Journal of AI and Data Mining
Vol 8, No 3, 2020, 391-407.
DOI: 10.22044/JADM.2020.7826.1941

High-Dimensional Unsupervised Active Learning Method

V. Ghasemi[1*], M. javadian[1], and S. Bagheri Shouraki[2]

1. Department of Computer Engineering, Kermanshah University of Technology. Kermanshah, Iran.
2. Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran.

Received 09 February 2019; Revised 31 December 2019; Accepted 26 February 2020
**Corresponding author: v.ghasemi@kut.ac.ir (V. Ghasemi).*

Abstract
In this work, a hierarchical ensemble of projected clustering algorithm is proposed for high-dimensional data. The basic concept of this algorithm is based on the *active learning method* which is a fuzzy learning scheme, inspired by some behavioral features of human brain functionality. High-dimensional unsupervised active learning method is a clustering algorithm, which blurs the data points as 1D ink drop patterns in order to summarize the effects of all data points, and then applies a threshold to the resulting vectors. It is based on an ensemble clustering method that performs 1D density partitioning to produce ensemble of clustering solutions. Then it assigns a unique prime number to the data points that exist in each partition as their labels. Consequently, a combination is performed by multiplying the labels of every data point in order to produce the absolute labels. The data points with identical absolute labels are fallen into the same cluster. The hierarchical property of the algorithm is intended to cluster complex data by zooming in each already formed cluster to find further sub-clusters. The algorithm is verified using several synthetic and real-world datasets. The results obtained show that the proposed method has a promising performance, compared to some well-known high-dimensional data clustering algorithms.

Keywords: *Ensemble Clustering, High-Dimensional Clustering, Hierarchical Clustering, Unsupervised Active Learning Method.*

1. Introduction

Soft computing algorithms that were originally developed for simulation of a human inference system are now widely used to solve complex problems in various fields of sciences such as system control, modeling, function approximation, prediction, decision-making process, classification, and clustering. One of the major fields of soft computing is fuzzy logic. The term "fuzzy logic" was introduced in 1965 by Lotfi A. Zadeh in his proposal on the fuzzy set theory [1, 2]. Fuzzy logic has many applications in the control theory, system modeling, machine learning and data mining.

Active learning method (ALM) was originally proposed by Shouraki et al. as one of the most effective algorithms in the fuzzy logic field [3]. ALM was developed based on the capability of the human brain in confronting with complex problems. It breaks a complex problem into several simplex problems, and then aggregates the results of these sub-problems. ALM is a powerful recursive fuzzy modeling without a considerable computational complexity, which has been used in many applications such as function modeling [4, 5], classification [6, 7], clustering [8-10] and control [11-13], with outstanding performance reports. In this paper, a high-dimensional clustering algorithm is introduced based on the concept of ALM.

Clustering is a data mining tool to uncover the existed but previously unknown patterns in a large dataset by grouping similar objects into the same cluster. Various clustering algorithms with different capabilities and different structures have been designed: full-dimensional versus sub-space high-dimensional, hierarchical versus non-hierarchical, crisp versus fuzzy, density-based versus grid-based, and methods based on partitioning [14-18].

Many clustering applications are subjected to high-dimensional data, in which each object is described by a large number of attributes. Examples of these data types can be found in the areas of computer vision applications, pattern recognition, and molecular biology [19]. Conventional clustering algorithms face many problems with high-dimensional datasets, and they do not scale well for efficient clustering [16, 20, 21]. The most prominent problem comes up when the distance between any two data points becomes almost the same, due to the natural sparsity of this type of data. Therefore, it is difficult to distinguish similar data points from the dissimilar ones. The time- and space-complexity of these algorithms to cluster high-dimensional datasets is another bottleneck, which usually results in failing to cluster such types of datasets. Apart from these two problems, high-dimensional data contains many irrelevant attributes, meaning that clusters are embedded in the sub-spaces of data space. As a result, high-dimensional specific clustering algorithms are essential to overcome these problems.

The difficulty that conventional clustering algorithms encounter in dealing with high-dimensional datasets motivates the concept of sub-space clustering and projected clustering [22-24], where the main goal is to find clusters ingrained within sub-spaces of the entire feature space, each with their own associated features. Sub-space clustering is the task of detecting all clusters in all sub-spaces. This means that a point may belong to multiple clusters, each of which exists in a different sub-space [25, 26]. This task results in overlapping clusters. Projected clustering, on the other hand, tries to assign each point to a unique cluster, resulting in non-overlapping clusters. Each algorithm has its own drawbacks. The interpretation of sub-space clustering algorithm results is hard due to the overlapping clusters and a large number of produced clusters. Projected clustering algorithm that produces non-overlapping clusters generally suffers from two common limitations. First of all, they usually have problems with sub-space clusters of significantly different dimensionality. Secondly, they cannot find clusters of different shapes and densities [27]. In addition, some of these algorithms suffer from time complexity, which grows dramatically by increasing the dimensions of datasets. Hence, sub-space clustering and projected clustering have some limitations with high-dimensional data.

The limitations of sub-space clustering algorithms show that a flexible and general high-dimensional clustering algorithm is required. Ensemble clustering has emerged as an important elaboration of the classical clustering problems. It overcomes the challenges of high-dimensional data and gives a high performance on the real world datasets. Therefore, in this work, we propose a hierarchical ensemble of projected clustering algorithm for high-dimensional data called HUALM (*high-dimensional unsupervised active learning method*) based on the ALM concepts, which has a linear time complexity with respect to the number of features and data points. The algorithm is run over some well-known datasets and it is compared with four different high-dimensional clustering algorithms: CLIQUE [22], PROCLUS [24], DP-clustering [28], and HDDC [29]. HUALM breaks the feature space into several single-feature spaces, and then blurs the data points as multiple 1D ink-drop patterns. This process is called "*ink drop spread*" (*IDS*). As individual ink drop patterns overlap, the density of overlapping portions becomes significantly higher. Therefore, they form a density vector named IDS-vector, with some hills and dales on the density bar graph. A threshold is applied on the IDS-vector, and then those data points that fall between each two adjacent dales are labeled with a unique prime number. This labeling procedure is done for all features. Finally, the labels of all features related to each data points are multiplied, and data points with the same multiplication results form the clusters. The proposed algorithm shows acceptable evaluation results, offering a better time complexity than CLIQUE. Furthermore, unlike sub-space clustering, it produces non-overlapping clusters so interpretation of the results is easy. The proposed algorithm can also find sub-space clusters of significantly different dimensionality. Moreover, the zooming process of the algorithm helps to find clusters with different densities. Consequently, HUALM solves the problems associated with sub-space and projected clustering algorithms.

The main contributions and advantages of the presented work can be summarized as follows:

- Presenting a novel algorithm for clustering high-dimensional data based on ALM.
- Hiring the idea of ink drop spread to address uncertainty in the data.
- Presenting a hierarchical clustering algorithm with a shallower depth tree structure, compared with the other common hierarchical methods, where a decision is made for a single data point at each iteration.
- Using prime-number labeling to ensemble the clusters in order to decrease the time/space complexity.

The remainder of this paper is organized as follows. In Section 2, the related works are discussed. The ALM concepts are reviewed in Section 3. In Section 4, HUALM algorithm is described. An experimental setup to assess the quality of our clustering algorithm is designed in Section 5. To show the performance of our approach in noisy environments, synthetic datasets with a variable amount of noise has been used in simulations, and HUALM is compared with four high-dimensional algorithms, namely CLIQUE, PROCLUS, DP-clustering, and HDDC. Finally, Section 6 provides a summary and conclusion.

2. Related works

In this section, we review the related clustering algorithms in the following categorizations: *sub-space* versus *projected*, *bottom-up* versus *top-down*, *cell-based* versus *density-based*, and *ensemble* clustering. In each part, the characteristics of our proposed algorithm are investigated.

2.1 Sub-space vs. projected

Sub-space clustering and projected clustering are extensions of traditional clustering, which are developed to cluster high-dimensional datasets. In sub-space clustering algorithms, an object might be reported to belong to several clusters in different sub-space projections, while in projected clustering algorithms each object belongs to one cluster in a sub-space. Sub-space clustering was first proposed by Aggarwal et al. in the CLIQUE approach [22] and after that, several sub-space clustering algorithms have been designed. MAFIA [23], SUBCLU [30], PreDeCon [31], FIRES [32], DUSC [33], and INSCY [34] are some examples of such algorithms. Projected clustering algorithms are partitioning methods that identify separate clusters in sub-space projections. PROCLUS (PROjected CLUstering) [24] is a modification of the K-Medoid algorithm for projected clustering. The PROCLUS algorithm discovers the sub-space dimensions of each cluster by evaluating the locality of the space adjacent to it. The number of clusters to be detected and the average dimensionality of the clusters are the objective functions that PROCLUS tries to satisfy. ORCLUS [35] adds the cluster-merging process to PROCLUS, and utilizes the principal components instead of the attributes. FINDIT [36] and SSPC [37] are variations of PROCLUS. FINDIT enhances efficiency and clustering accuracy by employing some heuristics. SSPC makes use of domain knowledge in the form of labeled objects and attributes in order to improve the clustering accuracy. FLOC [38], DOC [39], and MineClus [40] use the FP-tree for iterative-projected clustering. In DP-clustering [28], a projected clustering algorithm, at first, the data is projected to a $O(\log(n))$-dimensional space, and some candidate centers are calculated in the projected space. Then, a discrete clustering algorithm is hired to privately find k centers out of the candidate set. In HDDC [29], the Gausian mixture models (GMMs) are applied to sub-spaces with high data density. This approach could be counted as a sub-space clutering method. P3C [41], StatPC [42], FASTDOC [43], HARP [44], EPCH [45], and PCKA [46] are some other projected clustering algorithms.

