International Journal of Innovative Computing, Information and Control
Volume **11**, Number **4**, August **2015**

ICIC International ©2015 ISSN 1349-4198
pp. 1325–1339

A NEW DYNAMIC ALGORITHM FOR UNSUPERVISED LEARNING

Amina Dik[1], Abdelaziz El Moujahid[1], Khalid Jebari[1,2]
and Aziz Ettouhami[1]

[1]Conception and Systems Laboratory
Faculty of Sciences
Mohammed V University
Rabat, Morocco
a.dik70@yahoo.fr; { elmoujahid; khalid.jebari; Ettouhami.aziz }@gmail.com

[2]Multidisciplinary Faculty El Jadida
Bouchaib Doukkali University
Jadida, Morocco

Received July 2014; revised May 2015

Abstract. *This paper introduces a new dynamic method for unsupervised learning, aiming at discovering and representing structures of homogeneous clusters within unlabeled training data, where the number of clusters is algorithmically estimated with no assumption about the compactness and the separation of clusters. Assuming that the training data are originated from at least two different clusters, and that a minimum average degree of similarity exists between objects of each cluster, the learning process is initiated by creating two clusters around the least similar objects according to a given measure of inter-points similarities. The remaining objects are sequentially explored by analyzing their similarities with the mean points or centers that represent previously discovered clusters. For each of these objects a new learning rule is used for (1) creating a new cluster around this object, or (2) using the information carried by the object for updating representative points of existing clusters, or (3) deferring consideration of this object until either of the two previous decisions can be made with enough confidence. The method is dynamic in that the decision rule depends upon the number of clusters, which varies during the learning process. The effectiveness of this method is assessed on four real benchmark datasets in comparison to four other methods that require the number of clusters as an input, namely k-means, iterative self-organizing data analysis technique (ISODATA), fuzzy c-means (FCM), possibilistic c-means (PCM), and an unsupervised fuzzy learning method (UFL) that tries to automatically determine the number of clusters, and whose the proposed method constitutes an improved version (IUFL).*

Keywords: Cluster analysis, Unsupervised learning, Fuzzy clustering, Similarity measure

1. Introduction.

Being recognized as a fundamental unsupervised mode of learning, clustering has been widely utilized in several different application fields [1,2]. The aim of data clustering is mainly to find structure in data by grouping objects into homogeneous clusters such that the objects of each cluster should be more similar to each other compared to objects belonging to distinct clusters. Clustering is found in the literature under different appellations, such as unsupervised learning in pattern recognition, numerical taxonomy in biology and typology in social science [3].

As a corollary, several algorithms have been intensively researched in the literature [4] among which stands the k-means [5], or hard c-means clustering, as a very popular clustering algorithm. This algorithm assigns each point of data set to a unique cluster with a degree of membership equal to one [6]. As a result, clusters are disjointed and have well-defined boundaries. However, boundaries between clusters are not always definite in real world data sets [3]. The fuzzy c-means (FCM) algorithm was proposed to model uncertainty of belonging [7,8]. Indeed, based on fuzzy set theory, FCM is able to assign each data point to every cluster with different degrees of membership. As such, this algorithm is more efficient than the k-means where the boundaries of clusters remain overlapping and not well defined. However, this algorithm is sensitive to initialization and needs that the number of clusters be user specified in advance. Therefore, if the number of clusters is not known, k-means and FCM cannot be used [9]. To deal with this problem, an unsupervised learning technique (UFL) for automatic detection of the number of clusters with their prototypes was proposed [3]. UFL does not require initialization of prototypes as the learning procedure explores sequentially the learning base. Using a learning rule, UFL exploits the information carried by each current object to determine the prototypes even if this information is small. This may cause fuzziness in the final partition.

This paper proposes a new learning rule based on a measure of inter-points similarities. It consists in leaving uninformative objects that cannot be "easily" recognized by the existing prototypes. We estimate that these objects do not carry enough information to make a decision. These objects are reported for a subsequent re-examination until other newly encountered objects are examined and their fuzziness is dispelled.

The remainder of the paper proceeds as follows. We present related work in Section 2 whilst, in Section 3, we introduce the proposed algorithm. Section 4 details and discusses test results. We present our conclusions in Section 5.

2. Related Work.

As explained in the previous section, the aim of data clustering is to find structure in dataset according to measured or perceived intrinsic characteristics or similarity. This method is recognized as an unsupervised learning process in that the nature of the clusters is not known a priori. That is to say, the method does not use any prior class identifiers.

Clustering can be classified as hard or fuzzy. Fuzzy clustering is a generalization of hard clustering that has the potential of dealing with overlapping clusters and with data points on ill-defined boundaries among clusters [8]. This generalization was introduced by the concept of membership degree u_{ik} which is interpreted as the degree to which the object i belongs to the k^{th} cluster ($1 \leq k \leq c$ and $1 \leq i \leq n$) [8,10]. Hence, in fuzzy clustering, as boundaries among clusters are usually not well-defined, each data point can be assigned to any cluster with different membership degrees [3]. Conversely, in case of hard clustering each data point has to belong exclusively to one cluster with a membership degree that is equal to one. Consequently, clusters are disjointed and their boundaries are well-defined.

