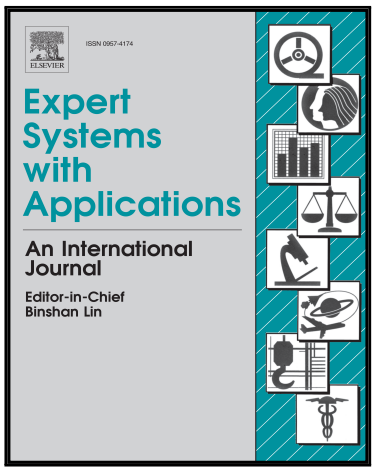


Attribute clustering using rough set theory for feature selection in fault severity classification of rotating machinery

Fannia Pacheco, Mariela Cerrada, René-Vinicio Sánchez,
Diego Cabrera, Chuan Li, José Valente de Oliveira

PII: S0957-4174(16)30659-5
DOI: 10.1016/j.eswa.2016.11.024
Reference: ESWA 10995

To appear in: *Expert Systems With Applications*

Received date: 20 June 2016
Revised date: 18 November 2016
Accepted date: 19 November 2016

Please cite this article as: Fannia Pacheco, Mariela Cerrada, René-Vinicio Sánchez, Diego Cabrera, Chuan Li, José Valente de Oliveira, Attribute clustering using rough set theory for feature selection in fault severity classification of rotating machinery, *Expert Systems With Applications* (2016), doi: 10.1016/j.eswa.2016.11.024

Highlights

- A novel algorithm is proposed for unsupervised feature selection
- The algorithm efficacy is evaluated through the accuracy of several classifiers
- Adequate attributes are effectively selected for several case studies
- The proposal presents better results than other attribute clustering algorithms
- The proposal provides similar results to supervised feature selection approaches

Attribute clustering using rough set theory for feature selection in fault severity classification of rotating machinery

Fannia Pacheco[a,*], Mariela Cerrada[b,a], René-Vinicio Sánchez[a], Diego Cabrera[a], Chuan Li[c,a], José Valente de Oliveira[d,a]

[a] *Mechanical Engineering Department, Universidad Politécnica Salesiana, Ecuador*
[b] *CEMISID, Universidad de Los Andes. Venezuela*
[c] *National Research Base of Intelligent Manufacturing Service, Chongqing Technology and Business University, Chongqing-China*
[d] *CEOT, Universidad do Algrave, Faro, Portugal*

Abstract

Features extracted from real world applications increase dramatically, while machine learning methods decrease their performance given the previous scenario, and feature reduction is required. Particularly, for fault diagnosis in rotating machinery, the number of extracted features are sizable in order to collect all the available information from several monitored signals. Several approaches lead to data reduction using supervised or unsupervised strategies, where the supervised ones are the most reliable and its main disadvantage is the beforehand knowledge of the fault condition. This work proposes a new unsupervised algorithm for feature selection based on attribute clustering and rough set theory. Rough set theory is used to compute similarities between features through the relative dependency. The clustering approach combines classification based on distance with clustering based on prototype to group similar features, without requiring the number of clusters as an input. Additionally, the algorithm has an evolving property that allows the dynamic adjustment of the cluster structure during the clustering

*Corresponding author
Email addresses: fannikaro@gmail.com (Fannia Pacheco), cerradam@ula.ve (Mariela Cerrada), rsanchezl@ups.edu.ec (René-Vinicio Sánchez), dcabrera@ups.edu.ec (Diego Cabrera), chuanli@21cn.com (Chuan Li), jvo@ualg.pt (José Valente de Oliveira)

process, even when a new set of attributes feeds the algorithm. That gives to the algorithm an incremental learning property, avoiding a retraining process. These properties define the main contribution and significance of the proposed algorithm. Two fault diagnosis problems of fault severity classification in gears and bearings are studied to test the algorithm. Classification results show that the proposed algorithm is able to select adequate features as accurate as other feature selection and reduction approaches.

Keywords: Attribute clustering, Rough set, Feature selection, Fault severity classification, Rotating machinery

1. Introduction

Machine learning tries to extract knowledge from a set of features (attributes), that represents the measurable properties of a process. The learning process is performed by training different models, i.e., classification, prediction or clustering; and their usage depend on the problem characteristics. One of the main difficulties that machine learning models might face, when a large number of features are given, is that they might be exposed to increasing computational burden, decreasing accuracy, increasing overfitting, among others. These problems are typically related to the curse of dimensionality. In particular, for machine learning-based fault diagnosis in mechanical rotating machinery, a sizable amount of features could be extracted from different monitoring signals which are processed to obtain condition parameters in time, frequency and time-frequency domain. Conversely, feature selection is widely applied in this field to select the most relevant features and improve the accuracy of machine learning diagnostic models (Cabrera et al., 2015; Cerrada et al., 2015a, 2016; Pacheco et al., 2016b).

Feature selection is considered as NP-complete problem, and several approximation algorithms have been proposed to find the near best feature subset in a reasonable time (Gheyas & Smith, 2010). Comprehensive surveys of feature selection algorithms are presented in (Liu & Yu, 2005; Liu & Motoda, 2008; Chandrashekar & Sahin, 2014), where the algorithms are grouped into Filter, Wrapper and Embedded approaches which in turn can be developed by supervised and unsupervised strategies. In the supervised strategy, the feature vector has a class label to guide the search process of relevant information. On the other hand, in the unsupervised strategy, the feature vector has no labels and it is used to determine relationships between

features to discover their relevance. Filter approaches give a score to each feature using a metric, in order to execute a ranking process, where the most relevant features are those ones that meet an accepted threshold. Correlation analysis is a simple filter approach where relationships between features are detected through a correlation coefficient. Other techniques used by the filter approach are Gini Index (Shang et al., 2007), Information Gain (Raileanu & Stoffel, 2004), Laplacian Score and Sparsity Scores (He et al., 2005; Liu & Zhang, 2016), among others. Unsupervised strategies are used to support the filter approach as recently proposed by (Wang et al., 2015; Zhou et al., 2016; Wang et al., 2016; Liu et al., 2016), in order to find the best basis features, or to select the features through structured sparsity regularization models in order to preserve the cluster structure of the instances composing the dataset (Chen, 2015; Shi et al., 2015; Maldonado et al., 2015; Wu et al., 2016). Wrapper approaches can use supervised learning to search relationships between the features. An objective function is defined in order to determine the effect of different feature sets on the model accuracy. Genetic algorithms and sequential search strategies are widely used as wrapper methods (Siedlecki & Sklansky, 1989; Klepaczko & Materka, 2010), even in the context of solving attribute clustering in a supervised manner (Hong et al., 2015), but they require a high computational cost for high-dimensional datasets. Finally, embedded methods are mostly composed of machine learning algorithms that include the feature selection process as a part of the model design such as regularized regression models and decision trees based models (Ng, 2004; Genuer et al., 2010). Recently, an intelligent approach using DE and Artificial Bee Colony has been proposed in (Zorarpac & Özel, 2016) for feature selection in an embedded method. The selection process is basically supervised.

In this work, we are interested in the filter approach using unsupervised techniques for attribute clustering. Attribute clustering is a trend dedicated to gather together similar features in order to split the characteristic space into clusters; only one feature is defined as the most representative for each cluster, and the remaining ones are discarded. Several works integrate clustering algorithms with different information measures, regarding the interdependence between attributes to perform feature selection in an unsupervised manner, most of them requiring the number of clusters as input parameter to be adjusted until reaching good performance to the particular problem to be solved (Jornsten & Yu, 2003; Au et al., 2005; Niu et al., 2008; Zhao et al., 2013; Song et al., 2013; Yuwono et al., 2015; Zhou & Chan, 2015).

On the other hand, some approaches have used Rough Set (RS) theory to

find relationships between features, based on the indiscernibility, lower and upper approximations and dependency degree concepts (Świniarski, 2001); these particular characteristics make RS suitable to propose similarity metrics for feature selection. Feature selection is mainly achieved by determining the lower approximation of an information system characterized by features and samples. RS is most commonly found in fuzzy approaches for feature selection. The main aim is maintained by using fuzzy-rough sets, which allows finding approximate reducts in the information system (Parthaláin & Jensen, 2013; Gu et al., 2015; Jensen & Parthaláin, 2015). Some other works combine RS theory with different feature selection approaches such as Neural Network (NN) with Fuzzy-RS (Ganivada et al., 2013), Expectation-Maximization (Fazayeli et al., 2008), and mutual information (Qian & Shu, 2015; Foithong et al., 2012), among others.

The use of the dependency degree as similarity measure in attribute clustering algorithms is reported in recent works. A quantitative measure to compute the similarity between two random variables, called fuzzy-rough supervised similarity, is defined by (Maji, 2011). The measure is based on the definition of the fuzzy positive region of fuzzy-rough sets, and incorporates the information of sample categories or class labels while measuring the similarity between attributes, working in a supervised manner. The clustering algorithm basically starts creating clusters with the attributes whose supervised similarity measure are inside of a radio. Next, the representative cluster is refined incrementally. The current representative cluster is merged and averaged with one single attribute such the augmented cluster representative increases the relevance value. The merging process is repeated until the relevance value can not be improved. The approach in (Hong et al., 2014) proposes a dissimilarity measure based on the dependency degree between a pair of attributes; the algorithm is called Most Neighbour First (MNF). MNF uses K-means to perform attribute clustering based on the proposed dissimilarity measure. The process starts randomly selecting k representative attributes as cluster medoids, then the dissimilarity measure is computed between the non-representative attributes. The non-representative attributes are allocated to their nearest medoids, and in the end of the process the medoid is updated with the attribute with more neighbours in its surroundings. One of the big MNF deficiencies is that the convergence of the algorithm is not assured, in consequence it has to be executed several times to find tendencies or patterns in the results. In (Cerrada et al., 2015b), a feature selection technique inspired by the MNF algorithm is presented for fault diagnosis in spur

gears. The search procedure is kept with slight modifications, and an hierarchical procedure speeds up the time-execution. The information system is divided into disjoint subsets, which in turn are evaluated by NMF. A subset of attributes is obtained for each level, and the result feedback the next level in the hierarchy. The resulting attributes are those ones at the hierarchical level where the best accuracy is obtained with a fault classifier. Maximum dependency attributes (MDA) is a feature selection technique based on RS presented by (Herawan et al., 2010). Strictly, MDA is not an attribute clustering approach but it proposes the concept of 'clustering attribute'. The first step is to determinate the equivalence classes using the indiscernibility relation on each attribute. Secondly, the dependency degree is computed for each attribute against the rest ones. The maximum dependency degree is selected for each attribute, and the selected attribute is the one with the maximum value regarding their highest dependency degree, and this attribute is called a 'clustering attribute'. This method has to compute the dependency degree of all the attributes, which in turn makes it time-consuming. The work in (Qin et al., 2012) improves the time execution of MDA, by introducing a soft model approach to select the clustering attribute. The dependency degree is computed over a soft set of an information system instead of the complete set. A similar algorithm called Maximum Attribute Relative using the concept of 'clustering attribute' is presented in (Mamat et al., 2013).

Attribute clustering is also considered NP-hard procedure, as the majority of feature selection algorithms, due to the similarity degree must be computed for all the pairs of attributes in order to arrange the clusters. Moreover, the number of clusters must be fixed for most of the approaches that use clustering for feature selection (Maji, 2011; Hong et al., 2014; Cerrada et al., 2015b). In this sense, advances in this field to decrease the computational burden of the clustering algorithms, and the dynamic adjustment of the number of cluster, are appreciated contributions.

