Knowledge-Based Systems 59 (2014) 33–47

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Incremental entropy-based clustering on categorical data streams with concept drift

Yanhong Li, Deyu Li *, Suge Wang, Yanhui Zhai

School of Computer and Information Technology, Shanxi University, Taiyuan, 030006 Shanxi, China
Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Taiyuan, 030006 Shanxi, China

ARTICLE INFO

Article history:
Received 7 January 2013
Received in revised form 16 January 2014
Accepted 1 February 2014
Available online 7 February 2014

Keywords:
Categorical data stream
Clustering
Data labeling
Concept drift detection
Cluster evolving analysis

ABSTRACT

Clustering on categorical data streams is a relatively new field that has not received as much attention as static data and numerical data streams. One of the main difficulties in categorical data analysis is lacking in an appropriate way to define the similarity or dissimilarity measure on data. In this paper, we propose three dissimilarity measures: a point-cluster dissimilarity measure (based on incremental entropy), a cluster–cluster dissimilarity measure (based on incremental entropy) and a dissimilarity measure between two cluster distributions (based on sample standard deviation). We then propose an integrated framework for clustering categorical data streams with three algorithms: Minimal Dissimilarity Data Labeling (MDDL), Concept Drift Detection (CDD) and Cluster Evolving Analysis (CEA). We also make comparisons with other algorithms on several data streams synthesized from real data sets. Experiments show that the proposed algorithms are more effective in generating clustering results and detecting concept drift.

1. Introduction

Many natural and artificial systems in practical applications such as real-time monitoring, stock market, and credit card fraud detection, continuously generate the temporally ordered, fast changing, massive and potentially infinite data streams. The research on data stream mining is becoming important and meaningful [1,2]. A data stream is defined as a real-time, continuous, ordered (implicitly by arrival time or explicitly by time-stamp) sequence of data items [3]. In recent years, several kinds of data mining researches have been explored for the data stream environment, including the summarization and statistics [4–6], data selection [7], change detection [8,9], sampling [10], data clustering [11–16] and data classification [17–20]. Ditzler and Polikar [21] discussed learning concept drift from imbalanced data. Ghazikhani et al. [22] proposed an online ensemble of classifiers for non-stationary and imbalanced data streams. Lu et al. [23] took the training case-base as an evolving data stream and proposed a new case-base editing method targeting competence enhancement under concept drifting environment.

Clustering is a widely used technique used to identify the cluster structure in an unlabeled data set by objectively organizing data into homogeneous groups and maximizing the within-group-object similarity as well as minimizing the between-group-object similarity [24]. Clustering techniques for data streams are very different from those for static data (i.e., data set that is unchanged in the clustering process), because it is difficult to control the order in which data items arrive, to store an entire data stream, or to scan through it multiple times due to its tremendous volume [1]. Another distinguishing characteristic of data streams is that they are time-varying. Changes in the hidden context can induce more or less radical changes in the target concept, generally known as concept drift [25]. As the concepts behind the data evolve with time, the underlying clusters may also change considerably with time [24]. Performing clustering on the entire time-evolving data not only decreases the quality of clusters but also disregards the expectations of users, who usually require the recent clustering results [26]. Thus, discovery of the concepts hidden in data streams imposes a great challenge upon cluster analysis.

Many researches on clustering data streams in the numerical domain have been reported [11–13,27,15,28–34]. Actually, categorical data streams prevalently exist in real data. In the categorical domain, however, the above algorithm is infeasible because the numerical characteristics of clusters are difficult to

* Corresponding author. Tel.: +86 13303408298; fax: +86 3517018176.
E-mail addresses: liyh@sxu.edu.cn (Y. Li), lidy@sxu.edu.cn (D. Li), wsg@sxu.edu.cn (S. Wang), chai_yanhui@163.com (Y. Zhai).

http://dx.doi.org/10.1016/j.knosys.2014.02.004
0950-7051/

define. Nasraoui et al. [35] presented a strategy to mine evolving user profiles in the Web and designed an algorithm for tracking evolving user profiles based on clustering results. Chen et al. [26] proposed a framework for clustering concept-drifting categorical time-evolving data. In their framework, a kind of cluster representative is defined based on the importance of the combinations of attribute values and an algorithm, named maximal resemblance data labeling, is then proposed to allocate each unlabeled data point into a corresponding appropriate cluster by utilizing cluster representative. In Chen's framework, the reclustering is performed in the current sliding window when quite a large number of outliers are found or quite a large number of clusters are varied in the ratio of data points in the current temporal clustering result obtained by data labeling. However, we claim that the reclustering is not necessary when quite a large number of clusters are varied in the ratio of data points, because every data point in the current sliding window has been properly labeled. By defining the distance between two sliding windows, Cao et al. [36] proposed an algorithm for detecting concept-drifting windows on the categorical time-evolving data. But in his framework, concept-drifting windows are detected based on the distance between adjacent sliding windows. When computing the distance, all data points in each window are regarded as a cluster without taking both cluster distribution and outliers into consideration. Thus, it is desired to devise an efficient method for clustering categorial data streams.

In this paper, we propose an integrated framework for clustering categorical data streams by using sliding window technique and data labeling technique. It consists of three parts: Minimal Dissimilarity Data Labeling (MDDL), Concept Drift Detection (CDD) and Cluster Evolving Analysis (CEA). In this framework, the initial clustering is performed on the first sliding window. MDDL marks an incoming data point in the current sliding window with a proper cluster label by referring to the clustering result of the previous sliding window, and the data points that cannot be exactly marked are regarded as outliers. There are two cases to be considered as concept drift. One case occurs when the outlier ratio in the current window is larger than a given threshold. In this case, a reclustering is performed in the current window. Another case occurs when the cluster distribution in the current window has a larger difference with that in the previous window. CDD is designed to explore the two cases and to find out the concept drift windows. In order to iconically show the cluster evolving process, the representative of a cluster and a dissimilarity measure between two clusters with adjacent time stamps are defined. CEA is designed to analyze the time-evolving trend of clusters at different time stamps. The comparative experiments validate the availability of the proposed framework.

The major contributions of this paper are the following:

- An integrated framework is proposed for clustering categorical data streams by using sliding window technique and data labeling technique.
- An effective data labeling algorithm is developed based on the point-cluster dissimilarity measure.
- The dissimilarity measure between two cluster distributions is employed to detect the concept drift.
- The cluster–cluster dissimilarity measure is employed to analyze the time-evolving trend of data stream.

This paper is set up as follows. In Section 2, the problem of clustering categorical data steams is formulated. In Section 3, a dissimilarity measure between a data point and a cluster is defined by incremental entropy and MDDL algorithm is proposed. In Section 4, a dissimilarity measure between two cluster distributions is defined, and CDD algorithm is designed. In Section 5, the cluster representative is defined, and CEA algorithm is proposed based on the dissimilarity measure between two clusters. Section 6 reports our experimental study on synthetic data sets generated from a few of real raw data sets. Section 7 concludes the paper with some remarks.

2. Problem description

Suppose that a set of categorical data points DS is given, where each data point $\mathbf{x}_i$ is a d-dimensional vector of attribute values, i.e., $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^d)$. Each component x_i^j $(1 \leqslant j \leqslant d)$ takes a value from the domain V_j of the jth attribute. It should be noticed that the data points in DS are ordered. Sliding window is an often-used technique for observing and analyzing a data stream. The size of sliding window usually indicates how large time scale or data granularity will be utilized by analysts to data analysis. When the window size N is given the data set DS is then separated into a series of continuous sliding windows S^t, where the superscript t is the identification number of the sliding window, also called time stamp.

The characteristics of continuation, speediness, order, changing, huge amount of data streams require a fast, real-time response of data analysis method. Data labeling technique is often adopted to improve the efficiency of clustering [26,36]. In our framework, let $C^{t-1} = \left\{c_1^{t-1}, c_2^{t-1}, \ldots, c_{k^{t-1}}^{t-1}\right\}$ be the clustering result of the sliding window S^{t-1}, where c_m^{t-1} $(1 \leqslant m \leqslant k^{t-1})$ is the mth cluster. Utilizing the cluster information of C^{t-1} we mark each data point in S^t with a proper label corresponding to a cluster of C^{t-1}. And the labeling result $C'^t = \left\{c_1'^t, c_2'^t, \ldots, c_{k^{t-1}}'^t, outliers'^t\right\}$ of S^t will be called the temporal clustering result, where $outliers'^t$ is the set of data points in S^t that cannot be marked with any proper cluster label of C^{t-1}.

3. Incremental entropy and data labeling

3.1. Some basic notions of entropy

As a kind of measure of the uncertainty of a random variable [37], Shannon entropy and its variants were widely applied to almost all disciplines such as pattern discovery [38], numerical clustering [39] and categorical data clustering [40–44]. Let x be a discrete random variable taking a finite number of possible values $v_1, v_2, \ldots, v_n$ with probabilities $p_1, p_2, \ldots, p_n$ respectively, such that $p_i \geqslant 0$ $(i = 1, 2, \ldots, n)$, and $\sum_{i=1}^n p_i = 1$. The entropy $H(x)$ of a discrete random variable x is defined by

$$H(x) = -\sum_{i=1}^{n} p_i \log_2 p_i. \tag{1}$$

Let $X = (x^1, x^2, \ldots, x^d)$ be a discrete random vector, a finite set V_j be the domain of x^j $(1 \leqslant j \leqslant d)$. $p(x^j = v)$ denotes the probability of the event $x^j = v$, where $v \in V_j$. If random variables x^j $(1 \leqslant j \leqslant d)$ are independent, the information entropy $H(X)$ of X is defined as [37]

$$H(X) = \sum_{j=1}^{d} H(x^j) = -\sum_{j=1}^{d} \sum_{v \in V_j} p(x^j = v) \log_2 p(x^j = v). \tag{2}$$

Entropy-based measures can evaluate the orderliness of a given cluster [43]. Also, entropy criterion is especially good for categorical data clustering because of the lack of intuitive distance definition for categorical values.

3.2. Dissimilarity between a data point and a cluster

Let $c \subseteq DS$ be a cluster. we regard an attribute x^j $(1 \leqslant j \leqslant d)$ as a discrete random variable taking its values from V_j. By c, we can construct a discrete probability distribution according to the following way:

$$p(x^j = v)|_{v \in V_j} = \frac{|\{\mathbf{x} \in c : x^j(\mathbf{x}) = v\}|}{|c|},$$

where $x^j(\mathbf{x})$ denotes the value of the point $\mathbf{x}$ under the attribute x^j, and $|\cdot|$ denotes the cardinality of a set. By denoting the random variable determined by this probability distribution as c^j, we then have a d-dimensional discrete random vector $(c^1, c^2, \ldots, c^d)$ denoted by c yet.

Definition 3.1 [44]. Let c_1 and c_2 be two clusters from DS. The incremental entropy of merging (mixing) two clusters c_1 and c_2 is defined by the following equation.

$$IE(c_1, c_2) = (|c_1| + |c_2|)H(c_1 \cup c_2) - |c_1|H(c_1) - |c_2|H(c_2). \tag{3}$$

Property 3.1 [44]. $IE(c_1, c_2) \geqslant 0$.