The proposed HUALM algorithm can be used for both sub-space and projected clustering but here, we introduce it as a projected clustering algorithm. In the empirical studies, we compare the proposed method with CLIQUE, PROCLUS, DP-clustering, and HDDC, implemented according to [22], [24], [28], and [29], respectively.

2.2 Bottom-up vs. top-down

Sub-space clustering algorithms could also be divided into two groups, the bottom-up search and the top-down search methods [21]. The bottom-up search method creates a histogram for each dimension, and selects those parts whose densities exceed a given threshold (that helps to reduce the search space), then combines those parts to form a multi-dimensional grid. The main idea is: if a unit in k dimensions is dense, then there are dense units in all (k-1) dimensional projections of it. Furthermore, the bottom-up approach could result in overlapping clusters; it means that, there can be zero or more clusters for an instance. Setting proper values for parameters such as the grid size and the density threshold is difficult for these algorithms. On the other hand, the top-down sub-space clustering approach, at first, tries to find an initial approximation of the clusters in the full feature space with equally weighted dimensions. After that, a weight is assigned to each dimension for each cluster. In an iterative manner, these assigned weights are used to regenerate clusters, and then are updated. This approach is highly resource-consuming in terms of time complexity because it requires multiple iterations of clustering algorithms in the full set of dimensions. Therefore, many implementations of this approach use a sampling technique to improve the performance. In the top-down algorithms, each instance is assigned to only one cluster [24]. The number of clusters and the size of the sub-spaces are the most important parameters for the top-down algorithms, which are

often very difficult to determine as the initial parameters. Furthermore, since the size of a sub-space is a parameter in the top-down clustering algorithms, they tend to find clusters in sub-spaces with the size of this parameter's value. For techniques that use sampling, the size of the sample is another critical parameter, and can play a critical role in the quality of the results.

CLIQUE, ENCLUS [47], MAFIA, CBF [48], CLTree [49], MINECLUS, DOC, EPCH, and SCHISM [50] are the bottom-up clustering algorithms. CLIQUE and ENCLUS use a static-sized grid for dividing each dimension, while the others act wisely to determine the cut-points by analyzing the data. MAFIA, CBF, and EPCH use histograms for analyzing the density of data, related to each dimension. CLTree makes use of decision trees to find the best cut point. DOC does a random search by a maximum width and minimum number of instances per cluster. It can be considered as an earlier version of MINECLUS. However, the FP-tree structures help MINECLUS achieve better runtimes compared with DOC. EPCH computes low-dimensional histograms, and dense regions are identified in each histogram, based on iteratively lowering a threshold that depends on a user-specified parameter. A signature that consists of the identifiers of the dense regions that the data object belongs to, is derived for each data object. By matching the coefficients of the signatures, the similarity between two objects is measured, and then the objects are grouped due to their similarity until the desired number of clusters is obtained. The SCHISM algorithm enhances CLIQUE by adapting the density threshold to the sub-space dimensionality.

PROCLUS, ORCLUS, FINDIT, COSA [51], P3C and STATPC are the top-down clustering algorithms. PROCLUS, ORCLUS, FINDIT, and COSA determine the weights of instances for each cluster [52]. PROCLUS samples the data, then selects a set of k medoids and iteratively improves the clustering, similar to CLARANS [53]. ORCLUS is a non-axis-aligned method, which is the extended version of the PROCLUS algorithm. FINDIT is similar in structure to PROCLUS, which uses the DOD (*dimension oriented distance*) distance measure. The algorithm counts the number of dimensions on which two instances are within a threshold distance of each other. COSA is an iterative algorithm that that uses the k-nearest neighbors for each instance in the dataset to determine the weights for each dimension for that particular instance. P3C uses the $\chi 2$ statistical test and expectation-maximization to find the optimal final clustering solution. STATPC defines sub-space clusters that are statistically significant to eliminate redundant clusters.

It should be mentioned that although FIRES uses 1D histograms, it is not a bottom-up approach. It makes use of 1D histogram information (called base clusters) to jump directly to the interesting sub-spaces. Our proposed algorithm works similar to FIRES from this viewpoint. It uses 1D IDS-vectors to label data points, and then aggregates all information and jumps directly to the composed clusters for further searching.

2.3. Cell-based vs. density-based

One more simple classification of high-dimensional clustering algorithms, reported in [46], includes cell-based, density-based and clustering-oriented approaches. Cell-based approaches such as CLIQUE, DOC, MINECLUS, and SCHISM find static or dynamic-sized grids in each dimension, and combine them to make sub-space clusters. This category is very similar to the bottom-up category of [21]. Density-based methods such as SUBCLU, FIRES, and INSCY start by finding dense instances instead of dense bins in each attribute. These approaches require expensive computation to find dense instances; therefore, they are usually computationally infeasible. Clustering-based approaches focus on optimizing the overall clustering results rather than finding separate sub-space clusters. For example, in PROCLUS, the total number of clusters and the average dimensionality of the clusters should be optimized. Other examples of clustering-based methods are P3C and STATPC.

Our proposed algorithm can be categorized as both cell-based and density-based. It finds dynamic-sized grids in each dimension similar to cell-based, and since it uses a fuzzy membership function for each data point, each specific instance is affected by the data points that exist in its neighborhood. Furthermore, because of using the data-labeling technique, it can find arbitrary shaped clusters in each cell. Consequently, it is neither completely cell-based nor completely density-based. However, it does not require expensive computations as the density-based methods do.

2.4 Ensemble clustering

Ensemble clustering has emerged as an important elaboration of the classical clustering problems [54]. The clustering ensembles combine multiple clustering solutions of a given dataset into a single (consensus) clustering solution [55] to improve the robustness, accuracy, and quality of the clustering result, which is known by many different names such as consensus clustering, aggregation of clustering and clustering combination [55-60]. It

overcomes the challenges created by high-dimensional data and gives a high performance on the real world datasets [61-65]. Every clustering ensemble method is made up of two steps. The first step takes a dataset as input, and outputs an ensemble of clustering solutions. Once a collection of ensemble members has been generated, a suitable integration function is applied to combine them to produce a final clustering in the second step. The major challenge of ensemble clustering is the second step [55, 66-68], for some reasons. First, there is no label associated with each object. The number of produced clusters may differ across the different base solutions as well. Cluster labels are also symbolic. Therefore, ensemble clustering requires solving these problems in order to combine partitions.

Many different strategies have been used to generate the ensemble members, for example, applying different clustering algorithms, changing initialization or other parameters of a clustering algorithm, and using non-identical sets of the features. Obviously, the strategy of projecting objects on different feature-spaces is somehow related to sub-space clustering and has been reported in different ensemble clustering approaches [69-73].

After generating the initial partitions, a consensus function is used to combine them and produce a final partition, which is the main step in any clustering ensemble algorithm. The definition of the correspondence of labels for different partitions is not simple. In spite of the difficulty associated with this issue, there are several functions generating the ensemble of partitions. The most usual functions are based on the co-association [69, 75, 76], graph [77-79], mixture models [80], mutual information [81], and majority voting [82, 83] techniques. Recently, a distributed ensemble clustering method has been proposed that hires an entropy-based consensus function [84].

Our proposed algorithm uses non-identical sets of features in order to generate the initial partitions for cluster ensemble, and to make diversity. It uses a single feature space with all the data points. Using different subsets of features is very useful for clustering of high-dimensional data, because, as it works with subsets of the overall feature space, it reduces the computation burden of the clustering and time complexity of the algorithm. This is one of the advantages of ensemble clustering that uses the simple clustering trials and then aggregates the results.

HUALM algorithm uses data-labeling with the prime numbers and then multiplying the labels, as a consensus function. The advantage of using this method is to overcome some difficulties related to combining the results of clustering phases. As a consequence, the problems associated with the combination of ensemble clustering are easily solved in our proposed clustering algorithm.

3. Active learning method (ALM)

ALM was developed for the purpose of human brain learning simulation [3, 4]. The idea resembles the brain activity that stores the behavior of data instead of the exact data. It also looks at data while considering some kind of uncertainty, just as the brain does. ALM breaks a complex problem into several easier and understandable problems, and then aggregates the results; this is similar to the human brain activity, as well.

ALM breaks a *multiple-inputs-single-output* (MISO) system into several *single-input-single-output* (SISO) sub-systems and aggregates the behavior of sub-systems to obtain the final output (Figure 1). Each SISO sub-system is expressed as a data plane (called IDS plane) resulting from the projection of the data on each input-output plane. The effect of each data point is simulated with a fuzzy membership function called an *ink*. Figure 2a depicts two such membership functions over-lapped and aggregated. An IDS plane after applying IDS operator to the five data samples is shown in figure 2b.

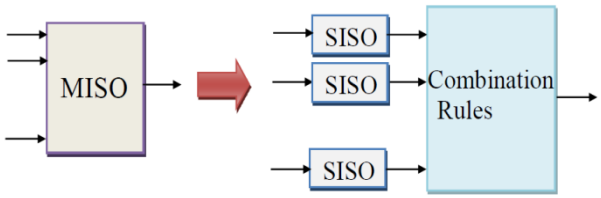


Figure 1. ALM breaks Multiple–Inputs–Single–Output (MISO) system into several Single–Input–Single–Output (SISO) subsystems and aggregates the behavior of subsystems to obtain the final output.