The authors already introduced a short version of this work in (Pacheco et al., 2016a), where several tests with classic data sets are shown. In the current paper a detailed formalization of the algorithm is presented, besides more interesting results that depict the working and performance of the proposal. Generally speaking, we propose an attribute clustering algorithm called Attribute Clustering Algorithm using Rough Set theory (ACARS), which aims to overcome the main disadvantage of the clustering algorithms ; that is, setting the number number of clusters to be created, as an input parameter. The relative dependency degree is used to find similarities between attributes

without considering the labels associated with the classes. Therefore, this is a filter approach that uses an unsupervised strategy. Our proposal properly combines clustering based on prototype and classification based on distance to take advantage of the main ideas of the classic K-means and K-NN algorithms to create new clusters, split or merging the existing ones. In this sense, our approach have evolving characteristics to allow the adjustment of the number of clusters during the algorithm execution. On the other hand, the whole set of n attributes is not treated by ACARS in each iteration as the classic K-means works, but the attributes are incoming one by one per iteration, and only the current subset of attributes is analysed. Then, only n iterations are needed to create the attribute clusters. This characteristic gives to our algorithm the property of incremental learning. Based on the mentioned properties, our main contributions by proposing ACARS are summarized as follows:

1. It is an unsupervised alternative for feature selection.
2. It combines the ideas of well-known and fast algorithms such as K-means and K-NN, without knowing the number of clusters as input.
3. The only input to the algorithm is the dissimilarity matrix based on the relative dependency between attributes.
4. The algorithm allows creating an evolving structure of the attribute clusters, as a consequence, the number of adequate clusters are adjusted during the clustering process.
5. It is an incremental learning algorithm analysing only one feature in every iteration, and new features can feed the current cluster structure without a retraining process.
6. The computational burden, regarding other similar works on attribute clustering, is quite improved.

Moreover, the proposal can be extended to solve general clustering problems by correctly defining the similarity metric between objects. Thus can represent a significant impact for upcoming new algorithms in expert and intelligent systems.

Two real world problems for fault diagnosis in rotatory machinery are evaluated, to test the ACARS effectiveness to select adequate features for

solving the classification problem of fault severity diagnosis. The accuracy of several classification models using the selected features is used as the performance metric of the proposed algorithm. Comparisons with classic feature selection and reduction techniques and other attribute clustering proposals, are also provided in order to show the performance of our algorithm.

The rest of the paper is organized as follows. Section 2 presents related works to attribute clustering. Section 3 reviews the theory used by our proposal. Section 4 introduces the proposed attribute clustering algorithm. Section 5 presents the cases of study for fault severity classification in gear and bearing devices. Section 6 shows the results, and finally Section 7 concludes the paper.

2. Related works

Intelligent systems and machine learning techniques have recently contributed to proposing unsupervised feature selection for attribute ranking and attribute clustering. In (Tabakhi et al., 2014; Dadaneh et al., 2016), Ant Colony Optimization (ACO) has been used to score each feature of the original set according to the times that the ants have visited the nodes (attributes) of the graph; the heuristic desirability of each edge is computed according to some similarity metric (Cosine similarity or Correlation, respectively). Finally, a certain subset of attributes is selected given a threshold. The main disadvantage of this approach is the random nature of the process, it must be run several times before deciding about the features ranking. In (Bhadra & Bandyopadhyay, 2015) the authors propose an improved version of the Differential Evolution (DE) technique to decrease the computational effort registered with other search techniques, such as ACO. A composite objective function is defined based on the average standard deviation of the selected feature subset, the average dissimilarity of the selected features, and the average similarity of non-selected features with respect to their first nearest neighbour in the selected feature set. Similarity is computed through the mutual information score. At the end of the process, two disjoint subsets of features are built: selected and not selected features. A certain number of features must be defined to set the size of the solution vectors from the population.

On the other hand, attribute clustering using unsupervised machine learning techniques has been used early in the bioinformatics field, where the characteristic of gene expression data often compromises the performance

of conventional clustering algorithms. In (Jornsten & Yu, 2003), the gene attribute clustering is based on a Gaussian mixture model; the algorithm called K-Modes Attribute Clustering (K-MAC) is used in (Au et al., 2005) to group interdependent attributes into clusters, by optimizing a criterion function derived from a multiple interdependence measure between attributes. The attribute with the highest value of interdependence regarding the other attributes in the cluster is selected as the significant one. The work in (Zhou & Chan, 2015) uses an optimized K-MAC algorithm, the distance between clusters are measured through the Maximal Information Coefficient (MIC). The attribute with the highest value of Multiple MIC (MMIC), regarding the rest of the attributes in the cluster, is selected as the significant attribute. The mentioned works are inspired in the classical K-means, therefore the main disadvantage is that the number of cluster must be defined as an input parameter to train the algorithm.

There are another approaches for attribute clustering in which the number of clusters is not required as an input. In (Niu et al., 2008), the selected features are obtained after applying an overlapping clustering algorithm over a relation matrix. This matrix allows measuring the correlation to evaluate the relevance of every two non-redundant attributes, regarding their subspace clusters along the dataset (instances). In this sense, the features selection aims at identifying the candidate attributes from all interesting subspaces. A well-known method called Affinity Propagation Clustering, with the similarity measure based on maximal information coefficient, is proposed in (Zhao et al., 2013). The clustering method considers all the features as centroid candidates, and the number of the final clusters created depends on the similarity matrix obtained by using MIC. In (Yuwono et al., 2015), Swarm Ensemble Rapid Centroid Estimation based on consensus clustering is used. The approach provides a way of determining the number and membership of possible attribute clusters. The important features are automatically selected from the original measurements based on the relative entropy between the low-frequency and high-frequency features by using the KullbackLeibler(KL) divergence, to measure the relative entropy between two distributions. The feature with the highest entropy in the cluster is selected as the significant one.

Other approaches for attribute clustering are based on graph clustering, where the number of clusters is defined by the algorithm after representing the entire set of original features in a graph. The work in (Song et al., 2013) proposes a two step approach for graph based attribute clustering. First,

the non-relevant features are removed from the original set by applying the concept of T-relevance. Then, the remaining features are divided into clusters through graph-theoretic clustering; specifically minimum-spanning tree after deleting the edges with low weights regarding the T-relevance. Finally, the concept of R-features is used to select the most representative features in the clusters; which are strongly related to target classes by using the metric of symmetric uncertainty based on entropy. In (Moradi & Rostami, 2015), the proposed method for graph based attribute clustering works in three steps. In the first step, the original feature set is represented as a weighted graph. Next, the features are divided into several clusters using a community detection algorithm. This algorithm detects communities in the graph by maximizing a specific modularity function. Finally, a novel iterative search strategy based on node centrality is developed to select the final subset of features. A similar work using community detection in graph is presented in (Boobalan et al., 2016). The approach allows grouping the nodes based on the neighbourhood of the core node and Attribute Similarity is achieved using Similarity Score taking into account the high intracluster and the low intercluster similarities.

The related works show that the main disadvantages of the recent attribute clustering algorithms are: (i) The number of clusters must be an input to the algorithm, (ii) The entire set of original features is analysed in each iteration, and a retraining process is needed when new attributes are considered, (iii) Some approaches require search strategies to rank the best features in the clusters, and (iv) The execution time can involve significant computational burden. In this sense, approaches overcoming these aspects are well received as contributions to solve this important problem of attribute clustering in data mining applications. Our proposed ACARS aims to improve the former disadvantages.

3. Background

ACARS is based on a set of statements that properly combines the principles of two algorithms for classification and clustering such as K-NN and K-means. RS is used to find relationships between features through their relative dependency degrees. In the following, RS based on dependency degree, and the classic algorithms used by our proposal are briefly presented.

3.1. Relative Dependency

A rough set (RS) consists of a lower and upper approximation of an information system I. A lower approximation is a subset containing all the elements that certainly belong to I, and an upper approximation contains those elements that might belong to I. For attribute clustering, the lower approximation (*reducts*) of an information system can be viewed as a reduced space defined by attributes and samples. The RS properties, such as core attributes, projection operators and dependency degree, are used to find adequate *approximate reducts* of an information system.

An information system is defined as $I = (U, A)$, where $U = \{x_1, \cdots, x_n\}$ is a finite set of objects, and A is a finite set of attributes. In this sense, it is stated that $a : U \rightarrow f_a(x)$ for every $a \in A$, $f_a(x_i)$ is the value that attribute a takes given the object x_i.

Definition: The indiscernibility relation $IND(B)$ for any subset $B \subset A$ is given by the set in expression 1,

$$IND(B) = \{(x, y) \in U \times U | \forall a \in B, f_a(x) = f_a(y)\} \tag{1}$$

Indiscernibility relation defines a relation between two or more objects, and additionally the following properties are derived:

- If $IND(A) = IND(B)$, then B is a *reduct* of A
- B is a *minimal reduct* if the following condition is satisfied,

$$IND(B) = IND(A) \quad \text{and} \quad IND(B') \neq IND(A) \quad \forall B' \subseteq B$$

A metric to find *approximate reducts*, called dependency degree, is introduced using both discernibility and projection relations (Han et al., 2004). The relative dependency degree is a similarity metric between two attributes,

Definition: Given a pair of attributes (A_i, A_j), the relative dependency degree $Dep(A_i, A_j)$ is given by Eq. 2,

$$Dep(A_i, A_j) = \frac{\left|\prod_{A_i}(U)\right|}{\left|\prod_{(A_i, A_j)}(U)\right|} \tag{2}$$

where $\prod_{A_i}(U)$ is the projection of U on A_i. This projection is computed: (1) by defining the set $C = A - B$, and (2) by merging the remaining objects that are indiscernible.

By definition, a similarity metric has to be symmetric, i.e, $d(A_i, A_j) = d(A_j, A_i)$; however, this is not always true for relative dependency. The average dependency $d(A_i, A_j)$ between A_i and A_j is computed as follows to compensate this issue,

$$d(A_i, A_j) = \frac{1}{avg\{Dep(A_i, A_j), Dep(A_j, A_i)\}} \tag{3}$$

Eq. 3 will be used by our proposal as a dissimilarity measure to perform attribute clustering.

3.2. Classification based on distance

K-Nearest Neighbors (K-NN) is one of the simplest, and most effective algorithms based on distance, where a small distance between samples may mean high similarity. The distance value between two samples can be measured by different metrics, e.g, Euclidean, cosine, Pearson, Mahalanobis or city-block, among others. The samples are arranged according with their similarities. After that, the samples with high similarities are grouped or classified using a supervised or unsupervised learning strategy (Dunham, 2003).
Given a dataset X, and a set $C = \{C_k\}$ of classes or clusters with $k = 1, ..., N_c$; for each sample $x_i \in X$:

1. Compute the distance d between x_i and the rest of samples in X, and obtain the set X_d^i of attributes ordered from the minimum to maximum distance, by using the function *rank*. That is,

$$X_d^i = rank_{d_j}\left(\{x_j | d_{j=1,\cdots,|X|}(x_i, x_j)\}\right), \quad i \neq j \tag{4}$$

   where $rank_{d_j}$ denotes the rank function regarding the distance d_j.

2. Select the K first elements from X_d^i, which in turn compose the set X_{nn}^i with the K nearest neighbours of the sample x_i.

3. Identify the class or cluster where each $x_j \in X_{nn}^i$ belongs, that is create the sets $X_{nn_{C_k}}^i$:

$$X_{nn_{C_k}}^i = \{x_j \in X_{nn}^i | x_j \in C_k\} \tag{5}$$

4. Assign the sample x_i to the class or cluster such that x_i have the higher number of neighbours, that is,

$$C(x_i) = C_m \tag{6}$$

   where $C(x_i)$ denotes the class or cluster of x_i, and C_m is the class for which $|X_{nn_{C_k}}^i|$ is maximum.

K-NN by itself is suitable for new pattern detection, based on the premise that samples related with known patterns have always close neighbours; on the contrary, samples related with unknown patterns or outliers are located far from their neighbours. One of the K-NN's deficiency is its slowness in the training process when the amount of samples is large, because the searching procedure must be done over the whole set of available samples.