In mathematical terms, clustering or partitioning a learning base $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$ into c fuzzy clusters can be defined by c fuzzy sets $E_1, \ldots, E_c$ and a membership function [11] assuming values in the interval $[0,1]$ such as:

$$E_k = \{\mu_k(x_i)/\ x_i \in X,\ 1 \leq i \leq n\} \tag{1}$$

$$\forall i, k \quad \mu_k : \begin{cases} X \to [0,1] \\ x_i \to \mu_k(x_i) = u_{ik} \end{cases} \tag{2}$$

Therefore, a $(c \times n)$ fuzzy membership matrix $U = [u_{ik}]$ can be used to represent the partition result of a cluster analysis of X. The k^{th} row of this matrix contains values of the k^{th} membership function μ_k of the subset E_k. Elements u_{ik} satisfy the following condition:

$$0 \leq u_{ik} \leq 1; \quad 1 \leq k \leq c; \quad 1 \leq i \leq n \tag{3}$$

$$0 < \sum_{i=1}^{n} u_{ik} < n \quad 1 \leq k \leq c \tag{4}$$

As unsupervised methods, clustering algorithms do not require prior class tags. The learning procedure is determined solely from data set without additional knowledge. The most widely used clustering algorithms are k-means [5], FCM [8], ISODATA [12] and PCM [13]. These are highlighted in what follows.

2.1. *K*-means algorithm.

The k-means is recognized as a very popular hard clustering technique which aims to partition a data set X on n objects into k separated clusters. The method begins by randomly choosing k objects of X as initial cluster centers. Each object is assigned only to the nearest cluster and the mean for each cluster is recomputed. Then, objects can move from one cluster to another. The process stops when the centers of the clusters stop changing.

Mathematically, k-means algorithm minimizes an objective function that represents the sum of squared distances between all points and the cluster center:

$$J_k(X) = \sum_{j=1}^{k} \sum_{i \in C_j} d^2(x_i, c_j) \tag{5}$$

where x_i is a vector object representing the i^{th} object, c_j is the centre of the cluster C_j, $d(x_i, c_j)$ is the distance between the j^{th} centre and the i^{th} vector object.

The process of k-means consists of the following steps.

```
Input:
        unlabeled Dataset $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$;
        The number of cluster $k$;
Output
            $k$ clusters with centres.
Steps:
 1 – Initialize $K$ centres.
 2 – Repeat
            – Assign each $x_i$ to its nearest centre.
            – Re-compute the centre of each cluster using the following equation:
                    $c_j = \frac{\sum_{i=1}^{n_j} x_i}{n_j}; \quad 1 \leq j \leq k$        (6)
            // $n_j$ represents the number of objects in the $j^{\text{th}}$ cluster
Until the centres do not change.
```

Figure 1. K-means algorithm

However, the k-means algorithm is not adequate for application on real world data sets in which cluster boundaries are not well-defined. To overcome this challenge, FCM was proposed.

2.2. Fuzzy *c*-means algorithm.

Originally proposed by Bezdek, FCM is a generalization of the hard clustering k-means algorithm. K-means assigns each vector object x_i to a unique cluster with a degree of membership equal to one. As a consequence, in case of k-means clusters are disjointed and have well-defined boundaries. Conversely, in case of FCM each data point is assigned to every cluster with different degrees of membership and the boundaries between the generated clusters are likely to be not well-defined. FCM is more efficient compared to the crisp clustering in which clusters may remain overlapping and be not well separated.

A partition of $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$ into c fuzzy clusters can be defined by a fuzzy membership matrix $U = [u_{ik}]$ satisfying in more of Equations (3) and (4) the following condition:

$$\sum_{i=1}^{c} u_{ik} = 1; \quad 1 \leq k \leq n \tag{7}$$

where u_{ik} is the degree to which the pattern x_k belongs to the i^{th} cluster ($1 \leq i \leq c$ and $1 \leq k \leq n$).

The first constraint reflects the generalization of the characteristic function which assumes values in $\{0, 1\}$. For a given vector object, a value close to 1 indicates a high grade of belonging to the cluster. Inversely, a value close to 0 indicates a low grade of belonging to the cluster. The second constraint guaranties that no cluster is empty or totally equal to X. The last constraint assures that the membership of each object is distributed over all the c clusters.

FCM is an iterative procedure that optimizes an objective function J_m. This objective function depends on the distances of the data to the cluster centres weighted by the membership degrees. By varying the distance function, different forms of cluster in data sets can be detected.

The objective function J_m is defined by:

$$J_m(U, V; X) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m d^2(x_k, v_i) \tag{8}$$

where m ($1 < m < \infty$) is a weighting exponent used to control the relative contribution of each object vector x_i and the fuzziness degree of the final partition. $V = (v_1, v_2, \ldots, v_c)$ represents a c-tuple of prototypes, and each prototype characterizes one of the c clusters. $d(x_k, v_i)$ is the distance between the i^{th} prototype and the k^{th} data point.

Bezdek proved that FCM converges to an approximate solution under two conditions [3]:

$$u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{d(x_k, v_i)}{d(x_k, v_j)}\right)^{2/m-1}\right]^{-1}; \quad 1 \leq i \leq c; \quad 1 \leq k \leq n \tag{9}$$

$$v_i = \frac{\sum\limits_{k=1}^{n} (u_{ik})^m x_k}{\sum\limits_{k=1}^{n} (u_{ik})^m}; \quad 1 \leq i \leq c \tag{10}$$

The pseudo-code of FCM algorithm is given in Figure 2.

In FCM algorithm, the membership degrees are relative (Equation (7)). According to Zadeh [11], the membership degrees should only belong to the interval $[0, 1]$. To overcome this constraint, PCM was proposed.

