Autonomous Data-driven Clustering for Live Data Stream

Xiaowei Gu and Plamen P. Angelov
Data Science Group, School of Computing and Communications,
Lancaster University,
Lancaster, LA1 4WA, UK.
E-mail: p.angelov@lancaster.ac.uk

***Abstract*— In this paper, a novel autonomous data-driven clustering approach, called *AD_clustering*, is presented for live data streams processing. This newly proposed algorithm is a fully unsupervised approach and entirely based on the data samples and their ensemble properties, in the sense that there is no need for user-predefined or problem-specific assumptions and parameters, which is a problem most of the current clustering approaches suffer from. Moreover, the proposed approach automatically evolves its structure according to the experimentally observable streaming data and is able to recursively update its self-defined parameters using only the current data sample, meanwhile, discards all the previous data samples. Experimental results based on benchmark datasets exhibit the higher performance of the proposed fully autonomous approach compared with the comparative approaches with user- and problem- specific parameters to be predefined. This new clustering algorithm is a promising tool for further applications in the field of real-time streaming data analytics.**

Keywords*— *fully unsupervised clustering; live data streams; ensemble properties; recursive update; streaming data analytics

I. INTRODUCTION

Due to the fast progress of information and hardware technologies, in the past few years, astronomical amount of streaming data is being generated from various aspects of human societies and daily lives, such as network flows, sensor data and website traffic. Analyzing these data is becoming increasingly important because of the valuable information with the data streams. Facing the potential and challenge at the same time, data analytics is now a rapidly growing field involving the efforts of researchers and scholars from all over the world.

Currently, clustering algorithms require a number of assumptions and parameters to be known in advance [1-9], for instances, number of clusters in *k-means* clustering algorithm [1], the kernel size in *mean shift* clustering algorithm [2], which usually are impractical for users to decide in reality. In addition, most of the well-known clustering algorithms [1-5] as well as many recently proposed clustering algorithms [8, 10] are restricted to offline data processing and not applicable to the live data streams with potential changes of data patterns [11, 12]. In order to handle the huge amount of data samples continuously arriving with the data streams, the clustering algorithm needs to be memory- and computation- efficient [13]. The system structure of the clustering algorithm also should have the ability to evolve, which means the system structure will be changing all the time to follow the shifts and drifts of the data streams [11].

In this paper, a novel autonomous data-driven clustering algorithm, named *AD_clustering*, is proposed for live data streams processing. This algorithm is entirely based on the data and their mutual distribution in the data space. No predefined specific parameter or assumption is needed in *AD_clustering* clustering. The proposed approach starts "from scratch", self-decides its structure as well as all parameters according to the density and the ensemble features of the data samples in the live data streams. Once the structure and parameters are defined, the algorithm keeps re-adjusting them to follow the time-varying data pattern.

In addition, the proposed approach sequentially learns from the current data sample and automatically discards all the previous data samples. Therefore, all the parameters of the proposed approach are updated recursively and only the meta-information is stored in memory, which ensures *AD_clustering* the advantages of computation-and memory-efficiency.

This remainder of this paper is organized as follows. Section II introduces the theoretical basis of the proposed algorithm. The proposed algorithm is described in section III and the summary is given in section IV. Section V presents the numerical experiments and section VI concludes the paper.

II. THEORETICAL BASIS

In this section, the theoretical basis of the proposed *AD_clustering* approach will be introduced. The main quantities representing ensemble properties of the data samples of the live data stream will be firstly introduced followed by their corresponding recursive calculation expressions of these quantities. The well-known Chebyshev inequality will be briefly recalled in the end of this section.

A. Main Quantities

First of all, considering the real Hilbert space $\boldsymbol{R}^d$, we assume the live data stream with continuous arrival of new data samples is denoted as $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_k, ...\}$, where the subscripts indicate the time instance at which each data sample arrives and each data sample has d dimensions.

1) Cumulative proximity

This work was supported by The Royal Society (Grant number IE141329/2014)

Cumulative proximity is used to measure how close/similar of a particular data sample is to all other existing data samples [14].

The centrality of the data sample $\boldsymbol{x}_i$ is calculated as:

$$\pi_k(\boldsymbol{x}_i)=\sum_{j=1}^{k} d^2(\boldsymbol{x}_i,\boldsymbol{x}_j) \quad i=1,2,...,k \tag{1}$$

where $d(\boldsymbol{x}_i,\boldsymbol{x}_j)$ denotes the distance between the two data samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. The distance can be of Euclidean, Mahalanobis type and other metrics based on cosine [15]. In this paper, we only use the most representative Euclidean distance for derivation, namely $d(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left\|\boldsymbol{x}_i-\boldsymbol{x}_j\right\|$.