Next, we define the dissimilarity measure between a point and a cluster by the incremental entropy.

The structural characteristic of a data set is determined by its value frequencies in each column (i.e., the domain of x^j). Intuitively, putting a data point into a cluster whose most data points are similar to the point will not significantly change the value frequencies. On the contrary, putting a data point into a cluster whose most data points are dissimilar to the point will evidently change the value frequencies. Thus, though the similarity between a data point and a cluster cannot be directly measured by entropy, it can be observed by putting the data point into the cluster and examining the change of entropy caused by putting a data point into a cluster.

Definition 3.2. Let $c \subseteq DS$ be a cluster, and $\mathbf{x} \in DS$ a data point. We define

$$d(\mathbf{x}, c) = IE(\{\mathbf{x}\}, c) = (|c| + 1|)H(c \cup \{\mathbf{x}\}) - |c|H(c) \tag{4}$$

as the dissimilarity measure between a point $\mathbf{x}$ and a cluster c.

Property 3.2. *Let c be a cluster. We have the following properties.*

1. $d(\mathbf{x}, c)$ *takes its maximum if and only if*
$$x^j(\mathbf{x}) = \begin{cases} v \in V_j \setminus x^j(c), & \text{if} \quad V_j \setminus x^j(c) \neq \emptyset; \\ v \in \left\{ \arg\min_{v' \in V_j} \left| (x^j)_c^{-1}(v') \right| \right\}, & \text{otherwise}, \end{cases}$$
where $x^j(c) = \{x^j(\mathbf{x}) \in V_j : \mathbf{x} \in c\}$ *and* $(x^j)_c^{-1}(v') = \{\mathbf{x} \in c : x^j(\mathbf{x}) = v'\}$,
2. $d(\mathbf{x}, c)$ *takes its minimum if and only if* $x^j(\mathbf{x}) \in \left\{ \arg\max_{v' \in V_j} \left| (x^j)_c^{-1}(v') \right| \right\}$,
3. $d(\mathbf{x}, c) = 0$ *if and only if* $c = \{\mathbf{x}' = \mathbf{x} | \mathbf{x}' \in c\}$, i.e., *an arbitrary point in c has the same presentation with* $\mathbf{x}$.

3.3. Data labeling algorithm

By the dissimilarity measure in Definition 3.2 and the clustering result C^{t-1} of the window S^{t-1} we can mark each point in S^t with a temporal label of the cluster that achieves the minimal dissimilarity value among all clusters in C^{t-1}. However, even if the minimal dissimilarity value of a point in S^t is large, the point perhaps should not be marked with any cluster label of C^{t-1}. Such a data point will be treated as an outlier.

A group of thresholds $\lambda_1^{t-1}, \lambda_2^{t-1}, \ldots, \lambda_{k^{t-1}}^{t-1}$ are set to determine whether the data point is an outlier. Let $C^{t-1} = \{c_1^{t-1}, c_2^{t-1}, \ldots, c_{k^{t-1}}^{t-1}, outliers^{t-1}\}$. We use the data points in c_m^{t-1} to decide the threshold λ_m^{t-1} $(1 \leqslant m \leqslant k^{t-1})$. The maximum dissimilarity value in c_m^{t-1} is set as λ_m^{t-1}. For c_m^{t-1}, we define

$$\lambda_m^{t-1} = \max_{\mathbf{x} \in c_m^{t-1}} d(\mathbf{x}, c_m^{t-1}). \tag{5}$$

For a point $\mathbf{x} \in S^t$, let $M = \left\{ m | d(\mathbf{x}, c_m^{t-1}) \leqslant \lambda_m^{t-1}, 1 \leqslant m \leqslant k^{t-1}) \right\}$ and $m^* \in \arg\min_{m \in M} d(\mathbf{x}, c_m^{t-1})$. The labeling function is defined as follows:

$$Label(\mathbf{x}) = \begin{cases} c'^t_{m^*}, & M \neq \emptyset; \\ outliers'^t, & otherwise. \end{cases}$$

An algorithm MDDL (Minimal Dissimilarity Data Labeling) to mark points in the current sliding window S^t using the clustering result C^{t-1} of S^{t-1} is described in Table 1.

The time complexity of MDDL is analyzed as follows. For higher execution efficiency, the number of all attribute values of all attributes within c_m^{t-1} is recorded. For computing $H(c_m^{t-1} \cup \{\mathbf{x}\})$ after putting a data point $\mathbf{x}$ into c_m^{t-1}, we only modify the number of corresponding attribute values. And the time complexity of computing the dissimilarity between data point and a cluster is linear with respect to d and q, where $q = \max_j |V_j|$. Therefore the time complexity of data labeling is $O(k * N * d * q)$.

The following simple example demonstrates MDDL algorithm.

Table 1
Minimal Dissimilarity Data Labeling MDDL.

Algorithm MinimalDissimilarityDataLabeling(C^{t-1}, S^t, C'^t)
Begin
  $C'^t = outliers'^t = \emptyset$;
  For $m = 1$ to k^{t-1}
    $c'^t_m = \emptyset$;
    Calculate $H(c_m^{t-1})$;
  End for;
  For $m = 1$ to k^{t-1}
    For all data points $\mathbf{x} \in c_m^{t-1}$
      Calculate the dissimilarity $d(\mathbf{x}, c_m^{t-1})$;
    End for;
    $\lambda_m^{t-1} = max_{\mathbf{x} \in c_m^{t-1}} d(\mathbf{x}, c_m^{t-1})$;
  End for;
  For all data points $\mathbf{x} \in S^t$
    For $m = 1$ to k^{t-1}
      Calculate the dissimilarity $d(\mathbf{x}, c_m^{t-1})$;
    End for;
    $M = \left\{ m | d(\mathbf{x}, c_m^{t-1}) \leqslant \lambda_m^{t-1}, 1 \leqslant m \leqslant k^{t-1} \right\}$;
    If $M \neq \emptyset$ then
      $Label(\mathbf{x}) = c'^t_{m^*}$, $m^* \in \arg\min_{m \in M} d(\mathbf{x}, c_m^{t-1})$;
      $c'^t_{m^*} = c'^t_{m^*} \cup \{\mathbf{x}\}$;
    Else
      $outliers'^t = outliers'^t \cup \{\mathbf{x}\}$;
    End if;
  End for;
  For $m = 1$ to k^{t-1}
    $C'^t = C'^t \cup \{c'^t_m\}$;
  End for;
  $C'^t = C'^t \cup \{outliers'^t\}$;
  Return $|outliers'^t|$;
End algorithm;

Example 1. A categorical data set is given in Table 2, where, $DS = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{18}\}$, and $X = \{x_1, x_2, x_3\}$ is the attribute set. Give $N = 6$, we have $S^1 = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_6\}$, $S^2 = \{\mathbf{x}_7, \mathbf{x}_8, \ldots, \mathbf{x}_{12}\}$ and $S^3 = \{\mathbf{x}_{13}, \mathbf{x}_{14}, \ldots, \mathbf{x}_{18}\}$. Suppose that the clustering result of S^1 is $C^1 = \left\{c_1^1, c_2^1, outliers^1\right\}$, where $c_1^1 = \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_6\}$, $c_2^1 = \{\mathbf{x}_2, \mathbf{x}_4\}$ and $outliers^1 = \emptyset$. By Definition 3.2, the dissimilarity values between each data point in S^2 and each cluster in C^1 are computed and shown in Table 3.

According to Eq. (5), the thresholds are set to $\lambda_1^1 = 1.6096$ and $\lambda_2^1 = 0.7549$. Then the labeling result of points in S^2, i.e., the temporal cluster result, is as follows: $C'^2 = \left\{c_1'^2, c_2'^2, outliers'^2\right\}$, where $c_1'^2 = \emptyset$, $c_2'^2 = \{\mathbf{x}_7, \mathbf{x}_9\}$ and $outliers'^2 = \{\mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}\}$.

4. Concept drift detection

So far, there is not yet a recognized definition for concept drift in a data stream. In general, concept drift means an obvious change occurring between two adjacent observed regions of samples. The motive of concept drift detection in this paper is to detect the difference of cluster distributions between the current sliding window and the previous sliding window, and to decide whether reclustering is required in the current sliding window.

Let S^t be the current sliding window and S^{t-1} the previous. We think that concept drift should at least include the following two cases.

Case 1. There are so much outliers in the current window that we have to consider new clusters different from any cluster of C^{t-1}.
Case 2. Although the number of outliers in the current window is bearable, an obvious change has occurred in C'^t when compared with C^{t-1}.

In this section, we will discuss the dissimilarity between two cluster distributions. A method and the corresponding algorithm for concept drift detection are presented. The time complexity of the algorithm is analyzed as well.

4.1. Dissimilarity between two cluster distributions

In order to characterize the cluster distribution of a clustering result, the concept of a vector space is introduced. A vector used to represent the cluster distribution of a clustering result consists of the ratio of each cluster and outliers within the clustering result. Each entry of the vector is the ratio of the number of points in a cluster or in the outliers to the number of all points in a sliding window. Based on the cluster distribution space, the cluster distribution vector of a clustering result C is formally defined as follows.

Table 2
A categorical data set.

Data point	x^1	x^2	x^3	Data point	x^1	x^2	x^3
$\mathbf{x}_1$	A	F	T	$\mathbf{x}_{10}$	C	P	G
$\mathbf{x}_2$	X	F	R	$\mathbf{x}_{11}$	C	P	D
$\mathbf{x}_3$	A	F	C	$\mathbf{x}_{12}$	C	P	D
$\mathbf{x}_4$	Y	F	R	$\mathbf{x}_{13}$	X	F	R
$\mathbf{x}_5$	A	F	T	$\mathbf{x}_{14}$	X	F	R
$\mathbf{x}_6$	A	F	T	$\mathbf{x}_{15}$	I	N	T
$\mathbf{x}_7$	X	F	R	$\mathbf{x}_{16}$	X	F	R
$\mathbf{x}_8$	C	P	D	$\mathbf{x}_{17}$	C	P	D
$\mathbf{x}_9$	X	F	R	$\mathbf{x}_{18}$	X	F	R

Table 3
Dissimilarity between each data point in S^2 and each cluster in C^1.

	$\mathbf{x}_7$	$\mathbf{x}_8$	$\mathbf{x}_9$	$\mathbf{x}_{10}$	$\mathbf{x}_{11}$	$\mathbf{x}_{12}$
c_1^1	7.2193	10.8289	7.2193	10.8289	10.8289	10.8289
c_2^1	0.7549	8.2647	0.7549	8.2647	8.2647	8.2647

Definition 4.1. The cluster distribution vector $\overline{C}$ of a clustering result $C = \{c_1, c_2, \ldots, c_k, outliers\}$ is defined as

$$\overline{C} = \frac{1}{N}(|c_1|, |c_2|, \ldots, |c_k|, |outliers|). \tag{6}$$

To detect the difference of cluster distributions between two clustering results C'^t and C^{t-1}, we only need to compare $\overline{C'^t}$ with $\overline{C^{t-1}}$. In at least two cases, obvious change will occur in $\overline{C'^t}$ when compared with $\overline{C^{t-1}}$.

