Computer Modeling in Engineering & Sciences
DOI:10.32604/cmes.2020.010791
Article

Tech Science Press

Hybridization of Fuzzy and Hard Semi-Supervised Clustering Algorithms Tuned with Ant Lion Optimizer Applied to Higgs Boson Search

Soukaina Mjahed[1,*], Khadija Bouzaachane[1], Ahmad Taher Azar[2,3], Salah El Hadaj[1] and Said Raghay[1]

[1]Faculty of Sciences and Technology, Department of Applied Mathematics and Computer Sciences, Cadi Ayyad University, Marrakech, 40000, Morocco
[2]Robotics and Internet-of-Things Lab (RIOTU), Prince Sultan University, Riyadh, 12435, Saudi Arabia
[3]Faculty of Computers and Artificial Intelligence, Benha University, Benha, 13511, Egypt
*Corresponding Author: Soukaina Mjahed. Email: soukaina.mjahed@gmail.com
Received: 30 March 2020; Accepted: 31 July 2020

Abstract: This paper focuses on the unsupervised detection of the Higgs boson particle using the most informative features and variables which characterize the "Higgs machine learning challenge 2014" data set. This unsupervised detection goes in this paper analysis through 4 steps: (1) selection of the most informative features from the considered data; (2) definition of the number of clusters based on the elbow criterion. The experimental results showed that the optimal number of clusters that group the considered data in an unsupervised manner corresponds to 2 clusters; (3) proposition of a new approach for hybridization of both hard and fuzzy clustering tuned with Ant Lion Optimization (ALO); (4) comparison with some existing metaheuristic optimizations such as Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). By employing a multi-angle analysis based on the cluster validation indices, the confusion matrix, the efficiencies and purities rates, the average cost variation, the computational time and the Sammon mapping visualization, the results highlight the effectiveness of the improved Gustafson–Kessel algorithm optimized with ALO (ALOGK) to validate the proposed approach. Even if the paper gives a complete clustering analysis, its novel contribution concerns only the Steps (1) and (3) considered above. The first contribution lies in the method used for Step (1) to select the most informative features and variables. We used the t-Statistic technique to rank them. Afterwards, a feature mapping is applied using Self-Organizing Map (SOM) to identify the level of correlation between them. Then, Particle Swarm Optimization (PSO), a metaheuristic optimization technique, is used to reduce the data set dimension. The second contribution of this work concern the third step, where each one of the clustering algorithms as K-means (KM), Global K-means (GlobalKM), Partitioning Around Medoids (PAM), Fuzzy C-means (FCM),

Gustafson–Kessel (GK) and Gath–Geva (GG) is optimized and tuned with ALO.

Keywords: Ant lion optimization; binary clustering; clustering algorithms; Higgs boson; feature extraction; dimensionality reduction; elbow criterion; genetic algorithm; particle swarm optimization

1 Introduction

Knowledge discovery mechanisms promote combination within or across machine-learning algorithms. There are two common techniques to extract knowledge: classification and clustering. Clustering identifies the distribution of patterns in the data sets by grouping the similar data points. The objects from the same group are more comparable to those of distinct groups. Some clustering algorithm results are highly dependent on initial cluster centroids which make their high probability of trapping into local optima [1].

Particle Swarm Optimization (PSO) [2], Firefly Algorithm [3], Ant Colony Optimization (ACO) [4], Spider Monkey Optimization (SMO) [5], Artificial Bee Colony (ABC) [6], Grey Wolf Optimizer (GWO) [7], Whale Optimization Algorithm (WOA) [8], Moth-fame Optimization (MFO) [9] and Ant Lion Optimizer (ALO) [10] are some metaheuristic methods that help to efficiently explore the search space and avoid becoming stuck in local optima. The Salp Swarm Algorithm (SSA), which is an inspiration of the swarming behavior of salps, was introduced in [11]. A harmonized salp chain-built optimization was presented in [12] to improve the search mechanism of the classical SSA using the opposition-based learning and LVF search concepts. A hybrid multi-objective GWO was used in [13] for a dynamic welding scheduling problem. To minimize the potential energy of molecules, a Hybrid Grey Wolf Optimizer and Genetic Algorithm (HGWOGA) has been designed in [14]. A binary version of GWO for feature selection was presented in [15]. The work done in [16] presents five variants of GWO and analyses their performance using a fuzzy hierarchical operators. In [17], aGWO for multi-objective optimization problems was developed. A novel Random Walk Grey Wolf Optimizer (RW-GWO) is proposed to increase the potential of GWO algorithm in [18]. A study proposed a memory-based Grey Wolf Optimizer for global optimization tasks (mGWO) [19]. A Hybrid Grey Wolf Optimizer with mutation operator was presented in [20].

Combined to these metaheuristics, many clustering approaches have been proposed in the literature in order to reduce the issue of local optimum trap. A clustering analysis using a novel locality-informed grey wolf-inspired clustering approach was defined in [21]. A novel clustering method using enhanced Grey Wolf Optimizer (GWO) and Map Reduce is presented in [22]. An Ant Lion-based Random Walk Differential Evolution algorithm for optimization and clustering was proposed in [23]. An improved Particle Swarm Optimization (PSO) based K-means applied to Fault signal diagnosis is cited in [24]. A review on clustering with Genetic Algorithms (GA) was done [25]. An engine fault signals diagnosis using Genetic Algorithm and K-means based clustering is defined in [26]. A Whale Optimization Algorithm (WOA) approach for clustering is introduced in [27]. Clustering was implemented in various areas with the development of data mining, such as graphical recognition [28], machine learning [29], market analysis [30], medical diagnosis [31,32] and high-energy physics [33,34].

The primary concern of the field of high-energy physics is the interactions of the essential constituents of matter. The modern accelerators are the primary tools used in high energy physics experiments, allowing the creation of new particles that only result in extremely high energy.

A particularly interesting particle is the Higgs boson, created by physicist Peter Higgs to argue the validity of the standard model [35]. This particle was the subject of the 2013 Nobel Prize in Physics awarded jointly to Peter Higgs and François Englert, after its discovery by the CERN (European Nuclear Research Organization) at Large Hadron Collider (LHC) [36]. Some of these researches were ATLAS [37] and CMS which claimed the Higgs boson discovery [38]. The Higgs boson has several distinct decaying procedures. Putting the original discovery aside, the study of all decaying modes assures the validity of the theory and allows distinguishing new elements.

It should be noted according to Higgs boson recognition complexity, that high-performance methods such as neural networks as well as other multidimensional methods have been used [39–42]. However, the use of clustering methods in the high energy physics and the Higgs boson search is restricted. For the high energy physics data classification, some clustering methods [43] and similar techniques that focus on a space-filling curve are applied [44]. A dynamic nearest neighbor graph implemented with computational geometry was first used in the study of Jet clustering in particle physics event properties [33]. An approach based on Fuzzy clustering algorithms is proposed for the recognition of particles based on pulse shape evaluation [45]. Studying the H to tau–tau channel was the topic of the ATLAS challenge [46]. Using a collection of deep neural networks, the winning method [47] had costly computational complexity but achieved results greater than the runner-up with a margin that is statistically important. There is also an advanced Decision Tree technique that offers a solid balance between efficiency and simple implementation [48]. An attempt to prove that Cartesian Genetic Programming (CGP) could be an interesting solution to learn from the Higgs big data set was presented in [49]. The work done in [50] shows a practical implementation for the study of non-resonant production of Higgs boson pairs in the context of extensions of the standard model with anomalous couplings of the Higgs bosons.

To identify the Higgs boson signals and achieve a better classification performance, it is important that the features selected contain necessary discriminative information. Good classification results can be achieved if the input is well prepared. This promotes the use of simple algorithms which resulting functions are easy to interpret compared with complex schemes such as neural networks or support vector machines [41]. There isn't any ideal algorithm for clustering that is suitable for all areas. Successfully applying a specific technique in one domain does not necessarily imply that it can be deployed in another domain with the same degree of satisfaction. Several considerations can affect the decision of the best technique of clustering such as the understanding of the clustering objectives, the comparison of the algorithms results and the use of the appropriate clustering performance measurements [29]. The interaction between the data to be clustered and the efficiency of distinct clustering algorithms is an interesting issue that has yet to be answered in clustering. An attempt to locate the optimum cluster number, while comparing between hard and fuzzy clustering algorithms of the applied thyroid diseases data set, is presented. Our work is a continuity of the one presented in [31].

The main contribution of the approach presented in this paper is the hybridization of fuzzy and hard semi-supervised clustering algorithms tuned with ALO to distinguish a Higgs signal from a background in an unsupervised way. The clustering algorithms used in this hybridization are K-means [51], Global K-means [52,53], K-medoids [54–56], Fuzzy C-means [57,58], Gustafson–Kessel [57,58] and Gath–Geva [57,58] algorithms. To bring to a close the performances of this hybridization, a comparative study is done with some metaheuristic methods such as GA [26,59] and PSO [60,61] using multiple validity indexes. In addition, an in-depth investigation of the proposed algorithms parameters was carried to achieve better results.

This paper presents another contribution when selecting the most informative features from the dataset using a feature engineering technique based on Higgs data quantities ranking using T-statistic method and K-means algorithm optimized using PSO, before excluding highly correlated features using SOM and then features dimensionality reduction using K-means algorithm optimized by PSO.

The Machine Learning dataset (HiggsML)—based on ATLAS events and characterized by a list of selected attributes and variables—is used to confirm the proposed approach [62].

The elbow criterion [63] is applied to find the optimal number of clusters that group the considered data in an unsupervised way.

The rest of this paper sections is organized as follows. Section 2 explains the dataset setting and gives the physics background of this work. Section 3 introduces the clustering and meta-heuristic algorithms used in this work. Section 4 is dedicated to our proposed approach. Section 5 reports the performance of the experimental analyses and comparisons of the proposed clustering algorithms. Section 6 concludes the analysis and discusses future research directions.

2 Data

As part of the mechanism that gives mass to other elementary particles, the Higgs Boson particle was theoretically anticipated to exist almost 50 years ago. Its importance lies in the fact that it is the last ingredient of the particle physics Standard Model for fundamental particles and forces.

In the beginning, an LHC event is initiated by a proton–antiproton collision. By using detectors surrounding the intersection zone, it can be observed that the collision energy forms new particles. Most events, however, do not generate interesting particles like the top quark or the Higgs boson. A successful interpretation of data relies on the efficient distinction between events that produce interesting particles (signal) as well as those who produce other particles (background).

In the past, human knowledge (naive-Bayes-like "cut-based" methods) designed the region of interest. Event selection in high-energy physics is an important topic for machine learning as multiple background kinds can match the distinct signature of the signal.

From the point of view of machine learning, the concern can be formally considered as binary classification problem where the events generated in the collider are pre-processed and reflected as feature vectors. The concern is to classify events as a signal (i.e., an event of interest, in this work a decay of H to tau–tau) or background (an event produced by processes already known). More specifically, the classifier is used as a technique of selection that specifies a signal-rich region (not necessarily connected) in the space of the feature. While the objective is to find a new phenomenon, in the real data, the labeled actual signal events are not available. Alternatively, an extensive simulator is used to simulate events by producing artificial events based on the Standard Model and a detector model, taking into consideration possible artifacts and noise. These simulations are used to evaluate the classifiers.

The classes are very imbalanced in the real data (around two signal events after pre-selection in a thousand events). For this reason, in signal events, the simulated data used in this work is enhanced. Weights are applied to all events of significance to reflect their chances of event occurring to compensate for this bias.

CERN's ATLAS experiment provided simulated data used in many types of Higgs boson research. A portion of these data was publicly released.

The work presented in this paper is based on the data provided by the Higgs Boson Machine Learning experiment (HiggsML) organized in 2014 by the ATLAS physicist's team and computer scientists [62].

As introduced above, this work considers two classes of Higgs Boson: signals (denoted H class) and background (B class). Notice that, each class consists of several sub-classes reflecting the various Higgs decay channels and various other typical reactions. These different subclasses are not considered in this work.

The collected data amounted to 200,000 signals, with 100,000 learning signals and 100,000 test signals. In the first batch of signals, the class H participates with 34,170 events and the class B participates with 65,830 events. As a test batch of signals, the class H participates with 35450 events and the class B participates with 64,550 events.