2) Standardized eccentricity

Standardized eccentricity is directly derived from eccentricity introduced in [14] as a measure that represents the association of the data point with the tail of the distribution and the property of being an outlier/anomaly. The expression of standard eccentricity of the data sample $\boldsymbol{x}_i$ is as follows:

$$\varepsilon_k(\boldsymbol{x}_i)=\frac{2k\pi_k(\boldsymbol{x}_i)}{\sum_{j=1}^{k}\pi_k(\boldsymbol{x}_j)}=\frac{2k\sum_{v=1}^{k}\left\|\boldsymbol{x}_i-\boldsymbol{x}_v\right\|^2}{\sum_{j=1}^{k}\sum_{v=1}^{k}\left\|\boldsymbol{x}_j-\boldsymbol{x}_v\right\|^2} \quad i=1,2,...,k \tag{2}$$

Notice that : $\sum_{i=1}^{k}\varepsilon_k(\boldsymbol{x}_i)=2k$.

3) *Density*

Density is the inverse of the standardized eccentricity expressed as follows, $i=1,2,...,k$ [14]:

$$D_k(\boldsymbol{x}_i)=\frac{1}{\varepsilon_k(\boldsymbol{x}_i)}=\frac{\sum_{j=1}^{k}\pi_k(\boldsymbol{x}_j)}{2k\pi_k(\boldsymbol{x}_i)}=\frac{\sum_{j=1}^{k}\sum_{v=1}^{k}\left\|\boldsymbol{x}_j-\boldsymbol{x}_v\right\|^2}{2k\sum_{v=1}^{k}\left\|\boldsymbol{x}_i-\boldsymbol{x}_v\right\|^2} \tag{3}$$

B. Recursive Expressions of Calculation

The memory- and calculation-efficiencies of the algorithm for online data stream processing mainly rely on the recursive expressions of calculation of the main quantities used in the algorithm.

The recursive expression of $\boldsymbol{x}_i$'s cumulative proximity is [15]:

$$\pi_k(\boldsymbol{x}_i)=k\left(\left\|\boldsymbol{x}_i-\boldsymbol{\mu}_k\right\|^2+X_k-\left\|\boldsymbol{\mu}_k\right\|^2\right) \tag{4}$$

where $\boldsymbol{\mu}_k=\frac{k-1}{k}\boldsymbol{\mu}_{k-1}+\frac{1}{k}\boldsymbol{x}_k \quad \boldsymbol{\mu}_1=\boldsymbol{x}_1$ (5)

$$X_k=\frac{k-1}{k}X_{k-1}+\frac{1}{k}\left\|\boldsymbol{x}_k\right\|^2 \quad X_1=\left\|\boldsymbol{x}_1\right\|^2 \tag{6}$$

The sum of the cumulative proximity of all the existing data samples can also be updated recursively using (7) [15]:

$$\sum_{i=1}^{k}\pi_k(\boldsymbol{x}_i)=\sum_{i=1}^{k-1}\pi_{k-1}(\boldsymbol{x}_i)+2\pi_k(\boldsymbol{x}_k) \tag{7}$$

Therefore, by utilizing (4) and (7), the online recursive expressions of standardized eccentricity and density are as follows [15]:

$$\begin{cases}\varepsilon_k(\boldsymbol{x}_i)=\dfrac{2k^2\left(\left\|\boldsymbol{x}_i-\boldsymbol{\mu}_k\right\|^2+X_k-\left\|\boldsymbol{\mu}_k\right\|^2\right)}{\sum_{i=1}^{k-1}\pi_{k-1}(\boldsymbol{x}_i)+2k\left(\left\|\boldsymbol{x}_k-\boldsymbol{\mu}_k\right\|^2+X_k-\left\|\boldsymbol{\mu}_k\right\|^2\right)}\\ D_k(\boldsymbol{x}_i)=\dfrac{\sum_{i=1}^{k-1}\pi_{k-1}(\boldsymbol{x}_i)+2k\left(\left\|\boldsymbol{x}_k-\boldsymbol{\mu}_k\right\|^2+X_k-\left\|\boldsymbol{\mu}_k\right\|^2\right)}{2k^2\left(\left\|\boldsymbol{x}_i-\boldsymbol{\mu}_k\right\|^2+X_k-\left\|\boldsymbol{\mu}_k\right\|^2\right)}\end{cases} \tag{8}$$

C. Chebyshev Inequality

The well-known Chebyshev inequality states that, in any probability distribution, no more than $\frac{1}{n^2}$ of the values of the distribution can be n standard deviations away from the mean [16]. For Euclidean type of distance, the Chebyshev inequality has the following form:

$$P\left(\left\|\boldsymbol{x}_i-\boldsymbol{\mu}_k\right\|^2\le n^2\delta_k^2\right)\ge 1-\frac{1}{n^2} \tag{9}$$

where δ_k is the standard deviation of all the existing data samples to their mean, and there is $\delta_k^2=X_k-\left\|\boldsymbol{\mu}_k\right\|^2$. In this approach, the Chebyshev inequality will be used to decide the radius of the cluster.

III. The Proposed AD_custering Algorithm

AD_clustering is a novel online algorithm based entirely on the ensemble properties of the data samples. The proposed algorithm does not require any user- or problem- specific assumption or parameters to be predefined. Only a few meta-information is needed to be kept in memory and all the parameters can be updated recursively, which makes the algorithm efficient and suitable for live data streams.

The global meta-parameters need to be kept in memory are as follows:

1) Mean of the data samples: $\boldsymbol{\mu}_k$.

2) Mean of scalar product of the data samples: X_k.