Case 1. A certain degree of change have occurred in the radios of the majority of clusters in C'^t.
Case 2. Though the number of changed clusters in C'^t can be tolerated, the radios of the minority of clusters in C'^t have significantly changed.

Here, we will define a dissimilarity measure to characterize the above two cases.

Let $\overline{C'^t} - \overline{C^{t-1}} = \frac{1}{N}\left(|c_1'^t| - |c_2^{t-1}|, |c_1'^t| - |c_2^{t-1}|, \ldots, \left|c_{k^{t-1}}'^t\right| - \left|c_{k^{t-1}}^{t-1}\right|, |outliers'^t| - |outliers^{t-1}|\right)$. We denote the component of the vector $\overline{C'^t} - \overline{C^{t-1}}$ by cd_i $(1 \leqslant i \leqslant k^{t-1} + 1)$. It is obvious that cd_i takes its value in the range -1 to 1. Let $S = \{cd_i | 1 \leqslant i \leqslant k^{t-1} + 1\}$ be the set of all components of $\overline{C'^t} - \overline{C^{t-1}}$. Let $\overline{cd}$ denote the mean of S, i.e., $\overline{cd} = \frac{1}{k^{t-1}+1}\sum_{i=1}^{k^{t-1}+1} cd_i$. Obviously, $\overline{cd} = 0$.

Let s be the sample standard deviation of S, i.e.,

$$s = \sqrt{\frac{1}{k^{t-1}}\sum_{i=1}^{k^{t-1}+1}(cd_i - \overline{cd})^2} = \sqrt{\frac{1}{k^{t-1}}}d(\overline{C'^t}, \overline{C^{t-1}}),$$

where $d(\overline{C'^t}, \overline{C^{t-1}})$ is Euclidean distance between $\overline{C'^t}$ and $\overline{C^{t-1}}$. Evidently, $0 \leqslant s \leqslant \sqrt{\frac{2}{k^{t-1}}}$.

Obviously, sample standard deviation s can measure the centralization and decentralization degrees of S with respect to the mean value of samples. The smaller the sample standard deviation, the more concentrated the value of random variables. On the other hand, it is easy to see that s is proportional to Euclidean distance between $\overline{C'^t}$ and $\overline{C^{t-1}}$. This means that s can represent the total change in all components of $\overline{C'^t}$ and $\overline{C^{t-1}}$. Therefore, s can evaluate the dissimilarity of cluster distributions between clustering results C'^t and C^{t-1}.

Definition 4.2. Given two clustering results C'^t and C^{t-1}, we define

$$d(C^{t-1}, C'^t) = \frac{s}{\sqrt{\frac{2}{k^{t-1}}}} \tag{7}$$

as the dissimilarity of cluster distributions between clustering results C'^t and C^{t-1}.

Obviously, we have $0 \leqslant d(C^{t-1}, C'^t) \leqslant 1$.

4.2. An approximate solution of the density function of $d(C^{t-1}, C'^{t})$

By the discussion of Section 4.1, $d(C^{t-1}, C'^{t})$ can be regarded as a function of the random vectors $\overline{C'^{t}}$ and $\overline{C^{t-1}}$, which randomly take their values from the unit cube of the $k^{t-1}+1$ dimensional real space $R^{k^{t-1}+1}$ under the restriction $x_1 + x_2 + \cdots + x_{k^{t-1}+1} = 1$. Fig. 1. gives a diagrammatic drawing in the case of $k^{t-1}+1 = 3$. Now we estimate the density function of $d(C^{t-1}, C'^{t})$ by Monte Carlo method.

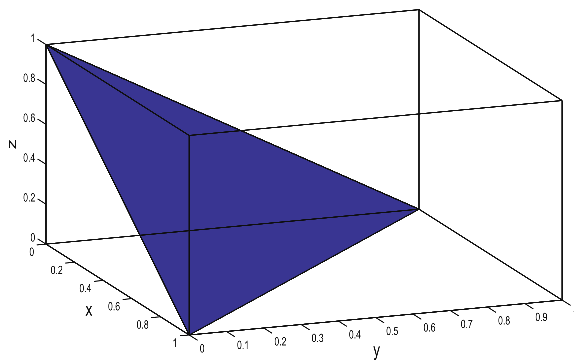

Fig. 1. Unit cube of the 3 dimensional real space R^3.

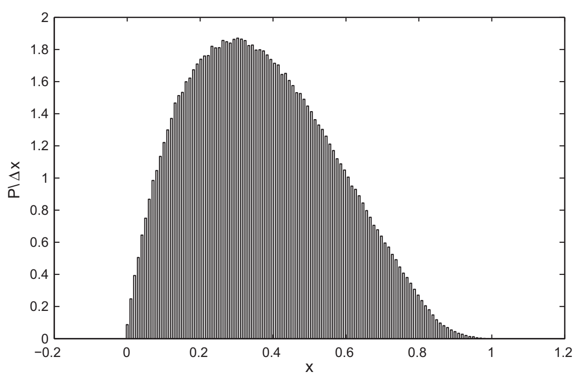

Fig. 2. Frequency distribution of $d(C^{t-1}, C'^{t})$, $k^{t-1} = 3$.

The following random experiment is performed.

Step 1. Randomly select two points as the two vectors $\overline{C'^{t}}$ and $\overline{C^{t-1}}$ in the region $x_1 + x_2 + \cdots + x_{k^{t-1}+1} = 1$ of $R^{k^{t-1}+1}$, and then compute the value of $d(C^{t-1}, C'^{t})$.

Step 2. Repeat Step 1 with P times, e.g., $P = 10^6$.

Step 3. Count the frequency of the values of $d(C^{t-1}, C'^{t})$ falling in the each equilong small interval (e.g., the length of a small interval $\Delta x = 0.01$), and draw the histogram of the frequency of $d(C^{t-1}, C'^{t})$ values.

By the experiment described above we can obtain an approximation to the density function of $d(C^{t-1}, C'^{t})$. Fig. 2 shows the case with $k^{t-1} = 3$, $\Delta x = 0.01$ and $P = 10^6$. The more experimental results of the density function of $d(C^{t-1}, C'^{t})$ are shown in Fig. 3. An important observation is that the density function of $d(C^{t-1}, C'^{t})$ is almost unchanged when $k^{t-1} \geqslant 3$.

4.3. Selecting an expected threshold of the distribution dissimilarity

How large the value of dissimilarity of two cluster distributions means an obvious change of a cluster distribution with respect to the previous? An applicable threshold of the dissimilarity between two cluster distributions is needed as a judgment standard of concept drift.

In general, a user maybe have an expected level for exploring out all concept drifts in a data stream. In other words, we hope to explore concept drifts under a probabilistic level guarantee.

Let α denote the expected level of concept drift detection. By using the approximate solution of the density function of $d(C^{t-1}, C'^{t})$, we select a threshold η_α of dissimilarity such that $P(d(C^{t-1}, C'^{t}) \geqslant \eta_\alpha) = \alpha$. Some dissimilarity thresholds η_α under the expected levels $\alpha = 0.9$ and $\alpha = 0.95$ are shown in Table 4.

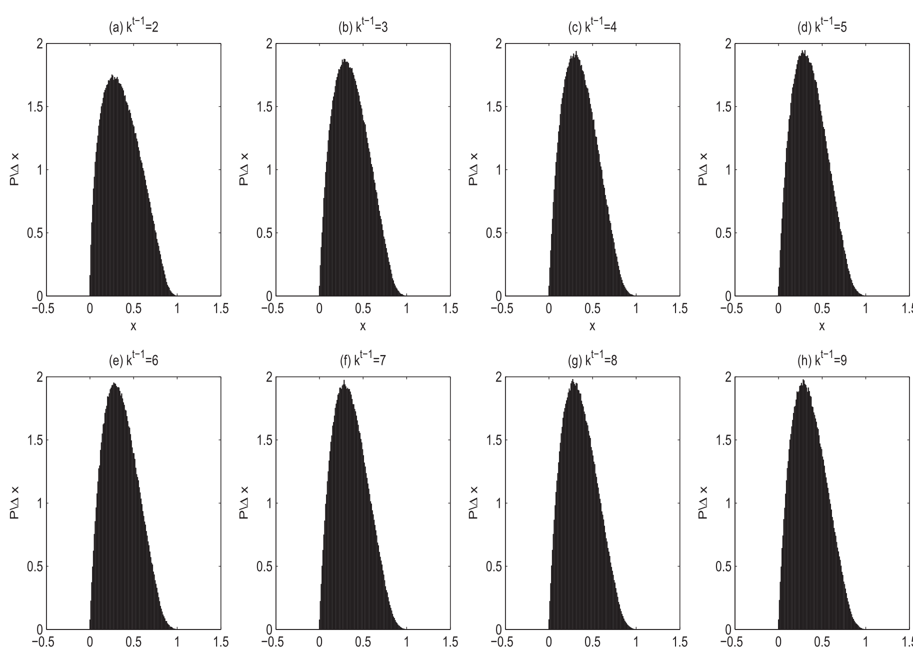

Fig. 3. Frequency distribution of $d(C^{t-1}, C'^{t})$, $2 \leqslant k^{t-1} \leq 9$.

Table 4
Some dissimilarity thresholds $\eta_{0.9}$ and $\eta_{0.95}$.

k^{t-1}	$\eta_{0.9}$	$\eta_{0.95}$	k^{t-1}	$\eta_{0.9}$	$\eta_{0.95}$
2	0.12	0.08	6	0.13	0.09
3	0.13	0.09	7	0.13	0.09
4	0.13	0.09	8	0.13	0.09
5	0.13	0.09	9	0.13	0.09

Example 2 (*Continued from Example 1*). Consider the categorical data set in Table 2. Suppose the outlier threshold $\theta = 0.2$. By setting $|outliers'^2| = 4$, the ratio of outliers in S^2 is $\frac{4}{6} > \theta$. Therefore, S^2 is considered as a concept drifting window, and the data in S^2 must be reclustered. Suppose that the reclustering result of S^2 is $C^2 = \left\{c_1^2, c_2^2, outliers^2\right\}$, where $c_1^2 = \{\mathbf{x}_7, \mathbf{x}_9\}$, $c_2^2 = \{\mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}\}$ and $outliers^2 = \emptyset$. According to Definition 3.2 (or Eq. (4)), we compute the dissimilarity values between each data point in S^3 and each cluster in C^2 and show the results in Table 5.

According to Eq. (5), the thresholds are set to $\lambda_1^2 = 0$ and $\lambda_2^2 = 1.6096$. From Table 5, we obtain that $c_1'^3 = \{\mathbf{x}_{13}, \mathbf{x}_{14}, \mathbf{x}_{16}, \mathbf{x}_{18}\}$, $c_2'^3 = \{\mathbf{x}_{17}\}$ and $outliers'^3 = \{\mathbf{x}_{15}\}$. Therefore, we have $C'^3 = \left\{c_1'^3, c_2'^3, outliers'^3\right\}$, $|outliers'^3| = 1$ and the ratio of outliers in S^3 is $\frac{1}{6} \leqslant \theta (\theta = 0.2)$. The cluster distribution vectors are $\overline{C^2} = (\frac{1}{3}, \frac{2}{3}, 0)$ and $\overline{C'^3} = (\frac{2}{3}, \frac{1}{6}, \frac{1}{6})$.