Each selected event is described using a list of features. The quantities that were selected by the physicists of ATLAS to select regions of interest are [62]:

- $\boldsymbol{F_1}$: The estimated mass m_H of the Higgs boson candidate, obtained through probabilistic phase space integration.
- $\boldsymbol{F_2}$: The transverse mass between the missing transverse energy and the lepton.
- $\boldsymbol{F_3}$: The invariant mass of the hadronic tau and the lepton.
- $\boldsymbol{F_4}$: The modulus of the vector sum of the transverse momentum of the hadronic tau, the lepton and the missing transverse energy vector.
- $\boldsymbol{F_5}$: The absolute value of the pseudorapidity separation between the two jets.
- $\boldsymbol{F_6}$: The invariant mass of the two jets.
- $\boldsymbol{F_7}$: The product of the pseudorapidities of the two jets.
- $\boldsymbol{F_8}$: The R separation between the hadronic tau and the lepton.
- $\boldsymbol{F_9}$: The modulus of the vector sum of the missing transverse momenta and the transverse momenta of the hadronic tau, the lepton, the leading jet.
- $\boldsymbol{F_{10}}$: The sum of the moduli of the transverse momenta of the hadronic tau, the lepton, the leading jet.
- $\boldsymbol{F_{11}}$: The ratio of the transverse momenta of the lepton and the hadronic tau.
- $\boldsymbol{F_{12}}$: The centrality of the azimuthal angle of the missing transverse energy vector with respect to the hadronic tau and the lepton.
- $\boldsymbol{F_{13}}$: The centrality of the pseudorapidity of the lepton with respect to the two jets.
- $\boldsymbol{F_{14}}$: The transverse momentum $\sqrt{{p_x}^2+{p_y}^2}$ of the hadronic tau.
- $\boldsymbol{F_{15}}$: The pseudorapidity of the hadronic tau.
- $\boldsymbol{F_{16}}$: The azimuth angle of the hadronic tau.
- $\boldsymbol{F_{17}}$: The transverse momentum $\sqrt{{p_x}^2+{p_y}^2}$ of the lepton (electron or muon).
- $\boldsymbol{F_{18}}$: The pseudorapidity of the lepton.
- $\boldsymbol{F_{19}}$: The azimuth angle of the lepton.
- $\boldsymbol{F_{20}}$: The missing transverse energy $\vec{E}_T$.
- $\boldsymbol{F_{21}}$: The azimuth angle of the missing transverse energy.
- $\boldsymbol{F_{22}}$: The total transverse energy in the detector.
- $\boldsymbol{F_{23}}$: The number of jets (integer with a value of 0, 1, 2 or 3; possible larger values have been capped at 3).

- $\boldsymbol{F_{24}}$: The transverse momentum $\sqrt{{p_x}^2+{p_y}^2}$ of the leading jet, i.e., the jet with the largest transverse momentum.
- $\boldsymbol{F_{25}}$: The pseudorapidity of the leading jet.
- $\boldsymbol{F_{26}}$: The azimuth angle of the leading jet.
- $\boldsymbol{F_{27}}$: The transverse momentum $\sqrt{{p_x}^2+{p_y}^2}$ of the jet, i.e., the jet with second largest transverse momentum.
- $\boldsymbol{F_{28}}$: The pseudorapidity of the subleading jet.
- $\boldsymbol{F_{29}}$: The azimuth angle of the subleading jet.
- $\boldsymbol{F_{30}}$: The scalar sum of the transverse momentum of all the jets of the events.

Each event is labeled as signal or background. Since the aim of this paper is an unsupervised classification, the event label is not used as an input feature to the clustering classifier, but as validation information. To avoid bias with a wider range, all the considered features are normalized.

3 Methodology

The following subsections present an overview of the clustering algorithms and metaheuristic techniques as well as some validation indices and performance metrics used in this work.

3.1 Clustering Algorithms

It is possible to categorize clustering algorithms into partition-based algorithms, hierarchical-based algorithms, density-based algorithms, grid-based algorithms, fuzzy-based algorithms and model-based algorithms [64–69]. These algorithms are used to measure the level of resemblance within and between clusters.

Various clustering algorithms have been proposed and designed, including K-means [51], Global K-means [52,53], K-medoids [54–56] and Fuzzy methods [57,58].

3.1.1 K-Means Algorithm

K-means (KM) clustering algorithm [51] is a well-known clustering method dividing samples into K clusters by updating the center of clusters in an iterative way until the criteria for convergence is met. Elements with highest resemblance are allocated to the same group and those with reduced resemblance are allocated to distinct groups [31].

Algorithm 1: KM Algorithm

1: **Initialize** a set of K items as first centroids
2: **Repeat**
3: Structure K clusters by allocating all items to the adjacent centroid
4: Recompute every cluster's centroid
5: **Until** no cluster position changes
6: **Return** K clusters positions

K-means clustering algorithm effectiveness is highly reliant on the randomly chosen initial cluster centroids. Few options are suggested to fix this issue of initialization. One of those techniques is the Global K-means clustering.

3.1.2 Global K-Means Algorithm

The global K-means (GlobalKM) clustering is an efficient global clustering technique which objective is to optimize the clustering error based on the K-means method as a local search strategy. It is an iterative clustering method that adds one cluster sequentially by applying N (equal to the data size) executions of the K-means algorithm as a global search process based on non-random initial positions [52,53].

Algorithm 2: GlobalKM Algorithm

1: **Define** the first cluster by computing the global data center.
2: $i = 2$
3: **Repeat**
4: **For each** point of the dataset
5: Grant the current data point as the i^{th} cluster
6: Allocate data elements to the nearest cluster from the i clusters.
7: Compute the root square quality per cluster using Eq. (1).

$$J = \sum_{i=1}^{K} \sum_{x_{j \in C_i}} \|x_j - c_i\| \tag{1}$$

8: **End for**
9: Memorize the C_i *cluster* that optimizes the root square quantity as the i^{th} center.
10: $i = i + 1$
11: **until** $i >= K$
12: Apply the KM algorithm (Algorithm 1) to the identified clusters.
13: **Return** K clusters positions

3.1.3 K-Medoids Algorithm

K-medoids [54–56] is a development of K-means by handling separate data. It considers the closest data point to the data centroid as the appropriate cluster. In comparison with K-means, it is a solid method for noise and anomalies. This is due to the fact that K-medoids reduces the sum value of dissimilitude between the cluster and the data points belonging to it, instead of the sum of squared Euclidean distances.

Several approaches using the K-medoids method have been proposed, including those suggesting novel multi-centroid, multi-run sampling schemes [55] and new search strategies for efficient K-medoids-based algorithms [56].

The most prevalent technique of K-medoids clustering is the Partitioning Around Medoids (PAM) algorithm [31,70]. This is also the algorithm that was adopted in our work as suggested below (Algorithm 3).

Algorithm 3: PAM Algorithm

1: **Select** randomly K elements of the n data points as the medoids.
2: **Repeat**
3: Allocate each data element to the neighboring medoid.

(Continued)

Algorithm 3 (Continued)

4: **For each** medoid m from the K selected ones
5: **For each** data element o allocated to m
6: Swap m and o.
7: Calculate the total cost corresponding to the average difference between o and all the m-related data points.
8: **End for**
9: **End for**
10: **until** no cluster position changes
11: **Return** K clusters positions

3.1.4 Fuzzy C-Means Algorithm

The core idea of Fuzzy c-means (FCM) is to define the membership of each element to each cluster using the optimization of the objective function. The membership summation of every element of the data needs to be equal to 1 [31,71–79].

The cluster centroids are revised after each iteration based on the Eq. (2).

$$c_i = \frac{\sum_{j=1}^{N} {\mu_{ij}}^m x_j}{\sum_{j=1}^{N} {\mu_{ij}}^m} \tag{2}$$

where:

- N is the number of data points.
- c_i corresponds to the i^{th} cluster center with $i \in \{1, \ldots, K\}$ and K is the number of clusters.
- M is the fuzziness index with $m \in [1, \infty[$.
- K is the number of cluster center.
- μ_{ij} is the membership of the i^{th} data to j^{th} cluster center. It is calculated using the Eq. (3):

$$\mu_{ij} = \frac{1}{\sum_{l=1}^{K} \frac{d_{ij}}{d_{il}}\left(\frac{2}{m-1}\right)} \tag{3}$$

d_{ij} is the Euclidean distance between the i^{th} data and the j^{th} cluster center. It corresponds to Eq. (4):

$$d_{ij} = \left\| x_i - c_j \right\| \tag{4}$$

The FCM algorithm's primary goal is to optimize the objective function J detailed in the formula Eq. (5):

$$J(U, V) = \sum_{i=1}^{N} \sum_{j=1}^{K} {\mu_{ij}}^m {d_{ij}}^2 \tag{5}$$

The FCM algorithm is given below. It can be ended if the objective function J can no longer be minimized or the change of the cluster centers position becomes very small.

Algorithm 4: FCM Algorithm

1: **Select** randomly K cluster centers.
2: $i = 1$
2: **Repeat**
3: Compute the fuzzy membership μ_{ij} using Eq. (3).
4: Calculate the fuzzy centers c_i using Eq. (4).
5: $i = i + 1$
6: **until** the minimum value J presented in Eq. (5) is reached or $\left\| U^{(i+1)} - U^{(i)} \right\| < \beta$, where:

- i is the iteration step,
- β the termination criterion belonging to [0, 1],
- $U = (\mu_{ij})_{N \times K}$ is the fuzzy membership matrix.

3.1.5 Gustafson–Kessel Algorithm

Gustafson and Kessel were the first to propose the Gustafson–Kessel fuzzy clustering (GK) algorithm [57,58]. This latter associates the cluster to its centroid and its covariance. Unlike the FCM algorithm that considers clusters to be spherical, this restriction does not apply on the GK algorithm that can identify ellipsoidal clusters [31,58].

The steps of the GK algorithm are given in Algorithm 5.

Algorithm 5: GK Algorithm

1: $i = 1$
2: **Repeat**
3: Calculate the clusters centroids positions based on Eq. (6).

$$c_i = \frac{\sum_{j=1}^{N} {\mu_{ij}}^m x_j}{\sum_{j=1}^{N} {\mu_{ij}}^m} \tag{6}$$

4: Compute the cluster covariance matrices using Eq. (7).

$$V_i = \frac{\sum_{j=1}^{N} (\mu_{ij})^m (x_j - c_i)(x_j - c_i)^T}{\sum_{j=1}^{N} {\mu_{ij}}^m} \tag{7}$$

5: Compute the distances using Eq. (8).

$${D_{ijA_i}}^2 = (x_j - c_i)^T \left[\rho_i det(V_i)^{\frac{1}{N}} {V_i}^{-1} (x_j - c_i) \right] \tag{8}$$

6: Update the partition matrix Eq. (9).

$$\mu_{ij} = \frac{1}{\sum_{l=1}^{k} \frac{D_{ij}}{D_{lj}} \left(\frac{1}{m-1} \right)} \tag{9}$$

7: $i = i + 1$

(Continued)

Algorithm 5 (Continued)

8: **until** the minimum value J presented in Eq. (10) is reached or $\left\| U^{(i+1)} - U^{(i)} \right\| < \beta$, where:

- β is the termination criterion between [0, 1].
- $U = \left(\mu_{ij}\right)_{N \times K}$ is the fuzzy membership matrix.

The objective function to be minimized in this algorithm is as given in Eq. (10).

$$J(X, U, V) = \sum_{i=1}^{K} \sum_{j=1}^{N} {\mu_{ij}}^{m} {D_{ij}}^{2} \tag{10}$$

3.1.6 Gath–Geva Algorithm

Gath and Geva generalize the highest likelihood estimate of the Fuzzy clustering. The clustering algorithm Fuzzy maximum likelihood estimates (FMLE) is based on the FMLE distance norm [31,58]. This norm distance decreases quicker than the one used in the GK algorithm. The difference between Gath–Geva (GG) and GK algorithms is that for GG the distance norm calculation includes an exponential term, which implies its faster decrease. This makes GG algorithm tend to the closest local value. This can be resolved using efficient initialization [31,58].

3.2 Metaheuristic Algorithms

Many metaheuristic algorithms exist in literature as Genetic Algorithms (GA) [25,60], Particle Swarm Optimization (PSO) [61] and Ant Lion Optimization (ALO) [10]. This section will briefly introduce the metaheuristic algorithms used in this work.

3.2.1 Genetic Algorithm

GA Principle the genetic algorithm (GA) is a search heuristic inspired by the theory of natural evolution of Charles Darwin [26].

It has three main operators: selection, crossover and mutation. A GA starts iteration with an initial population.

A GA process is initiated with a random set of individuals, considered as solutions to the problem to solve, called a Population. Each individual or solution is a chromosome identified by a set of joined parameters called Genes and is evaluated then assigned to a fitness value. In the selection procedure, some criterion is applied to select a certain number of strings, namely parents, from this population according to their fitness values. Strings with lower fitness values have more opportunities to be selected for reproduction in next step. This paper use a rank selection scheme that allows the control of the selection pressure, denoted s_p.

The most significant phase of the genetic algorithm is Crossover where the fittest individuals for reproduction are selected to create offspring of the next generation based on the process of natural selection [26,60]. This is an iterating process which ends to find a generation with the fittest individuals. The crossover probability of a selected individual to go through a crossover process is denoted p_c.

Some of the genes of the formed offspring can be subjected to a mutation with a low random probability p_m. It occurs to prevent premature convergence and maintain diversity within the population [25,59,60].

The algorithm generally terminates when the population does not produce offspring significantly different from the previous generation.

GA operates on a population (a number of potential solutions n_{pop}). The population at time t is represented by the time-dependent variable $S(t)$, with the initial population of random estimates being $S(0)$. Algorithm 6 shows the GA structure.

Algorithm 6: GA Algorithm

1: **Initialize** initial random population of n_{pop} elements.
2: $t = 1$
2: **Repeat**
3: Compute the fitness $S(t)$ of the chromosome and remember the highest fitness value.
4: Select the parents of the next generation from the generated population.
5: Crossover the parent chromosomes.
6: Mutate the chromosomes.
7: $i = i + 1$
7: **until** the end criterion is satisfied.

GA Based Clustering In the GA based clustering applications (Sections 4.4 and 5), the fitness $S(t)$ cited in Step 3 of the GA Algorithm corresponds to the optimal distance of the considered clustering algorithm. The GA end criterion is satisfied in this work when the maximum number of iterations is reached.