3) The sum of the cumulative proximity of all the existing data samples: $\sum_{i=1}^{k}\pi_k(\boldsymbol{x}_i)$.

4) Number of the existing clusters: N_k.

Local meta-parameters (for each cluster) need to be kept in memory are:

1) The support of the cluster (number of members of the cluster): S_j.

2) Center of the cluster (mean of all the data samples in the cluster): $\boldsymbol{c}_j = \frac{1}{S_j}\sum_{i=1}^{S_j}\boldsymbol{x}_{j,i}$, where $\boldsymbol{x}_{j,i}$ denotes the i^{th} data sample of the j^{th} cluster.

3) The mean of square of the data samples belonging to the cluster: $L_j = \frac{1}{S_j}\sum_{i=1}^{S_j}\left\|\boldsymbol{x}_{j,i}\right\|^2$.

There is no need to keep the radii of the clusters because the radii can be obtained directly from $\boldsymbol{c}_j$ and L_j using the Chebyshev inequality given in section II:

$$r_j = \left\|\boldsymbol{x}_{j,r} - \boldsymbol{c}_j\right\| = n\sqrt{L_j - \left\|\boldsymbol{c}_j\right\|^2} \quad j = 1,2,\cdots,N_k \tag{10}$$

where $\boldsymbol{x}_{j,re}$ is the assumed farthest member for the j^{th} cluster.

Here $n = 2$ is used in our algorithm because it ensures the cluster to have enough tolerance to normal deviations but still be sensitive to abnormal data samples.

In rest of this section, we will demonstrate the three stages of the proposed algorithm.

A. Global Update Stage

For each new data sample, the first stage of the proposed algorithm focuses on the updating of global parameters and finding out the cluster that the new coming data sample is associated to.

At each time a new data sample denoted as $\boldsymbol{x}_{k+1}$ comes, the global parameters $\boldsymbol{\mu}_k$ and X_k are updated firstly to $\boldsymbol{\mu}_{k+1}$ and X_{k+1} using (5) and (6) respectively.

Then, the cumulative proximity $\pi_{k+1}(\boldsymbol{x}_{k+1})$ of $\boldsymbol{x}_{k+1}$ is calculated using (4) and the sum of cumulative proximity of all existing data samples $\sum_{i=1}^{k}\pi_k(\boldsymbol{x}_i)$ is updated using (7). With the new $\pi_{k+1}(\boldsymbol{x}_{k+1})$ and $\sum_{i=1}^{k+1}\pi_{k+1}(\boldsymbol{x}_i)$, $\boldsymbol{x}_{k+1}$'s density $D_{k+1}(\boldsymbol{x}_{k+1})$ is obtained using (8).

The densities of the centers of all the existing clusters are also needed to be updated to help decide whether there is new cluster appearing:

$$D_{k+1}(\boldsymbol{c}_j) = \frac{\sum_{i=1}^{k+1}\pi_{k+1}(\boldsymbol{x}_i)}{2(k+1)^2\left(\left\|\boldsymbol{c}_j - \boldsymbol{\mu}_{k+1}\right\|^2 + X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2\right)} \tag{11}$$

where $j = 1,2,...,N_k$; N_k is the number of existing clusters at the time instance k; $\boldsymbol{c}_j$ is the center of the j^{th} existing cluster.

After all the global parameters have been updated, we come to the second stage of the proposed approach.

B. Local Update Stage

At the beginning of this stage, we need to check Condition A [17]:

Condition A:

$$IF\left(D_{k+1}(\boldsymbol{x}_{k+1}) > \max_{j=1}^{N_k} D_{k+1}(\boldsymbol{c}_j)\right) OR \left(D_{k+1}(\boldsymbol{x}_{k+1}) < \min_{j=1}^{N_k} D_{k+1}(\boldsymbol{c}_j)\right) THEN\ (N_{k+1} \leftarrow N_k + 1) \tag{12}$$

If Condition A is satisfied, the new data sample is associated with a new cluster, $N_{k+1} \leftarrow N_k + 1$. $\boldsymbol{x}_{k+1}$ is set to be the original center $\boldsymbol{c}_{N_{k+1}}$ of the new cluster and other local parameters are set to be $S_{N_{k+1}} = 1$ and $L_{N_{k+1}} = \left\|\boldsymbol{x}_{k+1}\right\|^2$.

On the contrary, if Condition A is not satisfied, $\boldsymbol{x}_{k+1}$ is a potential member of one of the existing clusters and $N_{k+1} \leftarrow N_k$. The cluster that $\boldsymbol{x}_{k+1}$ possibly belongs to is decided by the following equation:

$$Clusterlabel = \arg\min_{j=1}^{N_k}\left\|\boldsymbol{c}_j - \boldsymbol{x}_{k+1}\right\| \tag{13}$$

