Application of Data Stream Pattern Evolution Analysis in Energy Control System

Wenjuan Wang, Zhihui Ye, Chengting Zhang and Yong Li*
Ningbo Cigarette Factory, China Tobacco Zhejiang Industrial Co. LTD., Ningbo 315504, Zhejiang, China

*Corresponding author email: liy@zjtobacco.com

Abstract. Aiming at the data stream obtained in the energy control system of a cigarette factory, a data stream pattern analysis method is proposed which provides support for anomaly detection and other applications by detecting and tracking the pattern and extracting the evolution between the patterns. In this paper, the concept and definition of pattern tracking method for data streams are proposed, as well as the measurement criteria of pattern similarity. On this basis, the paper introduces in detail how to generate and cluster the hypercube grids, store the grid, generate the pattern and track the pattern on the real-time data stream. The paper also defines and describes the dynamic process of the generating, retreating, mutating, dividing and merging of the data stream pattern. The algorithm in this paper is applied to the real data stream collected in the energy control system of Ningbo cigarette factory, identifying and analyzing various feature of the data stream pattern, which can effectively describe the physical changes of the energy system.

Keywords: Data stream; Pattern evolution; Grid.

1. Introduction
The stability of energy system has a direct impact on the quality of the products of manufacturing enterprises. The energy costs account for an increasing proportion of the total cost. So reducing energy consumption is an important means to improve the economic benefits of enterprises. In tobacco industry, the State Tobacco Monopoly Administration has put forward higher and higher demand for energy conservation and emissions reduction. This requires that the tobacco industry enterprise to build an energy control system with advanced technology and complete function. By the system, the energy supply process can be monitored comprehensively, all kinds of energy can be scheduled efficiently, and so the energy consumption can be reduced gradually[1]. Ningbo Cigarette Factory has conducted technology reform and promotion when migrating into the new plant building. Among them, the energy monitoring system was also basically realized, and the preliminary analysis over the monitoring data was also carried out. But the data was very diverse, so the applications are fragmented and couldn’t be integrated, also it’s impossible to make intelligent decision. Nowadays thousands of sensors are installed in the factory, and a large amount of real-time data is generated every second to form data streams. The distribution of these data changes over time, which is quite different from the business data in the traditional database system with fixed relation and random access[2]. Therefore, special data processing methods are to be studied for their characteristics. If the thinking of macro energy and big data is integrated, the information system will be enhanced to support intelligent management and control of energy system[3].

Pattern-based data analysis model refers to extracting pattern sets from large-scale data sets, and then performing subsequent work such as classification, anomaly detection, and knowledge discovery[4]. The
pattern representation and similarity measurement are the core of the model[5]. In recent years, researchers have paid attention to the classification, clustering, and pattern mining in data streams[6, 7]. Because the real-time information in data streams are characterized by rapid changes and continuous growth, traditional pattern mining algorithms are difficult to be directly applied to the new big data environment. For the data in the data stream, there are three window selection models: boundary marker window, attenuation window and sliding window, which corresponds to different pattern mining algorithms. For example, Liu X et al. proposed a DSM-FI method based on the boundary marker window model, scanning the data internally, storing the patterns in a tree structure and mining them from top to bottom[8]. Cohen E et al. introduced a time decay function to calculate the number of itemset support, so that the data closer to the current time has a higher weight[9]. Tanbeer SK et al. proposed a frequent itemsets mining algorithm CPS based on sliding window, saving the transaction itemsets in the window to the CPS-Tree tree, and using the FP-Growth algorithm to mine the frequent itemsets in each window[10]. The patterns obtained by the above researches are mainly used for classification, and the evolution analysis of the dynamic patterns in the data stream is not involved and cannot be accurately located. In fact, the energy supply of a cigarette factory is a continuous process, but the process requirements, external environment, energy equipment, and personnel operations at different times will affect the energy supply and thus the product quality. Based on the large amount of real-time data obtained from the energy control system of the cigarette factory, this paper draws on the design ideas in the above research, analyzes and establishes the data stream patterns, and tracks its changes, providing support for product quality inspection, state switching, abnormal detection, etc. 

In the rest of the paper, in Section 2, concepts of data stream are presented first. Then some measurements on pattern similarity are defined. Section 3 articulates the procedure of generating and tracking the data stream pattern. In Section 4 we apply the method on the real data, analyze the output the algorithm, and demonstrate the performance affected by some predefined constants. We conclude with Section 5 by summarizing major findings.

2. Basic Concepts and Definitions

2.1. Definition of Data Stream Pattern

Suppose that $D = \{d_1, d_2, \ldots, d_n\}$ is a data stream, where the data item $d_i$ is a value in the q-dimensional space $S$, and the corresponding moments are $t_1, t_2, \ldots, t_n$. $S = A_1 \times A_2 \times \cdots \times A_q$, where $A_i$ is the value set of each dimension. The $i$-th dimension is equally divided into $\beta$ segments, so each segment can be expressed as $[l_i, h_i]$, where $l_i$ and $h_i$ are split points of each segment. Each segment is given a mark from 1 to $\beta$. Therefore the space $S$ could be transformed into a standardized hypercube $G$ in the q-dimensional space, which is divided into $\beta^q$ disjoint grids, and each grid $g$ can be expressed with a vector $(x_1, x_2, \ldots, x_q)$, where $x_i$ is the mark of the corresponding segment of the grid $g$ on the $i$-th dimension.