The dissimilarity between clustering results C^2 and C'^3 is

$$d(C^2, C'^3) = \frac{1}{\sqrt{2}}\sqrt{\left(\frac{1}{3} - \frac{2}{3}\right)^2 + \left(\frac{2}{3} - \frac{1}{6}\right)^2 + \left(0 - \frac{1}{6}\right)^2} = \frac{1}{\sqrt{2}}\sqrt{\frac{7}{18}} = 0.44.$$

Suppose that the expected level α is set 0.95, and according to Table 4, the cluster distribution threshold $\eta_{0.95}$ is set to $0.08(k^{t-1} = 2)$. For $d(C^2, C'^3) > 0.08$, S^3 is considered as a concept drift window. But S^3 need not perform reclustering, because every data point (except outliers) has been properly labeled. Therefore the clustering result of S^3 is $C^3 = \left\{c_1^3, c_2^3, outliers^3\right\}$, where $c_1^3 = \{\mathbf{x}_{13}, \mathbf{x}_{14}, \mathbf{x}_{16}, \mathbf{x}_{18}\}$, $c_2^3 = \{x_{17}\}$ and $outliers^3 = \{\mathbf{x}_{15}\}$.

4.4. Concept drift detection algorithm

Following the discussion above, concept drift detection mainly depends upon two indexes, i.e., the ratio of outliers and the dissimilarity measure between two cluster distributions. Two kinds of technologies, clustering and data labeling are employed in the detecting process. According to the flow as shown in Example 2, an algorithm, called CDD (Concept Drift Detection) is designed for concept drift detection and shown in Table 6.

The time complexity of CDD is analyzed as follows. According to Section 3.3, the time complexity of data labeling algorithm MDDL is $O(k * N * d * q)$. Checking the ratio of outliers and comparing the dissimilarity between two cluster distributions are not time-consuming. Only when the ratio of outliers in the temporal clustering result C'^t is larger than the preestablished outlier threshold θ, reclustering is performed. Because the time complexity of most clustering algorithms is $O(N^2)$, the real bottleneck of the execution time in CDD occurs on the reclustering step, i.e. Step 7. Therefore, if the time complexity of reclustering algorithm is $O(N^2)$, then the time complexity of CDD is $O(N^2)$.

Table 5
Dissimilarity between each data point in S^3 and each cluster in C^2.

	$\mathbf{x}_{13}$	$\mathbf{x}_{14}$	$\mathbf{x}_{15}$	$\mathbf{x}_{16}$	$\mathbf{x}_{17}$	$\mathbf{x}_{18}$
c_1^2	0	0	8.2647	0	8.2647	0
c_2^2	10.8289	10.8289	10.8289	10.8289	0.3645	10.8289

Table 6
Concept Drift Detection CDD.

1 Function ConceptDriftDetection($C^{t-1}, C^t, S^t, \theta, \eta_\alpha$)
2 Begin
3 Drifting = false;
4 out^t = DataLabeling(C^{t-1}, S^t, C'^t);
5 If $\frac{out^t}{N} > \theta$
6 Drifting = true;
7 C^t is obtained by calling the reclustering algorithm on S^t;
8 Return Drifting;
9 Else
10 Generate $d(C^{t-1}, C'^t)$ according to Definition 4.2;
11 If $d(C^{t-1}, C'^t) > \eta_\alpha$
12 Drifting = true;
13 End if;
14 $C^t = C'^t$;
15 End if;
16 Return Drifting;
17 End function;

5. Cluster evolving analysis

In many applications, a user may want to know not only if concept drift happened in the current window with respect to the previous one, but also how it happened. In this section, a measure of the dissimilarity between two clusters with adjacent time stamps is defined for analyzing cluster evolving process. Furthermore, an algorithm of Cluster Evolving Analysis, named CEA, is proposed to explain the concept drift by analyzing the relation between two clustering results with adjacent time stamps.

5.1. Dissimilarity between two clusters

The key of cluster evolving analysis is to judge where a cluster in the current window is from. One cluster in the current window maybe newly emerges. And another one perhaps can be regarded to evolve from some clusters in the previous window if they are enough similar to each other. To this end, we need a measure to quantize the similarity (or equivalently dissimilarity) between two clusters.

Definition 5.1. Let C^{t-1} and C^t be the clustering results of S^{t-1} and S^t respectively, and $c_m^{t-1} \in C^{t-1}$ and $c_n^t \in C^t$ are two clusters, where $1 \leqslant m \leqslant k^{t-1}$ and $1 \leqslant n \leqslant k^t$. We call

$$\begin{aligned} d\left(c_m^{t-1}, c_n^t\right) &= IE\left(c_m^{t-1}, c_n^t\right) \\ &= \left(\left|c_m^{t-1}\right| + \left|c_n^t\right|\right)H\left(c_m^{t-1} \cup c_n^t\right) - \left|c_m^{t-1}\right|H\left(c_m^{t-1}\right) \\ &\quad - \left|c_n^t\right|H\left(c_n^t\right) \end{aligned} \tag{8}$$

the dissimilarity measure between c_m^{t-1} and c_n^t.

5.2. Cluster representative

In order to intuitively show the cluster evolving process in a diagram, we hope to construct a representative for each cluster by synthesizing the information of all samples in the cluster. A representative of a cluster may be either a real sample from the cluster or a fictitious sample.

Let $C^t = \left\{c_1^t, c_2^t, \ldots, c_{k^t}^t\right\}$ be the clustering result of S^t. For a cluster $c_m^t \in C^t$, if a real (or constructive) data point $RP(c_m^t)$ is regarded as a representative of c_m^t, it should have the following characteristics (denoted each component of $RP(c_m^t)$ by $x^j(RP(c_m^t))$ $(1 \leqslant j \leqslant d)$):

(1) The data points whose codomain of x^j is $x^j(RP(c_m^t))$ should appear with a higher frequency in c_m^t.
(2) The frequency of the data points whose codomain of x^j is $x^j(RP(c_m^t))$ in c_m^t should have a larger portion in the total frequency of the data points that the codomain of x^j is $x^j(RP(c_m^t))$ occurring in all clusters of C^t.
(3) The frequencies of the data points whose codomain of x^j is $x^j(RP(c_m^t))$ occurring in each cluster of C^t should be as inhomogeneous as possible.

For c_m^t and $v \in V_j$, we denote $c_{m,v}^t = \{\mathbf{x} \in c_m^t : x^j(\mathbf{x}) = v\}$ and $C_v^t = \cup_{m=1}^{k^t} c_{m,v}^t = \left\{\mathbf{x} \in \cup_{m=1}^{k^t} c_m^t : x^j(\mathbf{x}) = v\right\}$. Let $u_m^t(v) = \frac{|c_{m,v}^t|}{|c_m^t|}$, $v_m^t(v) = \frac{|c_{m,v}^t|}{\sum_{m'=1}^{k^t}\left|c_{m',v}^t\right|} = \frac{|c_{m,v}^t|}{|C_v^t|}$, and $H^t(v) = -\sum_{m=1}^{k^t}\left(v_m^t(v)\right)\log\left(v_m^t(v)\right)$.

Based on the discussion above, we define $RP(c_m^t)$ as follows.

Definition 5.2. Let $C^t = \left\{c_1^t, c_2^t, \ldots, c_{k^t}^t\right\}$ be the clustering result of S^t. For a cluster $c_m^t \in C^t$, denote $x^j(RP(c_m^t)) = \arg\max_{v \in V_j} \frac{u_m^t(v) * v_m^t(v)}{H^t(v)}$. We call $RP(c_m^t) = (x^j(RP(c_m^t)))_{j=1}^d$ the representative of c_m^t.

5.3. Cluster evolving analysis algorithm

The cluster evolving analysis algorithm have two main tasks. One is to judge where a cluster in the current window is evolved from, and the another is to compute the representative of a cluster. According to CDD described in Section 4, two cases are needed to process.

Case 1. The current sliding window S^t does not perform reclustering process, i.e., the outliers ratio is lower than the outlier threshold θ in the temporary clustering result C'^t. In this case, we can exactly know which cluster in C^{t-1} has evolved into the target cluster in $C'^t (C^t = C'^t)$ by its label.

Case 2. The reclustering process is performed in the current sliding window S^t, i.e., the outlier ratio is higher than the outlier threshold θ in the temporary clustering result C'^t.

In this case, we need to compute the dissimilarity for each pair of clusters from the adjacent sliding windows to find out the cluster in the previous window, from which the target cluster is evolved, and then compute the representative of each cluster in the current window.

A Cluster Evolving Analysis algorithm CEA, shown in Table 7, is designed to intuitively analyze cluster evolving process between two adjacent sliding windows. The time complexity of CEA is $O(N * d * q)$, where q is the number of distinct attribute values of a domain.

The following example illustrates the cluster evolving process with the algorithm CEA.

Example 3 (*Continued from Examples 1 and 2*). Suppose the threshold of cluster evolving $\xi = 10$. According to Definition 5.2, the representatives of clusters in sliding windows S^1, S^2 and S^3 are shown in Table 8.

5.3.1. The cluster evolving process from S^1 to S^2

From Example 1, the clustering result in S^1 is $C^1 = \left\{c_1^1, c_2^1, outliers^1\right\}$, where $c_1^1 = \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_6\}$ with its representative (A, F, T), $c_2^1 = \{\mathbf{x}_2, \mathbf{x}_4\}$ with its representative (X, F, R), and $outliers^1 = \emptyset$. Because the ratio of outliers of $S^2\left(\frac{4}{6}\right)$ is larger than the outlier threshold $\theta = 0.2$, concept drift occurs in the sliding window S^2. So the reclustering process must be performed in S^2 and the reclustering result is $C^2 = \left\{c_1^2, c_2^2, outliers^2\right\}$, where $c_1^2 = \{\mathbf{x}_7, \mathbf{x}_9\}$ with its representative (X, F, R), $c_2^2 = \{\mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}\}$ with its representative (C, P, D), and $outliers^2 = \emptyset$. According to Case 2 of cluster evolving, using Eq. (8), we compute the dissimilarity values of each cluster pair (c_m^1, c_n^2) $(1 \leqslant m, n \leqslant 2)$. The result is shown in Table 9. From Table 9 and $\xi = 10$, we know that the cluster c_1^2 evolves from the cluster c_2^1 of the sliding window S^1, and the cluster c_2^2 is a prominently emerged cluster in S^2. In fact,

Table 7
Cluster Evolving Analysis CEA.

```
Procedure ClusterEvolvingAnalysis(C^{t-1}, C^t, ξ)
Begin
  For m = 1 to k^{t-1}
    Counting RP(c_m^{t-1});
    Drawing a circle with the center location (t - 1, m) for c_m^{t-1};
  End;
  If outliers^{t-1} ≠ ∅
    Drawing a circle with the center location (t - 1, m + 1) for outliers^{t-1};
  end
  For m = 1 to k^t
    Counting RP(c_m^t);
    Drawing a circle with the center location (t, m) for c_m^t;
  End;
  If outliers^t ≠ ∅
    Drawing a circle with the center location (t, m + 1) for outliers^t;
  end
  If the reclustering process is performed in S^t
    For m = 1 to k^{t-1}
      For n = 1 to k^t
        If d(c_m^{t-1}, c_n^t) ⩽ ξ
          Connect c_m^{t-1} and c_n^t with line with an arrow;
        End if;
      End for;
    End for;
  End if;
End procedure;
```

Table 8
Representative of each cluster.