By using the previous clustering algorithms (defined in Section 3.1), we obtain the following GA based algorithms: GAKM, GAGlobalKM, GAPAM, GAFCM, GAGK and GAGG. The results of GA and clustering combinations are listed and analyzed in Section 5.

3.2.2 Particle Swarm Optimization

PSO Principle Particle swarm optimization (PSO) is a population based stochastic optimization process [61]. It was implemented effectively in many fields such as system control, function optimization, artificial neural network training and other areas.

A PSO process is initialized with a random population of solutions. The prospective solutions, called particles, move via the problem space by following the current optimal solutions [24,60,61].

Every particle has a fitness value to evaluate by the objective function, and a velocity which directs its flying. In every iteration, the particle swarm optimizer updates its velocity and position using the two best attributes: the best solution it has reached named P_{best} and the global best value named G_{best} as presented in Eqs. (11) and (12).

$$V_p = wV_p + c_1R_1\left(P_{best} - C_p\right) + c_2R_2\left(G_{best} - C_p\right) \tag{11}$$

$$C_p = C_p + V_p \tag{12}$$

where,

- C_p is the position of a particle and V_p its velocity.
- c_1 and c_2 are constants labeled as acceleration learning variables representing the weighting of stochastic terms pulling each particle to P_{best} and G_{best} positions respectively.
- R_1 and R_2 are obtained through a standardized distribution in the interval [0 1].

The inertia weight w is calculated for each iteration using Eq. (13).

$$w_t = 0.05 w_{t-1} w_{damp} \tag{13}$$

w_{damp} is the inertia weight damping ratio and t the current iteration.

Algorithm 7: PSO Algorithm

1: **Initialize** population.
2: **Repeat**
3: Evaluate individual fitness.
4: Update personal best P_{best}.
5: Update global best G_{best}.
6: Generate a new population.
7: Update velocity using the Eq. (11).
8: Update position using the Eq. (12).
9: **until** the end criterion is satisfied.

PSO Based Clustering in this work, the fitness cited in Step 3 of the PSO Algorithm corresponds to the optimal distance of each clustering algorithm defined in Section 3.1. The PSO end criterion is satisfied in this work when the maximum number of iterations is reached.

The results of PSO based clustering combinations giving the PSOKM, PSOGlobalKM, PSOPAM, PSOFCM, PSOGK and PSOGG algorithms are listed and analyzed in Section 5.

3.2.3 Ant Lion Optimization

Ant lion optimization (ALO) algorithm [64] is based on ant lions hunting mechanism. It consists on random walk exploration and random agent selection based on five main hunting steps: random walk of agents, building traps, entrapment of ants in the trap, catching prey and rebuilding traps. The ALO optimizer roulette wheel and random ants walks can eliminate local optima.

The random walk of ants is given by:

$$X(t) = [0, cumsum(2r(t_1) - 1), cumsum(2r(t_2) - 1), \ldots, cumsum(2r(t_n) - 1)] \tag{14}$$

where cumsum is calculating cumulative sum, n is maximum number of iterations and t is a step of the random walk

$$r(t) = \begin{cases} 1 & if\ rand > 0.5 \\ 0 & if\ rand < 0.5 \end{cases} \tag{15}$$

rand is a random number generator between [0, 1]. The random walk can be produced within the search space with Eq. (16).

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i^t - c_i^t)}{(b_i - a_i)} + c_i^t \tag{16}$$

b_i, a_i are the minimum and maximum values of random walk of i^{th} variable.

c_i^t, d_i^t are the minimum and maximum of i^{th} variable in the t^{th} iteration.

Roulette wheel is used to increase the probability for catching ants. It identifies the fittest ant lions.

The mathematical equations for trapping are given by Eqs. (17) and (18).

$$c_i^t = Antlion_j^t + c^t \tag{17}$$

$$d_i^t = Antlion_j^t + d^t \tag{18}$$

where c^t, d^t are respectively the minimum and the maximum of all variables at t^{th} iteration. c_i^t, d_i^t are respectively the minimum and the maximum of all variables for i^{th} ant. Antlion_j corresponds to the ant position of chosen ant lion.

Antlions shoot sand outward to push ants into them. It is possible to design the mathematical model for the above action:

$$C^t = \frac{c^t}{I} \tag{19}$$

$$D^t = \frac{d^t}{I} \tag{20}$$

where $I = 10^w \frac{t}{T}$, t is current iteration and T is the maximum number of iterations.

The final phase of ant lions hunting conduct is to catch an ant that enters the bottom of a pit and then the following equation must update its position to the recent situation.

$$\text{Antlion}_\text{j}^\text{t} = \text{Ant}_\text{i}^\text{t} \quad if\ \text{f}\left(\text{Ant}_\text{i}^\text{t}\right) > f\left(\text{Antlion}_\text{i}^\text{t}\right) \tag{21}$$

It is essential to keep the best solution in the evolution algorithm. This can be set up as:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{22}$$

where R_A^t is the random walk around the antlion selected by the roulette wheel at t^{th} iteration, R_E^t is the random walk around the elite at t^{th} iteration, and Ant_i^t indicates the position of i^{th} ant at t^{th} iteration.

The detailed ALO algorithm is described in Algorithm 8.

Algorithm 8: ALO Algorithm

1: **Initialize** randomly the first population of Ants and Antlions.
2: Calculate the fitness value of Ants and Antlions.
3: Find the best Antlion which has the optimal fitness and save it as the elite.
4: **Repeat**
5: **For each** Ant
6: **Select** an Antlion using Roulette wheel.
7: Update c and d using Eqs. (19) and (20).
8: Create a random walk and normalize it using Eqs. (14) and (15).

(Continued)

Algorithm 8 (Continued)

9: Update the position of ant using Eq. (21).
10: **End for**
11: Calculate the fitness of all Ants.
12: Replace an Antlion with its corresponding Ant_i^t if becomes fitter Eq. (22).
13: Update the elite if an Antlion has fitter value than it.
14: **until** the end criterion is satisfied.

3.3 Cluster Validation Indices and Performance Metrics

It is possible to use cluster validity indexes to validate the efficiency of a clustering method and evaluate its fitness of data partitions. Usually these effectiveness indicators are autonomous from clustering algorithms. Several cluster validity indexes were suggested in the literature for clustering algorithms. The partition coefficient (PC) was the first suggested cluster validity index. Subsequently, partition entropy (PE) was proposed as a normalization of PC. The separation coefficient (SC) was the first validity index to take the data geometrical properties in consideration.

3.3.1 Partition Coefficient (PC)

The partition coefficient, in Eq. (23), is described as the Frobenius norm of the membership matrix, divided by the data size [31].

$$PC(U) = \frac{1}{N}\sum_{i=1}^{k}\sum_{j=1}^{N}{\mu_{ij}}^2 \tag{23}$$

where μ_{ij} is the membership of data point j in cluster i. The optimum cluster number corresponds to the PC's highest value which shows the cluster's compactness. This value is between 0 and 1 [31].

3.3.2 Classification Entropy (CE)

Classification entropy, Eq. (24) [31], like the partition coefficient, measures the fuzziness of the cluster partition. They are both computed only using the membership matrix components.

$$CE(U) = \frac{-1}{N}\sum_{i=1}^{k}\sum_{j=1}^{N} log\left(\mu_{ij}\right) \tag{24}$$

The clustering is considered as efficient when PC index tends to 1 and CE index tends 0.

3.3.3 Partition Index (PI)

Partition index, in Eq. (25), is the proportion of clusters compactness sum in comparison with their segregation [31].

$$PI(U) = \sum_{i=1}^{k}\frac{\sum_{j=1}^{N}{\mu_{ij}}^m \left\|x_j - v_i\right\|^2}{N_i\sum_{l=1}^{k}\left\|x_l - v_i\right\|} \tag{25}$$

The partition is better when PI value is higher.

3.3.4 Separation Index (S)

On the opposite of the partition index (*PI*), the separation index *S* Eq. (26) uses a minimum-distance separation [31]. The partition is considered as optimal when separation index value *S* is small.

$$S(U) = \sum_{i=1}^{k} \frac{\sum_{j=1}^{N} {\mu_{ij}}^2 \left\| v_j - v_i \right\|^2}{N min_{i,l} \left\| v_l - v_i \right\|^2} \tag{26}$$

3.3.5 Xie-Beni Index (XB)

The Xie-Beni index is a fuzzy clustering validity measure that can be applied to crisp clustering. It is identified as the ratio between the quadratic error average and the minimum square distance between the clusters elements as given in Eq. (27) [31].

$$XB(U) = \frac{\sum_{i=1}^{k} \sum_{j=1}^{N} {\mu_{ij}}^m \left\| x_j - v_i \right\|^2}{N min_{i,j} \left\| x_j - v_i \right\|^2} \tag{27}$$

The optimum fuzzy partition is attained by reducing *XB* with respect to $c = 2, \ldots, c_{max}$.

3.3.6 Dunn's Index (DI)

Dunn proposed a crisp clustering efficiency index used for the identification of compact and strongly distinguished clusters. The Dunn's index *DI* is defined by Eq. (28) [31].

$$DI(U) = min_{i \in C} \left[min_{j \in c, i \neq j} \left\{ \frac{min_{x \in C_i, y \in C_j} d(x,y)}{max_{k \in C} \left\{ max_{x,y \in C} d(x,y) \right\}} \right\} \right] \tag{28}$$

where *d* is a distance function and C_i the set whose elements are allocated to the i^{th} cluster.

3.3.7 Alternative Dunn's Index (ADI)

Alternative Dunn's index is an improvement of the initial Dunn's index, Eq. (30). The computation becomes simpler when the dissimilarity function between two clusters ($min_{x \in C_i, y \in C_j}\ d(x,y)$) is evaluated in Beneath value by triangle-nonequality [31]:

$$d(x,y) \geq \left| d(y, v_j) - d(x, v_j) \right| \tag{29}$$

where v_j is the cluster center of the *j*th cluster.

$$ADI(U) = min_{j \in c, i \neq j} \left\{ \frac{min_{x \in C_i, y \in C_j} \left| d(y, v_j) - d(x_i, v_j) \right|}{max_{k \in C} \left\{ max_{x,y \in C} d(x,y) \right\}} \right\} \tag{30}$$

3.3.8 Efficiency and Purity

To measure the efficiency of the used algorithms, external criterions that evaluate the clustering matching of the actual classes are computed.

These performance parameters are the efficiency γ_i, and the purity β_i of classifications. They are calculated from the confusion matrix $N(N_{ij})$, (N_{ij} being the value of signals of genuine class C_i classified as class C_j) (Tab. 1).

Table 1: Confusion matrix

Actual class	Predicted class	
	H	B
H: N_1	N_{11}	N_{12}
B: N_2	N_{21}	N_{22}

For each class C_i, we have:

$$\gamma_i = \frac{N_{ii}}{N_i} \tag{31}$$

$$\beta_i = \frac{N_{ii}}{\sum_j N_{ji}} \tag{32}$$

The global performance parameters are then calculated as given in Eqs. (33) and (34).

$$\gamma = \frac{\sum_{i=1}^{2} N_i \gamma_i}{\sum_{i=1}^{2} N_i} \tag{33}$$

$$\beta = \frac{\sum_{i=1}^{2} N_i \beta_i}{\sum_{i=1}^{2} N_i} \tag{34}$$

4 Proposed Approach

As previously introduced, the purpose of this work is to improve the classification of events into Higgs Boson signal or background using the hybridization between fuzzy and hard semi-supervised clustering algorithms tuned with ALO. Towards this end, this work has been done based on 4 major processes:

- improved feature engineering approach,
- definition of the optimal number of clusters,
- new hybridization of fuzzy and hard clustering algorithms tuned with ALO and then
- PSO and GA based clustering used for a comparative analysis.

The details of each process are presented in the next sections.

4.1 Proposed Features Engineering Approach

4.1.1 Features Ranking Using T-Statistic Method and PSOKM

This feature selection method utilizes *t*-Statistic [80,81] where each element can be categorized either into class of signals C_1 or to class of backgrounds C_2. For each feature F_i, PSOKM (KM algorithm optimized using PSO) is applied on the data set (executed 50 times) to define the two clusters and calculate *t*-Statistic as in Eq. (35).

$$t(F_i) = \frac{|\mu_{i1} - \mu_{i2}|}{\sqrt{\frac{{\sigma_{i1}}^2}{n_1} + \frac{{\sigma_{i2}}^2}{n_2}}} \tag{35}$$

where μ_{ij} and σ_{ij} denote respectively the mean and the standard deviation of i^{th} feature F_i for class C_j, $j = \{1, 2\}$, using PSOKM algorithm.

The P value is described as the likelihood of achieving a result equivalent to or better the observed one. When the p value tends to 1, no distinction is suggested between groups other than by chance. If the p value is close to 0, the observed distinction is unlikely due to chance [81].

The t-statistic values and p-value are reported in Tab. 2 in an ascending order of the p-value.