2.3. Possibilistic c-means.

Krishnapuram and Keller proposed the possibilistic c-means (PCM) clustering [13] to remedy the drawbacks of FCM. PCM relaxes the objective function (Equation (8)) by dropping the sum to 1 (Equation (7)) and introduces a possibilistic type of membership function to describe the degree of belonging [14]. Thereby, degrees of membership became independent.

```
Store unlabeled Dataset $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$;
Choose
    • $1 < c < n$;
    • $m > 1$;
    • $t_{\max}$ (iteration limit);
    • the $\varepsilon$ (tolerance bound);
    • norm for clustering criterion $J_m$;
    • norm for termination error $E_t = ||V_t - V_{t-1}||_{err}$;
Initialize
    • prototypes $V_0 = (v_{1,0}, v_{2,0}, \ldots, v_{c,0}) \in \Re^{c \times p}$
    • $t = 0$; (iteration index)
do {t++;
    • Calculate $U_t$ using $V_{t-1}$ and (Equation (9));
    • Calculate $V_t$ using $U_t$ and (Equation (10));
} while $(||V_t - V_{t-1}||_{err} > \varepsilon)$ and $(t < t_{\max})$);
$U^* = U_t$; $V^* = V_t$;
Use $U^*$ and/or $V^*$;
```

FIGURE 2. FCM algorithm

PCM optimizes the objective function J_m defined as:

$$J_m(U, V; X) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m d^2(x_k, v_i) + \sum_{i=1}^{c} \eta_i \sum_{k=1}^{n} (1 - u_{ik})^m \tag{11}$$

where η_i $(1 < i < c)$ is the scale parameter defined as

$$\eta_i = K \frac{\sum\limits_{k=1}^{n} (u_{ik})^m d^2(x_k, v_i)}{\sum\limits_{k=1}^{n} (u_{ik})^m}, \quad K > 0 \tag{12}$$

and u_{ik} is defined as:

$$u_{ik} = \left[1 + \left(\frac{d^2(x_k, v_i)}{\eta_i} \right)^{1/m-1} \right]^{-1}; \quad 1 \leq i \leq c; \quad 1 \leq k \leq n \tag{13}$$

These algorithms need to specify the number of clusters. ISODATA was proposed as a method of clustering that does not require the number of clusters.

2.4. **Fuzzy ISODATA.** This algorithm is based on the k-means algorithm. To improve the clustering process, the algorithm employs three processes or operations: eliminating, splitting, and merging [12].

The algorithm starts with K_{init} centres, where K_{init} is a user-given initial number of clusters that eliminates clusters by distance or by size. As in FCM, ISODATA assigns the first K_{init} object vector to cluster centers. The other objects vectors are assigned to the clusters by a minimum distance principle and all clusters are considered in eliminating, splitting or merging clusters. In the eliminating phase, clusters that have less than the minimum cluster size $n_{\min}$ objects are deleted and their object vectors are reassigned. In the splitting phase, a cluster is divided into more clusters if its standard deviation is greater than a threshold value (for example, the variance). In the merging phase, two or more clusters are merged if the distances among these clusters are less than a threshold value (minimum distance between centers δ). Then, the number of clusters "c" is ultimately determined.

The pseudo-code of ISODATA algorithm is given in Figure 3.

```
Store unlabeled Dataset $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$;
Choose
  • $K_{init}$; $1 < c < n$;
  • $t_{\max}$ (iteration limit);
  • $n_{\min}$: minimum objects in clusters;
  • $\delta$: minimum distance between the centres;
  • $V_{\max}$: maximum variance in a cluster;
Initialize
  • $c = K\_init$;
  • prototypes $V_0 = (x_1, x_2, \ldots, x_c) \in \Re^{c \times p}$
  • $t = 0$; (iteration index)
Repeat {t++;
  • For each centre $c_{k,t-1}$ do
    { $S_k = \{x_i / d(x_i, c_{k,t-1}) < d(x_i, c_{j,t-1})\}$ $\forall j \neq k$
      if (Card $(S_k) < n_{\min}$) Then Delete($c_k$)
      if ($\exists c_{j \neq k} / d(c_k, c_j) < \delta$) Then {Merge($c_k, c_j$); $c = c - 1$}
      if ($var(S_k) > V_{\max}$) Then {Split ($c_k$); $c = c + 1$}
      Calculate $V_t$
    }
} until ($\exists k \in [1, c] / c_{k,t} \neq c_{k,t-1}$ and ($t < t_{\max}$));
$V^* = V_t$;
```

Figure 3. ISODATA algorithm

However, ISODATA needs four threshold values for parameters K_{init}, $n_{\min}$, minimum distance between centers, and maximum variance authorized in a cluster. Other approaches have been proposed to automatically determine the optimal number of clusters. UFL is one of these approaches.

2.5. **UFL algorithm.** This algorithm attempts to combine advantages from both hierarchical and partitional clustering techniques [3]. It sequentially explores all the "n" objects of the learning base X and analyzes their similarities using the similarity measure given by Equation (16). It starts by generating a first cluster whose prototype is initialized with the first object. Afterwards, the other objects are successively and iteratively examined.

A threshold ξ is used to detect when a new coming object is not recognized and, thereby, dissimilar to all existing prototypes. In this case a new cluster is created and its prototype is initialized with the current object. This threshold represents the minimum of similarity that each object should have with its nearest prototype.