Cluster	Representative
$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_6\}$	$RP(c_1^1) = A, F, T$
$c_2^1 = \{\mathbf{x}_2, \mathbf{x}_4\}$	$RP(c_2^1) = X, F, R$
$c_1^2 = \{\mathbf{x}_7, \mathbf{x}_9\}$	$RP(c_1^2) = X, F, R$
$c_2^2 = \{\mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}\}$	$RP(c_2^2) = C, P, D$
$c_1^3 = \{\mathbf{x}_{13}, \mathbf{x}_{14}, \mathbf{x}_{16}, \mathbf{x}_{18}\}$	$RP(c_1^3) = X, F, R$
$c_2^3 = \{x_{17}\}$	$RP(c_2^3) = C, P, D$

Table 9
Dissimilarity between clusters c_m^1 and c_n^2.

	$c_1^1 = \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_6\}$	$c_2^1 = \{\mathbf{x}_2, \mathbf{x}_4\}$
$c_1^2 = \{\mathbf{x}_7, \mathbf{x}_9\}$	11.0196	1.2451
$c_2^2 = \{\mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}\}$	24	16.5293

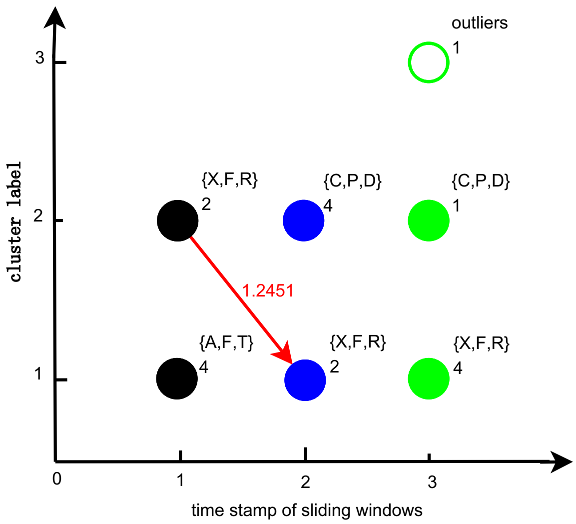

Fig. 4. Relationship between clusters at different time stamps.

this evolving process can also be clearly seen by the representatives of clusters in S^1 and S^2.

5.3.2. The cluster evolving process from S^2 to S^3

From Example 2, we know that the outlier ratio of S^3 $(\frac{1}{6})$ is lower than the outlier threshold $\theta = 0.2$, so the reclustering is unnecessary. However, since the difference of cluster distributions between C^2 and C^3 is larger than the cluster distribution threshold $\eta_{0.95} = 0.08(k^{t-1} = 2)$, concept drift happens in S^3. So the cluster evolving process from S^2 to S^3 belongs to Case 1. The clustering result $C^3 = \left\{c_1^3, c_2^3, outliers^3\right\}$ is obtained by data labeling according to C^2, where $c_1^3 = \{\mathbf{x}_{13}, \mathbf{x}_{14}, \mathbf{x}_{16}, \mathbf{x}_{18}\}$ with its representative (X, F, R), $c_2^3 = \{x_{17}\}$ with its representative (C, P, D), and $outliers^3 = \{\mathbf{x}_{15}\}$. Therefore, the cluster $c_m^3(m = 1, 2)$ in S^3 evolves from c_m^2 in S^2.

Fig. 4 intuitively shows the cluster evolving process from the sliding window S^1 to S^3 via S^2. In Fig. 4, the horizontal direction is the time stamp of sliding windows; the circles in a column indicate the clustering result of a sliding window. Note that the colors of circles are different. The window that includes black circles represents no concept-drifting in it; the window that includes blue circles represents concept-drifting in the case when the outlier ratio is higher than the outlier threshold; the window that includes green[1] circles represents concept-drifting in the case when the cluster distribution has an evident change. For each circle there are two kinds of information, the representative of the corresponding cluster and the number of the points in the cluster of the sliding window. The outliers are represented by the hollow circle. In addition, the weighted line with an arrow linking the related circles represents the cluster evolving relation. The weight over a line with an arrow is the dissimilarity between the clusters linked by the line.

6. Experimental results

In this section, we carry out some experiments to demonstrate the performance of the presented framework for categorical data streams. In Section 6.1, the test environment and the data source are described. The method for constructing test data streams is illustrated in Section 6.2. The evaluation indexes and experimental results on data labeling, on concept drift detection and on clustering result are presented in Sections 6.3–6.5 respectively. And Section 6.6 shows the visualizing of cluster evolution. Section 6.7 studies how the parameters θ and α affect the performance of CDD.

6.1. Test environment and data source

All experiments are conducted on a PC with Intel Pentium 2.66-GHz processor and 3.37-GB memory running Windows XP SP3 operation system. In all experiments, the *k*-modes algorithm [45] is chosen to execute the initial clustering and reclustering.

We synthesize various kinds of test data streams using two kinds of raw data. The first includes Soybean, Zoo, Dermatology and DNA, taken from the UCI's (University of California at Irvine) data repository [46]. The second is a text data set taken from corpus of the first session and the second session of Chinese orientation analysis evaluation (COAE). The text data set contains 271 binary attributes and 8 subjects (class labels). For simplicity, we name this data set as Subject text. The main features of the raw data, such as sample number, attribute number and class number are shown in Table 10.

The KDD-CUP'99 Network Intrusion Detection data set used by [26,36] does not be used in this paper, because the data in a sliding window with some window size belong to the same class (or cluster). So it is difficult to perform initial clustering and reclustering on the sliding window.

In Cao's framework[36], concept-drifting windows are detected based on the distance between adjacent sliding windows. When computing the distance, all data points in each window are regarded as a cluster without taking into consideration both cluster distribution and outliers. So we only compare our framework with Chen's framework [26] in evaluating on concept drift detection and clustering result of data streams.

6.2. Constructing test data streams

- *Generating a concept:* At first, determine the size of a concept, i.e., the number of samples in it. Next, for all classes of a raw data set, give an expected class distribution. At last, according to the expected class distribution we randomly extract samples from the known classes of a raw data such that they achieve the predefined concept size.
- *Generating a data stream with concept drift:* According to the expected size of the data stream, repeatedly extract concepts from the generated concepts several times, and then randomly arrange them. Concept drift happens when two different concepts are adjacently arranged.

6.3. Evaluation on data labeling

In this section, we design two experiments to evaluate the efficiency of MDDL algorithm. At first, we inspect the necessity of data labeling algorithm in clustering. For this end, we compare the

Table 10
Main features of the raw data.

Data set	Sample number	Attribute number	Class number
Soybean	47	35	4
Zoo	101	16	7
Dermatology	366	33	6
DNA	3190	60	3
Subject text	1280	271	8

[1] For interpretation of color in Fig. 4, the reader is referred to the web version of this article.

Table 11
Parameter settings of the sample concepts.

Concept	Class number	Class distribution in a concept	Size of a concept	Real data set
concept_1	4	(0.1,0.4,0.4,0.1)	28	Soybean
concept_2	4	(0.4,0.1,0.1,0.4)	28	Soybean
concept_3	7	(0.1,0.2,0.1,0.1,0.1,0.2,0.2)	61	Zoo
concept_4	7	(0.2,0.1,0.2,0.1,0.2,0.1,0.1)	61	Zoo
concept_5	6	(0.1,0.1,0.5,0.2,0.05,0.05)	220	Dermatology
concept_6	6	(0.2,0.2,0.2,0.1,0.2,0.1)	220	Dermatology
concept_7	3	(0.2,0.6,0.2)	1914	DNA
concept_8	3	(0.4,0.2,0.4)	1914	DNA
concept_9	8	(0.1,0.2,0.1,0.1,0.1,0.2,0.1,0.1)	768	Subject text
concept_10	8	(0.1,0.2,0.1,0.1,0.1,0.2,0.1,0.1)	768	Subject text

Table 12
Time and accuracy of MDDL and k-modes algorithm. The optimal values of each index of various methods on all data sets are in bold.

Reference concept	Target concept	MDDL		k-modes	
		Time	$Accuracy_l$	Time	$Accuracy_c$
concept_1	concept_2	**0.0118**	**0.9975**	0.0257	0.8825
concept_2	concept_1	**0.0126**	**1.0000**	0.0304	0.8500
concept_3	concept_4	**0.0165**	**0.9658**	0.0577	0.7308
concept_4	concept_3	**0.0158**	**0.9717**	0.0570	0.7775
concept_5	concept_6	**0.1196**	**0.9702**	0.4978	0.7386
concept_6	concept_5	**0.1234**	**0.9859**	0.5084	0.6911
concept_7	concept_8	**1.8819**	**0.9538**	10.7563	0.4605
concept_8	concept_7	**1.8546**	**0.9551**	9.4860	0.6027
concept_9	concept_10	**3.1805**	**0.9689**	28.4023	0.5151
concept_10	concept_9	**3.1852**	**0.9711**	26.1969	0.5036

labeling accuracy and the time cost of MDDL algorithm with those of k-modes algorithm which is used to reclustering. Next, we compare the labeling accuracy and the time cost of MDDL with those of several typical data labeling algorithms.

According to the method described in Section 6.2, we generate some sample concepts whose parameter settings are given in Table 11. Then we select the reference concepts and the corresponding target concepts from the sample concepts about the same real data. MDDL algorithm assigns a class label to each data point in the target concept based on the reference concept. At the same time, the target concept is reclustered by k-modes algorithm, where the k is appointed as the number of classes in the target concept, and the initial cluster centers are randomly selected.

Table 12 shows the average execution time and the average accuracy of MDDL algorithm and k-modes algorithm in 20 experiments. The conference concepts and target concepts are from the sample concepts shown in Table 11. From Table 12, we can see that the average clustering time consumed by k-modes algorithm is about 2–8 times the average labeling time by MDDL. This indicates that using data labeling algorithm can greatly accelerate the clustering process. The cluster labels of conference concepts are obtained by using the real class label of the data in conference concepts. The labeling accuracy $Accuracy_l$ is defined as

$$Accuracy_l = \frac{b}{a}, \tag{9}$$

Table 13
Accuracy and time of some data labeling algorithms. The optimal values of each index of various methods on all data sets are in bold.

