** Accuracy-Size Balance: Comparison between MCFS and CFS. The vertical axis represents the difference, as a percentage, between the accuracy obtained by MCFS and that obtained by CFS. If a point is over the horizontal axis it is interpreted as MCFS being more accurate. The horizontal axis represents the difference, as a percentage, between the number of features selected by MCFS and the number of features selected by CFS. A point far to the left of the vertical axis means a large reduction of MCFS for the corresponding dataset.

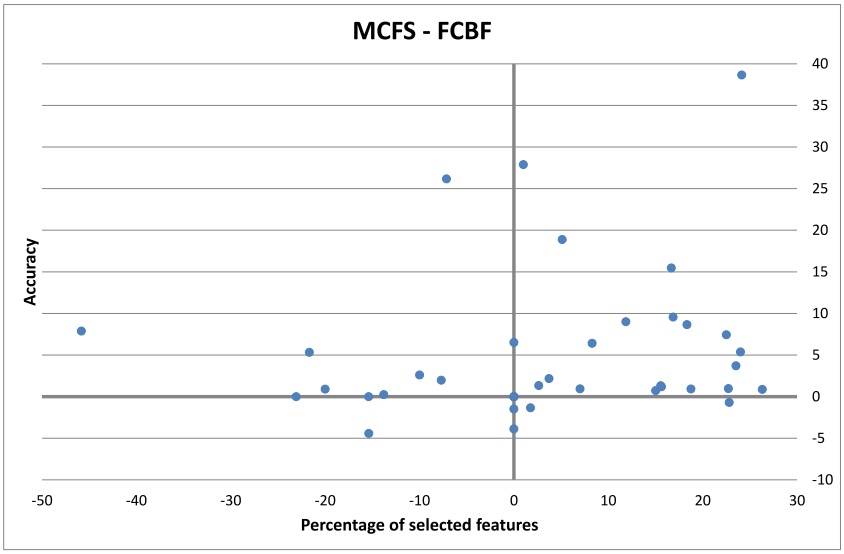


**Fig. 4.** Accuracy-Size Balance: Comparison between MCFS and FCBF. The vertical axis represents the difference, as a percentage, between the accuracy obtained by MCFS and that obtained by FCBF. If a point is over the horizontal axis it is interpreted as MCFS being more accurate. The horizontal axis represents the difference, as a percentage, between the number of features selected by MCFS and the number of features selected by FCBF. A point far to the left of the vertical axis means a large reduction of MCFS for the corresponding dataset.

significant differences between the accuracies of the techniques analysed (with $\alpha = 0.05$). The average ranks of the Friedman test are shown in Table 3.

Following Demšar [39] and García and Herrera [40], we performed a post-hoc analysis (Table 3). The meaning of the columns is as follows: "Avg. Rank" is the Friedman rank; "$R_i - R_{MCFS}$" is the difference between the Friedman rank of the technique and the Friedman rank of MCFS; "$z$"$= (R_i - R_{MCFS})/\sqrt{\frac{k(k+1)}{6N}}$ (where $k = 4$ is the number of techniques and $N = 36$ the number of databases); and "p(uni)" is the unilateral $p$-value for $z$.

According to the Bonferroni–Dunn correction, differences are significant if $p$ is lower than $\alpha/(k-1)$, that is, 0.0166667 for $\alpha = 0.05$. It can be observed that $p(uni)$ is below that value ($p = 0.00184$ for BASELINE, $p = 0.00002$ for FCBF, and $p = 0.01409$ for CFS), which means that the differences are statistically significant.

**Table 3**
Ranks of the Friedman test for accuracy values.

| | Avg. rank | $R_i - R_{MCFS}$ | $z$ | $p$ (uni) |
|---|---|---|---|---|
| BASELINE | 2.32 | −0.86 | −2.90369 | 0.00184 |
| FCBF | 1.97 | −1.21 | −4.08543 | 0.00002 |
| CFS | 2.53 | −0.65 | −2.19465 | 0.01409 |
| MCFS | 3.18 | 0.00 | 0.00000 | |

The ranges of the Friedman test are greater than those of the other three techniques (remember that, in accuracy, the higher, the better), and hence it can be stated that the accuracy obtained by MCFS is significantly better than those obtained by the other three techniques.