3.3. Clustering based on prototypes

A cluster is represented by a prototype (or centroid) usually a measure of centrality of the samples belonging to that cluster. K-means is a classic prototype based algorithm, which consecutively divides the dataset until finds K clusters that minimize an objective function. The process starts initializing K centroids randomly, then the data partition is reached following the next steps.
Let $C = \{C_1, \cdots, C_K\}$ be a set of prototypes (or centroids) each one representing a cluster P_j, $j = 1, ..., K$, and a set of samples X. For each $x_i \in X$:

1. Assign x_i to the cluster P_j, i.e, $P(x_i) = P_j$; where P_j is the cluster whose centroid fulfills the expression given in Eq. 7 ,

$$C_j = argmin_{j=1,\cdots,K}\{d(x_i, C_j)\} \tag{7}$$

2. Update the centroid C_j according to some procedure, e.g., computing the mean value of data in the cluster.

3. If the stopping criteria is met, end the process; otherwise, repeat the step 1.

The stopping criteria is generally related with an optimization problem, where the aim is to minimize an objective function J. In K-means the most common objective function is defined by the distance between the samples and the centroid where they were assigned.

Prototype based clustering is characterized for being faster than K-NN; however, its performance is lower in terms of accuracy (Hastie et al., 2009; Chapelle et al., 2006). Classically, both K-NN and K-means work over beforehand knowledge of all the dataset for the training process. Furthermore, they do not offer the possibility to create new classes or clusters.

4. Clustering principles of ACARS

Our proposal combines K-means and K-NN in order to overcome their deficiencies and take advantages of their benefits to solve the attribute clustering problem.

Particularly, the proposed algorithm runs over samples that are incoming continuously, since the proposed clustering mechanism is able to update or reconfigure clusters any time that a new sample x_i is evaluated. This is achieved through a search process that is performed only over a subset of the available samples in the clusters, that makes our approach be faster than using K-means or K-NN, separately. The on-line reconfiguration allows merging or splitting prototypes, as well as creating new prototypes; which in turn gives an evolving characteristic to our approach.

4.1. Fundamentals

This section shows the theoretical guidelines that support our proposal. First at all, some definitions are presented.

Definition 1. *A cluster P_j is described by the tuple given by the expression in 8,*

$$P_j = (C_j, t_j, \overline{X}_j, name) \tag{8}$$

where C_j is the centroid, t_j is the neighbour threshold, $\overline{X}_j$ is the set of samples x_i belonging to P_j and name is the cluster identifier.

In attribute clustering, the samples x_i are represented by attributes A_i and the centroid C_j which is the most representative attribute in the cluster and must be updated every time that a cluster is modified. In this work, we alternatively denote that $x_i \in P_j$ if $x_i \in \overline{X}_j$. The threshold t_j is an adjustable parameter that helps to take decisions to modify the structure of a cluster. The centroid and the threshold are defined as follows.

Definition 2. *The centroid C_j of the cluster P_j is the attribute $A_C \in \overline{X}_j$ obtained by Eq. 9,*

$$A_C = argmin_{A_i \in P_j |} \{d_{\mu i}(A_i, \overline{A}_m)\} \tag{9}$$

where $d_{\mu i}(A_i, \overline{A}_m)$ is the average dissimilarity of each attribute $A_i \in P_j$ against all the attributes in the set $\overline{A}_m = \overline{X}_j - A_i$, as defined in Eq. 10,

$$d_{\mu i}(A_i, \overline{A}_m) = avg_{A_k \in \overline{A}_m} \{d(A_i, A_k)\} \tag{10}$$

Remark: This definition aims to select the attribute that is more similar to rest of attributes in the cluster, based on a distance criterion. Conversely, A_C has the nearest attributes around itself.

Definition 3. *The threshold t_j of the cluster P_j is the value calculated by Eq. 11,*

$$t_j = avg_{A_i, A_j \in P_j}\{d(A_i, A_j)\}, \quad j \neq i \tag{11}$$

where $avg_{A_i, A_j \in P_j}$ is the average dissimilarity between all the possible combinations of attribute pairs (A_i, A_j) such as $A_i, A_j \in P_j$.

Definition 4. *The initial threshold t_{ini} for a new unitary cluster is defined by Eq. 12,*

$$t_{ini} = avg_{A_i, A_j \in A}\{d(A_i, A_j)\}, \quad \forall i, j, \quad j \neq i \tag{12}$$

where A is the set of all existing attributes to be clustered.

Remark: Note that t_{ini} is a reference value to start the attribute clustering algorithm. It is determined at the beginning of the process and assigned every time that a new cluster is created with a single attribute.

As mentioned, the main problem to be solved is the search for the best cluster where an attribute is assigned (cluster construction). The second problem is to perform cluster reconfigurations (merging or split), once an attribute is assigned to a cluster. In order to reach these goals, ACARS is based on two criteria that provide information for decision-making: the distance between an incoming attribute and the available prototypes, and the distance between the attribute and its neighbour attributes.

In the following, we state the main definitions and statements that support the three main functionalities of our algorithm: cluster construction, cluster merging and cluster split. The statements consider the similarity measure or distance between attributes (samples) given by Eq. 3.

4.1.1. Cluster construction

K-means and 1-NN are combined in order to find similarities between incoming samples and prototypes. This similarity is computed through two procedures in the following order.

1. The centroid similarity proposed by K-means is used to place the provisional cluster where an incoming sample (attribute) could be assigned. This allows identifying the region where its nearest neighbour attribute and neighbour cluster may be.

2. 1-NN is used to place the nearest sample to the incoming sample, but just considering the samples in its current provisional and neighbour clusters instead of checking the complete dataset, as proposed by the classic 1-NN algorithm.

At the end of the search, the incoming sample might be finally assigned to the cluster under evaluation (provisional or neighbour), or alternatively, a new prototype will be created. Several definitions and statements are given by our proposal to achieve the goals.

Statement 1. *A sample x_i provisionally belongs to an existing cluster P_r, i.e. $P(x_i) = P_r$, if the centroid C_r of P_r meets Eq. 7.*

Remark. Note that this statement aims to identify the nearest prototype such as stated in the K-means algorithm. K is the number of existing prototypes, and in our particular case is not previously defined as the K-means approach.

Definition 5. *The nearest neighbour of the sample x_i into the cluster P_j is the sample $x^i_{nn_{P_j}} \in P_j$ given by Eq. 13,*

$$x^i_{nn_{P_j}} = argmin_{x_j \in P_j}\{d(x_i, x_j)\} \tag{13}$$

Remark. Note that this definition states the 1-NN sample into the specified cluster P_j, and $x^i_{nn_{P_j}}$ is the single element of the set $X^i_{nn_{P_j}}$ defined by Eq. 5 using the corresponding cluster P_j.

Statement 2. *A new sample x_i is finally assigned to the provisional cluster P_r, if the distance between x_i and its nearest neighbour $x^i_{nn_{P_r}}$ accomplishes Eq. 14:*

$$d(x_i, x^i_{nn_{P_r}}) < t_r \tag{14}$$

Remark. Note that the nearest neighbour in this statement is established by Definition 5 using the corresponding cluster P_r.

Definition 6. *The resulting cluster after the assignment of x_i to P_r is given by the tuple in 15,*

$$P_r = \left(C_{new}, t_{new}, \overline{X}_r \cup \{x_i\}, name\right) \tag{15}$$

where C_{new} and t_{new} are properly updated, according to the definitions 2 and 3, respectively.

Definition 7. *The neighbour cluster P_v of the sample x_i, assigned to a cluster P_r, is represented by the centroid C_v given by the expression in 16,*

$$C_v = argmin_{k=1,\cdots,K}\{d(x_i, C_k)\}, \quad C_k \neq C_r \tag{16}$$

where K is the number of existing clusters, and C_r is the centroid of the cluster P_r.

Remark. Note that P_r is the provisional cluster defined in Statement 1.

Definition 7 is inspired by the hierarchical methods to construct clusters, specifically the agglomerative approach that is based on successive mergers regarding the proximity of a cluster pair. This proximity is measured through different ways, i.e., minimum distance between the centroids, minimum or maximum distance between two samples in the pair, and mean or average distance of all the samples belonging to a cluster against all the samples belonging to another one, among others (Tan et al., 2005). The hierarchical procedures can derive different clusters depending on the proximity metric used; the proximity between all the samples plus the input are updated in each iteration. Our proposal differs from that approach, and updates just the prototypes associated to the new input.

Statement 3. *When a sample x_i is not finally assigned to P_r, it may be finally assigned to its neighbour cluster P_v, by considering a new provisional cluster $P_r := P_v$.*

Remark. Statement 3 means that Statement 2 is not accomplished. In this case, the algorithm must apply Definition 5 and evaluate Statement 3 with $P_r := P_v$.

Statement 4. *If a new sample x_i is not similar to its nearest and neighbour cluster, then a new cluster P_{new} is created. That is,*

$$P_{new} = \left(C_{new} = x_i, t_{new}, \overline{X}_{new} = x_i, name = Cluster_{id}\right) \tag{17}$$

where $t_{new} = t_{ini}$, according to Definition 4.

Remark. Statement 4 implies that the statements 2 and 3 are not met.

Fig. 1 shows graphically three examples of the sample assignment in the cluster construction. We assume that the samples can be represented on a two-dimensional plane, in order to illustrate the idea: (1) example 1 shows when a new cluster P_2 has been created with x_i, (2) in example 2, the input sample belongs to the provisional prototype P_1 and is assigned to it, and (3) example 3 occurs when the input sample does not belong to its provisional prototype P_2, but it does to its neighbour P_1.

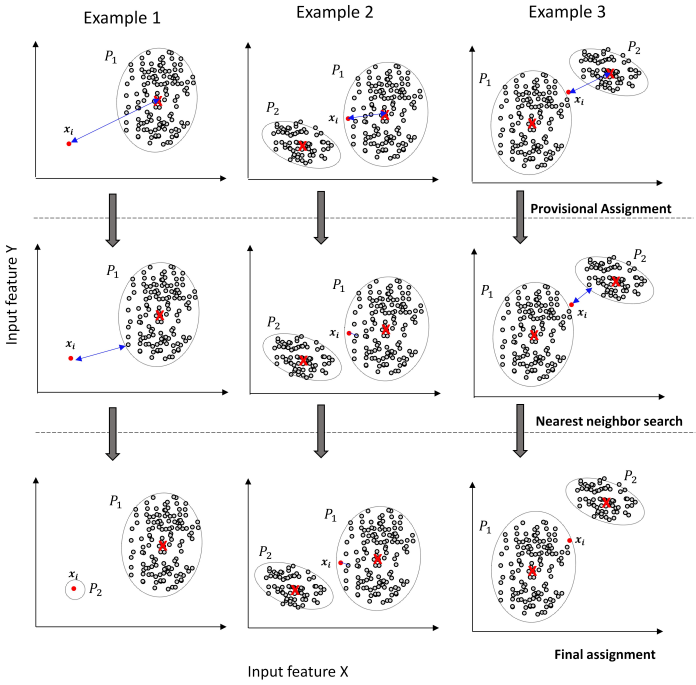


Figure 1: Graphical examples of the guidelines to assign x_i to a cluster

4.1.2. Cluster merging

Intuitively, two clusters could be merged if there is a high degree of overlapping. K-NN and K-means do not have this functionality. In our approach, the merging process is considered due to the characteristics of the final sample assignment defined by the cluster construction in the previous section. It is designed to change the cluster configuration given the effect caused by the new incoming sample. Particularly, for the attribute clustering problem, we consider that two neighbour clusters should be merged when a high density exists between their borders (Lughofer & Sayed-Mouchaweh, 2015). Let P_r and P_v be two neighbour clusters, the following statements define the overlapped area and the merging conditions.

Definition 8. *The 'degree of expansion' de_i allowed to a cluster P_i is defined by the proportion of the threshold t_i in the tuple 8, i.e., $de_i = \alpha * t_i$, where α is an adjustable parameter.*

Remark: The standard deviation of the dependency degree between the attributes might work as a reference value to define α and adjust de_i.

Definition 9. *The number of samples, in the overlapped area between two neighbour clusters P_r and P_v, are defined by N_I in 18,*

$$N_I = \{x_i \in \overline{X}_r \mid d(x_i, x^i_{nn_{P_v}}) < t_v + de_v\} \tag{18}$$

with $x^i_{nn_{P_v}}$ and t_v defined by Statement 5 and the tuple in 8, respectively.