After the label of the possible cluster that $\boldsymbol{x}_{k+1}$ could be assigned is obtained, it is necessary to check whether $\boldsymbol{x}_{k+1}$ is within the radius of influence of the cluster. Denoting the center of the selected cluster as $\boldsymbol{c}_v$, if the support of the cluster $S_v = 1$, we assign $\boldsymbol{x}_{k+1}$ to the cluster directly and update the corresponding parameters as follows:

$$S_{v\_new} = S_v + 1 \tag{14a}$$

$$\boldsymbol{c}_{v\_new} = \frac{\boldsymbol{c}_v + \boldsymbol{x}_{k+1}}{2} \tag{14b}$$

$$L_{v\_new} = \frac{1}{2}L_v + \frac{1}{2}\left\|\boldsymbol{x}_{k+1}\right\|^2 \tag{14c}$$

while, if $S_v > 1$, Condition B is needed to be checked:

Condition B:

$$IF\left(\left\|\boldsymbol{c}_v - \boldsymbol{x}_{k+1}\right\| > r_v\right)\ THEN\left(N_{k+1} \leftarrow N_k + 1\right) \tag{15}$$

where $r_v = n\sqrt{L_v - \left\|\boldsymbol{c}_v\right\|^2}$.

If Condition B is satisfied, $\boldsymbol{x}_{k+1}$ is in the location which is not covered by the current existing clusters, therefore, a new cluster is launched by $\boldsymbol{x}_{k+1}$. The original parameters of the new cluster are set in the same way as introduced at the beginning of this subsection.

In contrast, when Condition B is not met, $\boldsymbol{x}_{k+1}$ is a formal member of the existing cluster with center $\boldsymbol{c}_v$. The local parameters of this cluster are updated recursively:

$$S_{v\_new} = S_v + 1 \tag{16a}$$

$$\boldsymbol{c}_{v\_new} = \frac{S_v}{S_{v\_new}}\boldsymbol{c}_v + \frac{1}{S_{v\_new}}\boldsymbol{x}_{k+1} \tag{16b}$$

$$L_{v\_new} = \frac{S_v}{S_{v\_new}}L_v + \frac{1}{S_{v\_new}}\left\|\boldsymbol{x}_{k+1}\right\|^2 \tag{16c}$$

For other clusters that do not take in new member at the current time instance, their local parameters are kept as the same.

C. Adjustment Stage

In this stage, all the existing clusters will be examined and adjusted to avoid overlap.

Condition C is executed first.

Condition C:

$$IF\ \left(\left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\| < \max(r_i, r_j)\right) AND\ \left(\left\|\boldsymbol{c}_i - \boldsymbol{c}_l\right\| < \max(r_i, r_l)\right) AND...$$
$$THEN\ \left(Split\ \boldsymbol{c}_i\right)\ AND\ \left(N_k \leftarrow N_k - 1\right) \tag{17}$$

If the i^{th} cluster meets Condition C, it means that more than two other clusters have covered this cluster. In this case, the i^{th} cluster should be split according to the following rules:

$$S_{j\_new} = S_j + s_{ij} \quad S_{l\_new} = S_l + s_{il} \quad \cdots \tag{18a}$$

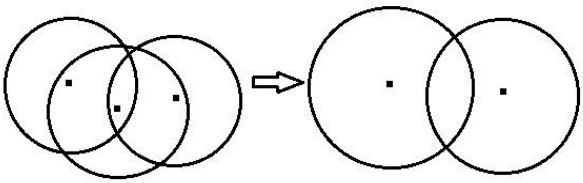


Fig. 1 Illustration of Condition C

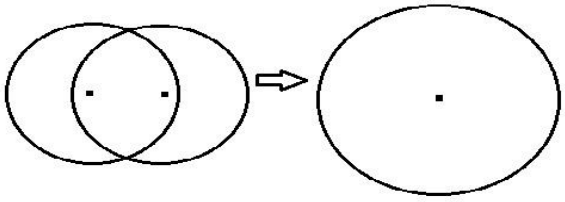


Fig. 2 Illustration of Condition D

$$\boldsymbol{c}_{l\_new} = \frac{S_l}{S_{l\_new}}\boldsymbol{c}_v + \frac{s_{il}}{S_{l\_new}}\boldsymbol{c}_i$$
$$\boldsymbol{c}_{j\_new} = \frac{S_j}{S_{j\_new}}\boldsymbol{c}_j + \frac{s_{ij}}{S_{j\_new}}\boldsymbol{c}_i \tag{18b}$$
$$\cdots$$

$$L_{j\_new} = \frac{S_j}{S_{j\_new}}L_j + \frac{s_{ij}}{S_{j\_new}}\left\|\boldsymbol{c}_i\right\|^2$$
$$L_{l\_new} = \frac{S_l}{S_{l\_new}}L_l + \frac{s_{il}}{S_{l\_new}}\left\|\boldsymbol{c}_i\right\|^2 \tag{18c}$$
$$\cdots$$

where $S_i = s_{ij} + s_{il} + \cdots$; (19a)

$$s_{ij} = Ro\left(\frac{\left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\|}{\left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\| + \left\|\boldsymbol{c}_i - \boldsymbol{c}_l\right\| + \cdots} \cdot S_i\right)$$
$$s_{il} = Ro\left(\frac{\left\|\boldsymbol{c}_i - \boldsymbol{c}_l\right\|}{\left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\| + \left\|\boldsymbol{c}_i - \boldsymbol{c}_l\right\| + \cdots} \cdot S_i\right); \tag{19b}$$
$$\cdots$$

$Ro(\cdot)$ denotes the operation - round to the nearest integer.