UFL uses the similarity measure Sim and its associated threshold ξ to construct classes. Two cases are considered:

a -
$$\underset{1 \leq k \leq c}{Max}(Sim(i, k)) < \xi \tag{14}$$

It means that the current element x_i does not meet any similarity criterion recognized in the previously detected prototypes [3] and will hence be defined to represent a new cluster. Thus, we put $c = c + 1$ and $v_c = x_i$.

b -
$$\underset{1 \leq k \leq c}{Max}(Sim(i, k)) \geq \xi \tag{15}$$

x_i is considered having the required minimum similarity to the previously detected clusters. So we do not need to create a new cluster.

Sim is defined by:

$$Sim(x_i, x_k) = 1 - \frac{\|x_i - x_k\|_A^2}{p} \tag{16}$$

where A is the positive definite $p \times p$ matrix defined by:

$$A_{jt} = \begin{cases} (r_j)^{-2}, & j = t \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

r_j represents the range of all possible values for the j^{th} feature associated with objects of X $(1 \le j \le p)$. It is defined by:

$$r_j = \max_{1 \le i \le n} \{x_{ij}\} - \min_{1 \le i \le n} \{x_{ij}\}, \quad 1 \le j \le p \tag{18}$$

The algorithm makes no assumptions about the number of clusters c. The choice of c depends automatically on the choice of the threshold ξ since the creation of a new class depends on its value. So the quality of the detected clusters depends mainly on this threshold [3]. By varying ξ between two values $\underset{\substack{1\le i,j\le n \\ i\ne j}}{Min}(Sim(i,j))$ and $\underset{\substack{1\le i,j\le n \\ i\ne j}}{Max}(Sim(i,j))$, different sets of c prototypes can be detected.

Prototypes of the previously created clusters are then updated according to the learning scheme:

$$v_k(i) = v_k(i-1) + \frac{Sim(x_i, v_k)}{n_k(i)}\left[x_i - v_k(i-1)\right] \quad 1 \le k \le c, \quad c \ge 2 \tag{19}$$

where $v_k(i)$, $v_k(i-1)$ are respectively the prototype of the k^{th} class before and after processing x_i.

$n_i(k)$ denotes the fuzzy cardinality of the k^{th} cluster after processing x_i, defined by:

$$n_i(k) = \sum_{j=1}^{k} Sim(x_i, v_k) \quad 1 \le k \le c, \;\; i \le n \tag{20}$$

The pseudo-code of UFL algorithm is given in Figure 4.

```
Store unlabeled Dataset $X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p$;
1-Choose
    • similarity measure, Sim [Equation (16)];
2-Initialize
    • $c = 1$
    • prototype $v_1 = x_1$
3-For $i = 2$ to $n$ {
            if ($\underset{1\le j\le c}{Max}(Sim(i,j)) < \xi$) { $c = c + 1$; $v_i = x_i$}
            else { Update all prototypes $v_j$, $1 \le j \le c$ (Equation (19)) }
4-Recalculate $U = U^*$ [Equation (9)], $V = V^*$ [Equation (10)]
5-Use $U^*$ and/or $V^*$;
```

Figure 4. UFL algorithm

3. **The Proposed Algorithm.** Assuming that the n available object vectors that form the learning database X are originated from at least two distinct categories, and given a measure of inter-points similarities, the proposed method starts by creating two clusters around the least similar objects [15]. For this, the representative points or prototypes of the created clusters, v_1 and v_2, are initialized using these two objects after swapping them with the two first objects x_1 and x_2. The $(n-2)$ remaining objects are then successively examined. For each object x_i – With i varying between 3 and n – A new learning rule is used that allows three different decisions to be made depending on the extent to which x_i is similar to the c prototypes v_1, v_2, …, v_c representing the so far discovered clusters.

The choice of the best decision to take is based on two quantitative criteria. The first criterion is the maximum value of the computed similarities between x_i and the c existing prototypes: $Max_1(x_i) = \underbrace{Max}_{1\leq p\leq c}(Sim(i,p))$. The second criterion is the difference between $Max_1(x_i)$ and the similarity degree of x_i to the second most similar prototype: $Max_1(x_i) - Max_2(x_i)$.

The first decision consists in creating a new cluster and initializing its prototype by x_i. This decision is taken when $Max_1(x_i)$ is less than a user-defined threshold ξ, whose values can theoretically vary between the limits $\xi_{\min} = \underbrace{Min}_{\substack{1\leq i,j\leq n\\ i\neq j}}(Sim(i,j))$ and $\xi_{\max} = \underbrace{Max}_{\substack{1\leq i,j\leq n\\ i\neq j}}(Sim(i,j))$. This decision means that x_i is not similar enough to the prototypes of the previously detected clusters in order to consider that it would come from one of these clusters. Therefore, a new cluster is created around x_i.

The second decision consists in deferring the exploration of x_i until other objects are examined. This decision is taken if $Max_1(x_i) \geq \xi$ and $Max_1(x_i) - Max_2(x_i) < \frac{1}{c+1}$. The first condition means that the level of similarity that x_i presents with the existing prototypes is sufficient to consider that it comes from one of the c already discovered clusters; the second condition means, however, that a big ambiguity exists concerning the cluster from which x_i may be originated. This ambiguity depends on the number c of existing clusters, with the worst case occurring when x_i presents the same level of similarity with the prototypes of all these clusters. $\frac{1}{c+1}$ is a threshold that the difference between the membership degrees of x_i to the clusters corresponding to its two most similar prototypes should exceed in order for the learning process to take account of the information carried by this object. Otherwise, exploration of this object is deferred until subsequent objects, if any, are explored, which may help reduce the observed ambiguity. Note that the similarity measure $Sim(x_i, v_k)$ between an object x_i and a prototype v_k can also be interpreted as the membership degree of x_i to the cluster represented by v_k [3].