Statement 5. *Two neighbour clusters P_r and P_v are merged, if the expression in 19 is met,*

$$|N_I| > \beta \left(|\overline{X}_r| + |\overline{X}_v|\right) \tag{19}$$

where $\overline{X}_r$ and $\overline{X}_v$ are defined in tuple 8, N_I is the set given by 18, and β is an adjustable parameter.

Remark. Note that with the expression in 19, the merging process is executed if the number of samples in the overlapped area N_I is higher than a proportion, defined by β, of the total samples in P_r and P_v.

Definition 10. *The resulting cluster P_{new} from the merging process is given by the tuple in 20,*

$$P_{new} = P_r \cup P_v = \left(C_{new}, t_{new}, \overline{X}_r \cup \overline{X}_v, name\right) \tag{20}$$

where C_{new} and t_{new} must be properly adjusted according to the definitions 2 and 3, respectively.

As an example, in Fig. 2(a) a new sample x_i was assigned to the cluster P_1; and it causes that the clusters P_1 and P_2 are very close to each other. Verifying the merging conditions, P_1 and P_2 could be merged and the result is showed in Fig. 2(b). We assume that the samples can be represented on a two-dimensional plane, in order to illustrate the idea.

4.1.3. Cluster split

Our attribute clustering algorithm can detect if samples in a cluster can be split and assigned to another one. Let P_m be a cluster where x_i was assigned with t_m its respective neighbour threshold, and P_v its neighbour cluster according to Definition 7 with neighbour threshold t_v. The statements to perform the split process are given as follows.

Statement 6. *The incoming sample x_i might generate new high similarities to others samples $x_j \in P_v$ if $t_m < t_v$.*

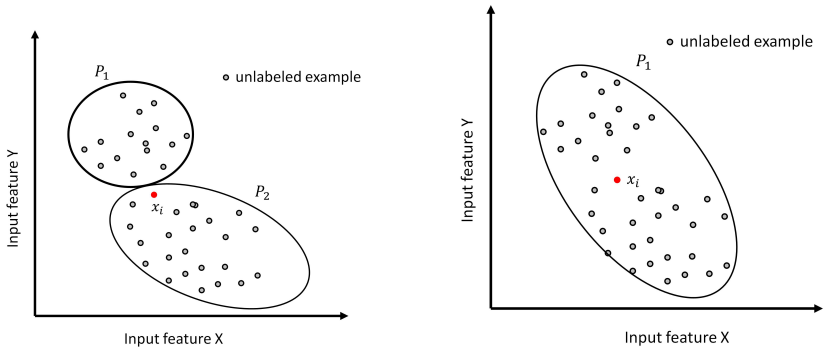


Figure 2: (a) New sample x_i assigned to the cluster P_1 (b) Merge of P_1 and P_2 after verifying Statement 5

Remark. The split process is considered just when the neighbour threshold of P_m is lower than the threshold of P_v, to assure that the samples separated from P_v are closer to its neighbour samples in P_m.

Statement 7. *The samples $x_j \in P_v$ are assigned to the cluster P_m, if $x_j \in X_{P_m}$ as defined in the set 21,*

$$X_{P_m} = \{x_j | d(x_i, x_j) < t_m\}, \quad x_j \in P_v \tag{21}$$

where $x_i \in P_m$.

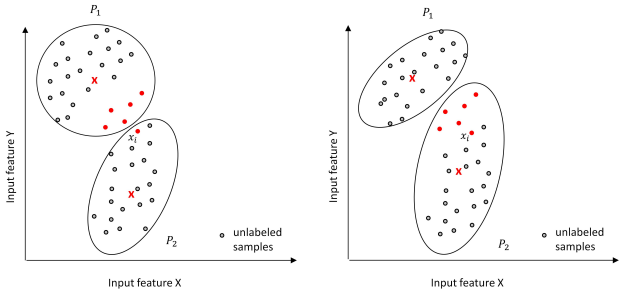
Definition 11. *The resulting clusters $P_{v_{new}}$ and $P_{m_{new}}$ from the split process are defined by the expressions 22 and 23, respectively,*

$$P_{v_{new}} = \left(C_{v_{new}}, t_{v_{new}}, \overline{X}_v - X_{P_m}, name\right) \tag{22}$$

$$P_{m_{new}} = \left(C_{m_{new}}, t_{m_{new}}, \overline{X}_m \cup X_{P_m}, name\right) \tag{23}$$

where $C_{v_{new}}$, $C_{m_{new}}$, $t_{v_{new}}$ and $t_{m_{new}}$ are updated properly according to Definition 2 and Definition 3.

Fig. 3(a) illustrates the split procedure. We know that the clusters P_1 and P_2 have their own thresholds t_1 and t_2. In consequence, if the t_2 is lower than t_1 (split condition in Statement 6), the instance x_i is separated from P_1 and also its nearest neighbours, as proposed in Statement 21 with $t_m = t_2$. The result is given by Fig. 3(b).



(a) (b)

Figure 3: (a) Neighbour to x_i that met t_2 (b) Separation of x_i and its neighbours from P_1

4.2. ACARS workflow

The clustering fundamentals in section 4.1 have been properly assembled in the proposed ACARS workflow, where an attribute A_i is the sample x_i . The aim is to perform attribute clustering using the attribute distance in Eq. 3 as similarity measure based on dependency degrees.

Let $A = \{A_1, \cdots, A_m\}$ be the set of attributes A_i, and A_d a set of possible pairs $((A_i, A_j), d_{ij})$ where d_{ij} denotes the dissimilarity based on the dependency degree between A_i and A_j. On the other hand, $P = \{P_j\}$ is the set of existing attribute clusters, with P_j defined in 1, and $P = \varnothing$ means that there are not available clusters.

The attribute clustering procedure is described in the pseudo-code in Table 1 and Fig. 4 depicts the main procedures detailed in Table 1.

As shown in Table 1 and its corresponding Fig. 4, ACARS is an evolving attribute clustering algorithm which states:

1. The assignment of A_i to an existing cluster is only performed considering the most similar and neighbour clusters. Then, only a subset of current attribute clusters is used.

2. The assignment of A_i to its most similar cluster based on the centroid similarity is accomplished, only if its nearest neighbour sample in the cluster is also similar, according to a threshold. That means that A_i could be either in the border of the cluster, or in its neighbour cluster or in a new cluster. This approach arises from the combination of K-means and 1-NN main ideas.

3. The merging process is only executed if the attribute A_i is finally assigned to its most similar cluster. As previously mentioned, this is inspired by the cluster construction of agglomerative hierarchical clustering.

Table 1: Pseudo code of ACARS

Macro algorithm

Input: A, A_d and $P = \varnothing$ sets.
Procedure:
For $i = 1$ to $\mid A \mid$,

1. If $P \neq \varnothing$,

   % Cluster construction

   1.1. Assign provisionally x_i to its most similar cluster, according to Statement 1, using the information in A_d and P.

   1.2. Search the 1-NN of A_i into the cluster P_r, according to Definition 5

   1.3. If the final assignment condition in Statement 2 is met,

   % Cluster merging

   1.3.1. Update the cluster P_r, according to Definition 6

   1.3.2. Place the neighbour cluster P_v, according to Definition 7

   1.3.3. If $\exists P_v$ and the merging condition in Statement 5 is met,

   1.3.3.1. Create a new merged cluster, acording to Definition 10

   1.3.3.2. Go to 3.

   1.3.4. Else, go to 3.

   1.4. Else, based on Statement 3, place the neighbour cluster P_v according to Definition 7

   1.5. If $\exists P_v$,

   1.5.1. Search the 1-NN of A_i into the cluster P_v, according to Definition 5

   1.5.2. If the final assignment condition in Statement 2 is met,

   % Cluster split

   1.5.2.1. Update the cluster P_v, according to Definition 6

   1.5.2.2. If the split condition in Statement 6 is met,

   1.5.2.2.1. Create the set of samples in Statement 7

   1.5.2.2.2. Update the new clusters, according to Definition 11

   1.5.2.3. Else, go to 3.

   1.5.3. Else, go to 2.

   1.6. Else, go to 2.

2. Else, according to Statement 4, create a new cluster P_{new} with the attribute A_i

3. End process

End For Output: Set of clusters P generated by the algorithm

4. The split process could be executed if A_i is not assigned to its most similar cluster based on the centroid similarity, but to its neighbour cluster. This is motivated by the scenario where attributes in the border of their most similar cluster could finally belong to their neighbour cluster.

5. Regarding the statements in section 4.1.3, it is highlighted that: (i) the cluster, where A_i is finally assigned in Table 1, is its neighbour cluster where $P_m := P_v$; and, in consequence, (ii) the neighbour cluster of P_m is the one where A_i was provisionally assigned in the first place, i.e., $P_v = P_r$.

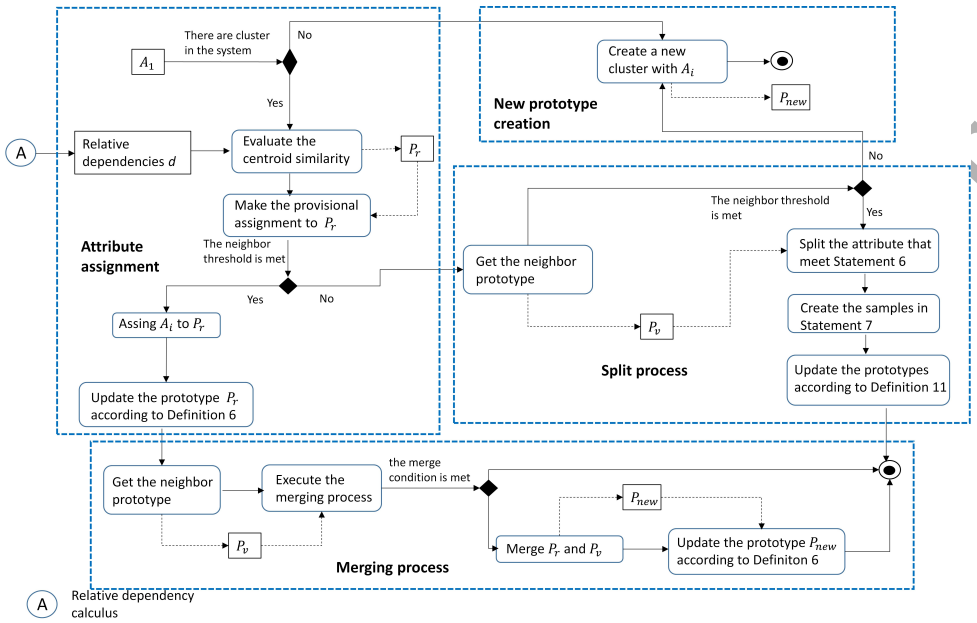


Figure 4: Activity diagram of the proposed attribute clustering algorithm

6. It is important to mention that the split and the merging processes are exclusive, this is due to the on-line cluster construction. We assume that, when an incoming sample is provisionally assigned to its nearest cluster P_r and meet the threshold condition with its nearest attribute (final assignment to P_r), the construction of the cluster P_r is adequate and only the merging process is verified. On the other hand, when the split process is activated because x_i is assigned to the neighbour cluster, only the neighbour samples to x_i in P_r are considered, as an indicator that P_r is not adequately constructed. Then, the merging is not applied.

4.3. Strengths and weaknesses of ACARS

One of the most important strengths of our proposal is that the cluster constructions are evolving, meaning that the clusters are dynamically created during the algorithm executions, under an unsupervised learning process. As previously mentioned in Section 2, most of the clustering algorithms need beforehand the number of clusters to be created, our proposal does not. The attributes are evaluated one by one simulating a streaming scenario, under an incremental learning scenario. The experiments performed in Section 6.4 shows that ACARS is faster than similar unsupervised attribute clustering approaches such as HUSFS and

MNF (Cerrada et al., 2015b; Hong et al., 2014). On the other hand, the most significant weakness of ACARS is the sensitivity to some adjustable parameters whose optimization mechanisms are left as future work. In summary, a general comparison is presented in Table 2, where the methods selected in the experimental evaluation (Section 5) are compared with the proposal.