Then, we check Condition D:

Condition D:

$$IF\ \left(\left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\| < \max(r_i, r_j)\right)$$
$$THEN\ \left(Merge\ \boldsymbol{c}_i\ and\ \boldsymbol{c}_j\right)\ AND\ \left(N_k \leftarrow N_k - 1\right) \tag{20}$$

If the i^{th} and the j^{th} cluster satisfy Condition D, the two clusters are very close to each other. We should merge them together according to the following rules:

$$S_{i\_new} = S_i + S_j \tag{21a}$$

$$\boldsymbol{c}_{i\_new} = \frac{S_i}{S_{i\_new}}\boldsymbol{c}_i + \frac{S_j}{S_{i\_new}}\boldsymbol{c}_j \tag{21b}$$

$$L_{i\_new} = \frac{S_i}{S_{i\_new}}L_i + \frac{S_j}{S_{i\_new}}L_j \tag{21c}$$

Here, the new i^{th} cluster becomes a combination of the old i^{th} cluster and j^{th} cluster.

After Condition C and Condition D are checked, the adjustment stage is finished, and the algorithm is ready for the next data sample.

In order to ensure the most comprehensive output, once a cluster is formed, the proposed algorithm will keep its meta-parameters in memory unless it is split or merged with others. Nonetheless, all the clusters will automatically be divided into three classes according to their support for easier analysis of the result.

1) ***Abnormal clusters***

For the clusters with support smaller than *3*, they are recognized as abnormal clusters. With the shift and drift of the data pattern [11], these clusters will possibly grow into minor clusters or major clusters later. Despite of their potential, at the time instance that output of the algorithm is carried out, they are meaningless, and can be viewed as noise.

2) ***Major clusters***

The clusters, which have supports above threshold $S_{threshold}$, can be viewed as major clusters. Namely, they are the most representative and important clusters in the data streams. The major clusters are the output of the *AD_clustering* algorithm because they represent the ensemble properties within the data samples in a relatively high degree.

The threshold $S_{threshold}$ is set to be a quarter of the average value of the numbers of data samples allocated to "meaningful" (non-abnormal) clusters and can be expressed as:

$$S_{threshold} = \frac{1}{4} \cdot \frac{\sum_{i=1}^{N_k'} S_i'}{N_k'} \tag{22}$$

where N_k' is the number of clusters with supports $S_j > 2$ ($j = 1,2,...,N_k$); $\{S_i'\}$ is the collection of clusters' supports that satisfy $S_j > 2$ ($j = 1,2,...,N_k$).

3) ***Minor clusters***

The clusters, which have the supports below threshold $S_{threshold}$, are recognized as minor clusters. Minor clusters are less important clusters in the clustering results at the time instance that output of the algorithm is carried out. If new data samples continue to arrive, they have potential to be major clusters due to the possible shift/drift of data pattern [11]. Minor clusters serve as the optional output if users want to see a more detailed result.

IV. Algorithm Summary

The overall algorithm process is summarized in this section presented in the form of pseudo-code.

AD_clustering Algorithm

While the data sample $\boldsymbol{x}_k$ is available (or until not interrupted)

I. **If** ($k = 1$) **Then**

1. $\boldsymbol{\mu}_1 \leftarrow \boldsymbol{x}_1$;
2. $X_1 \leftarrow \|\boldsymbol{x}_1\|^2$;
3. $\sum_{i=1}^{1} \pi_1(\boldsymbol{x}_i) \leftarrow 0$;
4. $N_1 \leftarrow 1$;
5. $\boldsymbol{c}_1 \leftarrow \boldsymbol{x}_1$;
6. $S_1 \leftarrow 1$;
7. $L_1 \leftarrow \|\boldsymbol{x}_1\|^2$;

II. **Else**

1. Update $\boldsymbol{\mu}_k$ and X_k using eq. (5) and eq. (6);
2. Calculate $\pi_k(\boldsymbol{x}_k)$ using eq. (4);
3. Update $\sum_{i=1}^{k} \pi_k(\boldsymbol{x}_i)$ using eq. (7);
4. Calculate $D_k(\boldsymbol{x}_k)$ using eq. (8);
5. Update the $D_k(\boldsymbol{c}_j)$ ($j = 1,2,...,N_k$) using eq. (11);
6. **If** (Condition A is met) **Then**
   a. $\boldsymbol{c}_{N_k+1} \leftarrow \boldsymbol{x}_k$;
   b. $S_{N_k+1} \leftarrow 1$;
   c. $L_{N_k+1} \leftarrow \|\boldsymbol{x}_k\|^2$;
   d. $N_k \leftarrow N_k + 1$;
7. **Else**
   a. Use eq. (13) to find the possible cluster *v* for $\boldsymbol{x}_k$;
   b. **If** ($S_v = 1$) **Then**
      i. Update S_v, $\boldsymbol{c}_v$ L_v using eq. (14);
   c. **Else**
      i. **If** (Condition B is met) **Then**
         *. Update S_v, $\boldsymbol{c}_v$ L_v using eq. (14);
      ii. **Else**
         *. $\boldsymbol{c}_{N_k+1} \leftarrow \boldsymbol{x}_k$;
         *. $S_{N_k+1} \leftarrow 1$;
         *. $L_{N_k+1} \leftarrow \|\boldsymbol{x}_k\|^2$;
         *. $N_k \leftarrow N_k + 1$;
      iii. **End If**
   d. **End If**
8. **End If**
9. **While** there are existing clusters meeting Condition C)
   a. Split the clusters using eq. (18)-(19);
10. **End While**
11. **While** there are existing clusters meeting Condition D)
   a. Merge the clusters using eq. (21);
12. **End While**