Table 2: Ascendant ranking of features based on the T-statistic method

Attributes	T-statistic	p-value
F_1	Inf	0.0000
F_{18}	1.6893	0.0956
F_{23}	1.6524	0.1029
F_{15}	−1.2478	0.2163
F_{24}	0.9466	0.3471
F_{30}	0.8054	0.4233
F_{17}	0.8054	0.4233
F_{14}	0.8054	0.4233
F_{10}	−0.8054	0.4233
F_8	−0.7647	0.4470
F_{12}	0.6476	0.5194
F_5	−0.5639	0.5747
F_{19}	−0.5585	0.5783
F_6	0.5189	0.6054
F_9	0.4959	0.6215
F_{21}	−0.4574	0.6488
F_{28}	0.3999	0.6904
F_{13}	0.3904	0.6975
F_4	−0.3828	0.7030
F_3	0.3276	0.7442
F_{16}	−0.2220	0.8250
F_{26}	−0.2241	0.8233
F_{27}	0.2113	0.8333
F_{29}	−0.2056	0.8377
F_7	0.1593	0.8739
F_{22}	−0.1470	0.8836
F_{20}	0.1282	0.8984
F_{25}	0.0830	0.9341
F_2	0.0462	0.9633
F_{11}	0.0231	0.9871

4.1.2 Highly Correlated Features Exclusion Using SOM Technique

From Tab. 2 and Fig. 1, the second step excludes the features that are highly correlated and allows having diversified discriminant information about events. In order to identify the correlation between these ranked features, the Self-Organizing Feature Map (SOM) technique [82–90] is used, by plotting the weight planes that shows the values in each map unit for each variable.

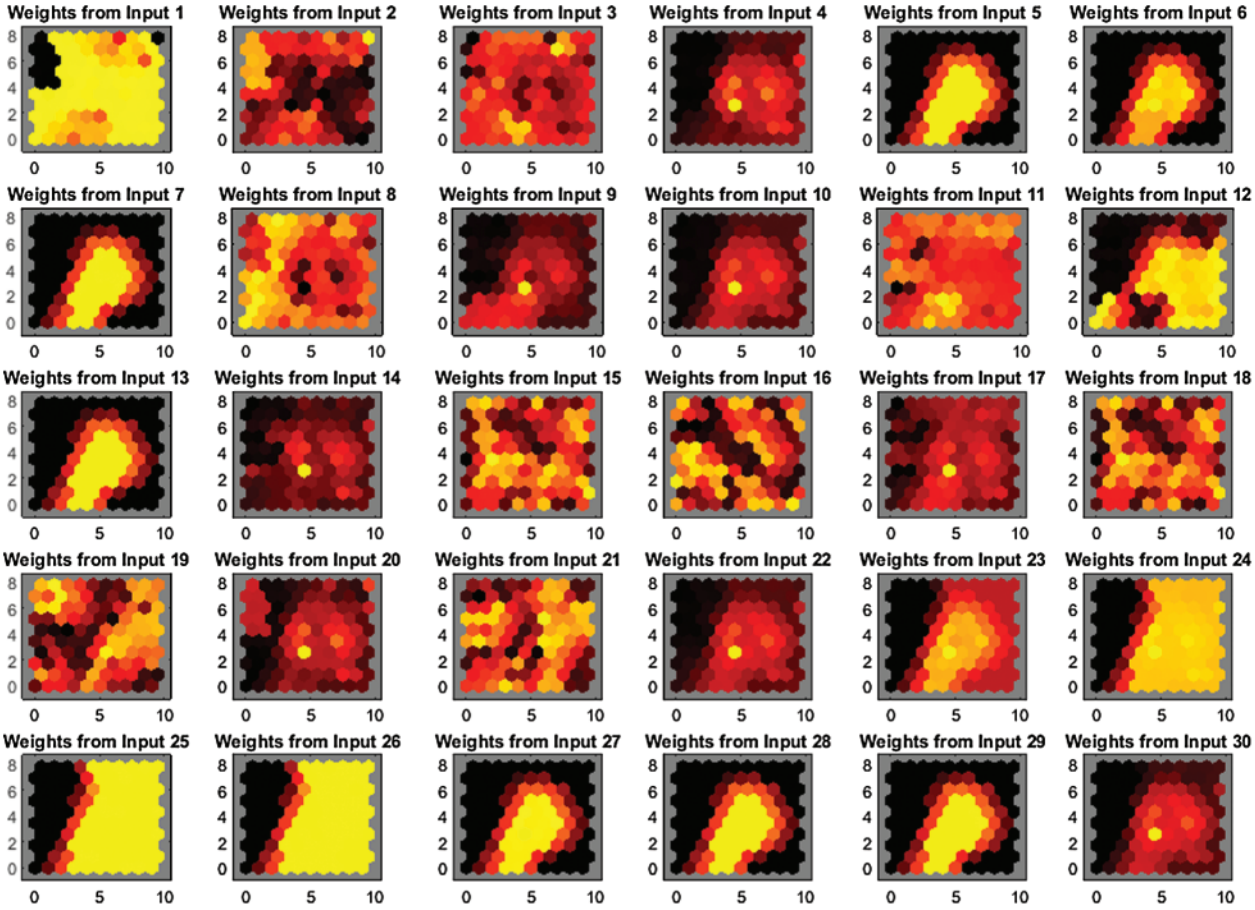

Figure 1: Features weight planes using SOM

The SOM technique is a non-linear mapping algorithm that aims to achieve a low-dimensional representation (usually 2D or 3D) of a set of points dispersed in high-dimensional pattern space. It enables the Euclidean distances between the points on the map to be as close as possible to the Euclidean distances in the high-dimensional pattern space between the respective points [82].

Fig. 1 presents the differences regarding the input variables. Lighter and darker colors represent respectively larger and smaller weights associated to each feature. If the patterns of two inputs are very close, they can be claimed that they are strongly correlated [83].

Then, the sets of features which are highly correlated and then very similar are:

—F_5, F_6, F_7, F_{13}, F_{27}, F_{28} and F_{29}.
—F_{24}, F_{25} and F_{26}.
—F_4, F_9 and F_{10}.
—F_{14} and F_{20}.

Using these sets and the feature ranking done in Tab. 2, only the first feature of each set is kept. The other similar features are excluded. The most significant features list for classification

in a descendant order is then: F_1, F_{18}, F_{23}, F_{15}, F_{24}, F_{30}, F_{17}, F_{14}, F_{10}, F_8, F_{12}, F_5, F_{19}, F_{21}, F_3, F_{16}, F_{22}, F_2, F_{11}.

The SOM parameters tuning that had given the most efficient results corresponds to a dimension of 20×20 and a number of epoch equal to 200.

4.1.3 Features Dimensionality Reduction Using PSOKM

The main idea of this section is to combine the softly correlated features, which leads to a reduction of the dimensionality of the feature space. Having this dimensionality as low as possible is very important to reduce exponentially the density of the dataset since the volume of the feature space grows exponentially with each dimension. In addition, the computing time of algorithms grows strongly with the number of dimensions.

The To define the best number of features to use from the list of attributes in the previous section, the PSOKM algorithm is used. An experiment was conducted on the dataset to evaluate the efficiency γ_i, and the purity β_i of the classification for each batch of features results are reported in Tab. 3.

Table 3: Efficiency and standard deviation using PSOKM with respect of features number

Features number	4	5	6	7	8	9	10	11
Accuracies average γ (%)	60.44	69.87	69.76	69.84	71.08	70.35	**73.92**	72.66
Standard deviation σ(%)	3.35	1.44	1.97	1.36	1.58	0.69	2.67	2.46
Features number	12	13	14	15	16	17	18	19
Accuracies average γ (%)	64.88	62.11	65.94	64.88	65.52	71.08	64.87	63.19
Standard deviation σ(%)	1.33	2.12	1.51	1.33	1.98	1.58	3.88	2.67

The PSOKM algorithm is executed 50 times. The averages of the experimental results of the dimensionality reduction are shown in Tab. 3. Each column of this table corresponds to a number of features taken from the first lines of Tab. 2, in the descendant order of their efficiency. The combination of the features that gives better efficiency corresponds to the first 10 ones from Tab. 2.

This work will then use the 10 features that was proven to be the most informative (F_1, F_{18}, F_{23}, F_{15}, F_{24}, F_{30}, F_{17}, F_{14}, F_{10}, F_8) as the system input matrix, to which an efficient comparative classification will be applied.

4.2 The Optimal Number of Clusters: Elbow Criterion

To find the optimal number of clusters, this work used the elbow criterion [63]. It consists on applying clustering on the data for different number of clusters and validates the correctness of the obtained results. The best number of clusters corresponds to the last one that adds sufficient information.

To answer this, the most informative feature values for the Higgs Boson signals, as defined in Section 4.1.3, are used as input for the clustering algorithms K-medoid (PAM) and Fuzzy c-means (FCM).

To validate the efficiency of these clustering methods and evaluate the data partitions fitness in each iteration, some common cluster validity indexes, as *PC, CE, PI, S, XB, DI*, and *ADI* defined in Subsection 3.3, are used.

There are several runs carried out with a distinct number of clusters ranging from 2 to 7.

Tab. 4 presents the validation indices values calculated for each run using the hard K-medoids (PAM) algorithm. Each run corresponds to a specific number of clusters (between 2 and 7 clusters). Tab. 5 shows the results found using FCM algorithm.

Table 4: Validation indices for PAM algorithm

Number of clusters	PC	CE	PI	S	XB	DI	ADI
2	–	–	**6.1221**	$6.1221.10^{-5}$	–	0.0065	0.0289
3	–	–	0.3482	$4.2423.10^{-6}$	–	0.0114	$3.0126.10^{-4}$
4	–	–	0.3874	$4.9631.10^{-6}$	–	0.0122	$3.1035.10^{-4}$
5	–	–	0.4431	$5.7112.10^{-6}$	–	0.0096	$3.0985.10^{-4}$
6	–	–	0.2793	$3.5406.10^{-6}$	–	0.0077	0.0015
7	–	–	0.2178	$2.4628.10^{-6}$	–	0.0073	$2.8420.10^{-5}$

Table 5: Validation indices for FCM algorithm

Number of clusters	PC	CE	PI	S	XB	DI	ADI
2	**0.8688**	**0.2472**	0.4629	$4.6286.10^{-6}$	**7.0395**	**0.6855**	**0.0357**
3	0.6707	0.5793	**0.5067**	$\mathbf{5.9964.10^{-6}}$	3.5061	0.0099	$8.9665.10^{-5}$
4	0.5674	0.7767	0.3968	$3.9794.10^{-6}$	3.0779	0.0078	0.0022
5	0.4828	0.9786	0.4058	$4.7727.10^{-6}$	2.6825	0.0078	$3.2168.10^{-4}$
6	0.4668	1.0605	0.3595	$3.6755.10^{-6}$	3.2977	0.0079	0.0011
7	0.4289	1.1836	0.3535	$3.9551.10^{-6}$	2.9973	0.0080	0.0012

PC and CE are not applicable for PAM since it is a hard clustering method [31]. In Tab. 4, PI and S using PAM reveals that the number of clusters could be defined to 2 and 3 respectively. XB index is infinity (Inf) for PAM as reported in Tab. 4. This reflects that an overflow happened leading to a too large to represent as a conventional floating-point value. All XB index values were equal to infinity. It may be caused by an initialization problem of the hard PAM clustering. The plots of Dunn's and alternative Dunn's indexes in Tab. 4 confirm that the adequate clusters number should be equal to 4 and 2 clusters respectively. According to this analysis, 2 clusters achieve the highest information partitioning of data. According to the FCM results of Tab. 5, PC and CE indexes reach the values 1 and 0 (respectively) when the number of clusters is rated to 2. This means that when the number of clusters is equal to 2, the FCM based clustering is efficient. The FCM results in Tab. 5 give more information about the optimal number of clusters using PI and S indexes where the local minimum is reached when the number of clusters is 3. For the XB index using FCM method, it is difficult to find the optimum number of clusters (Tab. 5). Either

3 and 6 can be seen as an elbow. Dunn's and the alternative Dunn's indexes have an elbow when the number of clusters is equal to 2. The hard PAM and the FCM algorithms based experimental results shown that the elbow corresponds to 2 clusters.