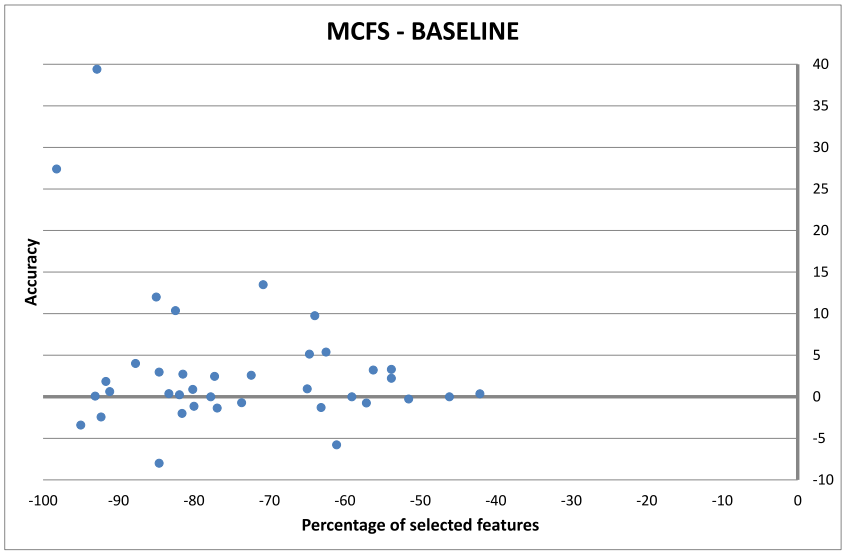


**Fig. 5.** Accuracy-Size Balance: Comparison between MCFS and BASELINE. The vertical axis represents the difference, as a percentage, between the accuracy obtained by MCFS and that obtained by BASELINE. If a point lies above the horizontal axis it is interpreted as MCFS being more accurate. The horizontal axis represents the difference, as a percentage, between the number of features selected by MCFS and the number of features selected by BASELINE. A point far to the left of the vertical axis means a large reduction of MCFS for the corresponding dataset. All points are to the left of the vertical axis since BASELINE applies no reduction and, therefore, the size of the datasets for this system is always 100%.

*4.3.2. Reduction*

Table 4 shows the size of the sets of instances selected by each technique in the datasets.

The Friedman test applied to the size values ($\chi_F^2 = 81.780$, $df = 3$, $p < 0.001$) shows significant differences between the algorithms. The average ranks of the Friedman test are shown in Table 5.

Following [39,40], a post-hoc analysis was again performed (Table 5). The meanings of the columns are the same as those in the previous table.

Recall that according to the Bonferroni–Dunn correction, differences are significant if $p$ is lower than $\alpha/(k-1)$, that is, 0.0166667 for $\alpha = 0.05$. The MCFS technique is significantly better than BASELINE ($p < 0.001$, and the Friedman range for BASELINE is greater than that for MCFS; in reduction, lower values indicate better results). The differences with the other two techniques are not significant ($p = 0.01670$ for FCBF, and $p = 0.50000$ for CFS).

*4.3.3. Summary of results and additional remarks*

It has been shown in previous sections that the Friedman test followed by post-hoc analysis establishes MCFS as being significantly better in accuracy than all the other algorithms analysed.

The accuracy of our algorithm is significantly greater than that obtained by the nearest-neighbour method (86.66 versus 83.52 on average, whereby the differences are statistically significant). This means that MCFS has strong editing capabilities: it removes features that worsen the classification, due to their provision of noise, mislabelling, etc.

## 5. Conclusions and future work

In this paper, we have presented a feature-selection algorithm based on a graph of features. We apply a maximum-flow algorithm to this graph in order to find the most relevant features of a dataset. Experiments and their statistical analysis have shown that this technique achieves significantly better accuracy than that of other, commonly used algorithms.