Reference concept	Target concept	MDDL		NIR		AVF		RMF	
		$Accuracy_l$	Time	$Accuracy_l$	Time	$Accuracy_l$	Time	$Accuracy_l$	Time
concept_1	concept_1	**1.0000**	**0.0102**	**1.0000**	0.0157	**1.0000**	0.0110	0.8100	0.0281
concept_1	concept_2	**1.0000**	**0.0102**	0.9850	0.0172	0.9975	0.0133	0.2075	0.0280
concept_2	concept_1	0.9950	0.0133	0.9950	0.0180	**0.9975**	**0.0125**	0.3675	0.0305
concept_2	concept_2	**1.0000**	0.0140	**1.0000**	0.0195	**1.0000**	**0.0125**	0.8875	0.0313
concept_3	concept_3	**0.9942**	**0.0156**	0.9417	0.0274	0.9658	0.0188	0.9125	0.0906
concept_3	concept_4	**0.9567**	0.0180	0.8625	0.0234	0.9217	**0.0117**	0.8325	0.0890
concept_4	concept_3	**0.9600**	**0.0180**	0.9033	0.0281	0.9275	0.0203	0.8208	0.0898
concept_4	concept_4	**0.9925**	**0.0165**	0.9208	0.0273	0.9492	0.0219	0.8983	0.0913
concept_5	concept_5	**0.9955**	0.1170	0.9786	0.1428	0.9795	**0.1154**	0.6923	0.5881
concept_5	concept_6	**0.9536**	0.1172	0.9091	0.1415	0.9234	**0.1167**	0.4230	0.5930
concept_6	concept_5	**0.9816**	0.1226	0.9764	0.1446	0.9636	**0.1204**	0.7214	0.6116
concept_6	concept_6	**0.9914**	0.1211	0.9911	0.1461	0.9689	**0.1202**	0.8243	0.6243
concept_7	concept_7	**0.9608**	1.8819	0.6633	1.8485	0.8472	**1.8233**	0.6003	4.7147
concept_7	concept_8	**0.9523**	1.8860	0.3140	1.8525	0.7043	**1.8282**	0.2007	4.7375
concept_8	concept_7	**0.9561**	1.8758	0.5473	1.8377	0.8376	**1.8064**	0.3384	4.7057
concept_8	concept_8	**0.9611**	1.8711	0.7632	1.8326	0.7154	**1.8038**	0.6177	4.7140
concept_9	concept_9	**0.9840**	**3.1783**	0.9785	3.8094	0.6774	3.2930	0.3807	31.8096
concept_9	concept_10	0.9707	**3.1743**	**0.9726**	3.8125	0.6490	3.2883	0.3775	31.8234
concept_10	concept_9	**0.9734**	**3.1759**	0.9721	3.8133	0.6641	3.2922	0.3786	31.8282
concept_10	concept_10	**0.9820**	**3.1804**	0.9797	3.8212	0.6839	3.2914	0.3795	31.8533

Table 14
Parameter setting of DS_1.

Concept	Class number	Class distribution in a concept	Size of a concept
concept_1	3	(0.2,0.3,0.5,0)	1000
concept_2	3	(0.6,0.1,0.3,0)	1000
concept_3	4	(0.2,0.6,0.1,0.1)	1000
concept_4	4	(0.1,0.1,0.2,0.6)	1000
concept_5	2	(0,0.5,0.5,0)	1000

Table 15
Parameter settings of data streams in the experiments.

Setting	Concept number	Class number in a concept	Attribute number	Size of a concept	Real data set
DS_1	20	2–4	35	1000	Soybean
DS_2	20	2–4	35	1000, 2000, 3000	Soybean
DS_3	20	3–7	16	1000	Zoo
DS_4	20	3–7	16	1000, 2000, 3000	Zoo
DS_5	20	3–6	33	1000	Dermatology
DS_6	20	3–6	33	1000, 2000, 3000	Dermatology
DS_7	20	2–3	60	20,000	DNA
DS_8	20	2–3	60	10,000, 20,000, 30,000	DNA
DS_9	20	3–8	271	1000	Subject text
DS_{10}	20	3–8	271	1000, 2000, 3000	Subject text

where a is the number of data points of the target concept and b is the number of data points that are correctly labeled by a labeling algorithm. The clustering accuracy $Accuracy_c$ of k-modes algorithm is defined as [47]

$$Accuracy_c = \frac{\sum_{m=1}^{k} a_m}{N}, \tag{10}$$

where N is the size of target concept, k is the number of clusters, and a_m is the number of data points in some real class that it is larger than the number of data points in any other real class ($1 \leqslant m \leqslant k$). From Table 12, we can see that the average labeling accuracy of MDDL algorithm is higher than that of k-modes algorithm. The reason is that the class information of reference concept is used by the MDDL algorithm.

Table 13 shows the comparison result of data labeling accuracy of MDDL with data labeling algorithm proposed by Chen et al. [26] (abbreviated as NIR) and two kinds of data labeling algorithms by Cao et al. [36] and Cao and Liang [48] (abbreviated as AVF and RMF respectively). The accuracy and time in Table 13 are the average of 20 experiments. The conference concepts and target concepts are from the sample concepts shown in Table 11. In order to exclude the interference of the clustering accuracy to labeling accuracy, the cluster labels of conference concepts are using the real class label of the data in conference concepts instead of calling clustering algorithm. The labeling accuracy $Accuracy_l$ is defined by Eq. (9).

Generally speaking, the labeling accuracy of MDDL is higher than that of other data labeling algorithms regardless of synthetic concepts in Table 13. The experimental results demonstrate that the execution times of MDDL, NIR and AVF are almost the same. The RMF algorithm is relatively time-consuming.

Table 16
Precision and recall on concept drift detection. The optimal values of each index of various methods on all data sets are in bold.

Data stream	Window size	Concept-drift number	CDD		Chen's framework	
			Precision	Recall	Precision	Recall
DS_1	300	27.7000	**0.8460**	**0.8931**	0.7836	0.4965
	400	24.7000	**0.9105**	**0.9275**	0.8535	0.5097
	500	15.7500	0.8730	**0.9805**	**0.8755**	0.8646
DS_2	300	21.2500	0.5626	**0.8749**	**0.5843**	0.4605
	400	19.3500	**0.6322**	**0.9089**	0.6082	0.5417
	500	13.2500	0.6172	**0.9964**	**0.6703**	0.8598
DS_3	300	26.9500	**0.8829**	**0.9322**	0.6329	0.4810
	400	24.6500	**0.9591**	**0.9699**	0.7503	0.6264
	500	16.4500	**0.9458**	**1.0000**	0.7477	0.8826
DS_4	300	22.6500	**0.6608**	**0.9361**	0.4062	0.5378
	400	19.0000	**0.6866**	**0.9777**	0.4524	0.5619
	500	13.2000	**0.6959**	**1.0000**	0.4328	0.8491
DS_5	300	25.9500	**0.8843**	**0.8507**	0.7834	0.3247
	400	23.5500	**0.9264**	**0.8948**	0.8440	0.3981
	500	15.6500	**0.8914**	**0.9579**	0.8595	0.5771
DS_6	300	21.5500	**0.7368**	**0.9155**	0.7278	0.3903
	400	18.8000	0.7173	**0.9033**	**0.7174**	0.3819
	500	13.3500	**0.7520**	**1.0000**	0.6753	0.6871
DS_7	3000	27.2000	**0.7750**	0.5139	0.2066	**0.9542**
	4000	16.1500	**0.7450**	0.8020	0.1659	**0.9358**
	5000	15.8500	**0.7516**	0.8238	0.2011	**0.9454**
DS_8	3000	17.8500	**0.5053**	0.5033	0.1349	**0.9589**
	4000	16.6500	**0.5275**	0.5544	0.1677	**0.9619**
	5000	11.3000	**0.5424**	0.7467	0.1445	**0.9478**
DS_9	300	27.2000	**0.7317**	**0.7861**	0.4947	0.2125
	400	24.0000	**0.7886**	**0.8876**	0.6808	0.2884
	500	16.0000	**0.7332**	**0.8741**	0.6133	0.3853
DS_{10}	300	20.1500	**0.5664**	**0.7834**	0.3072	0.1636
	400	19.4000	**0.6090**	**0.9042**	0.4176	0.2099
	500	13.0000	**0.5467**	**0.9201**	0.4665	0.3509

Table 17
Clustering accuracy and time on data streams. The optimal values of each index of various methods on all data sets are in bold.

Data stream	Window size	Proposed framework		Chen's framework	
		$Accuracy_{DS}$	$Time_{DS}$	$Accuracy_{DS}$	$Time_{DS}$
DS_1	300	0.8841	**7.3789**	**0.8929**	15.2235
	400	**0.8832**	**7.6312**	0.8731	16.3992
	500	0.8809	**8.5547**	**0.8871**	18.9118
DS_2	300	**0.9179**	**9.9157**	0.9032	19.6533
	400	**0.9029**	**10.6452**	0.8969	21.9668
	500	0.8999	**11.2194**	**0.9067**	23.9077
DS_3	300	0.8205	**4.3472**	**0.8331**	9.2450
	400	**0.8264**	**5.2596**	0.8190	10.4656
	500	**0.8298**	**5.6831**	0.8265	10.7576
DS_4	300	0.8398	**5.9392**	**0.8499**	14.1804
	400	0.8414	**6.2432**	**0.8473**	14.4982
	500	0.8328	**6.9414**	**0.8360**	15.7385
DS_5	300	**0.8113**	**5.6628**	0.7937	16.6149
	400	**0.8070**	**6.7461**	0.7818	18.2662
	500	**0.8110**	**8.4839**	0.8059	19.6862
DS_6	300	0.8402	**7.2607**	**0.8461**	24.5631
	400	**0.8278**	**7.8724**	0.8270	25.9727
	500	0.8240	**9.9515**	**0.8387**	29.4253
DS_7	3000	**0.7712**	**155.7305**	0.5432	2.2510e+03
	4000	**0.7668**	**209.5383**	0.5471	2.1862e+03
	5000	**0.7540**	**260.4493**	0.5467	2.2789e+03
DS_8	3000	**0.7779**	**146.5806**	0.5442	2.2560e+03
	4000	**0.7531**	**196.7648**	0.5431	2.3053e+03
	5000	**0.7429**	**249.4422**	0.5467	2.2971e+03
DS_9	300	**0.6832**	**29.6531**	0.5773	137.8438
	400	**0.6820**	**36.7274**	0.5936	159.3646
	500	**0.6572**	**35.2187**	0.5928	157.5071
DS_{10}	300	**0.7344**	**39.7274**	0.6084	179.3926
	400	**0.7442**	**46.5954**	0.5896	184.4702
	500	**0.7329**	**71.9421**	0.6034	214.4103

6.4. Evaluation on concept drift detection

We generate test data streams by the following steps: (1) We create some sample concepts as the base of generating data streams with concept drift using every real data set. For example, the parameter setting of sample concepts used to generate the data stream DS_1 is shown in Table 14. (2) We then randomly extract some sample concepts from the same real data set several times and combine them to generate a data stream with concept drift. (3) In a data stream, a concept drift is obtained by adjacently combining two different sample concepts. The parameter settings of 10 data streams utilized in the experiments are shown in Table 15.

We adopt the two widely used indexes: precision and recall to evaluate CDD algorithm. They are defined as

$$Precision = \frac{c}{b} \tag{11}$$

and

$$Recall = \frac{c}{a}, \tag{12}$$

where a is the number of concept-drifting windows, b is the number of concept-drifting windows detected by a concept drift detection algorithm, and c is the number of concept-drifting windows that are correctly detected by the algorithm.