III. **End If**

End While

V. Numerical Experiments

In this section, numerical experiments with several benchmark datasets are conducted to test the performance of the newly proposed *AD_clustering* algorithm.

A. Benchmark Datasets

Seven benchmark datasets are used in the experiments with Euclidean type of distance:

1) Fisher Iris Dataset [18]:

2) Climate Dataset [19]:

3) A1 Dataset [20];

4) A2 Dataset [20];

5) A3 Dataset [20];

6) S1 Dataset [20];

7) S2 Dataset [20];

Details of the seven datasets are given in Table I. Although, these datasets were primarily for offline processing, they are transformed into live data streams according to the rule that at each time instance only one data sample is used as the input to the clustering approach.

B. Results

Due to the fact that the proposed *AD_clustering* approach is for live data streams, the algorithm will not keep the members of the existing clusters in the memory because each time after the newly coming data sample is processed, it is discarded. Nonetheless, we still can measure the performance of the algorithm by using the standard measures as detailed below:

TABLE I. BENCHMARK DATASET DESCRIPTION

Data set	Details				
	Number of Data Samples	***Number of clusters***	***Number of Attributes***	***Maximum Cluster size***	***Minimum Cluster size***
FI [a]	150	3	4	50	50
C [b]	938	2	6	479	459
A1	3000	20	2	150	150
A2	5250	35	2	150	150
A3	7500	50	2	150	150
S1	5000	15	2	350	300
S2	5000	15	2	350	300

[a.] Fish Iris; [b.] Climate.

TABEL II (a). EXPERIMENTAL RESULTS

	UI [a]	Data set	Measures					
			N_{final}	N_{maj}	K_{maj}	S_{max}	S_{min}	t_{exe}
AD [b]		FI	9	**3**	**137**	**60**	**32**	0.11
MS [c]	0.1		145	1	3	3	3	0.04
	0.5		20	9	137	36	4	0.02
	1		2	2	150	100	50	0.02
EC [d]	0.05		4	3	149	70	18	0.02
	0.5		4	2	139	70	69	0.02
AD		C	3	**2**	**936**	**475**	**461**	0.32
MS	1		920	1	3	3	3	0.82
	5		98	11	775	334	9	0.13
	10		8	2	929	494	435	0.06
EC	10		8	6	908	309	49	0.10
	100		5	5	938	411	58	0.10

[a.] *User Input;* [b.] *AD_clustering Algorithm;* [c.] *Mean Shift Algorithm;* [d.] *eClustering Algorithm.*

TABEL II (b). EXPERIMENTAL RESULTS

	UI	Data set	Measures					
			N_{final}	N_{maj}	K_{maj}	S_{max}	S_{min}	t_{exe}
AD		A1	27	**20**	**2989**	**154**	**143**	8.39
MS	100		2756	23	70	4	3	9.65
	1000		236	132	2783	100	5	0.27
	5000		17	17	3000	324	139	0.04
EC	10		3	3	3000	2274	340	0.28
	1000		1	1	3000	3000	3000	0.25
AD		A2	53	**35**	**5213**	**160**	**140**	39.99
MS	100		4791	45	144	5	3	44.68
	1000		408	217	4854	101	5	0.7
	5000		28	28	5250	393	75	0.07
EC	10		3	2	4909	3774	1135	0.50
	1000		1	1	5250	5250	5250	0.43
AD		A3	71	**50**	**7456**	**160**	**140**	111.74
MS	100		6851	59	185	5	3	114.34
	1000		558	296	7000	131	5	1.24
	5000		41	40	7477	417	79	0.10
EC	10		3	2	7159	4269	2890	0.72
	1000		1	1	7500	7500	7500	0.61
AD		S1	42	**15**	**4763**	**349**	**218**	35.31
MS	500		4787	33	124	7	3	39.51
	5000		2038	466	3030	166	3	7.09
	25000		111	21	4678	347	21	0.28
EC	10		2	2	5000	2737	2263	0.47
	1000		2	2	5000	2737	2263	0.47
AD		S2	31	**15**	**4849**	**360**	**240**	20.55
MS	500		4831	23	91	10	3	39.62
	5000		2460	410	2514	94	3	9.98
	40000		196	57	4528	327	9	0.33
EC	10		2	2	5000	3685	1315	0.46
	1000		2	2	5000	3685	1315	0.47

1) Number of clusters generated from all the data, N_{final};

2) The number of major clusters, N_{maj};