**Table 4**
Size of the set of features selected by each technique and MCFS; the values are the ratio between the number of selected features and the total number of features, expressed as a percentage. This means that a lower value represents a higher reduction capacity. Recall that stratified 10-fold cross-validation has been used.

| Database | BASELINE | FCBF | CFS | MCFS |
|---|---|---|---|---|
| Ads | 100.00 | 4.81 | 4.62 | 19.83 |
| Anneal.ORIG | 100.00 | 15.79 | 15.79 | 18.42 |
| Arcene | 100.00 | 0.39 | 0.53 | 12.25 |
| Autos | 100.00 | 12.00 | 20.00 | 36.00 |
| Colic | 100.00 | 22.73 | 22.73 | 22.73 |
| Colic.ORIG | 100.00 | 14.81 | 14.81 | 18.52 |
| Credit-g | 100.00 | 15.00 | 15.00 | 5.00 |
| Cylinder-band | 100.00 | 10.26 | 15.38 | 15.38 |
| Flags | 100.00 | 3.45 | 3.45 | 27.59 |
| Gisette | 100.00 | 0.56 | 1.54 | 8.84 |
| Heart (Cleveland) | 100.00 | 53.85 | 53.85 | 46.15 |
| Heart (Hungarian) | 100.00 | 38.46 | 46.15 | 23.08 |
| Heart (Long Beach VA) | 100.00 | 15.38 | 15.38 | 15.38 |
| Heart (Statlog) | 100.00 | 46.15 | 53.85 | 46.15 |
| Heart (Swiss) | 100.00 | 23.08 | 23.08 | 7.69 |
| Hepatitis | 100.00 | 36.84 | 52.63 | 36.84 |
| Image segmentations | 100.00 | 31.58 | 36.84 | 57.89 |
| Ionosphere | 100.00 | 11.76 | 41.18 | 35.29 |
| Led creator + 17 | 100.00 | 75.00 | 45.83 | 29.17 |
| Madelon | 100.00 | 0.80 | 1.80 | 1.80 |
| Multifeatures | 100.00 | 20.03 | 22.65 | 42.84 |
| Mushroom | 100.00 | 18.18 | 18.18 | 40.91 |
| Musk2 | 100.00 | 1.20 | 6.02 | 18.07 |
| Optdigits | 100.00 | 32.81 | 59.38 | 48.44 |
| Page blocks | 100.00 | 40.00 | 60.00 | 30.00 |
| Promoters | 100.00 | 10.53 | 10.53 | 17.54 |
| Schizo | 100.00 | 14.29 | 14.29 | 7.14 |
| Sick | 100.00 | 20.69 | 20.69 | 6.90 |
| Solar flare 1 | 100.00 | 8.33 | 25.00 | 8.33 |
| Solar flare 2 | 100.00 | 16.67 | 16.67 | 16.67 |
| Sonar | 100.00 | 16.67 | 31.67 | 35.00 |
| Spambase | 100.00 | 24.56 | 26.32 | 26.32 |
| Splice | 100.00 | 36.67 | 36.67 | 15.00 |
| Sponge | 100.00 | 6.67 | 6.67 | 22.22 |
| Vehicle | 100.00 | 22.22 | 61.11 | 38.89 |
| Vote | 100.00 | 25.00 | 25.00 | 43.75 |
| Waveform | 100.00 | 15.00 | 37.50 | 37.50 |
| Wine | 100.00 | 76.92 | 84.62 | 53.85 |
| Average | 100.00 | 22.08 | 27.56 | 26.26 |

**Table 5**
Ranks of the Friedman test and post-hoc analysis for feature set size values.

| | Avg. rank | $R_i - R_{MCFS}$ | $z$ | $p$ (uni) |
|---|---|---|---|---|
| BASELINE | 4.00 | 1.79 | 6.04374 | <0.00001 |
| FCBF | 1.58 | −0.63 | −2.12712 | 0.01670 |
| CFS | 2.21 | 0.00 | 0.50000 | 0.50000 |
| MCFS | 2.21 | 0.00 | 0.00000 | |

The main conclusion drawn is that, if we find a way to express a problem in terms of graphs, then min-cut (and, in general, other graph algorithms) can provide a good option to help solve the problem. In the case of a machine-learning task, such as feature-selection, a transformation is required that enables a graph to be obtained from the training data. This way of interpreting a database through a graph is, in fact, the key aspect of our proposal. In this respect, there are several attractive lines of research for future exploration, such as the implementation of other transformation functions for edge weights, and the continued exploration of the potential of the graph representation of problems in other machine-learning tasks.