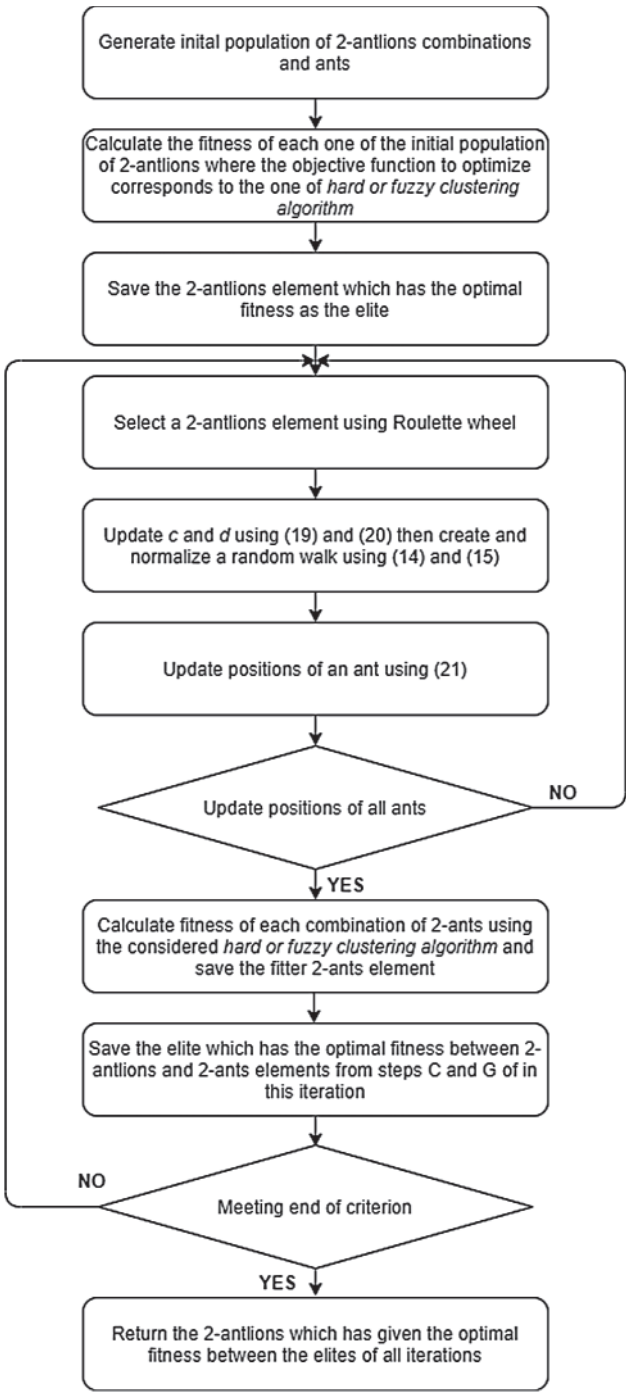

Figure 2: Scheme of the proposed hybrid clustering tuned with ALO giving ALOKM, ALOGlobalKM, ALOPAM, ALOFCM, ALOGK and ALOGG algorithms

4.3 Proposed Hybrid Clustering Tuned with ALO

Since ALO have the advantages of fast convergence, local optima avoidance, high efficiency and improved exploration space and in order to enhance clustering ability to find global minimum, a hybrid clustering approach is proposed. Fig. 2 presents the flowchart of proposed algorithms.

In the proposed approach, the objective function to minimize corresponds to the clustering algorithms used: KM, GlobalKM, PAM, FCM, GK and GG. Each one of the considered clustering algorithms is hybridized and tuned with the Ant Lion Optimizer giving respectively the ALOKM, ALOGlobalKM, ALOPAM, ALOFCM, ALOGK and ALOGG Algorithms.

Based on Section 4.2, the number of clusters to find for the considered data set is 2 clusters. The aim of clustering in our case is to assign the Higgs data points to two clusters where each one groups the data points with similar characteristics. The 10 most informative features selected in Section 4.1.3 are used.

Each cluster, in our approach, is considered as an Antlion and the elite of the ALO algorithm is a matrix of 2-anlions that optimizes the most the fuzzy and hard clustering algorithms.

In the first phase of the proposed approach, the best 2-anlions solution which optimizes the most the considered hard or fuzzy clustering algorithm is defined. Then, the positions of the initial population of ants are updated based on that. Thereafter, the fitness of each 2-ants is calculated. The fitness of the fitter 2-ants element is compared to the elite in order to return the best 2-clusters element. Many iterations are carried out in order to return the best results. The details of the proposed hybrid clustering tuned with ALO are presented in Algorithm 9.

Algorithm 9: Proposed ALO Based Algorithms

Initialize randomly the first population of 2-antlions combinations and ants.
Calculate the fitness value of the 2-antlions elements using the considered hard or fuzzy clustering algorithm in Section 3.1 as the objective function to optimize.
Find the best 2-antlions which has the optimal fitness and save it as the elite.
Repeat
For each ant
Select a 2-antlions element using Roulette wheel.
Update c and d using Eqs. (19) and (20).
Create a random walk and normalize it using Eqs. (14) and (15).
Update the position of ant using Eq. (21).
End for
Repeat
For each ant_1
Repeat
For each ant_2 different from ant_1
Calculate the fitness of the considered 2-ants using the considered hard or fuzzy clustering algorithm in Section 3.1.

(Continued)

Algorithm 9 (Continued)

16: **End for**
17: **End for**
18: Define the fitter 2-ants element.
19: Update the elite (defined in Step 3) if the 2-ants element (found in Step 18) has fitter value than it.
20: **until** the end criterion is satisfied.

4.4 Hybrid Clustering Combinations with GA and PSO

The performance of the proposed approach defined in Section 4.3 was compared to the hybridization of GA and PSO with the same considered clustering algorithms (KM, GlobalKM, PAM, FCM, GK and GG) giving respectively the GAKM, GAGlobalKM, GAPAM, GAFCM, GAGK, GAGG, PSOKM, PSOGlobalKM, PSOPAM, PSOFCM, PSOGK and PSOGG algorithms. These GA and PSO clustering combinations were used and applied to the field of Higgs boson for the first time in this work.

The 10 most informative features selected in Section 4.1.3 are used as an input to those algorithms in order to assign the Higgs data points to two groups with similar characteristics.

The Tab. 6 summarizes the offline parameter tuning of GA and PSO that gives the best efficiency applied to the considered clustering algorithm combinations.

Table 6: GA and PSO algorithms parameters tuning

Algorithm	Parameters tuning
GA	Number of population elements $(n_{pop}) = 100$
	Crossover probability $p_c = 0.8$
	Mutation probability $p_m = 0.3$
	Selection pressure $S_p = 8$
	Number of off springs $= 2 \times round(p_c \times n_{pop}/2)$
	Number of mutants $= round(p_m \times n_{pop})$
PSO	Number of population elements $(n_{pop}) = 100$
	Inertia weight $w = 0.72$
	Inertia weight damping ratio $w_{damp} = 0.99$
	Personal learning coefficient $c_1 = 1.49$
	Global learning coefficient $c_2 = 1.49$
	Lower bound velocity $(Vel_{Min}) = 0.1(Var_{Max} - Var_{Min})$
	Upper bound velocity $(Vel_{Max}) = -Vel_{Min}$

5 Results and Discussions

In this work, we have proposed a new approach for hybridization of both hard and fuzzy clustering tuned with ALO (Section 4.3) applied to the Higgs boson search and considering a selection of 10 most informative features (Section 4.1). The considered hard and fuzzy clustering algorithms (Section 3.1) as KM, GlobalKM, PAM, FCM, GG and GK are used as the objective function to optimize. The ALO based tuning leads to six algorithms: ALOKM, ALOGlobalKM, ALOPAM, ALOFCM, ALOGG and ALOGK.

Table 7: Confusion matrix obtained using clustering methods (KM, GlobalKM, PAM, FC, GG, GK) and ALO, GA and PSO based clustering method

		Predicted class					
	Method	KM		GlobalKM		PAM	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	**34170**	0	**34170**	0
	B: 65830	37379	28451	28584	37246	37381	28449
	Method	FCM		GG		GK	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	31548	2622	33074	1096
	B: 65830	26648	39182	14606	51224	14602	51228
	Method	ALOKM		ALOGlobalKM		ALOPAM	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	**34170**	0	**34170**	0
	B: 65830	25437	40393	24647	41183	29914	35916
	Method	ALOFCM		ALOGG		ALOGK	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	26088	8082	32893	1277
	B: 65830	20131	45699	7879	57951	8152	57678
	Method	GAKM		GAGlobalKM		GAPAM	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	**34170**	0	**34170**	0
	B: 65830	37383	28447	37383	28447	25437	40393
	Method	GAFCM		GAGG		GAGK	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	33664	506	33074	1096
	B: 65830	16748	49082	20842	44988	16493	49337
	Method	PSOKM		PSOGlobalKM		PSOPAM	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	**34170**	0	**34170**	0
	B: 65830	22317	43513	22317	43513	29914	35916
	Method	PSOFCM		PSOGG		PSOGK	
		H	B	H	B	H	B
Actual class	H: 34170	**34170**	0	28923	5247	32893	1277
	B: 65830	16748	49082	11576	54254	16144	49686

The results of this hybridization are compared with some evolutionary well-known stochastic methods as GA and PSO, especially that the field of Higgs boson is poor in term of meta-heuristic point of view analysis. Notice that, the hybridization of the six clustering methods

with GA and PSO led to the following algorithms: GAKM, GAGlobalKM, GAPAM, GAFCM, GAGG, GAGK and PSOKM, PSOGlobalKM, PSOPAM, PSOFCM, PSOGG and PSOGK.

The offline tuning parameters used for each clustering technique are listed in Sections 4.3.2 and 4.4.

The computation code of the different analyzes was compiled under the Matlab environment (R2017b) on an Intel Core i5 2,6 GHz processor, with 8 GB of RAM.

Tab. 7 reports the most efficient confusion matrix found in 50 runs executed for each one of the improved algorithm combinations used in this work. It is based on event's labels of the Higgs Machine Learning dataset used. According to this feature label, the considered 100000 events in the learning phase contain $N_1 = 34170$ of signals (class H) and $N_2 = 65830$ backgrounds (class B). Based on the confusion matrix, we can qualify a method as better that the others, as a part of a first layer analysis, when N_{11} (the number of signals found using the unsupervised clustering) tend to N_1 and N_{22} (the number of backgrounds found using the unsupervised clustering) tend to N_2.

Table 8: Efficiencies γ, purities rates β and computational time using clustering methods (KM, GlobalKM, PAM, FCM, GG, GK) and combined to ALO, GA and PSO (validation data)

Method	γ	β	Time ± standard deviation (min:s:ms)
KM	71.61 ± 3.36	73.87 ± 2.59	**00:00:21 ± 00:00:03**
GlobalKM	78.29 ± 1.42	77.22 ± 1.98	09:07:05 ± 00:02:33
PAM	71.77 ± 2.77	73.87 ± 1.76	00:02:08 ± 00:00:54
FCM	79.76 ± 0.33	78.09 ± 2.54	00:00:21 ± 00:00:04
GG	85.07 ± 0.76	81.74 ± 0.65	00:08:00 ± 00:00:28
GK	85.87 ± 1.56	83.63 ± 1.43	00:01:02 ± 00:00:07
ALOKM	80.68 ± 2.89	78.66 ± 2.66	04:06:19 ± 00:00:36
ALOGlobalKM	81.28 ± 3.45	79.04 ± 1.76	04:24:30 ± 00:00:18
ALOPAM	77.28 ± 1.56	76.66 ± 2.87	03:07:07 ± 00:00:08
ALOFCM	84.71 ± 2.65	81.46 ± 3.88	23:20:31 ± 01:21:49
ALOGG	82.19 ± 2.56	82.28 ± 2.59	28:18:31 ± 02:54:27
ALOGK	**91.94 ± 2.66**	**89.92 ± 5.67**	26:07:31 ± 01:58:37
GAKM	71.96 ± 2.32	73.87 ± 1.49	04:04:30 ± 00:31:43
GAGlobalKM	71.96 ± 3.65	73.87 ± 1.49	13:33:55 ± 00:29:48
GAPAM	80.68 ± 2.65	78.66 ± 1.87	04:06:19 ± 00:36:07
GAFCM	87.28 ± 1.12	83.56 ± 0.45	04:24:30 ± 00:24:37
GAGG	83.43 ± 3.29	80.33 ± 1.42	07:35:00 ± 00:53:43
GAGK	85.87 ± 4.53	82.28 ± 0.34	07:14:33 ± 00:51:09
PSOKM	83.05 ± 2.23	80.25 ± 2.54	02:59:08 ± 00:03:35
PSOGlobalKM	83.05 ± 1.34	80.25 ± 3.67	11:02:45 ± 00:01:52
PSOPAM	77.28 ± 0.33	76.66 ± 2.54	03:07:08 ± 00:00:43
PSOFCM	87.28 ± 2.38	83.56 ± 1.44	03:21:05 ± 00:00:51
PSOGG	83.52 ± 1.03	81.29 ± 2.67	07:47:01 ± 00:01:04
PSOGK	85.87 ± 2.37	82.28 ± 1.54	07:34:33 ± 00:00:22

In order to validate this first layer analysis and based on Eqs. (33) and (34), we are listing in Tab. 8 efficiency and purity performance average values and standard deviations found in the 50 runs done to each clustering algorithm combination. Based on this second level analysis, it can be clearly seen that the results found tend globally in favor to hybridization of fuzzy clustering (either using Fuzzy c-means, Gath–Geva or Gustafson–Kessel) tuned with ALO, GA and PSO). There is no specific fuzzy based clustering algorithm that worked best in all these combinations. The efficiencies of ALOGK, GA based FC-means and PSO based FC-means are the best. Moreover, based on Tab. 8, we can confirm the conclusion of the first layer analysis based on the confusion matrix (Tab. 7) that the ALOGK clustering is the most performing method used since it corresponds to the higher efficiency equal to 91.94% (with a standard deviation of ±2.66% in the 50 runs) and the higher purity equal to 89.92% (with a standard deviation of ±5.67% in the 100 runs).