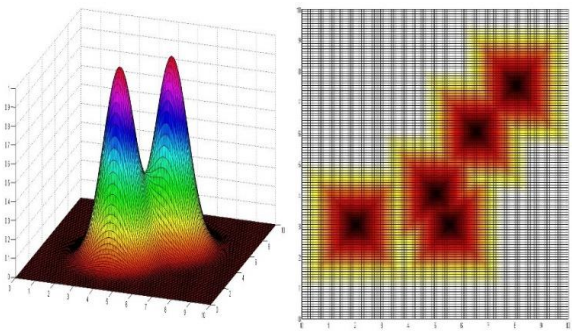


Figure 2. a) Ink stains with pyramid shape b) Five Inks are diffused on the plane.

The range of input variables should be quantized to *n* levels to gain a faster simulation time and less hardware for implementation. In order to have faster and more precise results, the algorithm requires to divide the input variables range into some intervals. Therefore, for each SISO sub-system there may be more than one IDS unit with a size less than the initial number of grids.

After mining some information from all of the IDS planes, the inference unit integrates this information. In inference unit of the ALM, a rule base is generated, while partial knowledge existing in the data samples are integrated.

4. High dimensional unsupervised active learning method

In this section, a novel projected clustering algorithm, called HUALM, which its fundamental concepts are driven from a powerful fuzzy modeling technique, ALM, is proposed. Sub-space and projected clustering methods are developed to deal with high-dimensional datasets. HUALM is an ensemble clustering method, which breaks high-dimensional data into several 1D data, and after analyzing each of them, aggregates the results to obtain the final clustering result; this concept is borrowed from the ALM algorithm. HUALM algorithm also uses the concept of diffusion of information that is one of the basic concepts of the ALM algorithm. This concept plays an important role in the HUALM algorithm. Each data point is associated with a fuzzy membership function, called ink-drop. Therefore, the projection of all data spots onto 1D spaces called IDS-vectors is smooth contrary to those clustering algorithms that consider the frequency of data (e.g. MAFIA, CBE, EPCH). In fact, considering a fuzzy membership function for each data point acts similar to running a low-pass filter over the data frequency. The histograms of the resulting IDS-vectors smoothly have some maxima and minima, which is similar to pictures of hills and dales. The smoothness of the resulting IDS-vectors makes it easy to find cutting points by searching for dales in the resulting hill and dale like IDS-vector. Therefore, HUALM is the extended version of ALM, which is adapted for clustering high-dimensional data.

There are other concepts in the HUALM algorithm, as well. It is a hierarchical clustering algorithm, which applies zooming process in already found clusters of upper levels. This zooming process can help the algorithm to find clusters with different densities and clusters that are not detected in the upper levels of the algorithm. The algorithm has a linear time complexity with respect to the number of features, number of data points, and number of levels, individually. However, usually no more than three or four levels are required. The algorithm also puts a threshold on each IDS-vector. Consequently, it can eliminate noise and outliers, and helps to ignore unrelated dimensions of the data. The idea of assigning a unique prime number to those data points that fall into a 1D cluster helps the algorithm to aggregate the results of all 1D clustering in an effective manner; this process is the aggregation part of our proposed ensemble clustering. Furthermore, although HUALM is a kind of cell-based approach, it neither assigns a cell to a cluster nor works with cells; it just assigns labels to data points and works with the labels. These concepts help the algorithm to be an effective high-dimensional clustering algorithm.

4.1. Algorithm

The pseudo-code and flowchart of HUALM algorithm are shown in figures 3 and 4, respectively. The body of the algorithm has three nested loops. The innermost loop finds out the dense partitions of each dimension and uniquely labels data points according to these partitions by prime numbers.

```
PROGRAM HUALM:
    Start
   Quantize input data;
   Initialize algorithm parameters;
   Labeling all data points with 1, and set num_clust(1)=1;
   i=1;
   Repeat
     For C=1: num_clust
       Datapoints=data points which their label is label©;
       For D=1:num_dim
         Spread ink drop for every Datapoint;
         Aggregate Ink-drop patterns on an IDS unit;
         Normalize IDS unit;
         Put threshold on IDS unit;
         Determine partitions by finding local minima of IDS
vectors;
         Labeling each partition's Datapoints with a unique
prime number;
       End for
       Multiplying labels of Datapoints;
     End for
     Update parameters (zoom in step);
     i=i+1;
     Num_clust(i)=count number of labels of all data points;
   Until(num_clust(i) = num_clust(i-1) or i>= max-level)
   Stop
End program.
```

Figure 3.The pseudocode of HUALM algorithm.

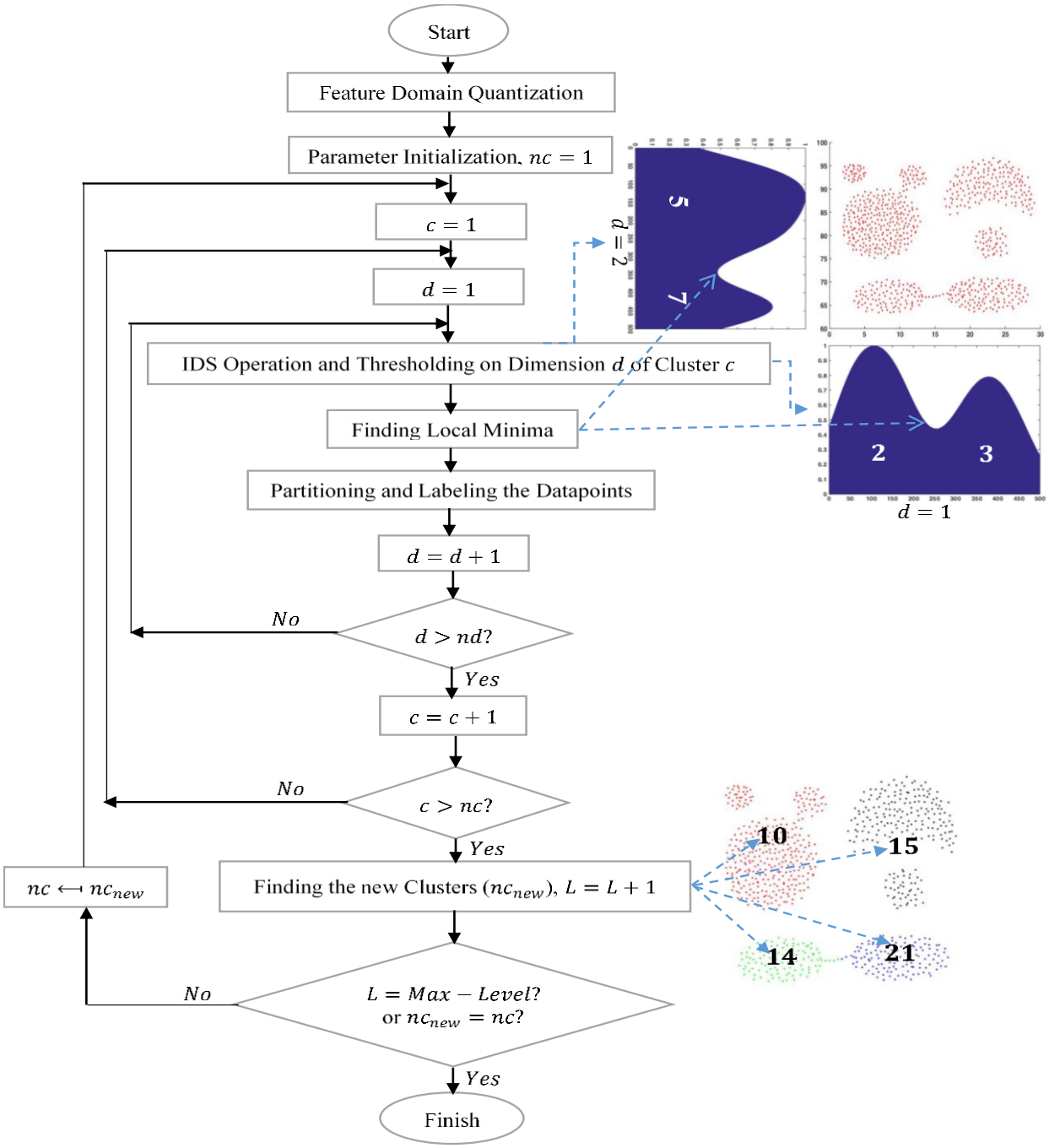


Figure 4. The flowchart of the proposed HUALM clustering algorithm.

Figure 5. Hierarchical structure of the HUALM algorithm.

The two inner loops in the algorithm form full dimensional clusters by assigning a unique label to every data point that belongs to one cluster. This process is done by multiplying the corresponding prime numbers that have been assigned to data points in the innermost loop of the algorithm. The third (and outer) loop separates data points with the same label more and more by zooming in each

cluster. This zooming process continues until for two iterations the number of produced clusters (labels) does not change or it exceeds the maximum number of levels. This step of the algorithm adds a hierarchical process to the algorithm (Figure 5). Consequently, the proposed algorithm is a hierarchical data-labeling ensemble of projected clustering algorithm for high-dimensional data. We describe the main steps of the algorithm in the following subsections.