## CRediT authorship contribution statement

**Carlos G. Vallejo:** Conceptualization, Methodology, Investigation, Formal analysis, Data curation, Software, Writing - original draft, Resources, Validation. **José A. Troyano:** Project administration, Investigation, Supervision, Data curation, Writing - original draft, Writing - review & editing. **Fernando Enríquez:** Writing - review & editing, Resources, Visualization, Funding acquisition. **F. Javier Ortega:** Writing - review & editing, Resources, Funding acquisition. **Fermín L. Cruz:** Writing - review & editing, Validation.

## Acknowledgements

This work has been funded by the Spanish Ministry of Economy and Business, Innovation, and Universities through the projects 'Energy Efficiency and Performance of Data Centers by Smart Virtualization and Deep Learning Event Detection' (RTI2018-098 062-A-I00) and 'Vision and Crowdsensing Technology for an Optimal Response in Physical-Security' (TIN2017-82113-C2-1-R).

## References

[1] C. Shi, Q. Ruan, G. An, Sparse feature selection based on graph laplacian for web image annotation, Image Vis. Comput. 32 (3) (2014) 189–201.
[2] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1) (1997) 273–324.
[3] O. Abedinia, N. Amjady, H. Zareipour, A new feature selection technique for load and price forecast of electrical power systems, IEEE Trans. Power Syst. 32 (1) (2017) 62–74.
[4] H. Liu, H. Motoda, R. Setiono, Z. Zhao, Feature selection: An ever evolving frontier in data mining, J. Mach. Learn. Res. - Proc. Track 10 (2010) 4–13.
[5] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, H. Liu, Advancing Feature Selection Research - ASU Feature Selection Repository, Technical Report, Arizona State University, 2010.
[6] A. Jović, K. Brkić, N. Bogunović, A review of feature selection methods with applications, in: 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), IEEE, 2015, pp. 1200–1205.
[7] M. Taradeh, M. Mafarja, A.A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, H. Fujita, An evolutionary gravitational search-based feature selection, Inform. Sci. 497 (2019) 219–239.
[8] Z.M. Hira, D.F. Gillies, A review of feature selection and feature extraction methods applied on microarray data, Adv. Bioinf. 2015 (2015).
[9] S. Van Landeghem, T. Abeel, Y. Saeys, Y. Van de Peer, Discriminative and informative features for biomolecular text mining with ensemble feature selection, Bioinformatics 26 (18) (2010) i554–i560.
[10] H. Faris, A.-Z. Ala'M, A.A. Heidari, I. Aljarah, M. Mafarja, M.A. Hassonah, H. Fujita, An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks, Inf. Fusion 48 (2019) 67–83.
[11] K. Belattar, S. Mostefai, A. Draa, A hybrid ga-lda scheme for feature selection in content-based image retrieval, Int. J. Appl. Metaheuristic Comput. (IJAMC) 9 (2) (2018) 48–71.
[12] W. He, X. Zhu, D. Cheng, R. Hu, S. Zhang, Unsupervised feature selection for visual classification via feature-representation property, Neurocomputing 236 (2017) 5–13, Good Practices in Multimedia Modeling.
[13] X. Zhu, H.-I. Suk, L. Wang, S.-W. Lee, D. Shen, A novel relational regularization feature selection method for joint regression and classification in ad diagnosis, Med. Image Anal. 38 (2017) 205–214.
[14] S. Alelyani, J. Tang, H. Liu, Feature selection for clustering: A review, in: Data Clustering, Chapman and Hall/CRC, 2018, pp. 29–60.
[15] Y. Saeys, I. Inza, P. Larrañaga, A review of feature selection techniques in bioinformatics, Bioinformatics 23 (19) (2007) 2507–2517.