Table 9.1: Cluster validation indices (Part 1) using clustering methods (KM, GlobalKM, PAM, FCM, GG, GK) and combined to ALO, GA and PSO (validation data)

Methods	PC	CE	PI	S
KM	–	–	1.012966 ± 0.556287	0.000010 ± 0
GlobalKM	–	–	2.213197 ± 0.778281	0.000022 ± 0.000001
PAM	–	–	1.024450 ± 0.077615	0.000010 ± 0
FCM	0.817041 ± 0.187297	0.302857 ± 0.006421	1.478514 ± 0.078261	0.000015 ± 0
GG	0.699337 ± 0.686366	0.438178 ± 0.055266	1.038487 ± 0.067528	0.000013 ± 0
GK	0.709337 ± 0.056366	0.448178 ± 0.065266	1.478487 ± 0.097528	0.000015 ± 0
ALOKM	–	–	0.967642 ± 0.089626	0.000009 ± 0.000001
ALOGlobalKM	0.799071 ± 0.196245	0.292667 ± 0.009987	1.441478 ± 0.086564	0.000012 ± 0.000003
ALOPAM	–	–	13.43907 ± 0.983766	0.000178 ± 0.000009
ALOFCM	0.839143 ± 0.076243	**0.273270 ± 0.099896**	**0.000000 ± 0**	**0.000000 ± 0**
ALOGG	0.849987 ± 0.056988	**0.273270 ± 0.006528**	**0.000000 ± 0**	**0.000000 ± 0**
ALOGK	0.845041 ± 0.036148	**0.273270 ± 0.006528**	**0.000000 ± 0**	**0.000000 ± 0**
GAKM	–	–	0.943431 ± 0.026753	0.000009 ± 0
GAGlobalKM	–	–	0.942978 ± 0.086257	0.000009 ± 0
GAPAM	–	–	0.967642 ± 0.097256	0.000009 ± 0
GAFCM	0.799071 ± 0.098682	0.292667 ± 0.026287	1.441478 ± 0.862758	0.000012 ± 0
GAGG	0.780562 ± 0.062683	0.279626 ± 0.007627	1.332875 ± 0.944264	0.000012 ± 0
GAGK	0.790562 ± 0.162683	0.289626 ± 0.008627	1.432875 ± 0.986264	0.000012 ± 0
PSOKM	–	–	13.43573 ± 0.862822	0.000134 ± 0.000007
PSOGlobalKM	–	–	13.43573 ± 0.086286	0.000134 ± 0.000007
PSOPAM	–	–	13.43907 ± 0.560007	0.000178 ± 0.000008
PSOFCM	0.799071 ± 0.098652	0.292667 ± 0.009828	1.441478 ± 0.000866	0.000012 ± 0
PSOGG	0.796543 ± 0.108257	0.279006 ± 0.016222	1.442875 ± 0.065765	0.000012 ± 0
PSOGK	0.790562 ± 0.098257	0.289626 ± 0.008222	1.432875 ± 0.055765	0.000012 ± 0

Tabs. 9.1 and 9.2 list the most common clustering validation indices as Partition Coefficient (PC), Classification Entropy (CE), Partition Index (PI), Separation Index (S), Xie-Beni Index (XB), Dunn's Index (DI) and Alternative Dunn's Index (ADI) applied to the considered data and calculated for each algorithm combination used. The *PC* and *CE* indexes are useless for the hard clustering. This includes that for all hard clustering algorithm combinations, the *PC* values are equal to 1 and the *CE* ones are not a number. Applied to the fuzzy clustering, the algorithm is efficient when the *PC* value tends to 1 and the *CE* value to 0. According to Tabs. 9.1 and 9.2,

this rule is validated whether we use the ALOGK or ALOFCM clustering. This means that these two methods are compact, belong to the crisp classification and produce clusters with less overlap. The minimal values of *PI* and *S* correspond to the ALOFCM, ALOGG or ALOGK. This means that they are the strongest algorithms based on the noise criterion and the separation of clusters. The GG basic clustering algorithm is the most compact and gives the best separated clusters according to the *XB* index. Both *DI* and *ADI* indexes tend toward the novel ALOFCM, ALOGK and ALOGG clustering algorithm. Using this third layer analysis based on the clustering validation indices, we can confirm that fuzzy based combinations are giving better results than hard ones, especially the combinations based on the ALO method.

Table 9.2: Cluster validation indices (Part 2) using clustering methods (KM, GlobalKM, PAM, FCM, GG, GK) and combined to ALO, GA and PSO (validation data)

Methods	XB	DI	ADI
KM	13.41202 ± 0.007782	0.002806 ± 0.000754	0.015605 ± 0.000077
GlobalKM	–	0.006039 ± 0.000662	0.015315 ± 0.000662
PAM	–	0.003243 ± 0.000785	0.016160 ± 0.007520
FCM	5.842667 ± 0.099726	0.003460 ± 0.000737	0.016367 ± 0.000752
GG	0.004055 ± 0.000772	0.004055 ± 0.000726	0.000819 ± 0.000076
GK	6.191479 ± 0.972661	0.002781 ± 0.000765	0.061754 ± 0.000727
ALOKM	–	0.004197 ± 0.000775	0.017003 ± 0.000672
ALOGlobalKM	5.842667 ± 0.426577	0.003460 ± 0.000587	0.016367 ± 0.000989
ALOPAM	22.13623 ± 1.887298	0.003722± 0.000862	0.037237± 0.008972
ALOFCM	28.45823 ± 2.919876	0.000624 ± 0.000068	**0.000002 ± 0**
ALOGG	28.45823 ± 1.979382	0.000624 ± 0.000068	**0.000002 ± 0**
ALOGK	28.45823 ± 0.986378	0.000624 ± 0.000068	**0.000002 ± 0**
GAKM	10.25958 ± 0.000826	0.003489 ± 0.000166	0.016406 ± 0.007862
GAGlobalKM	10.20232 ± 0.247675	0.003489 ± 0.000863	0.016402 ± 0.008962
GAPAM	–	0.004197 ± 0.000762	0.017003 ± 0.000726
GAFCM	5.842667 ± 0.007623	0.003460 ± 0.000024	0.016367 ± 0.000245
GAGG	5.916631 ± 0.986245	0.013560 ± 0.008762	0.016344 ± 0.000862
GAGK	5.916007 ± 0.775528	0.013300 ± 0.000752	0.016254 ± 0.000785
PSOKM	22.13412 ± 1.008987	0.003543 ± 0.000077	0.036947 ± 0.000268
PSOGlobalKM	22.13412 ± 0.076722	0.003543 ± 0.000072	0.036947 ± 0.000173
PSOPAM	22.13623 ± 0.000825	0.003722 ± 0.000008	0.037237 ± 0.000726
PSOFCM	5.842667 ± 0.997265	0.003460 ± 0.000087	0.016367 ± 0.000135
PSOGG	5.916631 ± 0.527517	0.013560 ± 0.000627	0.016344 ± 0.000820
PSOGK	5.916002 ± 0.277381	0.013300 ± 0.007273	0.016254 ± 0.007276

The execution times of the considered approaches are gathered in Tab. 8. The time spent using the hard clustering, especially the KM algorithm, is lower than the one using fuzzy clustering. This is due to the fact that we are using a large set of data and that this kind of clustering is the most suitable for this case.

As another angle of analysis, we are measuring the variation of the clustering cost by iterations for each algorithm combination which corresponds to the sum of the distances between each data point and its closest cluster centroid. The results are shown in the Figs. 3–6. According to these figures, the optimal cost is reached respectively using GK, ALOGK, GAGK and PSOGK. Again, we can conclude, as a part of comparison between hard and fuzzy clustering, that the

combinations done using the fuzzy clustering have given better results. It can also be concluded that the method that optimized the most the clustering cost was ALOGK algorithm. In term of fast convergence, we can see using Figs. 3–6 that the combinations with KM and PAM algorithms were faster to converge to their optimal costs. But, these optimal costs were at least doubly bigger than the optimal cost of each combination with GK algorithm.

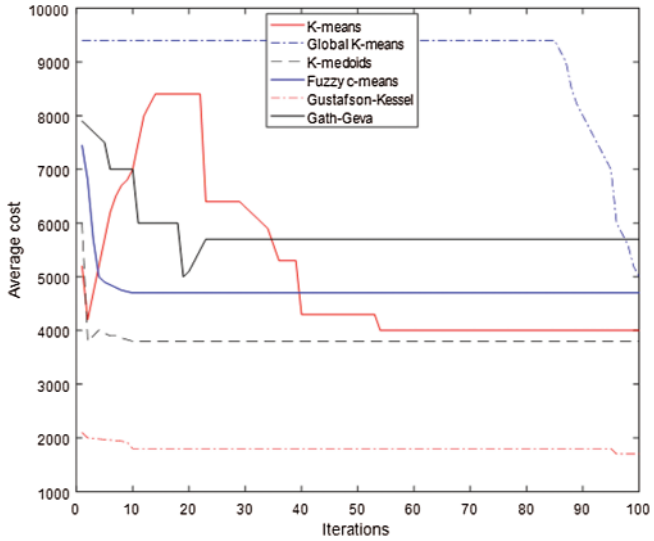

Figure 3: Average cost variation *vs*. iterations number for basic clustering KM, GlobalKM, PAM, FCM, GK and GG algorithms

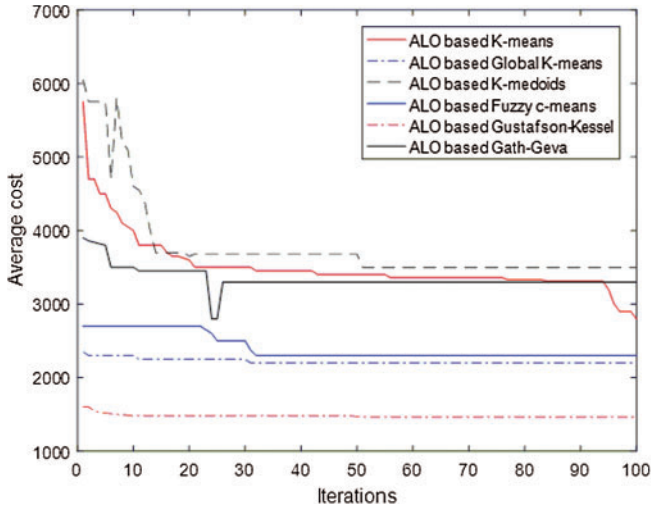

Figure 4: Average cost variation *vs*. iterations number for ALOKM, ALOGlobalKM, ALOPAM, ALOFCM, ALOGK and ALOGG algorithms

In another level of analysis, Figs. 7–10 present the Higgs Boson search clustering distribution using Sammon visualization [72], into signals and backgrounds based on the algorithm combinations that optimize the most the clustering cost according to the last analysis (GK, the

proposed ALOGK, GAGK, and PSOGK). The Sammon mapping used is based on the two most discriminant features obtained in the Section 4.1.3. This corresponds to the features F_1 and F_{18}. In Figs. 7–10, the distance measure of the cluster prototype is transformed into Euclidean distance in the projected two dimensional spaces where each cluster centroid is represented by a single red star. When the cluster is properly selected, the projected data fall close to the projected cluster center in an approximately spherically distributed cluster (shown as blue stars). As it can be seen in these figures, the data points lie much closer to the center of the cluster when the ALOGK algorithm is used.

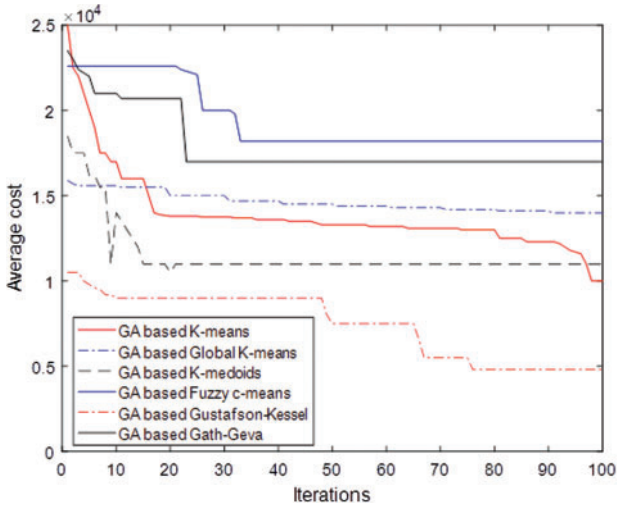

Figure 5: Average cost variation *vs*. iterations number for GAKM, GAGlobalKM, GAPAM, GAFCM, GAGK and GAGG algorithms

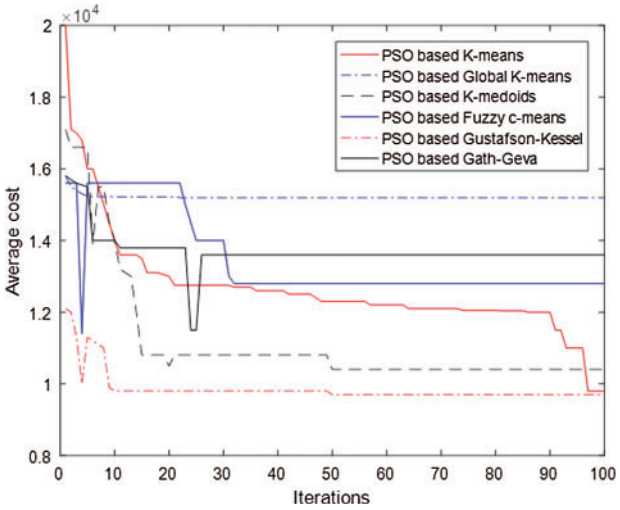

Figure 6: Average cost variation *vs*. iterations number for PSOKM, PSOGlobalKM, PSOPAM, PSOFCM, PSOGK and PSOGG algorithms