4.1.1 Feature Domain Quantization

In HUALM, at first, the range of each input variable (feature) is divided into n disjoint intervals of equal size, in order to gain a faster simulation time and less hardware for implementation. It should be noted that the IDS units in the algorithm are 1D vectors that contain the aggregated effects of spreading data points on each dimension. Therefore, each IDS unit is a vector with n elements. Selecting a value for n mainly depends on the data characteristics, and it can be different for each dimension.

4.1.2 Parameter Initialization

In the initialization phase of the algorithm, parameters of the algorithm including: the *ink-diameter* (Id), threshold value (Th), and an updating factor(s), are determined. The Id determines the influence of a point in its neighborhood. The effect of considering each data point with an Id, and adding the effects of all points is the same as applying a low-pass filter on frequency of data in each dimension. The size of ink is somehow related to the size of the filter window. The threshold can be used for noise elimination, outlier rejection, and specifying cluster boundaries. On the other hand, updating factor is used for adjusting the parameters in each level of hierarchy. Selecting proper values for the parameters of the algorithm is essential to find out clusters with high accuracy and efficiency.

The labels of all data points are set to 1, at the beginning of the algorithm, which means $nc = 1$, where nc is the number of clusters. Since there is no information about their real category, all data points are considered as one cluster at the beginning of the algorithm.

4.1.3 IDS Operation

In this step, the ink drops of data points that belong to a cluster spread over 1D IDS units, in which there is one IDS unit per each feature. Therefore, d IDS units are formed, where d is the feature size. A 2D fuzzy membership function is considered for ink-drop pattern, where any type of distribution functions can be used for it. Figure 6 shows some examples of this pattern. The ink-drop of data points of a cluster are combined in the IDS units; therefore, d vectors that represent the accumulation effects of all data points of the cluster are obtained.

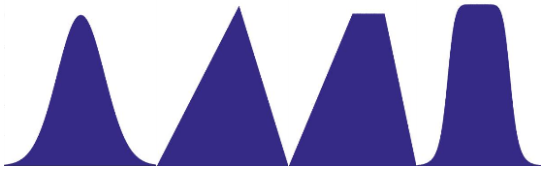


Figure 6. Some examples of ink drop patterns with the Ink-diameter of IDS.

In order to eliminate the features that do not belong to a cluster, or to remove noise or outliers, a threshold is applied on the IDS unit vector. Therefore, some data labels may be considered as (1). Normalization is necessary before applying the pre-defined threshold on the vector.

$$\exists x \in [1..n], D(x) < Th \Rightarrow D(x) = 0 \quad (1)$$

,where $D(x)$ is the IDS-vector.

Afterward, each IDS unit is partitioned via its local minima, and portions of data points that fall within each partition are labeled with a unique prime number. Therefore, each data point is labeled with at most d prime numbers, one label per feature. In order to find the local minima of an IDS-vector, we use the first derivative of the vector (i.e. $x'[n] = x[n] - x[n-1]$), then check the points where the slope turns from negative to positive (or maybe with some zeros between them).

Figure 7 shows an example of IDS-vector in a bar graph. After eliminating the unrelated data points by the thresholding process of (1), the IDS unit vector is partitioned with respect to the local minima of the vector. Then, data points that fall within the same partition are labeled with the same prime number.

4.1.4 Finding Clusters

The multiplication of D prime numbers of a data point determines the absolute label of it.

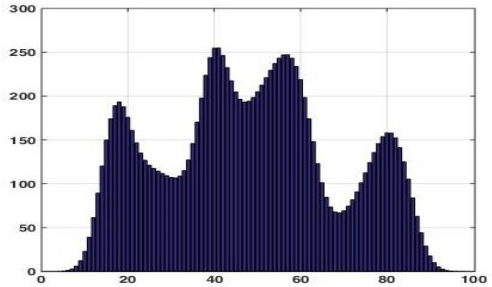


Figure 7. An IDS-vector, which has some hills (maxima) and dales (minima).

Data points with the same absolute label are considered as a cluster, which means that they are in the same partition of all features.
Subsequently, after finding clusters, the center of clusters can be found either by averaging their data points or by achieving the median, or by computing the median interval. Cluster centers can also be found from the generated IDS-vectors by defining the center of the cluster as the point that has maximum density in all dimensions.
After the first level of clustering, the algorithm zooms in each constituted cluster and clusters them again if possible, until either the number of clusters in two consecutive levels does not change or the algorithm reaches a predefined maximum number of levels. In most cases, no more than three levels are required to obtain an optimal result if the initial values of parameters are selected wisely. There could be other stopping criteria for the algorithm, for example, reaching an optimum clustering validation factor. Consequently, the proposed algorithm can discover clusters of data via a divisible-like algorithm. The hierarchical portion of the algorithm helps to find sub-clusters of an already found cluster by means of a zooming process.

4. 2. Time complexity

In order to compute the complexity of the proposed clustering algorithm, the two inner loops of algorithm are divided into three parts. The first part is spreading ink drops for N data points on D dimensions, which is $O(N \times D)$. The second part is scanning n arrays of the IDS-vectors and finding cutting points, which is $O(n \times D)$. The third part is multiplying data labels, which is $O(N \times (D-1))$. Thus the complexity of two inner loop of the algorithm is $O(2ND + nD - N)$, which equals to $O(N \times D)$. Therefore, the algorithm complexity is in the order of $O(N \times D \times L)$, where N is the size of dataset, D is the feature size, and L is number of zooming levels of the algorithm. The complexity shows that the proposed algorithm is linear in term of either dataset size, or feature size or number of levels of the algorithm, individually.
However, the overall time complexity of the algorithm for high-dimensional datasets is actually less than the proposed formula. Since in high-dimensional datasets, clusters exist in sub-spaces and thus the algorithm does not need to search the entire feature space after the first level. In addition, some data points are reported as outliers or noisy data points, so they will not involve in the clustering process of the next levels.

5. Experiments

In this section, a sequence of experiments are conducted for evaluating HUALM in terms of: scalability of the algorithm, noise and outlier immunity, and clustering quality. In addition, the hierarchical characteristics of the algorithm is shown by two experiments. The sensitivity analysis of the algorithm to its parameters is achieved by using different configurations of parameters and data conditions. Synthetic data are used for this part of the experiments. The ability of the proposed algorithm to work in noisy environments, detecting outliers, and finding clusters of sub-spaces has been investigated by some experiments. Clustering quality of the algorithm is measured by two clustering evaluation criteria, the accuracy, and F-measure. It is also compared with four different clustering algorithms including two sub-space clustering algorithms, CLIQUE and HDDC implemented according to [22] and [29], respectively, and two projected clustering algorithms, PROCLUS and DP-clustering, implemented according to [24] and [28], respectively. These algorithms have been chosen due to their fewer parameters and easier tuning compared to the others. Synthetic data and real databases are used for this part of the experiments. The characteristics of these datasets are summarized in table 1. The clustering evaluation methods can be divided into two categories: internal and external methods. Internal evaluation is attributed to the situation in which the clustering results are evaluated based on the data that is clustered itself. In external evaluation, the clustering results are evaluated based on the data that has known class labels. Using internal criteria in cluster evaluation is biased towards algorithms that use the same cluster model; consequently, the best way for unbiased evaluations so far has been the external evaluation measures [85, 86]. In this work, the external clustering measures accuracy and F-measure [87] are used to evaluate the proposed algorithm.
In this paper, accuracy is defined just the same as precision definition in [79], which measures the homogeneity of clusters with respect to priory known classes; it is given in (2). Having cluster C_i, let J_i denote the partition that contains the maximum number of points from C_i. It measures the fraction of points in C_i from the majority partition T_{Ji}.

$$Accuracy_i = 1/n_i \max_{j=1}^{k}\{n_{ij}\} = n_{ij}/n_i \quad (2)$$

The accuracy of clustering C is defined as the weighted sum of the cluster-wise accuracy values as in (3).

$$Accuracy = \sum_{i=1}^{r} n_i / n Accuracy_i \quad (3)$$

where, the ratio n_i/n denotes the fraction of points in cluster C_i.

The maximum value of accuracy is one, when each cluster comprises points from only one partition. Furthermore, when the number of clusters is equal to the actual number of partitions, then "accuracy=1" indicates perfect clustering, with a one-to-one correspondence between the clusters and partitions. However, a high accuracy can be achieved when the number of clusters is large; in particular, accuracy is 1 if each data point lies in a separate cluster. Thus, accuracy could not be used to trade-off the quality of the clustering against the number of clusters; however, F-measure allows us to make this trade-off. For a perfect clustering, when the number of clusters is equal to the actual number of partitions, the maximum value of F-measure is one.

5.1. Examples to show hierarchical property of proposed algorithm

The hierarchical behavior of the algorithm is shown by two examples. The datasets that are used for this purpose are aggregation and spiral datasets. As shown in figure 8, the HUALM algorithm can cluster the aggregation dataset after three levels. It shows that the algorithm works well for the clusters that have convex shapes. Likewise, figure 9 shows the result of the algorithm on the spiral dataset. For this dataset, the algorithm reaches the accuracy of one after three levels, although the number of

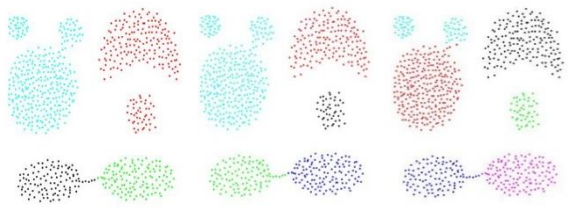


Figure 8. HUALM algorithm can thoroughly cluster the aggregation dataset after three levels, a) level-1, b) level-2, c) level-3.

generated clusters is greater than the real one.