When the difference $Max_1(x_i) - Max_2(x_i)$ is greater than or equal to the threshold $\frac{1}{c+1}$, and the maximum value of computed similarities $Max_1(x_i)$ is greater or equal to the threshold ξ, a third decision is made. It consists in exploiting the information carried by x_i in order to update the prototypes of the existing clusters according to the learning rule given by Equation (19).

Finally, by repeating this method for the same input data using different values of the similarity threshold ξ, different more or less acceptable clustering results can be obtained. To select the best result among all these candidate solutions, two validity criteria are used [3,15-17]. Namely, the partition entropy is defined by:

$$PE(U) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{c}\left[u_{ik}\log_a(u_{ik})\right] \tag{21}$$

and the partition coefficient is modified by Dave and defined by:

$$PC(U) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{c}(u_{ik})^2 \tag{22}$$

In terms of pseudo-code, the proposed method can be summarized as follows.

```
Input: Unlabeled data $X$
Output: The estimated number of clusters $c^*$, and the matrix of prototypes $V = (v_1, \ldots, v_{c^*})$
1 – Choose a similarity measure, sim [Equation (16)]
  – Find the least similar objects, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$
  – swap($\boldsymbol{x}_1$, $\boldsymbol{x}_i$); swap($\boldsymbol{x}_2, \boldsymbol{x}_j$)
  – Calculate:
        $Min(\xi) = \underset{\substack{1\le i,j\le n\\ i\ne j}}{Min}(Sim(i,j))$, $Max(\xi) = \underset{\substack{1\le i,j\le n\\ i\ne j}}{Max}(Sim(i,j))$, $\Delta(\xi)$[10%]
2- For ($\xi = Min(\xi)$, $\xi \le Max(\xi)$, $\xi+ = \Delta(\xi)$) {$\boldsymbol{c} = \boldsymbol{2}$; $\boldsymbol{v}_1 = \boldsymbol{x}_1$; $\boldsymbol{v}_2 = \boldsymbol{x}_2$;
  • Initialize
    Objects_Treated = 2;                       // Number of analyzed objects
    IsTreated[1] = 1; IsTreated[2] = 1;
    IsTreated[i] = 0 for $i = 3, 4, \ldots, n$
    NbTreated[i] = 0 for $i = 3, 4, \ldots, n$         // Number of times $x_i$ is analyzed
  • While (Objects_Treated < $n$)
  { for ($i$ = 3, $i$ <= $n$, $i$++)
      { if (IsTreated [i] == 0) do
          { if (NbTreated[i] <= 2) do
              { double $Max_1 = 0$, $Max_2 = 0$;
                if ($Sim(i,0) > Sim(i,1)${$Max_1 = Sim(i,0)$; $Max_2 = Sim(i,1)$;}
                else {$Max_1 = Sim(i,1)$; $Max_2 = Sim(i,0)$;}
                for ($j$ = 3; $j$ <= $c$; $j$++)
                    if ($Sim(i,j) > Max_1$) {$Max_2 = Max_1$; $Max_1 = Sim(i,j)$;}
                if ($Max_1 < \xi$) do {$c$++; $\boldsymbol{v}_c = \boldsymbol{x}_i$; IsTreated[i] = 1; Objects_Treated++;}
                else if (($Max_1 - Max_2$) >= $1/(c+1)$) {
                      Update all prototypes (Equation (19)); IsTreated[i] = 1; Objects_Treated++;}
                    else {NbTreated[i]++;}
              } else {Update all prototypes (Equation (19)); objects_Treated++; IsTreated[i] = 1;}
          }
      }
  }
  • Use $\boldsymbol{U}$ to $\boldsymbol{PE(U)}$ and $\boldsymbol{PC(U)}$.
}
3-Return $\boldsymbol{c}^*$ and $\boldsymbol{V}$.
```

Figure 5. Proposed algorithm

4. Results and Discussions.

To evaluate the performance of the proposed algorithm, a collection of experiments is conducted on four real-world datasets. These datasets are available from the UCI Machine Learning Repository [18]: Wine, Breast Cancer, Balance scale and Haberman's Survival (see Table 1). A segmentation of an MRI image of brain is also given.

Wine dataset is a result of a chemical analysis of wines from three different cultivars. There are 13 attributes and 178 samples from three classes corresponding to three different cultivars with respectively 59, 79, and 48 samples per variety.

Breast Cancer dataset is a 9-dimensional pattern classification problem with 699 samples from malignant (cancerous) class and benign (non-cancerous) class. The two classes contain respectively 458 and 241 points.

The third file is Balance scale dataset which contains 625 data objects. Each data object has 4 attributes: left weight, left distance, right weight, and right distance. There are 3 classes with 49 samples Balanced (B), 288 Left (L) and 288 Right (R).

The last example is Haberman's Survival dataset that is the result of a measure of 306 cases on the survival of patients who had undergone surgery for breast cancer. It is a 3-dimensional pattern classification problem from two classes.

Table 1 describes the considered data and gives information about the attributes, size and number of classes.