Table 2 shows the most important aspects for the comparison between the methods: (i) the properties required for the input, (ii) the characteristics of training process, and (iii) the properties of the output. A check sign means the method has the property. Among the input field, there are some input parameters or data preprocessing to start the training process such as: (a) do not need the number of desired attributes (NotNA) or clusters, and (b) special data treatment (DT), e.g. discretization and/or normalization. In the training process field, the properties are: (a) the algorithm is unsupervised (US); otherwise, it is assumed the algorithm is supervised or semi-supervised , (b) the algorithm needs tunning of parameters (TP) that is normally performed by the user, (c) the algorithm is fast (F), regarding the time needed to execute the training process in the experimental cases presented in Section 6, (d) the algorithm have the incremental learning (IL) characteristics; that is, the cluster model do not require a retraining when new features are integrated to the problem. Finally, in the output field we are concerned about: (a) the interpretability of the output (IO); that is, the selected feature has a physical meaning regarding the problem analysed, and (b) the evolving (E) characteristic. The evolving property is considering as a very important output, due to the model does not require: (a) The number of cluster or attributes as input, (b) a retraining process. Moreover, new attributes can feed the algorithm to accommodate the current cluster model.

Table 2: Theoretical comparison between ACARS and similar approaches

Method	**Input**		**Training**				**Output**	
	NotNA	DT	US	TP	F	IL	IO	E
RF		✓		✓	✓		✓	
LDA		✓		✓	✓			
PCA		✓		✓	✓			
NMF		✓		✓				
MNF		✓	✓	✓			✓	
HUSF		✓	✓	✓			✓	
ACARS	✓	✓	✓	✓	✓	✓	✓	✓

On the other hand, from a qualitative point of view regarding the unsupervised

algorithms discussed in Section 2, K-means based algorithms need the number of clusters as an input to the algorithm, instead graph-based algorithms do not. On the other hand, in our knowledge, the incremental learning and the evolving property are not widely reported for attribute clustering, and our approach has these two characteristics.

5. Experimental evaluation

This section presents two case studies for fault severity classification in rotatory machinery. For this particular case, fault diagnosis requires several sensors to acquire the vibration signals, and thereby to extract a large number of features. Then, this problem demands an adequate feature selection to developing models for on line implementation.

Different fault severity conditions were emulated in two rotating devices, i.e., gears and bearings. The experimental set-up to collect the data and the feature extraction process are detailed for each case. Basically, in both problems, vibration signal samples, related to a machinery condition (class label), are captured and stored in a dataset arranged in pairs (sample,label). From the vibration samples, features can be extracted to conform a useful dataset to build machine learning based fault diagnosers; they aim to extract knowledge through a classification model for classifying new machinery conditions.

5.1. Gear fault severity

A gearbox device is a power transmission mechanism, they are very common in low and medium speed applications. The use of condition-based monitoring in gearboxes is crucial, and it requires high reliability, effectiveness and accuracy from fault diagnosers.

5.1.1. Experimental test bed

The test rig showed in Fig. 5 was set up in order to emulate real conditions of faults. A three phase motor generates a motion at different speeds. This motion is transmitted to a gearbox composed of an input and an output gear. Next, a pulley receives the motion from the gearbox; the pulley is a part of a magnetic brake control for load regulation. Four accelerometers are placed to capture vibration signals in different locations. Two accelerometers are in the input gear, placed in a radial and axial position according to the shaft direction; in the same way, two accelerometers were placed in the output gear. A data acquisition device DAQ sends the vibration signals to a computer. Table 3 reports the setting of the experiment. Basically, five repetitions of the experiments were performed considering five rotation speeds and three different loads. Different fault severities

in the gear tooth listed in Table 4 are emulated in the input gear, some of them are shown in Fig. 6

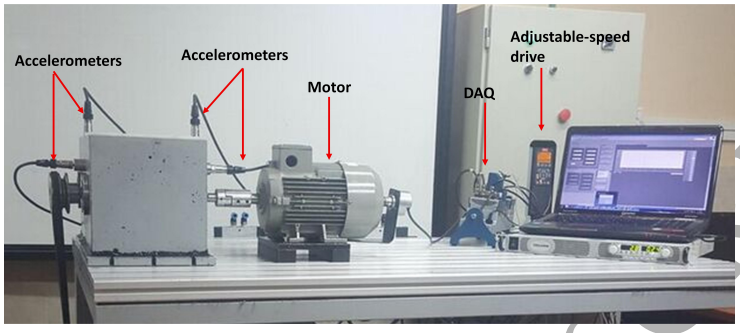


Figure 5: Experimental test bed for gear severity

Table 3: Experimental settings

Parameter	**Value**
Sampling frequency	50 kHz
Length of each sample	10 sec
Number of tests	5
Rotation Frequency (constant speed)	8 Hz, 12 Hz, 15 Hz
Range Frequency (variable speed)	8-15 Hz Sine wave 0.5Hz, 8-15 Hz Square wave 0.5Hz
Load (L)	No Load (L1), 10 V(L2), 30 V(L3)

Table 4: Fault severities in gear

Fault Label	**Description**
P1	Healthy Z1
P2	Tooth breaking level 1
P3	Tooth breaking level 2
P4	Tooth breaking level 3
P5	Tooth breaking level 4
P6	Tooth breaking level 5
P7	Tooth breaking level 6
P8	Tooth breaking level 7
P9	Tooth breaking level 8
P10	Tooth breaking level 9

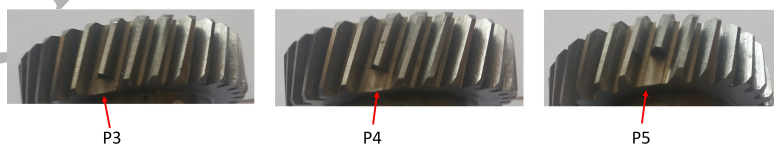


Figure 6: Gear with the faults P3, P4 and P5 induced to the test rig

Experimental data is acquired for each gearbox conditions in Table 5, resulting in a data set with 750 vibration signals for each accelerometer.

5.1.2. Feature extraction

The vibration signals are processed to extract features from the domains of time, frequency and time-frequency.

1. Time domain

   Mean μ, Standard Deviation ρ, Skewness (S), Root Mean Square (RMS), crest Factor (cf), kurtosis and kurtosis of the Teager's Energy Operator (KEO) were obtained over the whole raw vibration signal length. The mean of a vibration signal is described in Eq. 24. The standard deviation is computed using Eq. 25.

$$\mu_i = \frac{1}{N}\sum_{j=1}^{N} x_{ij} \tag{24}$$

$$\sigma_i = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(x_{ij} - \mu_i)^2} \tag{25}$$

   The skewness of a signal is given by Eq. 26, RMS is defined by Eq. 27, and cf is the maximum peak of the signal divided by RMS.

$$S_i = \frac{N\sum_{j=1}^{N}(x_{ij} - \mu_i)^3}{\rho^3} \tag{26}$$

$$RMS_i = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(x_{ij})^2} \tag{27}$$

   The kurtosis of the Teager's energy operator defined in Eq. 28,

$$KEO_i = \frac{N\sum_{i=1}^{N}(\triangle\, x_i - \triangle\, \overline{x}_i)^4}{(\sum_{i=1}^{N}(\triangle\, x_i - \triangle\, \overline{x}_i)^2)^2} \tag{28}$$

   where,

$$\triangle\, x_i = x_{(i+1)}^2 - x_i^2 \tag{29}$$

   and the kurtosis in Eq.30,

$$k\left[x_i\right] = \frac{N\sum_{j=1}^{N}(x_{ij}-\mu_i)^4}{\left[\sum_{j=1}^{N}(x_{ij}-\mu_i)^2\right]^4} \tag{30}$$

2. Frequency domain

   The vibration signals are transformed into frequency signals using Fast Fourier Transform (FFT), and the signal length in Hz is divided into 80 bands. RMS, mean, standard deviation, and kurtosis are calculated for each band, and 320 features were obtained from this analysis for each accelerometer.

3. Time-frequency domain

   Wavelet analysis in gear fault detection and diagnosis are widely used in this field (Cabrera et al., 2015; Cerrada et al., 2016). The raw vibration signals in time domain are decomposed by using the Wavelet Packet Decomposition (WPD). Six mother wavelets are considered for this analysis: Daubechies (db) 3, db 5 ,db 7, db 16, Symlet (sym) 3 and 5. WPD was performed up to four levels for db 3, db 5, db 7, sym 3 and sym 5; 2^4 coefficients are obtained for each one. For db 16, eight levels were used, and 2^8 coefficients are obtained. Finally, 336 features associated to the classic energy operator applied to the coefficients are extracted for each accelerometer.

Fig. 7 shows the complete feature extraction process performed over the vibration signals, getting as a result 663 features for each accelerometer. The process ended up with 2652 features, and 750 samples.

5.2. Bearing fault severity

A bearing is another important device that commonly fails in rotating machinery. This device is composed of an inner and outer race, inside which the rolling elements rotate. Faults in a bearing element can be placed in different components, next section describes the fault configuration for this case study.

5.2.1. Experimental test bed

Fig. 8 shows the experimental set-up for this case. Two bearings are coupled with a shaft as supporting device. A motor drives the shaft at different speeds, and flywheels can be disposed on the shaft in order to induce loads, when required. For monitoring the machinery state, vibration signals are collected through four accelerometers placed in different positions. One accelerometer is placed in a radial position, and other one in an axial orientation in the first bearing (*B1*); two more accelerometers are placed in the same positions for the second bearing

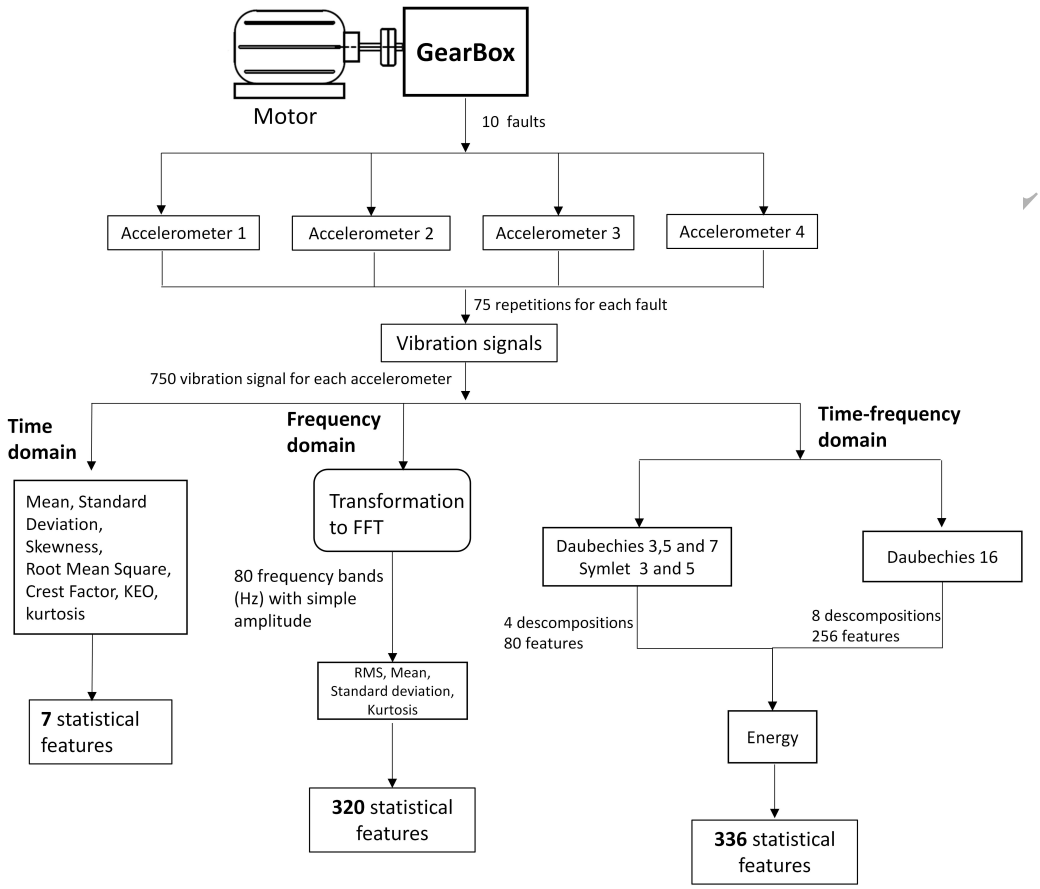


Figure 7: Feature Extraction on time, frequency and time-frequency domain

(*B2*). A data acquisition device collect the vibration signals and send them to a computer. Different fault severities are considered on the bearing components, as depicted in Table 3, and ten samples of vibration signals signal were collected in each condition. Additionally, the three loads vary in this case, i.e., with flywheel (L1), no load but with belt(L2), and 10V (L3).