In CDD, two thresholds, the outlier threshold θ and the expected level α need to be given. In the following experiments, we set the outlier threshold $\theta = 0.1$, the expected level $\alpha = 0.95$, and then $\eta_{0.95} = 0.8/0.9$ according to Table 4. In Chen's framework, three thresholds need to be given. The outlier threshold is set 0.1, the cluster variation threshold is set 0.1, and the cluster difference threshold is set 0.5. The precisions and recalls are shown in Table 16 with different window sizes. The precisions and recalls are the average of 20 experiments.

Generally speaking, CDD is superior to Chen's framework on all indexes on $DS_1, DS_2, \ldots, DS_6, DS_9$ and DS_{10} in Table 16. Especially, the recalls of CDD are higher than those of Chen's framework on $DS_1, DS_2, \ldots, DS_6, DS_9$ and DS_{10}. The precisions and recalls on concept drift detection of two frameworks are low on DS_7 and DS_8 because of the poor clustering performance of the k-modes algorithm [45] on the raw data DNA. The performances of two frameworks decrease a little when a data stream to be detected contains sample concepts in different sizes such as DS_2, DS_4, DS_6, DS_8 and DS_{10}. In addition, the recall values of the frameworks get a small increase with the increase in sliding window size.

6.5. Evaluation on clustering result

The clustering accuracy and the time cost are two important indexes to evaluate data stream clustering. For the current window S^t, the clustering accuracy is defined as [47]

$$Accuracy^t = \frac{\sum_{m=1}^{k^t} a_m^t}{N}, \tag{13}$$

where N is the size of window, k^t is the number of clusters, and a_m^t is the number of data points in some real class that it is larger than the number of data points in any other real class $(1 \leqslant m \leqslant k^t)$.

Moreover, the clustering accuracy of a data stream is defined as the average of all window accuracies, i.e.

$$Accuracy_{DS} = \frac{\sum_{t=1}^{M} Accuracy^t}{M}, \tag{14}$$

where M is the number of sliding windows used to partition a data stream.

The time cost of data stream clustering is evaluated by the average execution time on all sliding windows. It is defined as

$$Time_{DS} = \sum_{t=1}^{M} Time^t, \tag{15}$$

where M is the number of sliding windows and $Time^t$ is the execution time on the window S^t.

The clustering accuracies and time costs of the proposed framework and Chen's framework on the 10 data streams are shown in Table 17 with different window sizes. The accuracies and time costs are the average of 20 experiments.

From Table 17, we can see that the proposed framework and Chen's framework almost have the same clustering accuracies on $DS_1, DS_2, \ldots, DS_6$. The average processing time of the proposed framework is about half of that of Chen's framework on $DS_1, DS_2, \ldots, DS_6$. This means that our framework has a better time efficiency than Chen's framework. Because the clustering performance of the k-modes algorithm [45] on the raw data DNA and Subject text is poor, the clustering accuracy of two frameworks reduces on DS_7, DS_8, DS_9 and DS_{10}. In addition, since the time-consuming of reclustering of Chen's framework is quite large, so Chen's framework is very costing on DS_7, DS_8, DS_9 and DS_{10}.

Figs. 5 and 6 more intuitively compare the clustering accuracy and execution time of the proposed framework and Chen's framework in every sliding window on data streams DS_1 and DS_2 with window size 500. The thresholds in the two frameworks are same as those in the above experiments.

6.6. Visualizing cluster evolution

In order to iconically show the cluster evolving process, the clustering result of the first 10 sliding windows on the data stream DS_1 is graphed in Fig. 7. The cluster evolving threshold ξ is set 50.

From Fig. 7, it is clear that, in the first 10 sliding windows, concept drifting happened two times. They are in sliding windows 3

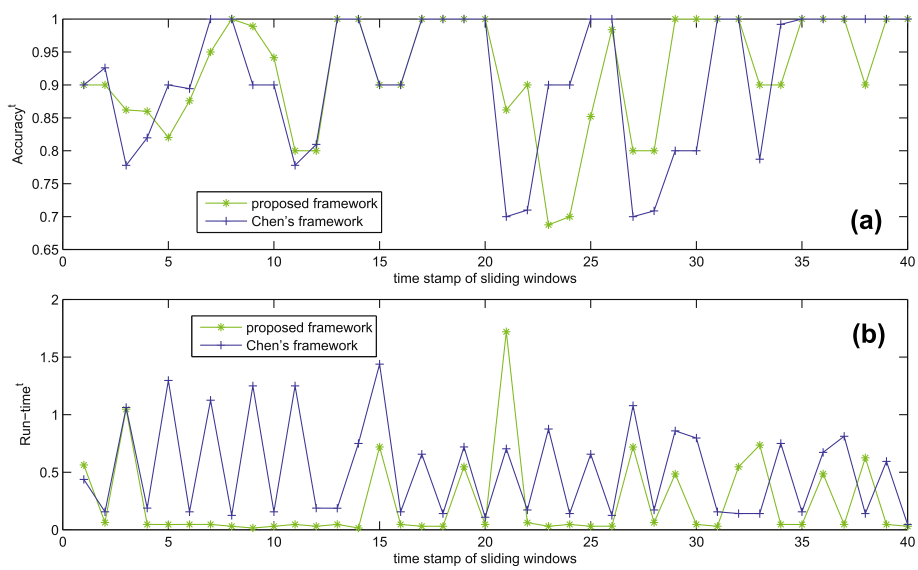

Fig. 5. $Accuracy^t$ and $Time^t$ of two frameworks in all windows on DS_1.

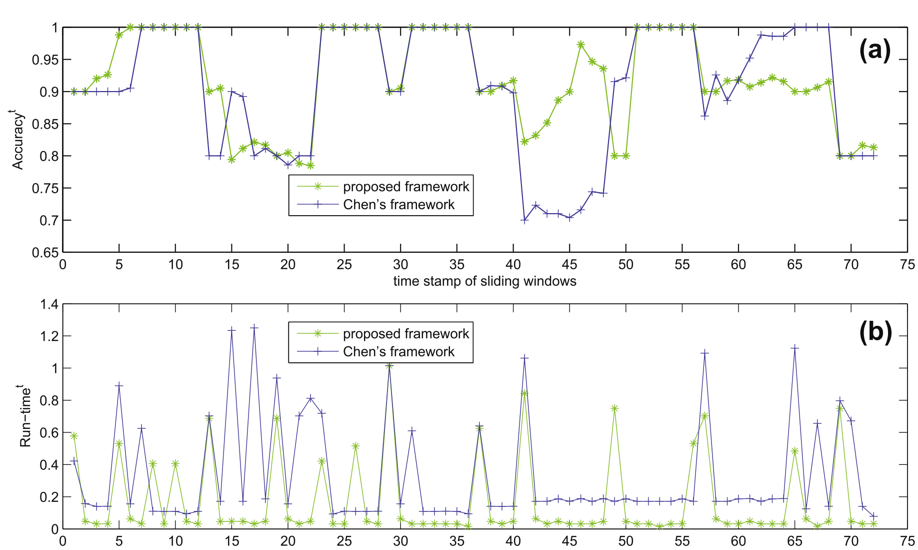

Fig. 6. $Accuracy^t$ and $Time^t$ of two frameworks in all windows on DS_2.

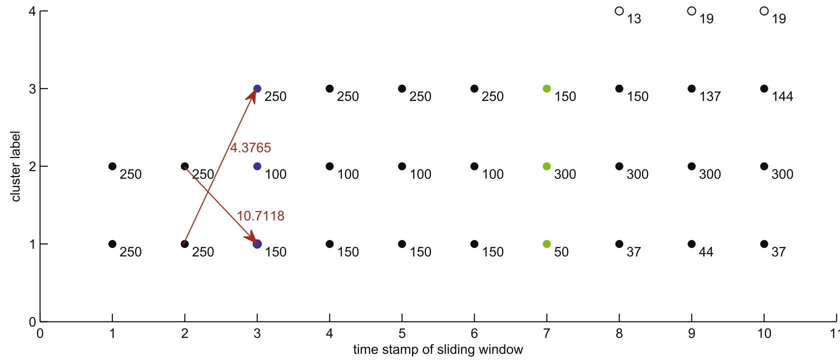

Fig. 7. Cluster evolving process on DS_1 in the first 10 sliding windows.

and 7 respectively. Furthermore, reclustering was performed in the sliding window 3. In the sliding window 7, although the outlier ratio was endurable, the cluster distribution had an evident change than that in the sliding window 6.

6.7. Effect of the parameters θ and α to CDD

We conduct the experiments on some data streams to study how the outlier threshold θ and the expected level α affect the performance of CDD with the window size $N = 300$. The procedure of the experiments follows the steps described in Section 6.4.

The parameters θ and α help us to detect concept drift from the views of the ratio of outliers in a window and the dissimilarity of two cluster distributions between two adjacent windows respectively. The experiment results are shown in Fig. 8.

From Fig. 8(a), we can see that for the 6 data streams the precision of CDD is in a stable state when $\theta \geqslant 0.1$. And from Fig. 8(b), we can see that for all of the 6 data streams the recall of CDD decreases with the increase of θ. In fact, for a detection task we prefer recall to precision, and thus, in a practical application, one should select the value of θ as small as possible to guarantee precision. In our experiments, it works well when $\theta = 0.1$.

From Fig. 8(c), we can see that, with the increase of α, the variation amplitudes of CDD precision are relatively small on all data streams except DS1. From Fig. 8(d), we can see that the recall of CDD increases on all of the 6 data streams with the increase of α. So we should select the value of α as large as possible to guarantee a higher recall of CDD. So, combining Fig. 8(c) and (d), our experiments suggest that the value of α is suitable when it is within [0.9,0.95].

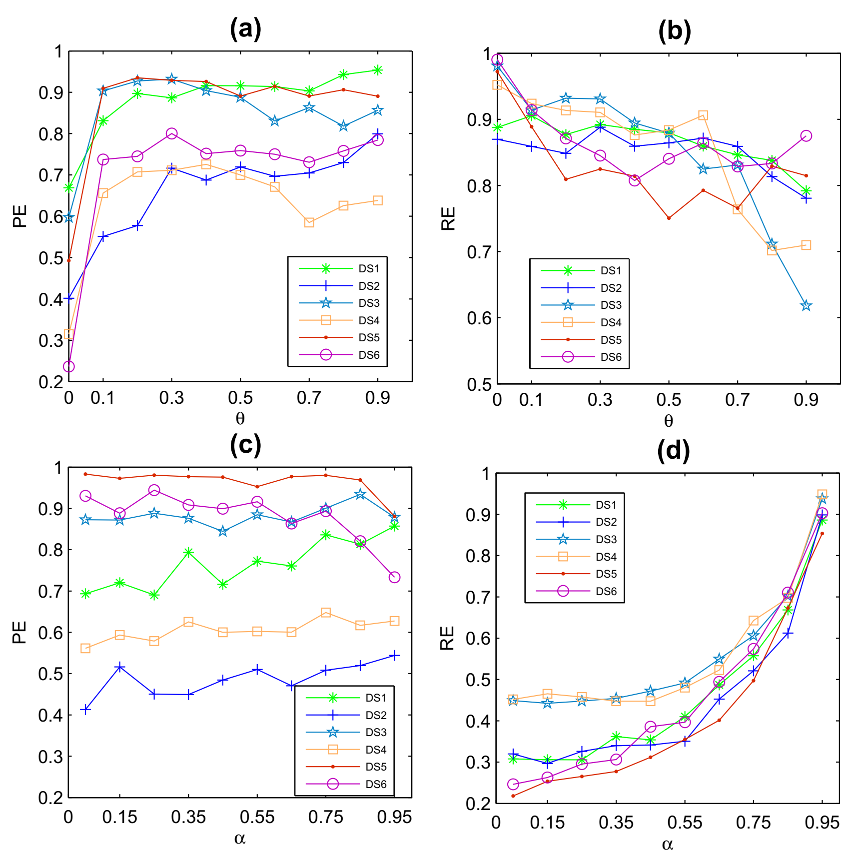

Fig. 8. Effect of the parameters θ and α to CDD.