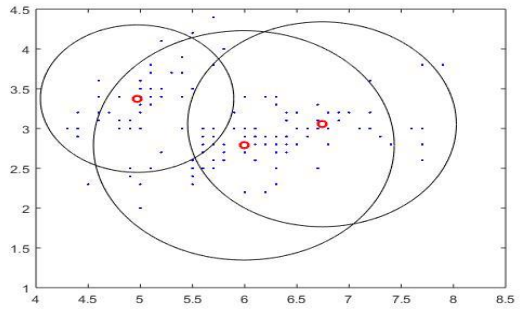
3) The number of data samples in major clusters, K_{maj};

4) Maximum support of the major clusters, $S_{\max}$;

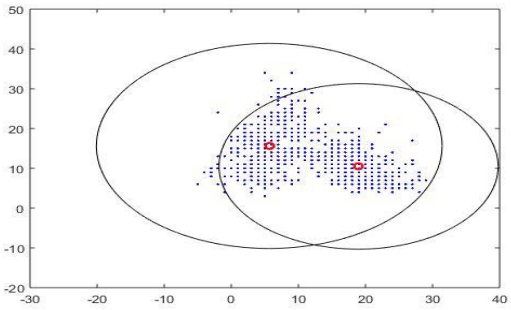
5) Minimum support of the major clusters, $S_{\min}$;

6) Execution time, t_e (in seconds).

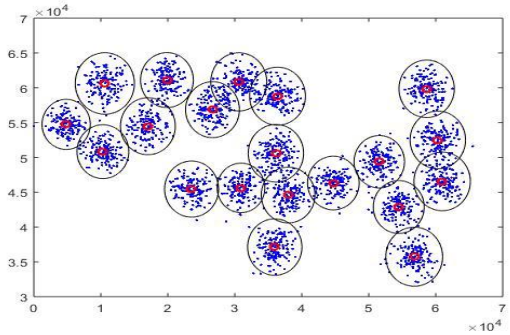
The detailed experimental results are depicted in Table II. The 2D figures of the clustering results are additionally depicted in Fig. 3.



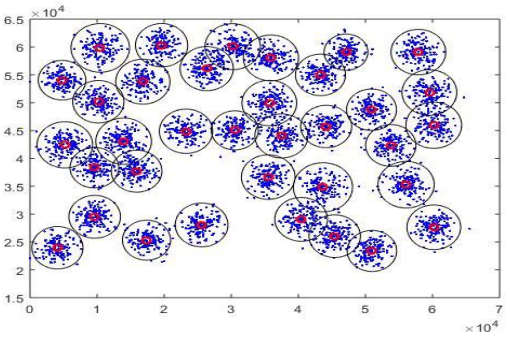
(a) *Fisher iris dataset*



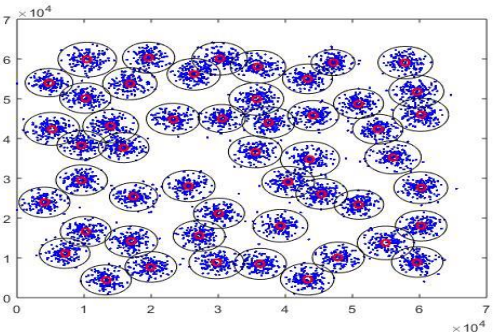
(b) *Climate dataset*



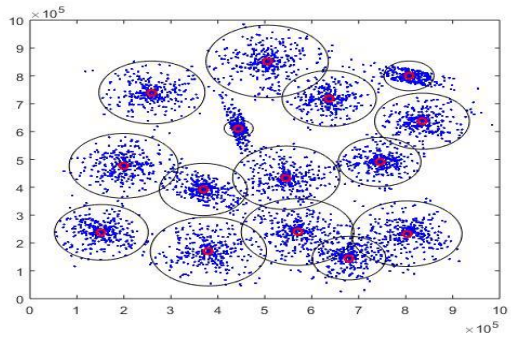
(c) *A1 dataset*



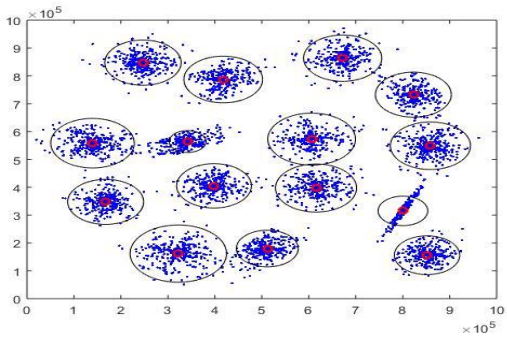
(d) *A2 dataset*



(e) *A3 dataset*



(f) *S1 dataset*



(g) *S2 dataset*

Fig. 3 Experimental results

For further discussion of the performance, the proposed *AD_clustering* algorithm is compared with two well-known clustering algorithms: *mean shift* clustering algorithm (offline) [2] and *eClustering* algorithm (online) [7]. To ensure the effectiveness of the comparison, we additionally apply the same rule measuring the importance of clusters described in Section III to the experimental results of the comparative algorithms

Because this paper is focused on a new method for live data streams analytics, the requirement of **user inputs** is also taken into consideration. For *mean shift* clustering algorithm, kernel size is the parameter needed to be predefined and 'for *eClustering* algorithm, user needs to predefine the initial radius.