Figure 10 shows the F-measure and accuracy measures for these two examples. As it shows, the two evaluation measures are almost the same for the aggregation dataset, because as the algorithm advances, the clusters are truly separated, and thus the clustering algorithm works well for this dataset. However, for the spiral dataset, as the algorithm advances, more clusters are composed and the purity of clusters increases, so the accuracy increases. However, as the number of clusters increases, the F-measure is decreased.

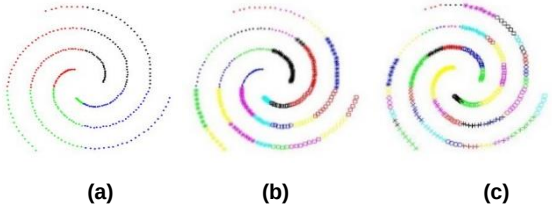


Figure 9. HUALM on spiral dataset reaches an accuracy of one after three levels, but with the number of clusters more than the actual one a) level-1, b) level-2, c) level-3.

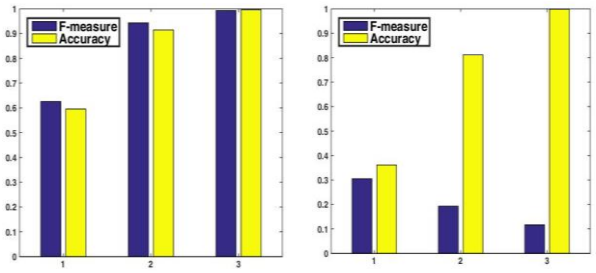


Figure 10. a) The accuracy and F-measure for aggregation dataset are near to one after running the three levels of the algorithm. b) Accuracy for the spiral dataset reaches one after running the three levels of the algorithm but the F-measure is low due to the large number of produced clusters.

5.2. Scalability of algorithm

This part of the experiment aims at checking the behavior of HUALM when the datasets have different numbers of instances and features. A data generator function has been written, which uniformly distributes clusters with Gaussian distribution in the feature space; the standard deviation of each cluster is random but restricted. This data generator function is used for evaluating the scalability of HUALM in terms of time and two quality measures, accuracy, and F-measure.

Figure 11 shows the scalability of the algorithm with respect to time. In all experiments, the number of clusters is constant and equal to 130. Figures 11a and 11b show that the execution time of the algorithm increases linearly with the number of data per cluster. They also show that the execution time of the algorithm is linearly correlated to the number of features. Therefore, as the complexity expression of the algorithm indicates, the time complexity of the algorithm is linear regarding the number of data points and the number of features.

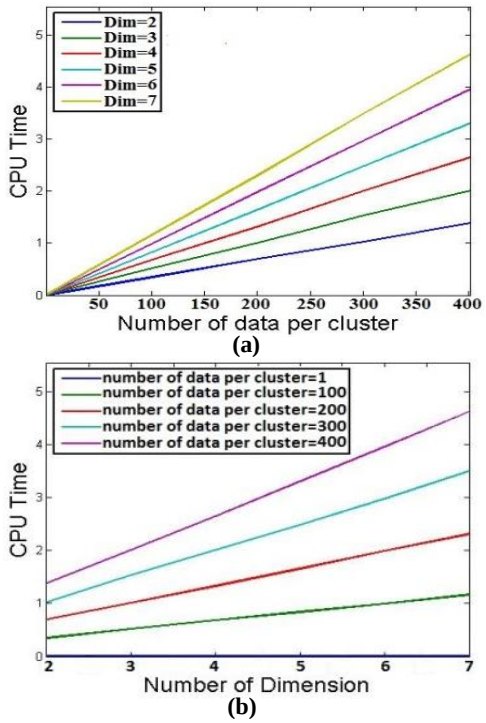


Figure 11. a) Time complexity of the algorithm is linearly related to the number of data per cluster b) The time complexity of the algorithm is linearly related to the number of features.

Figure 12 shows the time complexities of different levels of the algorithm. It indicates that the time complexities of different levels of the algorithm are almost the same. Therefore, as the complexity expression shows, the time complexity of the algorithm is linear regarding the number of levels.

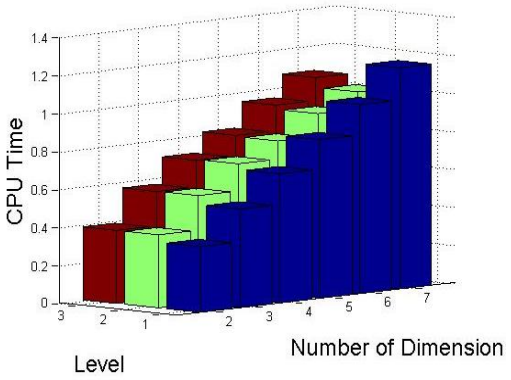


Figure 12. Time complexities of different levels of the algorithm are almost the same.

Figure 13 shows the scalability performance of the algorithm in terms of the accuracy and F-measure. The number of clusters is constant and equal to

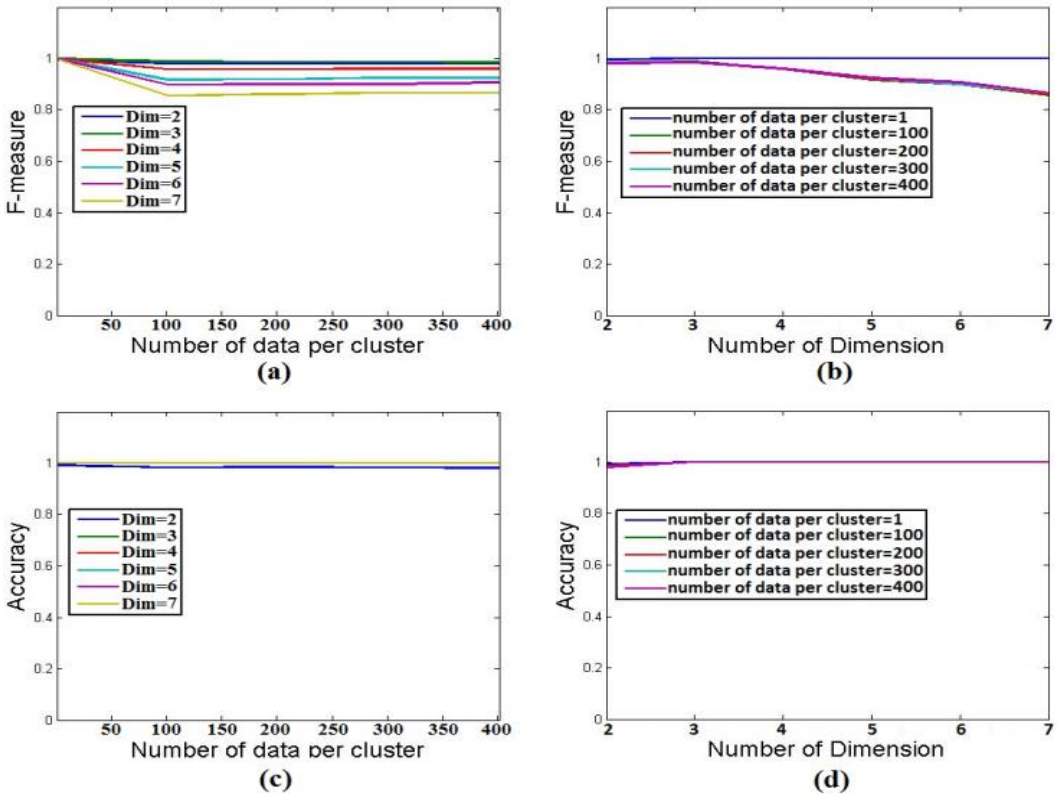
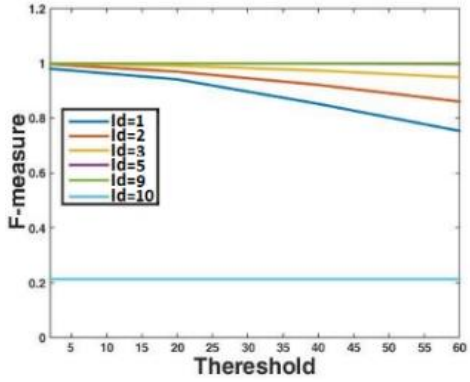


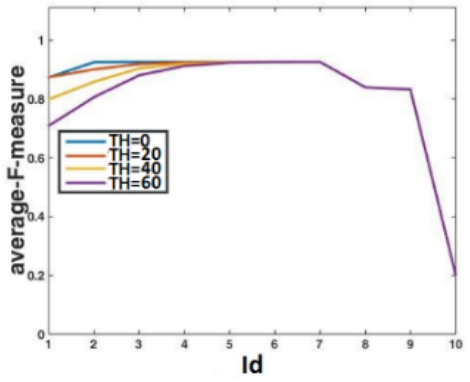
Figure 13. a) F-measure is almost constant when the number of data per clusters increases. Decreasing in F-measure as the number of dimensions increases is related to the generated dataset structure, and could improve by adjusting the algorithm parameters. b) If the algorithm parameters remain constant, the F-measure index will be degraded due to increase in feature size. c) Accuracy index does not change with the number of data per clusters. d) Accuracy does not change with the feature size of the generated dataset because in the HUALM algorithm, reaching an accuracy of one is almost always possible but usually with a lot of produced clusters.

130, and the parameters of algorithm are the same in all experiments.