7. Conclusions

In this paper, we proposed an integrated framework for clustering categorical data streams by using the sliding window technique and the data labeling technique. The point-cluster dissimilarity measure and the cluster–cluster dissimilarity measure are defined by means of incremental entropy. The dissimilarity measure between two cluster distributions is defined based on sample standard deviation. These measures are used to design the data labeling algorithm MDDL, the concept drift detection algorithm CDD and the Cluster Evolving Analysis algorithm CEA in our framework. A method for selecting the threshold of cluster distribution difference is also proposed based on an expected level and the approximate density function of the dissimilarity measure between two cluster distributions. Experimental results on several data streams show that the proposed algorithms are superior to the other algorithms both in generating clustering results and detecting concept drift.

It should be pointed out that the integrated framework introduced in this paper is only applicable to the categorical data streams. Since many real data may be mixed data (described by categorical and numerical variables) or multi-label data, it is expected to carry out the following work to cluster mixed data streams and multi-label data streams in the future:

- Developing point-cluster dissimilarity measures of mixed data and multi-label data and relative data labeling algorithms.
- Designing efficient concept drift detection algorithms and cluster evolving analysis algorithms to mixed data streams and multi-label data streams.

Acknowledgments

This work was supported by the National Natural Science Foundation (Nos. 61272095, 61175067, 71031006, 41101440, 61202365 and 61303107), Shanxi Foundation of Tackling Key Problem in Science and Technology (No. 20110321027-02), Shanxi Scholarship Council of China (2013-014), Project Supported by National Science and Technology (No. 2012BAH33B01), the Natural Science Foundation of Shanxi (No. 2013011066-4) and the Youth Foundation of ShanXi Province (Nos. 2012021015-1 and 2012021012-1). The authors of this paper would like to thank the reviewers for their valuable comments.

References

[1] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2006.
[2] S.-S. Ho, H. Wechsler, A martingale framework for detecting changes in data streams by testing exchangeability, IEEE Trans. Pattern Anal. Mach. Intell. 32 (12) (2010) 2113–2127.
[3] L. Golab, M.T. Özsu, Issues in data stream management, SIGMOD 32 (2) (2003) 5–14.
[4] A. Bulut, A.K. Singh, A unified framework for monitoring data streams in real time, in: Proceedings of the 21th International Conference on Data Engineering, IEEE Computer Society, 2005, pp. 44–55.
[5] M. Datar, A. Gionis, P. Indyk, R. Motwani, Maintaining stream statistics over sliding windows: (extended abstract), in: Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '02, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002, pp. 635–644.
[6] A. Bulut, A.K. Singh, SWAT: hierarchical stream summarization in large networks, in: Proceedings of the 19th International Conference on Data Engineering, IEEE Computer Society, 2003, pp. 303–314.
[7] W. Fan, Systematic data selection to mine concept-drifting data streams, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, ACM, New York, NY, USA, 2004, pp. 128–137.
[8] C.C. Aggarwal, On change diagnosis in evolving data streams, IEEE Trans. Knowl. Data Eng. 17 (5) (2005) 587–600.
[9] C.C. Aggarwal, P.S. Yu, Online analysis of community evolution in data streams, in: Proceedings of the Fifth SIAM International Conference on Data Mining, SIAM, 2005, pp. 56–67.
[10] T. Johnson, S. Muthukrishnan, I. Rozenbaum, Sampling algorithms in a stream operator, in: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05, ACM, New York, NY, USA, 2005, pp. 1–12.
[11] C.C. Aggarwal, J. Han, J. Wang, P.S. Yu, A framework for clustering evolving data streams, in: Proceedings of the 29th International Conference on Very Large Data Bases, VLDB, 2003, pp. 81–92.
[12] C.C. Aggarwal, J. Han, J. Wang, P.S. Yu, A framework for projected clustering of high dimensional data streams, in: Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB, 2004, pp. 852–863.
[13] B.-R. Dai, J.-W. Huang, M.-Y. Yeh, M.-S. Chen, Adaptive clustering for multiple evolving streams, IEEE Trans. Knowl. Data Eng. 18 (9) (2006) 1166–1180.
[14] S. Guha, N. Mishra, R. Motwani, L. O'Callaghan, Clustering data streams, in: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, IEEE Computer Society, 2000, pp. 359–366.
[15] L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, R. Motwani, Streaming-data algorithms for high-quality clustering, in: Proceedings of the 18th International Conference on Data Engineering, IEEE Computer Society, 2002, pp. 685–694.
[16] M. Albertini, R. Mello, Energy-based function to evaluate data stream clustering, Adv. Data Anal. Classif. 7 (4) (2013) 435–464.
[17] P. Domingos, G. Hulten, Mining high-speed data streams, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '00, ACM, New York, NY, USA, 2000, pp. 71–80.
[18] G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '01, ACM, New York, NY, USA, 2001, pp. 97–106.
[19] C.C. Aggarwal, J. Han, J. Wang, P.S. Yu, On demand classification of data streams, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, ACM, New York, NY, USA, 2004, pp. 503–508.
[20] L. Rutkowski, M. Jaworski, L. Pietruczuk, P. Duda, Decision trees for mining data streams based on the gaussian approximation, IEEE Trans. Knowl. Data Eng. 26 (1) (2014) 108–119.
[21] G. Ditzler, R. Polikar, Incremental learning of concept drift from streaming imbalanced data, IEEE Trans. Knowl. Data Eng. 25 (10) (2013) 2283–2301.
[22] A. Ghazikhani, R. Monsefi, H.S. Yazdi, Ensemble of online neural networks for non-stationary and imbalanced data streams, Neurocomputing 122 (2013) 535–544.
[23] N. Lu, G. Zhang, J. Lu, Modified blame-based noise reduction for concept drift, in: Proceedings of the 11th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases, AIKED'12, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2012, pp. 55–60.
[24] T.W. Liao, Clustering of time series data – a survey, Pattern Recognit. 38 (11) (2005) 1857–1874.
[25] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Mach. Learn. 23 (1) (1996) 69–101.
[26] H.-L. Chen, M.-S. Chen, S.-C. Lin, Catching the trend: a framework for clustering concept-drifting categorical data, IEEE Trans. Knowl. Data Eng. 21 (5) (2009) 652–665.
[27] O. Georgieva, F. Klawonn, Dynamic data assigning assessment clustering of streaming data, Appl. Soft Comput. 8 (4) (2008) 1305–1313.
[28] F. Cao, M. Ester, W. Qian, A. Zhou, Density-based clustering over an evolving data stream with noise, in: Proceedings of the Sixth SIAM International Conference on Data Mining, SIAM, 2006, pp. 326–337.
[29] D. Chakrabarti, R. Kumar, A. Tomkins, Evolutionary clustering, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06, ACM, New York, NY, USA, 2006, pp. 554–560.
[30] Y. Chi, X. Song, D. Zhou, K. Hino, B.L. Tseng, Evolutionary spectral clustering by incorporating temporal smoothness, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '07, ACM, New York, NY, USA, 2007, pp. 153–162.
[31] M.M. Gaber, P.S. Yu, Detection and classification of changes in evolving data streams, Int. J. Inform. Technol. Decis. Making 5 (4) (2006) 659–670.
[32] O. Nasraoui, C. Rojas, Robust clustering for tracking noisy evolving data streams, in: Proceedings of the Sixth SIAM International Conference on Data Mining, SIAM, 2006, pp. 618–622.
[33] M.-Y. Yeh, B.-R. Dai, M.-S. Chen, Clustering over multiple evolving streams by events and correlations, IEEE Trans. Knowl. Data Eng. 19 (10) (2007) 1349–1362.
[34] S. Guha, A. Meyerson, N. Mishra, R. Motwani, L. O'Callaghan, Clustering data streams: theory and practice, IEEE Trans. Knowl. Data Eng. 15 (3) (2003) 515–528.
[35] O. Nasraoui, M. Soliman, E. Saka, A. Badia, R. Germain, A web usage mining framework for mining evolving user profiles in dynamic web sites, IEEE Trans. Knowl. Data Eng. 20 (2) (2008) 202–215.
[36] F. Cao, J. Liang, L. Bai, X. Zhao, C. Dang, A framework for clustering categorical time-evolving data, IEEE Trans. Fuzzy Syst. 18 (5) (2010) 872–882.
[37] T.M. Cover, J.A. Thomas, Elements of Information Theory, Wiley-Interscience, 1991.

[38] M. Brand, An entropic estimator for structure discovery, in: Proceedings of the 1998 conference on Advances in neural information processing systems II, MIT, Cambridge, MA, USA, 1999, pp. 723–729.
[39] C.-H. Cheng, A.W. Fu, Y. Zhang, Entropy-based subspace clustering for mining numerical data, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '99, ACM, New York, NY, USA, 1999, pp. 84–93.
[40] P. Andritsos, P. Tsaparas, R.J. Miller, K.C. Sevcik, LIMBO: scalable clustering of categorical data, in: Advances in Database Technology – EDBT 2004, Lecture Notes in Computer Science, vol. 29, Springer Verlag, Berlin Heidelberg, 2004, pp. 531–532.
[41] D. Barbará, J. Couto, Y. Li, COOLCAT: an entropy-based algorithm for categorical clustering, in: Proceedings of the Eleventh International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2002, pp. 582–589.
[42] I.S. Dhillon, S. Mallela, D.S. Modha, Information-theoretic co-clustering, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, ACM, New York, NY, USA, 2003, pp. 89–98.
[43] T. Li, S. Ma, M. Ogihara, Entropy-based criterion in categorical clustering, in: Proceedings of the 21th International Conference on Machine Learning, ACM, New York, NY, USA, 2004, pp. 68–75.
[44] K. Chen, L. Liu, "Best K": critical clustering structures in categorical datasets, Knowl. Inform. Syst. 20 (1) (2009) 1–33.
[45] Z. Huang, Extensions to the k-means algorithm for clustering large data sets with categorical values, Data Mining Knowl. Discov. 2 (3) (1998) 283–304.
[46] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010. <http://archive.ics.uci.edu/ml>.
[47] M.K. Ng, M.J. Li, J.Z. Huang, Z. He, On the impact of dissimilarity measure in k-modes clustering algorithm, IEEE Trans. Pattern Anal. Mach. Intell. 29 (3) (2007) 503–507.
[48] F. Cao, J. Liang, A data labeling method for clustering categorical data, Expert Syst. Appl. 38 (3) (2011) 2381–2385.