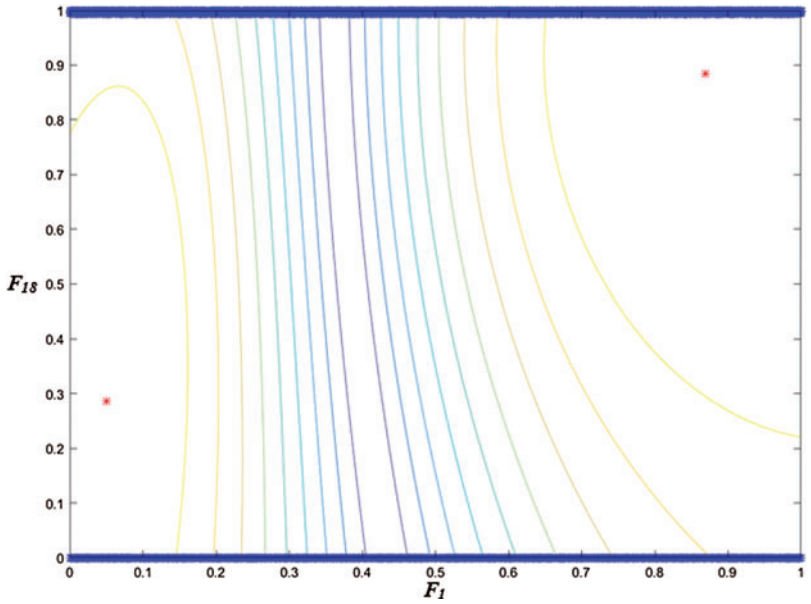

Figure 7: Higgs boson dataset Sammon mapping visualization into signal and background classes of GK algorithm

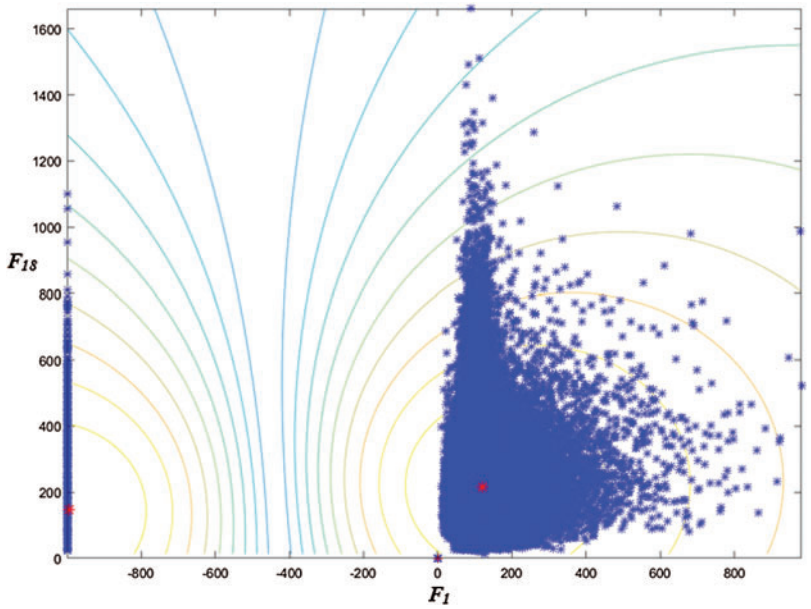

Figure 8: Higgs boson dataset Sammon mapping visualization into signal and background classes of ALOGK algorithm

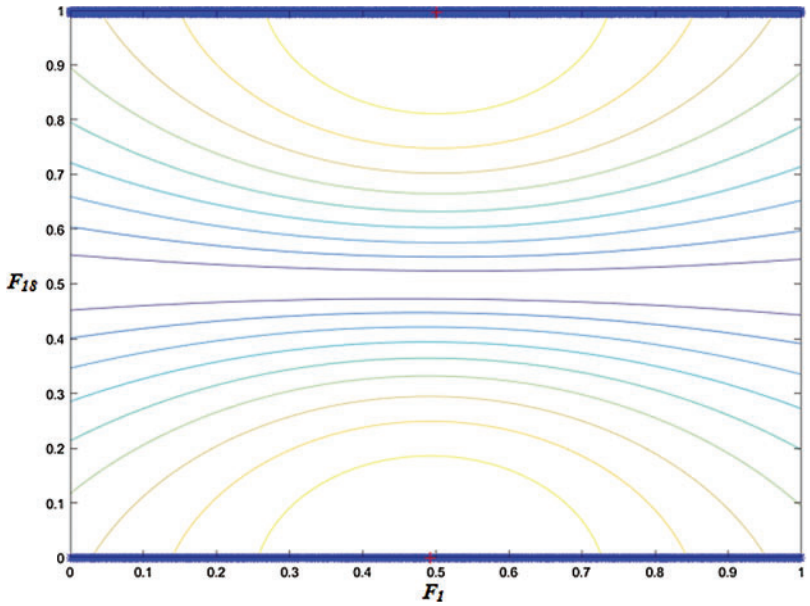

Figure 9: Higgs boson dataset Sammon mapping visualization into signal and background classes of GAGK algorithm

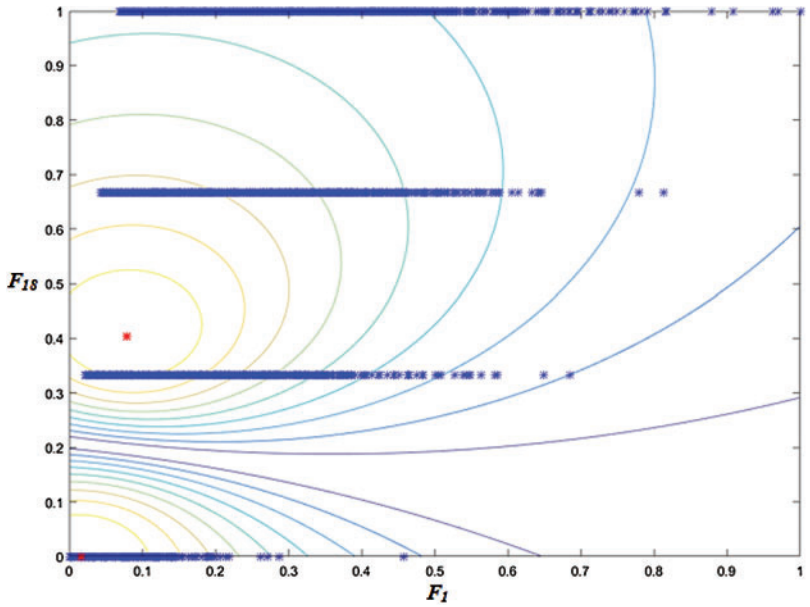

Figure 10: Higgs boson dataset Sammon mapping visualization into signal and background classes of PSOGK algorithm

6 Conclusion

This paper presents a hybridization of hard and fuzzy clustering tuned with Ant Lion Optimizer for the detection of the Higgs boson particle using the most informative features and variables which characterize the Higgs machine learning challenge 2014 data set.

The contribution of this work lies firstly in the approach used for these features and variables selection. The second contribution of this work is the new hybridized clustering combinations and tuning improvement, applied for the first time in the field of the Higgs boson search, where a metaheuristic technique such as ALO, optimizes various clustering methods as KM, GlobalKM, PAM, FCM, GK and GG. The results of the hybrid clustering technique tuned by ALO are compared with some exitising metaheuristic optimizations such as GA and PSO.

In order to choose the proper learning parameters for the experiment, an offline parameters tuning have been done for each one of the algorithms used in this work. The aim of each stochastic technique was to minimize the clustering algorithms.

Based on a multi-angle comparative analysis of the results found using each hybrid combination, the ALOGK clustering has proved its high truthfulness applied to Higgs boson search.

To confirm this result many scalar validity indexes are used in performances analysis as partition coefficient, classification entropy, partition index, separation index, Xie and Beni's index, Dunn's index, alternative Dunn's index, efficiency, purity computational time, average cost variation and Sammon mapping visualization.

As a perspective, we will improve and compare the 3 algorithms combinations that have given the best results in this work as well as other novel ones applied to an extended number of Higgs channels.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Kao, Y. T., Zahara, E., Kao, I. W. (2008). A hybridized approach to data clustering. *Expert Systems Applications, 34(3),* 1754–1762. DOI 10.1016/j.eswa.2007.01.028.
2. Eberhart, R., Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43. IEEE: Nagoya, Japan.
3. Yang, X. S. (2010). Firefly algorithm, Lévy flights and global optimization. In: Bramer, M., Ellis, R., Petridis, M., (Eds.), *Research and Development in Intelligent Systems XXVI*. pp. 209–218. London: Springer.
4. Dorigo, M., Birattari, M. (2010). *Ant colony optimization*. Berlin: Springer, 36–39.
5. Bansal, J. C., Sharma, H., Jadon, S. S., Clerc, M. (2014). Spider monkey optimization algorithm for numerical optimization. *Memetic Computing, 6(1),* 31–47. DOI 10.1007/s12293-013-0128-0.
6. Karaboga, D., Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artifcial bee colony (ABC) algorithm. *Journal of Global Optimization, 39(3),* 459–471. DOI 10.1007/s10898-007-9149-x.
7. Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69,* 46–61. DOI 10.1016/j.advengsoft.2013.12.007.
8. Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95,* 51–67. DOI 10.1016/j.advengsoft.2016.01.008.
9. Mirjalili, S. (2015). Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89,* 228–249. DOI 10.1016/j.knosys.2015.07.006.
10. Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software, 83,* 80–98. DOI 10.1016/j.advengsoft.2015.01.010.

11. Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H. et al. (2017). Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114,* 163–191. DOI 10.1016/j.advengsoft.2017.07.002.
12. Gupta, S., Deep, K., Heidari, A. A., Moayedi, H., Chen, H. (2019). Harmonized salp chain-built optimization. *Engineering with Computers,* 1–31. DOI 10.1007/s00366-019-00871-5.
13. Lu, C., Gao, L., Li, X., Xiao, S. (2017). A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry. *Engineering Applications of Artificial Intelligence, 57,* 61–79. DOI 10.1016/j.engappai.2016.10.013.
14. Tawhid, M. A., Ali, A. F. (2017). A hybrid grey wolf optimizer and genetic algorithm for minimizing potential energy function. *Memetic Computing, 9,* 1–13.
15. Emary, E., Zawbaa, H. M., Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing, 172,* 371–381. DOI 10.1016/j.neucom.2015.06.083.
16. Rodríguez, L., Castillo, O., Soria, J., Melin, P., Valdez, F. et al. (2017). A fuzzy hierarchical operator in the grey wolf optimizer algorithm. *Applied Soft Computing, 57,* 315–328. DOI 10.1016/j.asoc.2017.03.048.
17. Mirjalili, S., Saremi, S., Mirjalili, S. M., Coelho, L. S. (2016). Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Systems with Applications, 47,* 106–119. DOI 10.1016/j.eswa.2015.10.039.
18. Gupta, S., Kusum, D. (2018). A novel random walk grey wolf optimizer. *Swarm and Evolutionary Computation, 44,* 101–112. DOI 10.1016/j.swevo.2018.01.001.
19. Gupta, S., Kusum, D. (2020). A memory-based Grey Wolf optimizer for global optimization tasks. *Applied Soft Computing, 93,* 106367. DOI 10.1016/j.asoc.2020.106367.
20. Gupta, S., Kusum, D. (2019). Hybrid grey wolf optimizer with mutation operator. In: Bansal, J., Das, K., Nagar, A., Deep, K., Ojha, A., (Eds.), *Soft Computing for Problem Solving*. vol. 817. Singapore: Springer. DOI 10.1007/978-981-13-1595-4_75.
21. Aljarah, I., Mafarja, M., Heidari, A. A., Faris, H., Mirjalili, S. (2020). Clustering analysis using a novel locality-informed grey wolf-inspired clustering approach. *Knowledge and Information Systems, 62(2),* 507–539. DOI 10.1007/s10115-019-01358-x.
22. Ashish, T., Kapil, S., Manju, B. (2018). A novel clustering method using enhanced Grey Wolf optimizer and MapReduce. *Big Data Research, 14,* 93–100. DOI 10.1016/j.bdr.2018.05.002.
23. Liu, Y., Wang, Z., Xu, Z., Yang, Y., Gao, S. (2019). Ant Lion-based random walk differential evolution algorithm for optimization and clustering. *11th International Conference on Intelligent Human-Machine Systems and Cybernetics,* pp. 62–66. Hangzhou, China. DOI 10.1109/IHMSC.2019.10110.
24. Mjahed, S., El Hadaj, S. E., Bouzaachane, K., Raghay, S. (2018). Improved particle swarm optimization (PSO) based K-means clustering applied to fault signals diagnosis. *Proceedings of IEEE International Conference on Control Automation and Diagnosis,* pp. 1–6. Marrakech, Morocco.
25. Gupta, M. M. (2014). A review on clustering with genetic algorithms. *International Journal of Computer Science and Communication Networks, 4(3),* 94–98.
26. Mjahed, S., El Hadaj, S. E., Bouzaachane, K., Raghay, S. (2018). Engine fault signals diagnosis using genetic algorithm and K-means based clustering. *Proceedings of the International Conference on Learning and Optimization Algorithms: Theory and Applications,* Vol. 50, pp. 1–6. Association for Computing Machinery: New York, NY, USA. DOI 10.1145/3230905.32309.
27. Jhila, N., Farzin, K. (2018). A whale optimization algorithm (WOA) approach for clustering. *Cogent Mathematics & Statistics, 5(1),* 1–13.
28. Barbu, E., Héroux, P., Adam, S., Trupin, E. (2005). Clustering document images using graph summaries. *Machine Learning and Data Mining in Pattern Recognition, 3587,* 194–202.
29. Michie, D., Spiegelhalter, D. J., Taylor, C. C. (1995). *Machine learning, neural and statistical classification*. USA: Ellis Horwood.
30. Kamthania, D., Pahwa, A., Madhavan, S. S. (2018). Market segmentation analysis and visualization using K-mode clustering algorithm for e-commerce business. *Journal of Computing and Information Technology, 26(1),* 57–68. DOI 10.20532/cit.2018.1003863.