Fig. 9 shows three types of faults in the inner and outer race, as well as in the rolling element; which in turn represent the severity conditions in Table 5 (*P*3, *P*7 and *P*8, respectively). It is important to mention that all the faults were induced in the bearing *B*2.

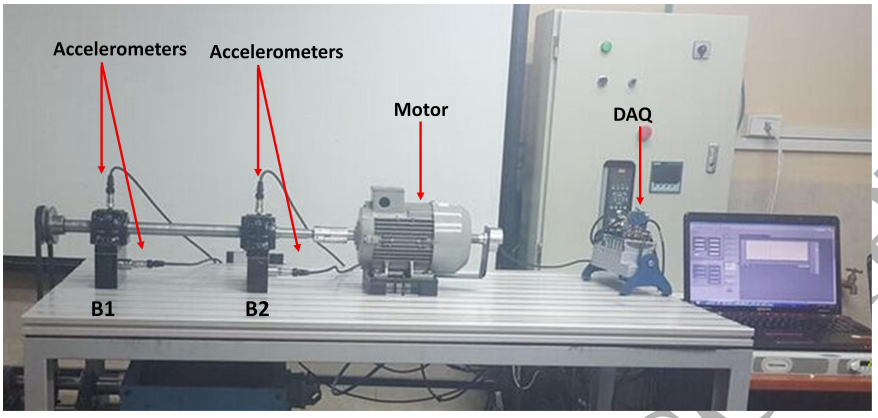


Figure 8: Experimental test bed for bearing severity

Figure 9: Bearing with the faults P3, P7 and P8 induced to the test rig

Table 5: Fault conditions in bearing

Label	Description	Diameter (mm)	Depth (mm)
P1	Healthy bering	-	-
P2	Inner race	0.5	0.3
P3	Inner race	0.9	0.3
P4	Inner race	1.3	0.3
P5	Outer race	0.5	0.3
P6	Outer race	0.9	0.3
P7	Outer race	1.3	0.3
P8	Rolling element	1	0.5
P9	Rolling element	0.9	0.5
P10	Rolling element	0.8	0.5
P11	Rolling element	1	0.3
P12	Rolling element	0.9	0.3
P13	Rolling element	0.8	0.3
P14	Rolling element	0.7	0.3
P15	Rolling element	0.6	0.3
P16	Rolling element	0.6	0.1
P17	Rolling element	0.5	0.1

5.2.2. Feature extraction

The feature extraction process for this case study was performed with the same procedure as described in section 5.1.2. The set of 663 features proposed in Fig. 7 were calculated from 120 samples of vibration signals collected for each fault condition. The resulting dataset has 2652 features and 2040 samples.

6. Experimental results

This section presents the experimental results of applying ACARS described in Table 1. First at all, a generic classification problem is evaluated by our proposal as an illustrative example, and the results of each step are depicted in order to show in detail how ACARS works. Next, the real cases about fault severity classification described in section 5 are analyzed, where a large set of features is available for tuning fault diagnosis models. The proposed algorithm is tested considering two clustering process: (i) without the merging process (denoted in the experiments as E_1), and (2) with the merging process (denoted as E_2), in order to see how the merging process improves the cluster updating. All E_2 experiments were developed with $\alpha = 0.01$ and $\beta = 0.7$ (see Definition 8 and Statement 5). The resulting reduced databases were evaluated by several classifiers such as Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), and K-nearest neighbours (KNN). The resulting datasets with the selected features are used to train several classifiers in order to assess the performance of ACARS through the classifier accuracies.

Classic feature selection and feature reduction techniques are also applied to be compared with ACARS. As a first step, correlation analysis (Corr) is executed over the original set of features, and features exhibiting a correlation higher than 95% with another one are removed. This analysis ended up with a reduced set of features that will be treated later by other techniques, including ACARS. The techniques used in this work are listed as: (i) Importance variables ranked by its information degree based on the entropy provided by the Random Forest (RF) algorithm (Genuer et al., 2010). Two thresholds were fixed, and features with information degree higher than 60% and 50% are used as inputs to the classifiers (denoted in the experiments as RF_1 and RF_2), (ii) LDA and PCA to create reduced artificial features (Chandrashekar & Sahin, 2014), (iii) MNF (Hong et al., 2014) and HUSFS (Cerrada et al., 2015b) as attribute clustering techniques to select the most representative attributes based on dependency degrees.

6.1. Illustrative example

A benchmark dataset associated to a classic classification problem is evaluated by our proposal, and most of the important steps that the algorithm executes are

described in this section. *Lvst* is a database related with voice rehabilitation, 14 participants were evaluated to detect whether voice rehabilitation treatment lead to phonations or not. A dataset with 126 samples and 309 features is obtained related to two classes, i.e. acceptable or unacceptable [1]. Firstly, the set A is stated as $A = \{A_1, A_2, ..., A_{309}\}$; then, the dissimilarity values between all the possible pairs of attributes by using Eq. 3 are computed. Table 6 reports some pairs composing the set A_d. In this particular case, the initial threshold is $t_{ini} = 5.9867$ according to Definition 4. The macro algorithm in Table 1 is executed, and some partial results are reported in Table 7. This table shows the iteration (it), the action and the result for each incoming attribute. The listed partial results are summarized as follows:

Table 6: Dissimilarity between the attributes of *Lvst* dataset.

Pair	d
(A_1, A_2)	5.7
(A_1, A_3)	5.8
(A_1, A_4)	5.1
.	.
.	.
.	.
(A_{109}, A_{113})	6.2
(A_{109}, A_{112})	4.7
.	.
.	.
.	.

- In the first iteration, there are not clusters in the system, and in consequence a new cluster is created with the attribute A_1.
- In the second iteration, the attribute A_2 is provisionally assigned to P_1; the final assignment condition is met and the merging process is not accomplished because there is only one available cluster.
- In the tenth iteration, the attribute A_{10} is assigned provisionally to P_1. The nearest neighbour sample of A_{10} is A_1, and the distance d between the attributes does not meet the neighbour threshold t_1, in consequence a new cluster with A_{10} as its prototype is created.
- In the eleventh iteration, the attribute A_{11} is provisionally assigned to P_1, and there is a neighbour cluster P_2. The conditions to merge are are not accomplished, thereby there is not a merge between clusters.

[1] http://archive.ics.uci.edu/ml

- For the thirty-seventh iteration, the attribute A_{37} is finally assigned to P_1, and it causes a merge with the cluster P_{19}.

This process continues until the whole set A is processed. As final result, 34 attribute clusters were obtained without the merging process and 12 attribute clusters with the merging process. The split process is never activated for this case study.

In order to evaluate the performance of ACARS in terms of the classification accuracy, $LSVT$ database is evaluated by several classifiers using additional reduced features (or attributes Att) obtained with the mentioned feature reduction techniques. In Table 8, the results are depicted where three important remarks can be highlighted: (i) The best result using ACARS is obtained without the merging process (black bold text), (ii) The best result using a feature selection technique different from a clustering technique is obtained with the importance variable ranking RF_2. This is an embedded approach under a supervised strategy (italic blue text), and (iii) The best result by using the clustering technique MNF is similar to the result of ACARS with the merging process (italic red text). The results show that our approach has similar performance comparatively to other supervised and clustering techniques for feature selection. For all the experiments, SVM is the best classifier as expected for binary classification.

Graphical analysis through a matrix heat map is illustrated in figures 10(a) and 10(b), where the dissimilarity is arranged in a matrix of attributes against attributes. The dissimilarity $d(A_i, A_j) = 1$ for all the pairs such as $i = j$; thus, all these pairs are on the diagonal. The result of ACARS in experiment E_1 is illustrated in Fig 10(a), where different clusters are built around the diagonal and, in general, we can notice that the dissimilarities values varies within 4 and 7. ACARS groups those attributes with low dissimilarity between them, as it is the pattern given by Fig. 10(a). Moreover, Fig. 10(b) shows the results in the experiment E_2, where we can notice the 12 clusters created. As additional information, we identified the number of attributes selected in the experiments E_1 and E_2 that agree with the attributes obtained with RF in the supervised scheme. For both E_1 and E_2, the number of attributes that agree with RF_1 and RF_2 are 21 and 9, respectively. In general, the number of selected features by ACARS and their performance with the SVM classifier are comparable with those results obtained with the RF based feature selection approach and the MNF based attribute clustering algorithm.

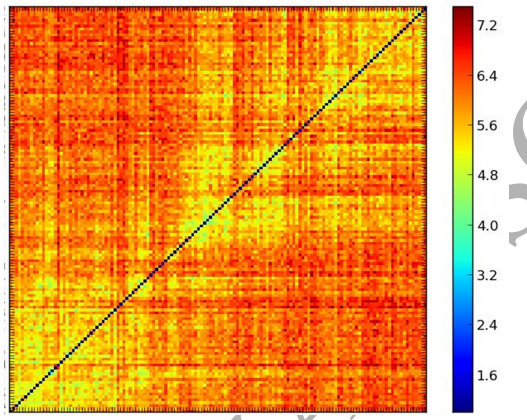
Additionally, other experiments have been developed with $\beta = 0.6$ and $\beta = 0.5$ in order to show through matrix heat maps how the proposal groups the attributes under different values of the parameter in the merging condition in Statement 5. Fig. 11 shows the results for each case. In Fig. 11(a), 9 clusters were constructed with $\beta = 0.6$; while Fig. 11(b) illustrates 4 clusters with $\beta = 0.5$. It is noticed that