Firstly, the cluster detection procedure is run for the values of ξ comprised between $Min(\xi) = \underset{\substack{1\le i,j\le n \\ i\ne j}}{Min}(Sim(i,j))$ and $Max(\xi) = \underset{\substack{1\le i,j\le n \\ i\ne j}}{Max}(Sim(i,j))$. The step varies with $\Delta\delta = 0.1$.

By varying ξ, different sets of c clusters can be detected. The role of this first exploration is to show how the number of detected clusters varies with ξ.

However, some values of c are skipped for some data sets. To obtain these values in BCW dataset for example, the considered dataset was explored with a step $\Delta\delta = 0.001$. Table 3 summaries the results of this exploration.

For each data set, Table 4 shows the smallest values of the threshold ξ that led to a number of clusters comprising between 2 and 6.

As depicted in Table 5, two validity indices are used to choose the best partition among a set of alternative partitions. Optimal values are displayed in bold. We can see that the best solution always corresponds to the actual number of clusters for all examples.

Both UFL and IUFL determine the number c of clusters presented in the data set. We run the two algorithms 30 times after changing randomly the order of elements. The optimal value of c corresponds to the minimum of PE(U) and the maximum of PC(U). Table 6 presents results of this exploration.

Table 1. Description of datasets

	BCW	***Wine***	***Balance***	***Haberman's Survival***
No. of Samples	699	178	625	306
No. of Attributes	9	13	4	2
No. of Classes	2	3	3	2

Table 2. Variation of c with the step $\delta = 0.1$ for BCW dataset

ξ (%)	c
21, 31	2
31, 31	10

Table 3. Variation of c with the step $\delta = 0.001$ for BCW dataset

ξ (%)	c
[12, 31-21, 61]	2
21, 71	3
[21, 81-23, 11]	4
[23, 21-24, 11]	5
[24, 21-25, 21]	6

Table 4. Lowest threshold values of ξ (%) for detecting different numbers of clusters c ($2 \leq c \leq 6$)

c	***BCW***	***Wine***	***Balance***	***Haberman's Survival***
2	12.31	22.19	*	39.83
3	21.71	22.69	33.33	43.83
4	21.81	22.99	33.53	44.83
5	23.21	23.39	34.73	47.83
6	24.21	23.59	*	48.83

Table 5. Validity indexes values for each detected number of clusters c ($2 \leq c \leq 6$)

c		***2***	***3***	***4***	***5***	***6***
BCW	E	**0.385**	0.511	0.536	0.575	0.627
	PC	**0.832**	0.694	0.627	0.562	0.492
Wine	E	0.971	**0.604**	0.647	0.770	0.809
	PC	0.519	**0.556**	0.525	0.371	0.305
Balance	E	*	**0.889**	0.926	0.940	*
	PC	*	**0.419**	0.306	0.239	*
Haberman's Survival	E	**0.965**	0.983	0.989	0.992	0.996
	PC	**0.523**	0.346	0.258	0.205	0.169

Table 6. Comparison of detected c by UFL and IUFL

Algorithm		BCW	Wine	Balance	Haberman
UFL	c correct	10	4	6	14
	c incorrect	20	26	24	16
IUFL	c correct	24	23	25	23
	c incorrect	6	7	5	7

IUFL reconstructs the classes which are presented in each data set by labeling the data. These bases are supervised but any information about classes is given to the algorithm. Thus, it is possible to determine both the number of misclassified objects and the recognition rate.

The experiments above aim to illustrate the usefulness of IUFL in comparison to the following algorithms: k-means, FCM, PCM, ISODATA and UFL. These algorithms assign every data point to clusters by the minimum distance assignment principle. This principle consists in assigning a new data point x_i to the cluster to which its membership value is the highest.

In this project, we implemented these algorithms in C++ programming language. At first, FCM with usual norms is used for partitioning data. These norms are Euclidean, Spearman, Manhattan and Chebychev, which are particular cases of Minkowski distances. We run the program at the same conditions but using different distances.

The other algorithms were applied to data sets and their performance was compared. The following table summarizes the results obtained for the considered algorithms.

Table 7 shows that IUFL can improve the performance of clustering considerably and can lead to an increase in accuracy for class discovery.

Table 7. Recognition rate for considered dataset

Dataset	*FCM*				*k-means*	*ISODATA*	*PCM*	*UFL*	*IUFL*
	Euclidian Distance	*Manhattan Distance*	*Spearman Distance*	*Chebychev Distance*					
BCW	65.96%	**66.39%**	49.22%	53.51%	**66.1%**	65.52%	61.52%	64.95%	**66.1%**
Wine	69.67%	69.67%	70.23%	69.67%	69.67%	66.30%	66.86%	69.67%	**71.92%**
Balance	53.92%	38.08%	36.8%	36.64%	48.16%	50.4%	50.72%	48.96%	**66.4%**
Haberman's Survival	49.02%	49.68%	51.64%	52.95%	49.02%	49.02%	50.33%	52.95%	**53.93%**