31. Azar, A. T., El-Said, S. A., Hassanien, A. E. (2013). Fuzzy and hard clustering analysis for thyroid disease. *Computer Methods and Programs in Biomedicine, 111(1),* 1–16. DOI 10.1016/j.cmpb.2013.01.002.
32. Belciug, S., Gorunescu, F., Salem, A., Gorunescu, M. (2010). Clustering-based approach for detecting breast cancer recurrence. *10th International Conference on Intelligent Systems Design and Applications*, Cairo. DOI 10.1109/ISDA.2010.5687211.
33. Salam, G. P., Cacciari, M. (2006). Jet clustering in particle physics, via a dynamic nearest neighbour graph implemented with CGAL. *LPTHE-06-02 Report*.
34. Mjahed, M. (2003). Clustering algorithm using space filling curves for the classification of high energy physics data. *Nuclear Physics B (Proc. Suppl.), 119,* 1024–1026. DOI 10.1016/S0920-5632(03)01751-1.
35. Chen, T., He, T. (2015). Higgs Boson discovery with boosted trees. *JMLR: Workshop and Conference Proceedings,* Vol. 42, pp. 69–80.
36. Abercrombie, D., Akchurin, N., Akilli, E., Maestre, J. A., Allen, B. et al. (2015). Dark matter benchmark models for early LHC run-2 searches: report of the ATLAS/CMS dark matter forum. *Physics of the Dark Universe, 27,* 1–63. DOI 10.1016/j.dark.2019.100371.
37. Aad, G., Abat, E., Abdallah, J., Abdelalim, A., Abdelouahab, A. et al. (2008). The ATLAS experiment at the CERN large hadron collider. *Journal of Instrumentation, 3,* 1–437. DOI 10.1088/1748-0221/3/07/P07007.
38. Chatrchyan, S., Hmayakyan, G., Khachatryan, V., Mousa, J. J., Adam, W. et al. (2008). The CMS experiment at the CERN LHC. *Journal of Instrumentation, 3(8),* 1–334.
39. Maggipinto, T., Nardulli, G., Dusini, S., Ferrari, F., Lazzizzera, I. et al. (1997). Role of neural networks in the search of the Higgs boson at LHC. *Physics Letters B, 409(1–4),* 517–522. DOI 10.1016/S0370-2693(97)00887-3.
40. Hakl, F., Hlavacek, M., Kalous, R. (2003). Application of neural networks to Higgs boson search. *Nuclear Instruments and Methods in Physics Research A, 502(2–3),* 489–491. DOI 10.1016/S0168-9002(03)00478-9.
41. Mjahed, M. (2005). Higgs search at LHC by neural networks. *Nuclear Physics B (Proc. Suppl.), 140,* 799–801. DOI 10.1016/j.nuclphysbps.2004.11.263.
42. Kumar, A., Ranjan, K. (2012). Search for standard model Higgs boson using the $H \rightarrow ZZ \rightarrow 2l2\nu$ channel in ppcollisions at CMS. *European Physics Journal Web of Conferences, 28,* 1–3.
43. Mjahed, M. (2006). The use of clustering techniques for the classification of high energy physics data. *Nuclear Instruments and Methods in Physics Research A, 559(1),* 199–202. DOI 10.1016/j.nima.2005.11.139.
44. Wirth, R., Fiori, E., Löher, B., Savran, D., Silva, J. et al. (2013). Particle identification using clustering algorithms. *Nuclear Instruments and Methods in Physics Research A, 717,* 77–82. DOI 10.1016/j.nima.2013.04.006.
45. Savran, D., Löher, B., Miklavec, M., Vencelj, M. (2010). Pulse shape classification in liquid scintillators using the fuzzy c-means algorithm. *Nuclear Instruments & Methods in Physics Research A, 624(3),* 675–683. DOI 10.1016/j.nima.2010.09.130.
46. The ATLAS Collaboration (2013). Evidence for Higgs boson decays to tau+tau-final state with the atlas detector. *Technical Report ATLAS-CONF-2013-108*. http://cds.cern.ch/record/1632191.
47. Melis, G. (2014). Dissecting the winning solution of the HiggsML challenge. *Proceedings of the NIPS Workshop on High-Energy Physics and Machine Learning,* Vol. 42, pp. 57–67.
48. Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B. et al. (2014). The Higgs boson machine learning challenge. *Proceedings of the International Conference on High-Energy Physics and Machine Learning,* Vol. 42, pp. 19–55.
49. Hmida, H., Hamida, S. B., Borgi, A., Rukoz, M. (2018). Scale genetic programming for large data sets: case of Higgs bosons classification. *International Conference on Knowledge Based and Intelligent Information and Engineering Systems,* Vol. 126, pp. 302–311. Belgrade, Serbia.
50. Dall'Osso, M., Dorigo, T., Gottardo, C. A., Oliveira, A., Tosi, M. et al. (2016). Higgs pair production: choosing benchmarks with cluster analysis. *Journal of High Energy Physics, 126,* 1–28.

51. Mac Queen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability,* Vol. 1, pp. 281–297.
52. Wang, X., Bai, Y. (2016). The global Minmax K means algorithm. *SpringerPlus, 5(1665*). DOI 10.1186/s40064-016-3329-4.
53. Likas, A., Vlassis, M., Verbeek, J. (2003). The global k-means clustering algorithm. *Pattern Recognition, 36(2),* 451–461. DOI 10.1016/S0031-3203(02)00060-2.
54. Park, H., Jun, C. (2009). A simple and fast algorithm for K-medoids clustering. *Expert Systems Applications, 36(2),* 3336–3341. DOI 10.1016/j.eswa.2008.01.039.
55. Chu, S. C., Roddick, J. F., Pan, J. S. (2004). Novel multi-centroid, multi-run sampling schemes for K-medoids-based algorithms. *International Journal of Knowledge-Based and Intelligent Engineering Systems, 8(1),* 45–56. DOI 10.3233/KES-2004-8106.
56. Chiang, C., Chu, S. C., Roddick, J. F., Pan, J. S. (2007). New search strategies and new derived inequality for efficient K-medoids-based algorithms. *Chinese Journal of Electronics, 16(1),* 82–87.
57. Miller, D. J., Nelson, C. A., Cannon, M. B., Cannon, K. P. (2009). Comparison of fuzzy clustering methods and their applications to geophysics data. *Applied Computational Intelligence and Soft Computing, 2009(3),* 1–17. DOI 10.1155/2009/876361.
58. Balasko, B., Abonyi, J., Feil, B. (2005). *Fuzzy clustering and data analysis toolbox: for use with MATLAB*. pp. 1–74. Math Works: Veszprem, Hungary.
59. Mjahed, M. (2010). Optimization of classification tasks by using genetic algorithms. *Proceedings of the 7th International Conference on Informatics and Systems,* pp. 1–4. Cairo.
60. Hewahi, N. (2017). A hybrid approach based on genetic algorithm and particle swarm optimization to improve neural network classification. *Journal of Information Technology Research, 10(3),* 48–68. DOI 10.4018/JITR.2017070104.
61. Neshat, M., Sargolzaei, M. (2012). A new kind of PSO: predator particle swarm optimization. *International Journal on Smart Sensing and Intelligent Systems, 5(2),* 521–539. DOI 10.21307/ijssis-2017-493.
62. ATLAS Collaboration (2014). Dataset from the atlas Higgs machine learning challenge 2014. *CERN Open Data Portal*. DOI 10.7483/OPENDATA.ATLAS.ZBP2.M5T8.
63. Bholowalia, P., Kumar, A. (2014). EBK-means: a clustering technique based on elbow method and K-means in WSN. *International Journal of Computer Applications, 105(9),* 17–24.
64. Mann, A. K., Kaur, N. (2013). Review paper on clustering techniques. *Global Journal of Computer Science and Technology Software & Data Engineering, 13(5),* 1–6.
65. Elavarasi, S. A., Akilandeswari, J. (2011). A survey on partition clustering algorithms. *International Journal of Enterprise Computing and Business Systems, 1(1),* 1–13.
66. Boley, D., Gini, M., Gross, R., Han, E., Hastings, K. et al. (1999). Partitioning-based clustering for web document categorization. *Decision Support Systems, 27(13),* 329–341. DOI 10.1016/S0167-9236(99)00055-X.
67. Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters, 31(8),* 651–666. DOI 10.1016/j.patrec.2009.09.011.
68. Velmurugan, T., Santhanam, T. (2011). A survey of partition based clustering algorithms in data mining: an experimental approach. *Information Technology Journal, 10(3),* 478–484. DOI 10.3923/itj.2011.478.484.
69. Xu, D., Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Annals of Data Science, 2(2),* 165–193. DOI 10.1007/s40745-015-0040-1.
70. Kaufman, L., Rousseeuw, P. J. (1987). Statistical data analysis based on the L1-norm and related methods. *Second International Conference on Statistical Data Analysis Based on the L1 -Norm and Related Methods,* pp. 405–416. North-Holland, Amsterdam.
71. Dunn, J. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters. *Journal of Cybernetics, 3(3),* 32–57. DOI 10.1080/01969727308546046.
72. Bezdek, J. (1981). Pattern recognition with fuzzy objective function algorithms, Springer: New York.

73. Bezdek, J., Ehrlich, R. (1984). FCM: the fuzzy c-means clustering algorithm. *Computers and Geosciences, 10(2–3),* 191–203. DOI 10.1016/0098-3004(84)90020-7.
74. Dave, R. N., Bhaswan, K. (1992). Adaptive fuzzy c-shells clustering and detection of ellipses. *IEEE Transactions on Neural Networks, 3(5),* 643–662. DOI 10.1109/72.159055.
75. Yager, R., Filev, D. (1994). Approximate clustering via the mountain method. *IEEE Transactions on Systems, Man and Cybernetics, 24(8),* 1279–1284. DOI 10.1109/21.299710.
76. Yang, M. S. (1993). A survey of fuzzy clustering. *Mathematical and Computer Modelling, 18(11),* 1–16. DOI 10.1016/0895-7177(93)90202-A.
77. Baraldi, A., Blonda, P. (1999). A survey of fuzzy clustering algorithms for pattern recognition. I. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 29(6),* 778–785. DOI 10.1109/3477.809032.
78. Höppner, F. (2000). Fuzzy cluster analysis: methods for classification, data analysis and image recognition. *Journal of the Operational Research Society, 51(6)*. DOI 10.2307/254022.
79. Babuska, R., Van der Veen, P. J., Kaymak, U. (2002). Improved covariance estimation for Gustafson–Kessel clustering. *FUZZ 2020: IEEE International Conference on Fuzzy Systems,* Vol. 2, pp. 1081–1085.
80. Zhou, N., Wang, L. (2007). A modified T-test feature selection method and its application on the HapMap genotype data. *Genomics, Proteomics & Bioinformatics, 5(3–4),* 242–249. DOI 10.1016/S1672-0229(08)60011-X.
81. Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers, 18(5),* 401–409. DOI 10.1109/T-C.1969.222678.
82. Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE, 78(9),* 1464–1480. DOI 10.1109/5.58325.
83. Carpenter, G., Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision Graphics and Image Processing, 37(1),* 54–115. DOI 10.1016/S0734-189X(87)80014-2.
84. Carpenter, G. A., Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *Computer, 21(3),* 77–88. DOI 10.1109/2.33.
85. Carpenter, G. A., Grossberg, S. (1987). ART 2: self-organization of stable category recognition codes for analog input patterns. *Applied Optics, 26(23),* 4919–4930. DOI 10.1364/AO.26.004919.
86. Carpenter, G. A., Grossberg, S. (1990). ART 3: hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks, 3(2),* 129–152. DOI 10.1016/0893-6080(90)90085-Y.
87. Kohonen, T., Schroeder, M. R., Huang, T. S. (2001). *Self-organizing maps,* 3rd edition. Berlin-Heidelberg: Springer-Verlag.
88. Kohonen, T., Oja, E., Simula, O., Visa, A., Kangas, J. (1996). Engineering applications of the self-organizing map. *Proceedings of IEEE, 84(10),* 1358–1384. DOI 10.1109/5.537105.
89. Vapola, M., Simula, O., Kohonen, T., Meriläinen, P. (1994). Representation and identification of fault conditions of an anaesthesia system by means of the Self-Organizing Map. *Proceedings of the International Conference on Artificial Neural Networks Sorrento,* Vol. I, pp. 350–353.
90. Ritter, H., Martinetz, T., Schulten, K. (1992). *Neural computation and self-organizing maps, an introduction*. Addison-Wesley: New York.