Figures 13a and 13b show that the algorithm has a good and almost constant F-measure in case of increasing the number of data points per clusters, although as the number of dimensions increases, the F-measure index decreases. A reason for decreasing F-measure is related to the dataset because in higher dimensional generated datasets, the data in each cluster is farther from each other than in the lower dimensional one; consequently, for higher dimensions of the generated dataset, the algorithm splits the clusters more and more. Therefore, F-measure index which is sensitive to the number of clusters, reduces by increasing the dimensions of the generated dataset. However, as Figure 14 indicates, the F-measure for these types of datasets could be improved by adjusting the algorithm parameters. As figures 13c and 13d show, the accuracy of the algorithm for the generated dataset is near one. They show the ability of the algorithm to work with high-dimensional datasets with a good accuracy measure.



(a)



(b)

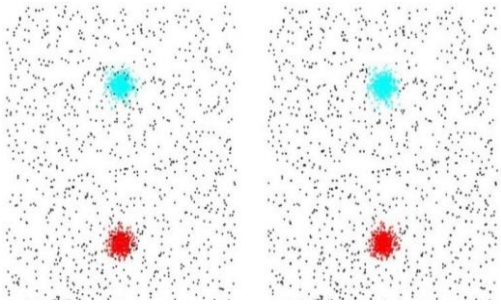
Figure 14. Effects of the algorithm parameters on the F-measure. a) As the threshold increases the F-measure decreases. b) For IDS between 4 and 8 the maximum F-measure value has been achieved.

Figure 14 shows the effects of the algorithm parameters on the quality of clustering by monitoring the F-measure evaluation index. In this experiment, the number of features is 6, the number of clusters is 130, and the number of data points in each cluster is 200. The algorithm has been run 10 times and the best and average F-measures have been reported in figures 14a and 14b respectively.

As shown in figures 13, the F-measure for higher dimensions decreases when the parameters are constant. However, in the experiment of figure 14 it is shown that by adjusting the parameters, better quality measures can be achieved. Figure 14.a shows that with an IDS from 5 to 9, the F-measure index can reach to its maximum value. However, for this dataset, with the IDS smaller than 5 and bigger than 9, the F-measure index is smaller than its maximum value. In addition, it shows the impact of the threshold on the F-measure index for this dataset, where it is reduced by increasing in the threshold. That is, as the threshold increases, more data points are eliminated and thus, the F-measure will be decreased. Therefore, by adjusting the parameters of the algorithm, better clustering results could be achieved.

5.3. Noise and outlier immunity

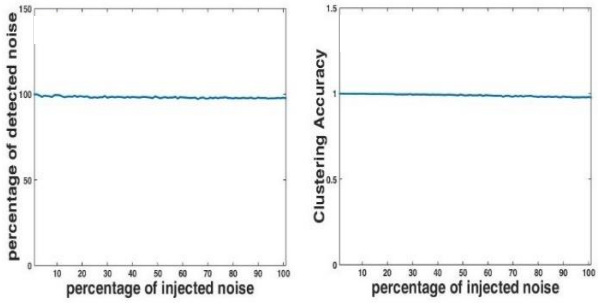
In order to show the invariance of our approach with respect to noise, a 2D dataset having two clusters with Gaussian distribution has been used, in which different percentages of noise ranging from 1% to 100% of the total number of data points have been injected. The white noise has been injected, and the algorithm has been run ten times over the datasets containing Gaussian clusters with random standard deviations. Figure 15.a shows the original dataset with 100% injected white-noise, and figure 15b shows the result of HUALM on this synthetic dataset. This example shows the power of HUALM in distinguishing clusters in noisy environments.



(a) (b)

Figure 15. HUALM result on a random Gaussian dataset with 100% of total number of data, injected white noise. a) Two clusters have been shown in red and cyan, and the white noise in black. b) Output of the algorithm shows the ability of the algorithm to eliminate noise and outliers.

Figure 16 shows the ability of the algorithm to find noisy data and outliers, while preserving a high accuracy. Figure 16.a, shows the percentage of detected noise vs. the percentage of injected noise; and figure 16b shows the HUALM clustering accuracy versus the percentage of injected noise.



(a) **(b)**

Figure 16. Performance of the algorithm to find noisy data and outliers, while keeping a high accuracy. a) The percentage of detected noise vs. the percentage of injected noise. b) Clustering accuracy vs. the percentage of injected noise.

5.4. Clustering Quality

In this part of the experiments, the clustering quality of HUALM is measured and compared with CLIQUE, PROCLUS, DP-clustering and HDDC. These algorithms are high-dimensional clustering methods. Synthetic and real datasets are used in this part of the experiments.

The characteristics of these datasets are summarized in table 1.

Table 1. Summary of the datasets.

Data set	Number of objects	Number of features	Number of classes
Aggregation	788	2	7
spiral	312	2	3
Chainlink	1000	3	2
Haberman	306	3	2
Seeds	210	7	3
Glass	214	9	7
Waveform	5000	21	3
Breast	198	33	2

The clustering quality measures have been calculated for these algorithms, and the results are summarized in table 2. It should be noticed that each algorithm has been run with different parameters and the best concurrent evaluation indices are reported.

According to table 2, the HUALM algorithm has the best accuracy compared to the other four clustering algorithms. As discussed earlier, HUALM can potentially gain an accuracy near one in exchange for the higher number of clusters. Since the number of clusters in our algorithm is usually greater than the real number of clusters, in some cases, the F-measure is less than the other four algorithms.

Table 2. Validity indexes for clustering methods.

	Accuracy					F-measure				
Data set	**HUALM**	**CLIQUE**	**PROCLUS**	**HDDC**	**dp-clustering**	**HUALM**	**CLIQUE**	**PROCLUS**	**HDDC**	**dp-clustering**
Aggregation	0.9970	0.9210	0.9940	0.9112	0.7272	0.9950	0.7082	0.6060	0.8345	0.6821
Spiral	1.0000	0.9020	0.9500	0.9359	0.8820	0.1596	0.1776	0.1410	0.1847	0.2171
Chainlink	1.0000	0.9952	1.0000	1.0000	0.9060	0.2800	0.2800	0.360	0.2119	0.3935
Haberman	0.7605	0.7494	0.7544	0.7549	0.7353	0.8197	0.8092	0.5726	0.7779	0.8355
Seeds	0.9164	0.8900	0.8549	0.5238	0.8324	0.5751	0.4130	0.7136	0.5953	0.5322
Glass	0.7918	0.4631	0.7567	0.5047	0.5733	0.3986	0.4237	0.3589	0.4271	0.3927
Waveform	0.6906	0.3410	0.3457	0.5530	0.4121	0.4394	0.4880	0.3425	0.5632	0.3432
Breast	0.8807	0.7308	0.7574	0.7626	0.7530	0.4376	0.6281	0.5917	0.6054	0.6837

6. Conclusion

In this work, a novel clustering algorithm, namely HUALM, was introduced for high-dimensional data, based on the *active learning algorithm*.
The proposed algorithm clusters the 1D IDS-vectors, which are the density vectors of the grids after ink drop spread of data points and projecting the results. The algorithm then finds the cutting points by exploring the local minima of the IDS-vectors. It assigns labels to the data points according to their position in each dimension, with prime numbers. Eventually, it aggregates the results of 1D clustering by multiplying the assigned labels. Noise mitigation and outlier elimination are carried out through a thresholding mechanism. It also helps the algorithm to eliminate the unrelated dimensions of clusters to lower the complexities. In order to improve the performance of the proposed algorithm, it has a hierarchical zooming process, which helps the complex datasets clustering. The complexity of the algorithm was computed and confirmed via experiments. The effects of the parameters on the clustering results were also investigated. The simulation results confirmed the efficiency of the proposed algorithm.

References

[1] Stanford Encyclopedia of Philosophy Website (2016), Available: https://plato.stanford.edu/ entries/logic-fuzzy/, retrieved 2020-3-27.

[2] Zadeh, L. A. (1965s). Fuzzy sets. Information and control, vol. 8, no. 3, pp. 338-353.

[3] Shouraki, S. B. (1999). Recursive Fuzzy Modeling Based on Fuzzy Interpolation. Journal of Advanced Computational Intelligence, vol. 3, no. 2, pp. 114-125.

[4] Murakami, M., & Honda, N. (2007). A study on the modeling ability of the IDS method: A soft computing technique using pattern-based information processing. Int. Journal of Approximate Reasoning, vol. 45, no. 3, pp. 470-487.

[5] Afrakoti, I. E. P., Shouraki, S. B., Bayat, F. M., Gholami, M. (2017). Using a memristor crossbar structure to implement a novel adaptive real-time fuzzy modeling algorithm. Fuzzy Sets and Systems, vol. 307, no. 1, pp. 115-128.

[6] Murakami, M., & Honda, N. (2005). Classification performance of the IDS method based on the two-spiral benchmark. IEEE International Conference on Systems, Man and Cybernetics, Waikoloa, USA, 2005, vol. 4, pp. 3797-3803.

[7] Firouzi, M., Shouraki, S. B., & Afrakoti, I. E. P. (2014). Pattern analysis by active learning method classifier. Journal of Intelligent & Fuzzy Systems, vol. 26, no. 1, pp. 49-62.

[8] Javadian, M., Shouraki, S. B., & Kourabbaslou, S. S. (2017). A novel density-based fuzzy clustering algorithm for low dimensional feature space. Fuzzy Sets and Systems, vol. 318, no. 1, pp. 34-55.