[16] M.A. Hall, Correlation-based Feature Subset Selection for Machine Learning (Ph.D. thesis), University of Waikato, Hamilton, New Zealand, 1998.
[17] L. Yu, H. Liu, Feature selection for high-dimensional data: A fast correlation-based filter solution, in: 20th International Conference on Machine Learning, 2003, pp. 856–863.
[18] H. Kuswanto, R.Y. Nurhidayah, H. Ohwada, Comparison of feature selection methods to classify inhibitors in dud-e database, Procedia Comput. Sci. 144 (2018) 194–202.
[19] R. Ge, M. Zhou, Y. Luo, Q. Meng, G. Mai, D. Ma, G. Wang, F. Zhou, Mctwo: a two-step feature selection algorithm based on maximal information coefficient, BMC Bioinformatics 17 (1) (2016) 142.
[20] S.S. Gandhi, S. Prabhune, Overview of feature subset selection algorithm for high dimensional data, in: 2017 International Conference on Inventive Systems and Control (ICISC), IEEE, 2017, pp. 1–6.
[21] C.G. Vallejo, J.A. Troyano, F.J. Ortega, InstanceRank: Bringing order to datasets, Pattern Recognit. Lett. 31 (2) (2010) 133–142.
[22] X. Chang, F. Nie, Z. Ma, Y. Yang, Balanced k-means and min-cut clustering, 2014, arXiv preprint arXiv:1411.6235.
[23] L.R. Ford, D.R. Fulkerson, Flows in Networks, Princeton University Press, 1962.
[24] Z. Zhao, H. Liu, Semi-supervised feature selection via spectral analysis, in: Proceedings of the Seventh SIAM International Conference on Data Mining, 2007, pp. 641–646.
[25] G. Dantzig, D. Fulkerson, On the Max-Flow Min-Cut Theorem of Networks, RAND Corporation, 1964.
[26] M. Mansour, F. Jarray, An iterative solution for the coverage and connectivity problem in wireless sensor network, Procedia Comput. Sci. 63 (2015) 494–498.
[27] T.A. Johnson, R. Eigenmann, T.N. Vijaykumar, Min-cut program decomposition for thread-level speculation, in: PLDI'04, ACM, Washington, DC, USA, 2004, pp. 59–70.
[28] D. Greig, B. Porteous, A. Seheult, Exact maximum a posteriori estimation for binary images, J. R. Stat. Soc. Ser. B Stat. Methodol. 51 (2) (1989) 271–279.
[29] B. Pang, L. Lee, A sentimental education: Sentiment analysis using subjectivity summarization bases on minimum cuts, in: Proceeding of the ACL, 2004, pp. 271–278.
[30] A.V. Goldberg, R.E. Tarjan, A new approach to the maximum-flow problem, J. ACM 35 (4) (1988) 921–940.
[31] D.R. Karger, Minimum cuts in near-linear time, in: 28th Annual ACM Symposium on Theory of Computing, 1996, pp. 56–63.
[32] Y. Boykov, V. Kolmogorov, An experimental comparison of Min-Cut/Max-Flow algorithms for energy minimization in vision, IEEE Trans. PAMI 26 (9) (2004) 1124–1137.
[33] C.S. Chekuri, A.V. Goldberg, D.R. Karger, M.S. Levine, C. Stein, Experimental Study of Minimum Cut Algorithms, Technical Report 96–132, NECI TR 96–132, 1996.
[34] B.V. Cherkassky, A.V. Goldberg, On implementing push-relabel method for the maximum flow problem, Algorithmica 19 (1994) 390–410.
[35] H. Liu, L. Yu, Towards integrating feature selection algorithms for classification and clustering, IEEE Trans. Knowl. Data Eng. 17 (4) (2005) 491–502.
[36] A. Asuncion, D. Newman, UCI machine learning repository, 2007.
[37] S. Thrun, L.K. Saul, B. Schölkopf (Eds.), Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada], MIT Press, 2004.
[38] X. Song, J. Zhang, Y. Han, J. Jiang, Semi-supervised feature selection via hierarchical regression for web image classification, Multimedia Syst. 22 (1) (2016) 41–49.
[39] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[40] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (2008) 2677–2694.