Table 7: Feature selection using ACARS applied to an illustrative example

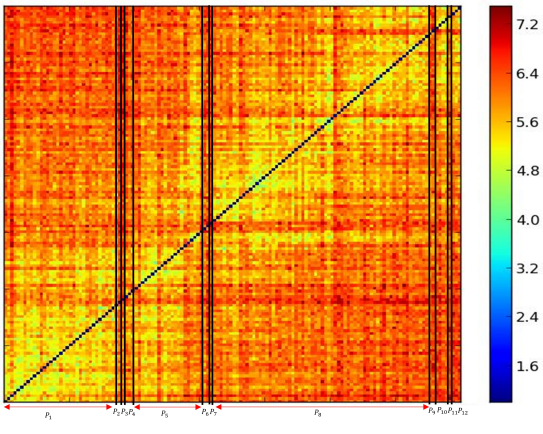
It	Input	Action	Result
1	A_1	• Step 1.: There are not clusters in the system • Step 2.: Create the cluster P_1 • Step 3.: End process	$P_1 = (A_1, 5.9864, \{A_1\})$
2	A_2	• Step 1.:There are clusters in the system – Step 1.1.: $d(A_2, A_1) = 5.7$ and $A_2 \in P_1$ provisionally – Step 1.2.: The neighbour attribute is $x^2_{nn_{P_1}} = A_1$ – Step 1.3.: $d(A_2, A_1) < t_1$ then $A_2 \in P_1$ ∗ Step 1.3.1.: $P_1 = (A_1, 5.7, \{A_1, A_2\})$ ∗ Step 1.3.2.: There is not neighbour cluster P_v ∗ Step 1.3.4.: Go to 3. • Step 3.: End process	$P_1 = (A_1, 5.7, \{A_1, A_2\})$
. . .	. . .	. . .	. . .
10	A_{10}	• Step 1.: There are clusters in the system – Step 1.1.: $d(A_{10}, A_5) = 5.5$ and $A_{10} \in P_1$ provisionally – Step 1.2.: The neighbour attribute is $x^{10}_{nn_{P_1}} = A_1$ – Step 1.3.: $d(A_{10}, A_1) > t_5$ then $A_{10} \notin P_1$ – Step 1.4.: There is not a neighbour cluster P_v – Step 1.6.: Go to 2. • Step 2.: Create the cluster P_2 • Step 3.: End process	$P_2 = (A_{10}, 5.9867, \{A_{10}\})$
11	A_{11}	• Step 1.: There are clusters in the system – Step 1.1.: $d(A_{11}, A_5) = 5.1$ and $A_{11} \in P_1$ provisionally – Step 1.2. : The neighbour attribute is $x^{11}_{nn_{P_1}} = A_2$ – Step 1.3.: $d(A_{11}, A_2) < t_1$ then $A_{11} \in P_1$ ∗ Step 1.3.1.: Update P_1 ∗ Step 1.3.2.: The neighbour cluster is $P_v = P_2$ ∗ Step 1.3.3.: $\exists P_v$ and $N_I = 0$ ∗ Step 1.3.4.: Go to 3. • End process	$P_1 = (A_5, 5.2422, \{A_1, ..., A_9, A_{11}\})$
. . .	. . .	. . .	. . .
131	A_{37}	• Step 1.: There are clusters in the system – Step 1.1.: $d(A_{37}, A_{15}) = 5.5$ and $A_{37} \in P_1$ provisionally – Step 1.2. : The neighbour attribute is $x^{37}_{nn_{P_1}} = A_3$ – Step 1.3.: $d(A_{37}, A_3) < 5.2833$ then $A_{37} \in P_1$ ∗ Step 1.3.1.: Update P_1 ∗ Step 1.3.2.: The neighbour cluster is $P_v = P_{19}$ ∗ Step 1.3.3.: $\exists P_v$ and $N_I = 23$, $23 > 0.7(5 + 26)$ ∗ Step 1.3.3.1.: Create a merged cluster with P_1 and P_{19} ∗ Step 1.3.3.2.: Go to 3. • Step 3.: End process	$P_1 = (A_{19}, 5.37, \{A_1, ..., A_{16}, A_{18}, ..., A_{26}, A_{28}, A_{32}, ..., A_{34}, A_{36}, A_{37}\})$
. . .	. . .	. . .	. . .

Table 8: Classification accuracy with different feature selection methods and classifiers, using $LSVT$ dataset

	all	*Corr*	RF_1	RF_2	*NMF*	*MNF*	E_1	E_2
Classifier	309Att	140Att	45Att	23Att	25Att	40Att	34Att	12Att
RF	0.8108	0.8108	0.8108	0.8649	0.8108	0.7568	0.8108	0.8649
DT	0.7838	0.7838	0.7838	0.7568	0.8378	0.7297	0.7838	0.8378
SVM	0.7838	0.8378	0.8378	*0.8919*	0.8108	*0.8108*	**0.8919**	*0.8108*
KNN	0.6216	0.5946	0.6486	0.7297	0.6486	0.5946	0.5946	0.6486



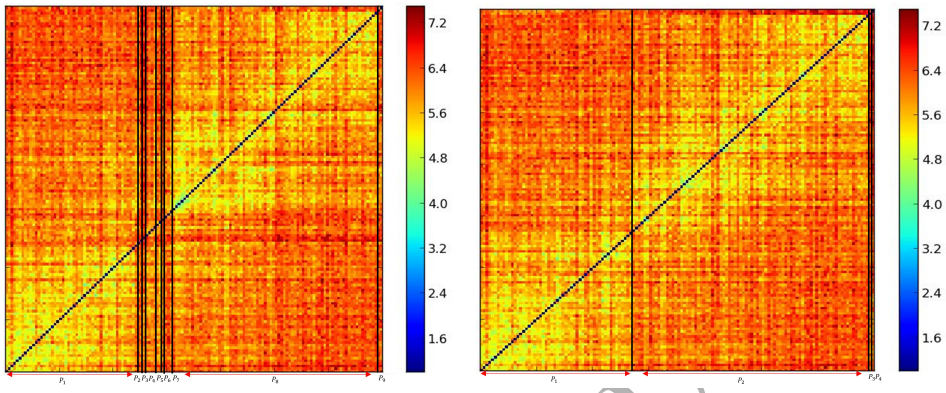
(a)



(b)

Figure 10: (a) Matrix heat map of the dissimilarities between the attributes grouped in the experiment E_1 (b) Matrix heat map of the dissimilarities between the attributes grouped in the experiment E_2, with 12 resulting clusters

the lower the value of β is, the more mergers are performed between the clusters, as expected from the fundamental guidelines.



(a) (b)

Figure 11: (a) Matrix heat map of the dissimilarities between the attributes grouped in the experiment E_2 with $\beta = 0.6$, with 9 resulting clusters. (b) Matrix heat map of the dissimilarities between the attributes grouped in the experiment E_2 with $\beta = 0.5$, with 4 resulting clusters

6.2. Fault severity classification in gears

ACARS was applied to perform feature selection, and its result is used to classify ten fault severity levels in gears. Table 9 reports the results using several feature selection techniques and classifiers, including the technique based on attribute clustering called HUSFS (an improved version of MNF) (Cerrada et al., 2015b). The number of attributes selected by ACARS in experiment E_1 and experiment E_2 were 410 and 342, respectively. Particular, for this case study, important results are obtained in this case study, and are summarized as follow for the SVM based classifier: (i) The best result using ACARS is obtained in E_1 without the merging process (bold black text), (ii) The result in E_2 is very close to E_1 with less number of features, (iii) The results after applying ACARS with E_1 and E_2 are better than those ones from HUSFS (italic red text), (iv) E_1 results are comparable with the results in experiment RF_1, with similar number of features. Finally, the number of attributes that agree with the attributes obtained through RF_1 in the supervised scheme are 104 and 102, in E_1 and E_2 respectively; in RF_2, 20 attributes agree with both E_1 and E_2. In general, ACARS with and without the merging process yields good results, and particularly these results are better than the obtained by the attribute clustering technique HUSFS.

Table 9: Classification accuracy with different feature selection methods and classifiers, using Gear severity dataset.

	all	*Corr*	RF_1	RF_2	*LDA*	*PCA*	*NMF*	*HUSFS*	E_1	E_2
Classifier	2652Att	1083Att	389Att	146Att	150Att	150Att	129Att	400Att	410Att	342Att
RF	1	1	1	0.9911	0.7946	0.7679	1	0.9756	0.9732	0.9643
DT	0.8661	0.8750	0.8839	0.8839	0.5000	0.7589	0.3482	0.6980	0.6696	0.6786
SVM	1	1	1	1	0.6786	0.9554	0.9643	*0.9821*	**1**	*0.9911*
KNN	0.5625	0.5179	0.6607	0.5625	0.5179	0.5536	0.3839	0.5080	0.2232	0.2500

6.3. Fault severity classification in bearings

The same analysis in section 6.2 was performed for fault severity classification in bearings. Table 10 shows the accuracy of several classifiers with different feature selection and reduction techniques. For both experiments E_1 and E_2 the number of selected features were 279, that means the merging process was not activated. Experiments with other values of β upper than 0.3 reports the same result. In summary, the main obtained results are: (i) The accuracy, after applying ACARS, is adequate even with or without merging process, and the best accuracy is exhibited by RF based classifier (bold black text), (ii) ACARS provides better results than HUSFS with the number of selected features, and accuracy as well (italic red text), (iii) Results in E_1 and E_2 are comparable with the supervised technique RF_1. As additional information, the number of attributes selected by E_1 and E_2 that agree with the attributes obtained with RF_1 and RF_2 was 39 attributes.

Table 10: Classification accuracy with different feature selection methods and classifiers, using Bearing severity dataset.

	all	*Corr*	RF_1	RF_2	*LDA*	*PCA*	*HUSFS*	E_1	E_2
Classifier	2652Att	1124Att	245Att	98Att	150Att	150Att	300Att	279Att	279Att
RF	1	1	1	1	0.9542	0.9739	*0.9954*	**0.9967**	**0.9967**
DT	0.9837	0.9636	0.9608	0.9771	0.8693	0.6569	0.9657	0.9542	0.9542
SVM	1	1	1	0.9967	0.9771	0.9967	0.9815	0.9869	0.9869
KNN	0.7386	0.6471	0.7124	0.6601	0.5490	0.6471	0.5237	0.2843	0.2843

6.4. Further analysis

We present as follow a comparison of the execution time obtained with the attribute clustering approaches based on rough set theory, i.e., ACARS, MNF and HUSFS. For this experiment, different datasets for classification were used, including the study cases assessed by this paper. It is important to mention that MNF and HUSFS algorithms were implemented by the authors as described in (Hong et al., 2014; Cerrada et al., 2015b), and the dependency degrees between all the possible attribute pairs are previously computed and stored for their further use by the three approaches. The computer used to perform the experiments is

The Titan with an Intel Core i7 processor 4.50 GHz. None optimization or parallel computing process was performed with the algorithms in order to have a plain and fair comparison between themselves.

Table 11 shows the dataset used with a brief description that includes the number of attributes (Att) and samples (S). The table reports the time (in seconds) needed to execute each of the attribute clustering approaches. It is notable HUSFS is faster than MNF. However, our proposal achieves the best results within the attribute clustering algorithms not only in time but in accuracy as it was previously presented in the experimental results.

Table 11: Comparison in term of execution time between ACARS, NMF and HUSFS

Dataset	Description	MNF [s]	HUSFS [s]	ACARS [s]
Ionosphere	34 Att and 351 S	$t_1 = 9.4839$	$t_1 = 2.5302$	0.002617
Libras Movement	56 Att and 361 S	$t_2 = 13.4694$	$t_2 = 8.5485$	0.004942
Lsvt	140 Att and 126 S	$t_3 = 4346.8184$	$t_3 = 14.8699$	0.06211
Isolet5	564 Att and 1559 S	$t_4 > 40 * t_3$	$t_4 = 3736{,}0278$	0.6052
Bearing severity	1083 Att and 750 S	$t_5 > 2 * t_4$	$t_5 > 10 * t_4$	53.0468
Gear severity	1124 Att and 750 S	$t_6 > 2 * t_4$	$t_6 > 10 * t_4$	28.2434

Overall, the experimental results reflect that ACARS accuracy is similar to other supervised approaches, which is an asset, as ACARS is unsupervised. In particular, we focused our attention on RF results and the difference with our approach in each experiment due to RF is one of the most accurate algorithms to measure the attribute importance given a labeled dataset. On the other hand, ACARS is faster than some other unsupervised approaches that use relative dependency such as MNF and HUSFS.

7. Conclusion

A novel algorithm called ACARS for attribute clustering based on an unsupervised strategy is proposed. The algorithm is based on the main ideas from classification based on distance and clustering based on prototypes. Particularly, ACARS is inspired by K-means and 1-NN techniques, that are properly combined into a workflow to analyse a subset of available clusters when inputs, in this case attributes, are continuously incoming one by one. This characteristic allows building new clusters when needed, giving to ACARS an evolving property under incremental learning scenario that do not need *a priori* knowledge of the number of cluster to be created. Therefore, this is one of our main contributions aside from the fact that the algorithm offers an unsupervised solution to extract features. With the evolving and incremental learning properties as new contributions, ACARS might impact new developments regarding unsupervised feature selection methods and

clustering algorithms in expert and intelligent systems, by properly considering the objects to be grouped and the similarity measure.

Dependency degree between attributes proposed by the rough set theory is used as dissimilarity measure to create or update the clusters. The feature selection algorithm was validated in classification problems related to real applications of fault severity diagnosis in rotating machinery. The results from the case studies show that ACARS provides better results than a similar attribute clustering technique called HUSFS. Moreover, ACARS results are comparable with the well-known supervised feature selection technique given by the RF algorithm which is based on the attribute importance ranking, with the advantage that ACARS is an unsupervised algorithm. Besides the classification problems analysed in this work, most of the real world problems do not necessarily have labels of their condition states to perform supervised strategies of feature selection. In fact, we can find several processes with a large number of unlabeled samples. Conversely, unsupervised feature selection techniques are valuable and important to study.