[9] Javadian, M., & Shouraki, S. B. (2017). UALM: Unsupervised active learning method for clustering low-dimensional data. Journal of Intelligent & Fuzzy Systems, vol. 32, no. 3, pp. 2393-2411.

[10] Javadian, M., Malekzadeh, A., Heydari, G., & Shouraki, S. B. (2019). A clustering fuzzification algorithm based on ALM. Fuzzy Sets and Systems, Vol. 389, no. 1, pp. 93-113.

[11] Shahdi, S. A., & Shouraki, S. B. (2002). Supervised active learning method as an intelligent linguistic controller and its hardware implementation. 2nd IASTEAD International Conference on Artificial Intelligence and Applications (AIA'02), Malaga, Spain, pp. 453-458.

[12] Sakurai, Y. (2005). A study of the learning control method using PBALM-a nonlinear modeling method. Ph.D. Dissertation, University of Electro-Communications, Tokyo, Japan.

[13] Firouzi, M., Shouraki, S. B., & Conradt, J. (2014). Sensorimotor Control Learning Using a New Adaptive Spiking Neuro-Fuzzy Machine, Spike-IDS and STDP. 24th International Conference on Artificial Neural Networks and Machine Learning (ICANN), Hamburg, Germany, pp. 379-386.

[14] Kaufman, L., & Rousseeuw, P. J. (2009). Finding groups in data: an introduction to cluster analysis. John Wiley & Sons Inc, Hoboken, NJ, pp- 126-162.

[15] Berkhin, P. (2006). A survey of clustering data mining techniques, In: Kogan, J., Nicholas, C., & Teboulle, M. Grouping multidimensional data, Springer, Berlin, Heidelberg, pp. 25-71.

[16] Aggarwal, C. C., & Reddy, C. K. (2014). Data clustering: Algorithms and applications. Chapman&Hall Press, Florida, pp. 107-320.

[17] Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques, 3rd ed. Morgan Kaufman Pubs, Massachusetts, pp. 443-493.

[18] Gan, G., Ma, C., & Wu, J. (2007). Data clustering: theory, algorithms, and applications. SIAM Press, Virginia, pp. 109-320.

[19] Jiang, D., Tang, C., & Zhang, A. (2004). Cluster analysis for gene expression data: a survey. IEEE

Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1370-1386.

[20] Jahirabadkar, S., & Kulkarni, P. (2013). Clustering for high dimensional data: density based subspace clustering algorithms. Int. Journal of Computer Applications, vol. 63, no. 20, pp. 29-35.

[21] Parsons, L., Haque, E., & Liu, H. (2004). Subspace clustering for high dimensional data: a review. ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 90-105.

[22] Agrawal, R., Gehrke, J. E., Gunopulos, D., & Raghavan, P. (1999). Automatic subspace clustering of high dimensional data for data mining applications. ACM SIGMOD International Conference on Management of data, Pennsylvania, USA, pp. 94-105.

[23] Goil, S., Nagesh, H., & Choudhary, A. (1999). MAFIA: Efficient and scalable subspace clustering for very large data sets. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, California, USA, Vol. 443, p. 452-463.

[24] Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., & Park, J. S. (1999). Fast algorithms for projected clustering. ACM SIGMoD Record, vol. 28, no. 2, pp. 61-72.

[25] Moise, G., Sander, J., & Ester, M. (2008). Robust projected clustering. Knowledge and Information Systems, vol. 14, no. 3, pp. 273-298.

[26] Clustering High-dimensional Data, Website (2020), available: http://en.wikipedia.org/wiki/ Clustering_high-dimensional_data, retrieved 2020-3-27.

[27] Achtert, E., Böhm, C., Kriegel, H.-P., Kröger, P., Müller-Gorman, I., & Zimek, A. (2007). Detection and visualization of subspace cluster hierarchies, In LNCS: Advances in Databases: Concepts, Systems and Applications, Springer Pubs. Berlin, Heidelberg, pp. 152-163.

[28] Balcan, M.-F., Dick, T., Liang, Y., Mou, W., & Zhang, H. (2017). Differentially private clustering in high-dimensional euclidean spaces. 34th International Conference on Machine Learning, Sydney, Australia, pp. 322-331.

[29] Bouveyron, C., Girard, S., & Schmid, C. (2007). High-dimensional data clustering. Computational Statistics & Data Analysis, vol. 52, no. 1, pp. 502-519.

[30] Kailing, K., Kriegel, H.-P., & Kröger, P. (2004). Density-connected subspace clustering for high-dimensional data. SIAM International Conference on data mining, Florida, USA, pp. 246-256.

[31] Bohm, C., Railing, K., Kriegel, H.-P., & Kroger, P. (2004). Density connected clustering with local subspace preferences. 4th IEEE International Conference on Data Mining (ICDM'04), Brighton, UK, pp. 27-34.

[32] Kriegel, H.-P., Kroger, P., Renz, M., & Wurst, S. (2005). A generic framework for efficient subspace clustering of high-dimensional data. 5th IEEE International Conference on Data Mining (ICDM'05), Houston, USA, pp. 1-8.

[33] Assent, I., Krieger, R., Müller, E., & Seidl, T. (2007). DUSC: Dimensionality unbiased subspace clustering. 7th IEEE International Conference on Data Mining (ICDM 2007), Omaha, NE, USA, pp. 409-414.

[34] Assent, I., Krieger, R., Müller, E., & Seidl, T. (2008). INSCY: Indexing subspace clusters with in-process-removal of redundancy. 8th IEEE International Conference on Data Mining, Miami, Florida, USA, pp. 719-724.

[35] Aggarwal, C. C., & Yu, P. S. (2000). Finding generalized projected clusters in high dimensional spaces. ACM SIGMOD Int. Conf. on Management of data, Dallas, Texas, USA, pp. 70-81.

[36] Woo, K.-G., Lee, J.-H., Kim, M.-H., & Lee, Y.-J. (2004). FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting, Information and Software Technology, vol. 46, no. 4, pp. 255-271.

[37] Yip, K., Cheung, D. W., & Ng, M. K. (2005). On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. 21st Int. Conf. on Data Engineering (ICDE'05), Tokoyo, Japan, pp. 329-340.

[38] Demirbas, M., Arora, A., Mittal, V., & Kulathumani, V. (2004). Design and analysis of a fast local clustering service for wireless sensor networks. 1st Int. Conf. on Broadband Networks, San Jose, CA, USA, pp. 700-709.

[39] Procopiuc, C. M., Jones, M., Agarwal, P. K., & Murali, T. (2002). A Monte Carlo algorithm for fast projective clustering, the ACM SIGMOD Int. Conf. on Management of Data, Madison Wisconsin, USA, pp. 418-427.

[40] Yiu, M. L., & Mamoulis, N. (2003). Frequent-pattern based iterative projected clustering, 3rd IEEE Int. Conf. on Data Mining, Washington DC, USA, pp. 689-692.

[41] Moise, G., Sander, J., & Ester, M. (2006). P3C: A robust projected clustering algorithm. 6th Int. Conf. on

Data Mining (ICDM'06), Hong Kong, China, pp. 414-425.

[42] Moise, G., & Sander, J. (2008). Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering, 14th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, pp. 533-541.

[43] Yiu, M. L., & Mamoulis, N. (2005). Iterative projected clustering by subspace mining. IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 2, pp. 176-189.

[44] Yip, K. Y., Cheung, D. W., & Ng, M. K. (2004). Harp: A practical projected clustering algorithm. IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1387-1397.

[45] Ng, E. K. K., Fu, A.-C., & Wong, R.-W. (2005). Projective clustering by histograms. IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 3, pp. 369-383.

[46] Bouguessa, M., & Wang, S. (2008). Mining projected clusters in high-dimensional spaces. IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 4, pp. 507-522.

[47] Cheng, C.-H., Fu, A. W., & Zhang, Y. (1999). Entropy-based subspace clustering for mining numerical data. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, San Diego, California, USA, pp. 84-93.

[48] Chang, J.-W., & Jin, D.-S. (2002). A new cell-based clustering method for large, high-dimensional data in data mining applications. ACM Symposium on Applied Computing, Madrid, Spain, pp. 503-507.

[49] Liu, B., Xia, Y., & Yu, P. S. (2000). Clustering through decision tree construction. 9th Int. Conf. on Information and Knowledge Management, McLean, Virginia, USA, pp. 20-29.

[50] Sequeira, K., & Zaki, M. (2004). SCHISM: A new approach for interesting subspace mining. 4th IEEE Int. Conf. on Data Mining (ICDM'04), Brighton, UK, pp. 137-160.

[51] Wang, H., Wang, W., Yang, J., & Yu, P. S. (2002). Clustering by pattern similarity in large data sets. ACM SIGMOD Int. Conf. on Management of Data, Madison, Wisconsin, USA, pp. 394-405.

[52] Vahdat, A. R. (2013). Symbiotic Evolutionary Subspace Clustering (S-ESC), Ph.D. Dissertation, Dalhousie University, Halifax, Canada.

[53] Ng, R. T., & Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 5, pp. 1003-1016.

[54] Zheng, L., Li, T., & Ding, C. (2010). Hierarchical ensemble clustering. 10th IEEE Int. Conf. on Data Mining, Sydney, Australia, pp. 1199-1204.

[55] Strehl, A., & Ghosh, J. (2002). Cluster ensembles---a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research, vol. 3, no. 1, pp. 583-617.