As we can see clearly from Table II, the *mean shift* clustering algorithm is somewhat faster than the proposed approach. However, its performance largely depends on the

proper user input, namely kernel size, which is often impractical in real cases because of the lack of *prior* knowledge. *eClustering* algorithm is the fastest and more stable than mean shift clustering algorithm. However, its accuracy is not as high as that of *AD_clustering*.

Comparatively, the clustering results of the newly proposed online *AD_clustering* algorithm are supreme compared with the other well-known clustering algorithms. Without the need of *prior* knowledge or pre-defined parameters from users, *AD_clustering* algorithm is entirely driven by the data streams and exhibits high purity results. Despite of the fact that the proposed algorithm costs more time that the two comparative algorithms, it is acceptable due to its excellent performance and the feature of being totally free of user inputs.

VI. Conclusion

A fully autonomous unsupervised clustering algorithm called *AD_clustering* is introduced in this paper. This newly proposed clustering algorithm is entirely free of any kind of user- and problem- specific restrictive assumptions and parameters. The algorithm is completely based on the data samples and their mutual distributions and is able to self-define and re-adjust its system structure as well as parameters. The recursive update of the parameters ensure the proposed algorithm to be calculation-efficient, and update parameters only based on the current data sample also give *AD_clustering* algorithm the advantage of memory-efficiency, which make this approach extremely suitable for online data stream processing. This newly proposed algorithm is a solid basis and promising tool for the area of streaming data analytics.

In the future, we intend to conduct the research on the performance of this algorithm using different distances and dissimilarities including Mahalanobis type and other metrics based on cosine, and will be further applied to other fields like natural language processing.

References

[1] J. MacQueen, “Some methods for classification and analysis of multi-variate observations”, in Porceedings of the Fifth Berkeley Synposium on Mathematical Statistics and Probability, Volue 1: Statistics, Berkeley, 1967, pp. 281-297.

[2] K. Fukunaga and L. Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition”, IEEE Transactions on Information Theory,1975, vol. 21(1), pp. 32-40.

[3] Yager, R. and D. Filev, “Generation of fuzzy rules by mountain clustering”, Journal of Intelligent & Fuzzy Systems, 1994, vol. 2(3), pp. 209-219.

[4] Chiu, S.L., “Fuzzy model identification based on cluster estimation”. Journal of Intelligent & Fuzzy Systems, 1994, vol.2(3), pp. 1064-1246.

[5] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise”, in 2nd International Conference on Knowledge Discovery and Data Mining. 1996. Portland, Oregan, vol. 96(34), pp. 226-231.

[6] J. Csirik, L. Epstein, C. Imreh, and A. Levin, " Online clustering with variable sized clusters", Algorithmica, 2013, vol.65(2), pp.251-27.

[7] P. Angelov, "An approach for fuzzy rule-base adaptation using on-line clustering", International Journal of Approximate Reasoning, 2004, vol.35(3), pp. 275-289.

[8] R. Hyde and P. Angelov, "A fully autonomous Data Density based Clustering technique", 2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS), 2014, Orlando, pp.116-123.

[9] C. Wang, J. Lai, D. Huang and W. Zheng, "SVStream: A support vector-based algorithm for clustering data streams", IEEE Transactions on Knowledge and Data Engineering, 2013, vol.25(6), pp.1410-1424.

[10] P. Angelov, X. Gu, G. G. Gutierez, J. A. Iglesias and A. S. de Miguel, “Autonomous data density based clustering method”, in IEEE World Congress on Computational Intelligence, 24-29 July 2016, Vancouver, Canada, *in press*.

[11] E. Lughofer and P. Angelov, "Handling drifts and shifts in on-line data streams with evolving fuzzy systems", Applied Soft Computing Journal, 2011, vol.11(2), pp.2057-2068

[12] P. Angelov, ‘Outside the Box: An Alternative Data Analytics Framework’. Journal of Automation, Mobile Robotics and Intelligent Systems, 2014, vol. 8(2), pp. 53-59.

[13] J. Silva, E. Faria, R. Barros, E. Hruschka, A. de Carvalho and J.Gama, "Data stream clustering: A survey", ACM Computing Surveys, 2013, vol.46(1), p.13.

[14] P. Angelov, X. Gu, J. Principe and D. Kangin, “Empirical data analysis: A new tool for data analytics”, unpublished.

[15] P. Angelov, X. Gu and D. Kangin, “TEDA: typicality based empirical data analysis”, unpublished.

[16] J. Saw, M. Yang, and T. Mo, "Chebyshev inequality with estimated mean and variance", The American Statistician, 1984, vol.38(2), pp.130-132

[17] P. Angleov, Autonomous Learning Systems from Data Stream to Knowledge in Real Time. West Sussex, United Kingdom: John Wiley & Sons, Ltd., 2012.

[18] https://archive.ics.uci.edu/ml/datasets/Iris

[19] http://www.worldweatheronline.com

[20] http://cs.joensuu.fi/sipu/datasets/