The algorithm is sensitive to both the threshold proposed in Definition 3, and to the parameters α and β given by Definition 8 and Statement 5, respectively. Consequently, future research works can be oriented to: (i) Tune the threshold value in Definition 3 through an optimization process in order to minimize the distance between the attributes assigned to a cluster. The optimization process would aim at improving the compactness of the clusters constructed. (ii) In the same vein, improve the selection of the parameter α in Definition 8 to optimally measure the separability between clusters. On the other hand, it is important to mention that the algorithm was tested using numeric data where the samples are obtained through a discretization process. In consequence, (iii) As an additional work, the paper can be evaluated using real world applications using categorical data, and (iv) Finally, the algorithm can be modified to obtain a fuzzy system, where the numeric data can be discretized through different fuzzy metrics; in this particular case, the dissimilarity metric in Section 3.1 has to be redefined in fuzzy terms.

Acknowledgments

The work was sponsored in part by the GIDTEC project No. 017-007-2015-11-05, and the Prometeo Project of the Secretariat for High Education, Science, Technology and Innovation (SENESCYT) of the Republic of Ecuador. The experimental work was developed at the Vibration Laboratory of GIDTEC, in the Universidad Politécnica Salesiana de Cuenca-Ecuador.

References

Au, W.-H., Chan, K. C. C., Wong, A. K. C., & Wang, Y. (2005). Attribute clustering for grouping, selection, and classification of gene expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *2*, 83–101.

Bhadra, T., & Bandyopadhyay, S. (2015). Unsupervised feature selection using an improved version of differential evolution. *Expert Systems with Applications*, *42*, 4042 – 4053.

Boobalan, M. P., Lopez, D., & Gao, X. (2016). Graph clustering using k-neighbourhood attribute structural similarity. *Applied Soft Computing*, *47*, 216 – 223.

Cabrera, D., Sancho, F., Sánchez, R. V., Zurita, G., Cerrada, M., Li, C., & Vásquez, R. (2015). Fault diagnosis of spur gearbox based on random forest and wavelet packet decomposition. *Frontiers of Mechanical Engineering*, *10*, 1–10.

Cerrada, M., Sánchez, R. V., Cabrera, D., Zurita, G., & Li, C. (2015a). Multi-stage feature selection by using genetic algorithms for fault diagnosis in gearboxes based on vibration signal. *Sensors*, *15*, 23903–23926.

Cerrada, M., Sánchez, R.-V., Pacheco, F., Cabrera, D., Zurita, G., & Li, C. (2015b). Hierarchical feature selection based on relative dependency for gear fault diagnosis. *Applied Intelligence*, *44*, 687–703.

Cerrada, M., Zurita, G., Cabrera, D., Sánchez, R.-V., Artés, M., & Li, C. (2016). Fault diagnosis in spur gears based on genetic algorithm and random forest. *Mechanical Systems and Signal Processing*, *70-71*, 87–103.

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

Chapelle, O., Scholkopf, B., & Zien, A. (2006). *Semi-Supervised Learning*. Cambridge MA: MIT Press.

Chen, C.-H. (2015). Feature selection for clustering using instance-based learning by exploring the nearest and farthest neighbors. *Information Sciences*, *318*, 14 – 27.

Dadaneh, B. Z., Markid, H. Y., & Zakerolhosseini, A. (2016). Unsupervised probabilistic feature selection using ant colony optimization. *Expert Systems with Applications*, *53*, 27 – 42.

Dunham, M. (2003). *Data Mining: Introductory and Advanced Topics*. Prentice Hall.

Fazayeli, F., Wang, L., & Mandziuk, J. (2008). Feature selection based on the rough set theory and expectation-maximization clustering algorithm. In *Rough Sets and Current Trends in Computing* (pp. 272–282). Springer Berlin Heidelberg volume 5306 of *Lecture Notes in Computer Science*.

Foithong, S., Pinngern, O., & Attachoo, B. (2012). Feature subset selection wrapper based on mutual information and rough sets. *Expert Systems with Applications*, *39*, 574 – 584.

Ganivada, A., Ray, S. S., & Pal, S. K. (2013). Fuzzy rough sets, and a granular neural network for unsupervised feature selection. *Neural Networks*, *48*, 91–108.

Genuer, R., Poggi, J.-M., & Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, *31*, 2225–2236.

Gheyas, I. A., & Smith, L. S. (2010). Feature subset selection in large dimensionality domains. *Pattern Recognition*, *43*, 5–13.

Gu, X., Li, Y., & Jia, J. (2015). Feature selection for transient stability assessment based on kernelized fuzzy rough sets and memetic algorithm. *International Journal of Electrical Power & Energy Systems*, *64*, 664–670.

Han, J., Hu, X., & Lin, T. (2004). Feature subset selection based on relative dependency between attributes. In *Rough Sets and Current Trends in Computing* (pp. 176–185). volume 3066 of *Lecture Notes in Computer Science*.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. New York: Springer.

He, X., Cai, D., & Niyogi, P. (2005). Laplacian score for feature selection. In *Advances in neural information processing systems* (pp. 507–514).

Herawan, T., Deris, M. M., & Abawajy, J. H. (2010). A rough set approach for selecting clustering attribute. *Knowledge-Based Systems*, *23*, 220 – 231.

Hong, T.-P., Chen, C.-H., & Lin, F.-S. (2015). Using group genetic algorithm to improve performance of attribute clustering. *Applied Soft Computing*, *29*, 371 – 378.

Hong, T.-P., Liou, Y.-L., Wang, S.-L., & Vo, B. (2014). Feature selection and replacement by clustering attributes. *Vietnam Journal of Computer Science*, *1*, 47–55.

Jensen, R., & Parthaláin, N. M. (2015). Towards scalable fuzzyrough feature selection. *Information Sciences*, *323*, 1 – 15.

Jornsten, R., & Yu, B. (2003). Simultaneous gene clustering and subset selection for sample classification via md. *BioinformaticsS*, *19*, 11001109.

Klepaczko, A., & Materka, A. (2010). Artifical intelligence and soft computing: 10th international conference, icaisc 2010, zakopane, poland, june 13-17, 2010, part ii. chapter Combining Evolutionary and Sequential Search Strategies for Unsupervised Feature Selection. (pp. 149–156). Berlin, Heidelberg: Springer Berlin Heidelberg.

Liu, H., & Motoda, H. (2008). *Computational Methods of Feature Selection*. Boca Raton, Florida: Chapman and Hall, Taylor and Francis Group.

Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, *17*, 491–502.

Liu, M., & Zhang, D. (2016). Feature selection with effective distance. *Neurocomputing*, *215*, 100 – 109.

Liu, Y., Liu, K., Zhang, C., Wang, J., & Wang, X. (2016). Unsupervised feature selection via diversity-induced self-representation. *Neurocomputing*, (pp. –). doi:http://dx.doi.org/10.1016/j.neucom.2016.09.043.

Lughofer, E., & Sayed-Mouchaweh, M. (2015). Autonomous data stream clustering implementing split-and-merge concepts towards a plug-and-play approach. *Information Sciences*, *304*, 54 – 79.

Maji, P. (2011). Fuzzy rough supervised attribute clustering algorithm and classification of microarray data. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *41*, 222–233.

Maldonado, S., Carrizosa, E., & Weber, R. (2015). Kernel penalized k-means: A feature selection method based on kernel k-means. *Information Sciences*, *322*, 150 – 160.

Mamat, R., Herawan, T., & Deris, M. M. (2013). MAR: Maximum Attribute Relative of soft set for clustering attribute selection. *Knowledge-Based Systems*, *52*, 11–20.

Moradi, P., & Rostami, M. (2015). A graph theoretic approach for unsupervised feature selection. *Engineering Applications of Artificial Intelligence*, *44*, 33 – 45.

Ng, A. Y. (2004). Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning* (p. 78). ACM.

Niu, K., Zhang, S., & Chen, J. (2008). Subspace clustering through attribute clustering. *Frontiers of Electrical and Electronic Engineering in China*, *3*, 44–48.

Pacheco, F., Cerrada, M., Sánchez, R.-V., Cabrera, D., Li, C., & Valente-de-Oliveira, J. (2016a). Clustering algorithm using rough set theory for unsupervised feature selection. In *2016 International Joint Conference on Neural Networks (IJCNN)* (pp. 3493–3499). doi:10.1109/IJCNN.2016.7727647.

Pacheco, F., Valente-de-Oliveira, J., Sánchez, R.-V., Cerrada, M., Cabrera, D., Li, C., Zurita, G., & Artés, M. (2016b). A statistical comparison of neuroclassifiers and feature selection methods for gearbox fault diagnosis under realistic conditions. *Neurocomputing*, *194*, 192 – 206.

Parthaláin, N. M., & Jensen, R. (2013). Unsupervised fuzzy-rough set-based dimensionality reduction. *Information Sciences*, *229*, 106 – 121.

Qian, W., & Shu, W. (2015). Mutual information criterion for feature selection from incomplete data. *Neurocomputing*, *168*, 210 – 220.

Qin, H., Ma, X., Zain, J. M., & Herawan, T. (2012). A novel soft set approach in selecting clustering attribute. *Knowledge-Based Systems*, *36*, 139 – 145.

Raileanu, L. E., & Stoffel, K. (2004). Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, *41*, 77–93.

Shang, W., Huang, H., Zhu, H., Lin, Y., Qu, Y., & Wang, Z. (2007). A novel feature selection algorithm for text categorization. *Expert Systems with Applications*, *33*, 1 – 5.

Shi, H., Li, Y., Han, Y., & Hu, Q. (2015). Cluster structure preserving unsupervised feature selection for multi-view tasks. *Neurocomputing*, *175, Part A*, 686697.

Siedlecki, W., & Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, *10*, 335 – 347.

Song, Q., Ni, J., & Wang, G. (2013). A fast clustering-based feature subset selection algorithm for high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, *25*, 1–14.

Świniarski, R. W. (2001). Rough sets methods in feature reduction and classification. *International Journal of Applied Mathematics and Computer Science*, *11*, 565–582.

Tabakhi, S., Moradi, P., & Akhlaghian, F. (2014). An unsupervised feature selection algorithm based on ant colony optimization. *Engineering Applications of Artificial Intelligence*, *32*, 112 – 123.

Tan, P., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining*. Addison Wesley.

Wang, D., Zhang, H., Liu, R., Liu, X., & Wang, J. (2016). Unsupervised feature selection through Gram-Schmidt orthogonalization-a word co-occurrence perspective. *Neurocomputing*, *173, Part 3*, 845 – 854.

Wang, S., Pedrycz, W., Zhu, Q., & Zhu, W. (2015). Unsupervised feature selection via maximum projection and minimum redundancy. *Knowledge-Based Systems*, *75*, 19 – 29.

Wu, Y., Wang, C., Bu, J., & Chen, C. (2016). Group sparse feature selection on local learning based clustering. *Neurocomputing*, *171*, 1118 – 1130.

Yuwono, M., Guo, Y., Wall, J., Li, J., West, S., Platt, G., & Su, S. W. (2015). Unsupervised feature selection using swarm intelligence and consensus clustering for automatic fault detection and diagnosis in heating ventilation and air conditioning systems. *Applied Soft Computing*, *34*, 402 – 425.

Zhao, X., Deng, W., & Shi, Y. (2013). Feature selection with attributes clustering by maximal information coefficient. *Procedia Computer Science*, *17*, 70 – 79. First International Conference on Information Technology and Quantitative Management.

Zhou, N., Cheng, H., Pedrycz, W., Zhang, Y., & Liu, H. (2016). Discriminative sparse subspace learning and its application to unsupervised feature selection. {*ISA*} *Transactions*, *61*, 104 – 118.

Zhou, P.-Y., & Chan, K. C. C. (2015). An unsupervised attribute clustering algorithm for unsupervised feature selection. In *IEEE International Conference on Data Science and Advanced Analytics* (pp. 1–7).

Zorarpac, E., & Özel, S. A. (2016). A hybrid approach of differential evolution and artificial bee colony for feature selection. *Expert Systems with Applications*, *62*, 91 – 103.