[56] Fern, X. Z., & Brodley, C. E. (2003). Random projection for high dimensional data clustering: A cluster ensemble approach. 20th Int. Conf. on Machine Learning (ICML-03), Washington, DC, USA, pp. 186-193.

[57] Fred, A. L., & Jain, A. K. (2002). Data clustering using evidence accumulation. Int. Conf. on Object Recognition Supported by User Interaction for Service Robots, Quebec, Canada, Vol. 4, pp. 276-280.

[58] Filkov, V., & Skiena, S. (2004). Integrating microarray data by consensus clustering. Int. Journal of Artificial Intelligence Tools, vol. 13, no. 04, pp. 863-880.

[59] Topchy, A. P., Law, M. H., Jain, A. K., & Fred, A. L. (2004). Analysis of consensus partition in cluster ensemble. 4th IEEE Int. Conf. on Data Mining (ICDM'04), Brighton, UK, pp. 225-232.

[60] Gionis, A., Mannila, H., & Tsaparas, P. (2007). Clustering aggregation. ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 1, no. 1, pp. 4-es.

[61] He, Y., Wang, J., Qin, L.-x., Mei, L., Shang, Y.-f., & Wang, W.-f. (2013). A HK clustering algorithm based on ensemble learning. IET Int. Conf. on Smart and Sustainable City, Shanghai, China, pp. 276-281.

[62] Paithankar, R., & Tidke, B. (2015). A HK clustering algorithm for high dimensional data using ensemble learning. Available online: https://arxiv.org/abs/1501.02431.

[63] Tidke, B., Mehta, R., & Rana, D. (2012). A novel approach for high dimensional data clustering. Int. Journal of Engineering Science and Advanced Technology (IJESAT), vol. 2, no. 3, pp. 264-267.

[64] Kong, Z., Wei, L., Yang, S., Guan, D., & Cai, Z. (2008). A novel clustering-ensemble approach. 2nd Int. Conf. on Bioinformatics and Biomedical Engineering, Shanghai, China, pp. 722-724.

[65] Viswanath, P., & Jayasurya, K. (2006). A fast and efficient ensemble clustering method. 18th Int. Conf. on Pattern Recognition (ICPR'06), Hong Kong, China, Vol. 2, pp. 720-723.

[66] Nisha, M., Mohanavalli, S., & Swathika, R. (2013). Improving the quality of clustering using cluster ensembles. IEEE Conf. on Information & Communication Technologies, Thuckalay, Tamil Nadu, India, pp. 88-92.

[67] Ghaemi, R., Sulaiman, M. N., Ibrahim, H., & Mustapha, N. (2009). A survey: clustering ensembles techniques. World Academy of Science, Engineering and Technology, vol. 50, pp. 636-645.

[68] Al-Razgan, M. S. (2008). Weighted clustering ensembles. Ph.D. Dissertation, George Mason University, Virginia, USA.

[69] Fern, X. Z., & Lin, W. (2008). Cluster ensemble selection. Statistical Analysis and Data Mining: The ASA Data Science Journal, vol. 1, no. 3, pp. 128-141.

[70] Fern, X. Z., & Brodley, C. E. (2003). Random projection for high dimensional data clustering: A cluster ensemble approach. 20th Int. Conf. on Machine Learning (ICML-03), Washington, DC, USA, pp. 186-193.

[71] Bertoni A., Valentini G. (2006) Ensembles Based on Random Projections to Improve the Accuracy of Clustering Algorithms. In: Apolloni B., Marinaro M., Nicosia G., Tagliaferri R. (eds) Neural Nets. WIRN 2005, NAIS 2005. Lecture Notes in Computer Science,. Springer, Berlin, Heidelberg, pp. 31-37.

[72] Topchy, A., Jain, A. K., & Punch, W. (2005). Clustering ensembles: Models of consensus and weak partitions. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 12, pp. 1866-1881.

[73] Kriegel, H.-P., & Zimek, A. (2010). Subspace clustering, ensemble clustering, alternative clustering, multiview clustering: what can we learn from each other. 1st Int. Workshop on Discovering, Summarizing and Using Multiple Clusterings (MultiClust), held in conjunction with the 16th ACM SIGKDD, Washington, DC, USA, pp. 1-6.

[74] Fred, A. (2001). Finding consistent clusters in data partitions. Int. Workshop on Multiple Classifier Systems, Cagliari, Italy, pp. 309-318.

[75] Fred, A. L., & Jain, A. K. (2002). Data clustering using evidence accumulation. Int. Conf. on Object Recognition Supported by User Interaction for Service Robots, Quebec, Canada, vol. 4, pp. 276-280.

[76] Ana, L., & Jain, A. K. (2003). Robust data clustering. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, Madison, WI, USA, vol. 2, pp. 1-6.

[77] Ghosh, J., & Acharya, A. (2011). Cluster ensembles. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 1, no. 4, pp. 305-315.

[78] Karypis, G., & Kumar, V. (1998). A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal of Scientific Computing, vol. 20, no. 1, pp. 359-392.

[79] Fern, X. Z., & Brodley, C. E. (2004). Solving cluster ensemble problems by bipartite graph partitioning. 21th Int. Conf. on Machine Learning, Alberta, Canada, pp. 36-43.

[80] Topchy, A., Jain, A. K., & Punch, W. (2004). A mixture model for clustering ensembles. SIAM Int. Conf. on Data Mining, Florida, USA, pp. 379-390.

[81] Topchy, A., Jain, A. K., & Punch, W. (2003). Combining multiple weak clusterings. 3rd IEEE Int. Conf. on Data Mining, Melbourne, FL, USA, pp. 331-338.

[82] Dudoit, S., & Fridlyand, J. (2003). Bagging to improve the accuracy of a clustering procedure. Bioinformatics, vol. 19, no. 9, pp. 1090-1099.

[83] Fischer, B., & Buhmann, J. M. (2003). Bagging for path-based clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 11, pp. 1411-1415.

[84] Owhadi-Kareshki, M. (2019). Entropy-based Consensus for Distributed Data Clustering. Journal of AI and Data Mining, vol. 7, no. 4, pp. 551-561.

[85] Zaki, M. J., Meira Jr, W., & Meira, W. (2014). Data mining and analysis: fundamental concepts and algorithms. Cambridge University Press, UK, pp. 333-461

[86] Rendón, E., Abundez, I., Arizmendi, A., & Quiroz, E. M. (2011). Internal versus external cluster validation indexes. Int. Journal of Computers and Communications, vol. 5, no. 1, pp. 27-34.

[87] Larsen, B., & Aone, C. (1999). Fast and effective text mining using linear-time document clustering. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, San Diego, California, USA, pp. 16-22.

نشریه هوش مصنوعی و داده کاوی

روش یادگیری فعال غیر نظارتی با ابعاد بالا

وحید قاسمی[1*]، محمد جوادیان و سعید باقری شورکی

[1] گروه مهندسی کامپیوتر، دانشگاه صنعتی کرمانشاه، کرمانشاه، ایران.

[2] گروه مهندسی الکترونیک، دانشگاه صنعتی شریف، تهران، ایران.

ارسال ۲۰۱۹/۰۲/۰۹؛ بازنگری ۲۰۱۹/۱۲/۳۱؛ پذیرش ۲۰۲۰/۰۲/۲۶

چکیده:

در این مقاله یک الگوریتم ترکیب سلسله مراتبی برای خوشه‌بندی تصویری داده‌های با ابعاد بالا پیشنهاد شده است. مفهوم پایه‌ای این الگوریتم مبتنی بر روش یادگیری فعال ا ست، که یک رویکرد یادگیری فازی الهام گرفته شده از برخی ویژگی‌های رفتاری عملکرد مغز ان سان می‌با شد. روش یادگیری فعال غیر نظارتی با ابعاد بالا یک الگوریتم خوشه‌بندی است که تاثیر نقاط داده را به شکل الگوهای قطره جوهر پراکنده کرده تا همه آن تاثیرات خلاصه سازی شده، و سپس یک حد آستانه را بر بردارهای حاصل اعمال می‌کند. این رویکرد مبتنی بر یک روش خوشه‌بندی ترکیبی است که از بخش بندی تراکمی یک بعدی برای ایجاد ترکیبی از راهکارهای خوشـــه‌بندی بهره می‌گیرد. ســـپس، به نقاط داده موجود در هر بخش یک عدد اول را، به عنوان برچسب، تخصیص می‌دهد. در ادامه، یک عمل ترکیب، با ضرب کردن برچسب‌های نقاط داده، انجام گرفته تا برچسب‌های قطعی ایجاد شوند. نقاط داده با برچسب‌های قطعی یکسان در یک خوشه قرار داده می‌شوند. ویژگی سلسله مراتبی این الگوریتم به منظور خوشه‌بندی داده‌های پیچیده در نظر گرفته شده است، تا با بزرگنمایی خوشه‌های تشکیل شده قبلی، زیر-خوشه‌های بیشتری به دست آیند. این الگوریتم با استفاده از چندین مجموعه داده تحلیلی و واقعی ارزیابی شده است. نتایج به دست آمده نشان می‌دهند که در مقایسه با برخی الگوریتم‌های خوشه‌بندی شناخته شده برای داده‌های با ابعاد بالا، روش پیشنهادی عملکرد مطلوبی از خود نشان می‌دهد.

کلمات کلیدی: خوشه‌بندی ترکیبی، خوشه‌بندی با ابعاد بالا، خوشه‌بندی سلسله مراتبی، روش یادگیری فعال غیر نظارتی.