Table 8. Actual centers and learned prototypes for considered dataset

Dataset	*Actual centres*	*Learned prototypes*
BCW	$\begin{pmatrix} 2,956 & 7,195 \\ 1,325 & 6,572 \\ 1,443 & 6,56 \\ 1,364 & 5,547 \\ 2,12 & 5,298 \\ 1,305 & 7,564 \\ 2,1 & 5,979 \\ 1,29 & 5,863 \\ 1,063 & 2,589 \end{pmatrix}$	$\begin{pmatrix} 3,005 & 7,239 \\ 1,294 & 7,035 \\ 1,424 & 6,961 \\ 1,325 & 5,971 \\ 2,077 & 5,54 \\ 1,301 & 7,986 \\ 2,098 & 6,255 \\ 1,241 & 6,291 \\ 1,087 & 2,627 \end{pmatrix}$
Wine	$\begin{pmatrix} 13,744 & 12,278 & 13,153 \\ 2,010 & 1,932 & 3,333 \\ 2,455 & 2,244 & 2,437 \\ 17,037 & 20,238 & 21,416 \\ 106,338 & 94,549 & 99,312 \\ 2,840 & 2,2588 & 1,678 \\ 2,9823 & 2,0808 & 0,781 \\ 0,29 & 0,363 & 0,447 \\ 1,899 & 1,630 & 1,153 \\ 5,528 & 3,086 & 7,396 \\ 1,062 & 1,056 & 0,682 \\ 3,157 & 2,785 & 1,683 \\ 1115,711 & 519,507 & 629,895 \end{pmatrix}$	$\begin{pmatrix} 13,817 & 12,509 & 12,962 \\ 1,89 & 2,454 & 2,512 \\ 2,44 & 2,288 & 2,398 \\ 16,89 & 20,775 & 19,773 \\ 105,206 & 92,318 & 103,763 \\ 2,866 & 2,074 & 2,135 \\ 3,026 & 1,787 & 1,612 \\ 0,287 & 0,387 & 0,388 \\ 1,915 & 1,457 & 1,521 \\ 5,788 & 4,08 & 5,679 \\ 1,078 & 0,945 & 0,886 \\ 3,093 & 2,498 & 2,388 \\ 1214,622 & 454,006 & 736,87 \end{pmatrix}$
Balance	$\begin{pmatrix} 2,938 & 3,611 & 2,399 \\ 2,938 & 3,611 & 2,399 \\ 2,938 & 2,399 & 3,611 \\ 2,938 & 2,399 & 3,611 \end{pmatrix}$	$\begin{pmatrix} 2,643 & 4,029 & 2,259 \\ 1,925 & 3,585 & 3,587 \\ 3,109 & 2,029 & 3,892 \\ 1,989 & 3,172 & 3,89 \end{pmatrix}$

IUFL provides c prototypes which are very close to the actual centres. Table 8 displays the actual centres calculated from original labelled data, and those produced by IUFL.

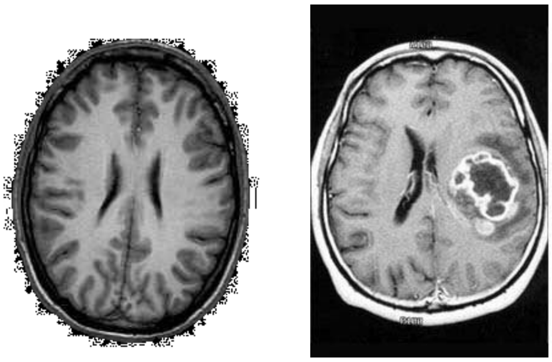
For the segmentation of MRI image, we consider Figures 6-8.

Table 9 presents minimum values of ξ corresponding to each detected number of clusters $(2 \leq c \leq 6)$ produced by IUFL and validity indices of segmenting Image1. The best partition corresponds to $c = 3$.

Table 10 presents the variation of the number of detected clusters for Image2, and validity indexes produced by IUFL. It shows that best partition is obtained for $c = 4$.

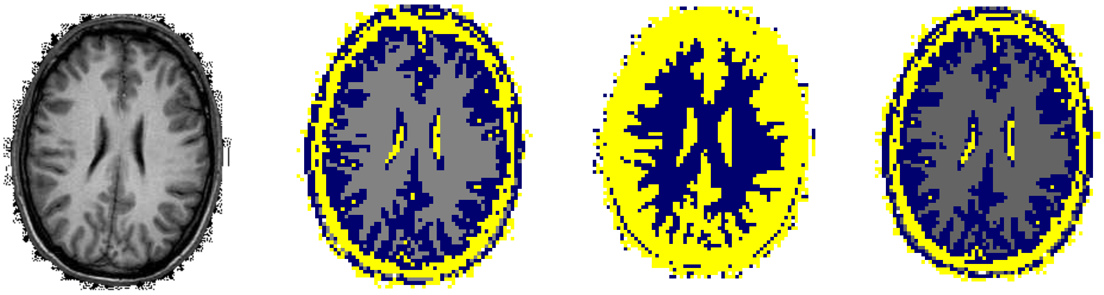
The learned prototypes by the New UFL are used to initialize FCM which gives a very good initialization of cluster centers compared to FCM.

5. **Conclusions.** In this paper, we have proposed a new method based on an unsupervised learning procedure which explores the learning database X in order to (1) discover its intrinsic number of clusters and to (2) provide a prototype for each detected cluster. To this end, a new learning rule was proposed. It consists in avoiding treatment



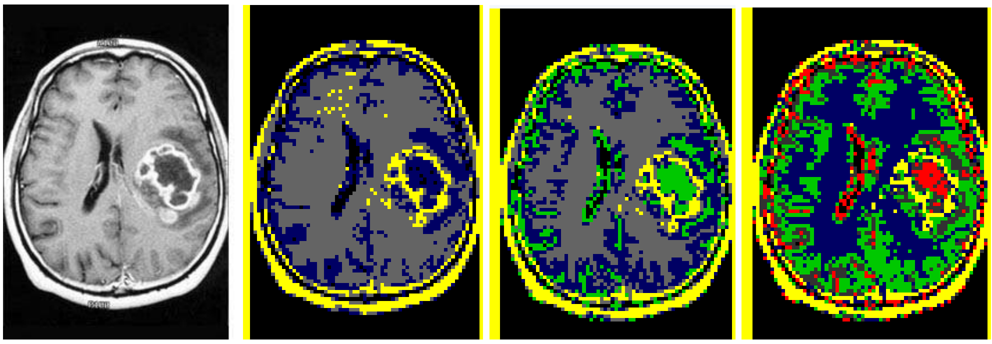
(a) Image1 (b) Image2

FIGURE 6. MRI images of brain



(a) (b) (c) (d)

FIGURE 7. Segmentation of Image1. (a) Original image, (b) FCM with $c = 3$, (c) ISODATA with $c = 2$, (d) New UFL + FCM with $c = 3$.



(a) (b) (c) (d)

FIGURE 8. Segmentation of Image2. (a) Original image, (b) FCM with $c = 4$, (c) FCM with $c = 5$, (d) FCM with $c = 6$.

of immediately some vector objects that cannot be "easily" recognized and that may cause fuzziness in the partition. These vector objects are reported until examining other newly encountered vector objects and dispelling the fuzziness or reducing the confusion. In other words, the information provided by the examined objects during the iterative process contributes to minimizing the fuzziness of the final partition.

Our evidence shows that incorporating the proposed rule in the process improves clustering accuracy. In our tests on the four datasets, an improvement of up to 29.76%

Table 9. Number of clusters and validity indexes values produced by New UFL for Image1

ξ (%)	c	PE	PC
20	2	0.666	0.516
60	2	0.556	0.530
70	**3**	**0.520**	**0.542**
71	4	0.540	0.482
72	5	0.543	0.473
75	6	0.531	0.459

Table 10. Number of clusters and validity indexes values produced by New UFL for Image2

ξ (%)	c	PE	PC
[10-50]	3	0.407	0.690
[60-70]	**4**	**0.353**	**0.713**
75	5	0.503	0.564
80	8	0.503	0.495

is realized relative to FCM with Chybechev distance and 29.6% relative to FCM with Spearman distance.

Unfortunately, our method is time consuming. This is because the proposed rule restricts the automatic treatment of vector objects and explores unlabeled vector objects twice.

The advantages of the algorithm are its simplicity and self-organization. It can be used as an initial process in other algorithms. We also envisage in future work merging or splitting clusters or eliminating small clusters using the automatically determined threshold.

REFERENCES

[1] X. B. Gao, *Fuzzy Cluster Analysis and Its Applications*, Xi'an Electronic Technology University Press, 2004.

[2] K. West, S. Cox and P. Lamere, Incorporating machine learning into music similarity estimation, *Proc. of the 1st ACM Workshop on Audio and Music Computing Multimedia*, pp.89-96, 2006.

[3] A. Bouroumi, M. Limouri and A. Essaïd, Unsupervised fuzzy learning and cluster seeking, *Intelligent Data Analysis*, vol.4, nos.3-4, pp.241-253, 2000.

[4] H. Bandemer and W. Näther, *Fuzzy Data Analysis*, Kluwer Academic Publishers, Dordrecht, Boston, London, 1992.

[5] J. B. Macqueen, Some methods for classification and analysis of multivariate observations, *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, vol.1, pp.281-297, 1967.

[6] B. S. Everitt, *Cluster Analysis*, Edward Arnold, 1993.

[7] J. C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, *Journal of Cybernetics*, vol.3, pp.32-57, 1973.

[8] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[9] R. Suganya and R. Shanthi, Fuzzy c-means algorithm – A review, *International Journal of Scientific and Research Publications*, vol.2, no.11, 2012.

[10] J. C. Bezdek, FCM: The fuzzy c-means clustering algorithm, *Computers & Geosciences*, vol.10, nos.2-3, pp.191-203, 1984.

[11] L. A. Zadeh, Fuzzy sets, *Inform. Contr.*, vol.8, pp.338-353, 1965.

[12] G. H. Ball and D. J. Hall, ISODATA: An iterative method of multivariate analysis and pattern classification, *Int. Commun. Conf.*, Philadelphia, PA, 1966.

[13] R. Krishnapuram and J. Keller, A possibilistic approach to clustering, *IEEE Trans. Fuzzy Syst.*, vol.1, no.2, pp.98-110, 1993.

[14] J. Yu, S.-H. Lee and M. Jeon, An adaptive ACO-based fuzzy clustering algorithm for noisy image segmentation, *International Journal of Innovative Computing, Information and Control*, vol.8, no.6, pp.3907-3918, 2012.

[15] M. Benrabh, A. Bouroumi and A. Hamdoun, A fuzzy validity-guided procedure for cluster detection, *Malaysian Journal of Computer Science*, vol.18, no.1, pp.31-39, 2005.

[16] M. R. Rezaee, B. P. F. Lelieveldt and J. H. C. Reiber, A new cluster validity index for the fuzzy c-means, *Pattern Recognition Letters*, vol.19, pp.237-246, 1998.

[17] M. Halkidi, Y. Batistakis and M. Vazirgiannis, On clustering validation techniques, *Journal of Intelligent Information Systems*, vol.17, pp.107-145, 2001.

[18] C. L. Blake and C. J. Merz, *UCI Repository of Machine Learning Databases*, University of California, Irvine, Department of Information and Computer Sciences, 1998.