Streaming data is constantly being imported, while the data item $d_i$ will eventually fall into a certain grid after comparing each dimension coordinate with the split point. Hence the items in each grid will be accumulated. However, different data items have different weight for the current moment. Generally, the earlier data items become less significant for the current moment[11]. With regard to this, the data point influence factor will be defined first. That is, if a data point $d$ arrives at the moment $t_0$, its influence factor decreases with time according to the attenuation model. At any subsequent time $t$, the data point influence factor can be expressed as:

$$\text{Inf}(d, t) = \lambda^{t-t_0} \quad \text{(1)}$$

where $\lambda$ is a constant attenuation factor, $0 < \lambda < 1$.

Since the grid can be regarded as a collection of data points, the grid influence factor can be defined as:

**Definition 1**: Grid influence factor
At a given moment \( t \), for any grid \( g = (v_1, v_2, \cdots, v_q) \), \( d(g, t) \) represents a set of data points mapped to \( g \), whose influence factor \( \text{Inf}(g, t) \) is the sum of the influence factors mapped to all data points in grid \( g \). The influence factor of grid \( g \) at moment \( t \) is expressed as follows:

\[
\text{Inf}(g, t) = \sum_{d \in d(g, t)} \text{Inf}(d, t)
\]

(2)

where \( d \) is a data point mapped to \( g \).

Since the influence factor is a function of time, and new data point arrives at every moment, and then the influence factor of a certain grid is updated, the influence factor of the grid in the hypercube changes constantly too. If the segment coordinates of two grids have only one difference among all the dimensions and the difference is only 1 segment, the grids are considered as adjacent grid. 

**Definition 2:** Grid features

The grid \( g \) and the including data points can have many features at time \( t \). We selects four representative quantities:

1. \( f_1 = (x_1, x_2, \cdots, x_q) \) is a vector of segment marks of the grid;
2. \( f_2 = \text{Inf}(g, t) \), which is the network influence factor calculated in Definition 1;
3. \( f_3 = \text{dsq} \), which is a mode formed by adjacent grids of the network;
4. \( f_4 = t \) is the arrival time of the latest data item in the network.

**Definition 3:** Data stream pattern

In the \( q \)-dimensional space \( S \), the adjacent \( k \) grids can be aggregated to a grid cluster through clustering algorithm. Each grid cluster is called a data stream pattern (DSP), which is expressed as \( \{g_1, g_2, \cdots, g_k\} \).

In the pattern, the feature vector of the element is defined in Definition 2. At the moment \( t \), the feature of each pattern is uniquely identified by \( f_1 \) and \( f_2 \) of the grids. For the feature \( f_1 \) (segment vector) in the two grid clusters, denoted as \( p_1 (g_{11}, g_{12}, \cdots, g_{1n}) \) and \( p_2 (g_{21}, g_{22}, \cdots, g_{2m}) \), Boolean operations can be applied on them according to set theory, including:

1. \( \cup(p_1, p_2) = \{g | g \in p_1 \text{ or } g \in p_2\} \);
2. \( \cap(p_1, p_2) = \{g | g \in p_1 \text{ and } g \in p_2\} \);
3. if \( m \geq n \), \( p_1 \subset p_2 = \{g | \forall g \in p_1, g \in p_2\} \);
4. \( |p_1| = n, |p_2| = m \)

**2.2. The Measurement on Similarity between Data Stream Patterns**

The measurement on similarity between data streams is the basis of data stream mining. The data stream is usually represented as a vector, and Euclidean distance or non-Euclidean distance can be used to measure the similarity between two vectors. Measurement on similarity between data stream patterns is studied here.

Suppose that a data stream pattern \( \text{dsp}_1 \) at moment \( t_1 \) transformed a new pattern \( \text{dsp}_2 \) at moment \( t_2 \) after the new data points are distributed to corresponding grids. Examining these two patterns in the same data space, the similarities between the two patterns can be compared with their features. Generally, the two patterns may have intersection in the hypercube. The intersection can be called a sub-pattern relative to the original pattern. The similarity of two patterns can be expressed by the similarity of the sub-patterns. The variation of patterns from moment \( t_1 \) to \( t_2 \) can be shown in the variation of the influence factor value of the sub-patterns. However, the difference between the influence factors of different space vector sets cannot be quantified. Therefore, a measurement criterion is proposed that the modulus and the influence factor of the pattern be considered. Here three metrics will be defined first:

1. Overlap ratio: the ratio of the modulus of the intersection to the union of the patterns \( p_1 \) and \( p_2 \), namely:
\[ \text{Overlap}(p_1, p_2) = \frac{|\cap(p_1, p_2)|}{|\cup(p_1, p_2)|} \]  

(3)

It’s clear that when \( p_1 = p_2 \), the overlap ratio is the highest, which is 1. When \( p_1 \) doesn’t intersect with \( p_2 \), the overlap ratio is the lowest, which is 0. When the two patterns are more similar, the overlap rate will be higher.

(2) Similarity weight: Suppose that the pattern at \( t_1 \) is \( p_1(\{g_{11}, g_{12}, \cdots, g_{1m}\}) \), and the pattern at \( t_2 \) is \( p_2(\{g_{21}, g_{22}, \cdots, g_{2m}\}) \). A common sub-pattern \( p_1'(\{g_{11}', g_{12}', \cdots, g_{1m}'\}) \) and \( p_2'(\{g_{21}', g_{22}', \cdots, g_{2m}'\}) \) can be created, whose vector sets are the same. But the influence factors have changed over time. The similarity weight of the patterns \( p_1 \) and \( p_2 \) is defined as the ratio of the average influence factors of the common sub-patterns \( p_1' \) and \( p_2' \). In the q-dimensional space, the vector of grid \( g_q \) is \( \{v_1^q, v_2^q, \cdots, v_n^q\} \). The average influence factors of the common sub-pattern at \( t_1 \) and \( t_2 \) are:

\[ \text{AverInf}(p_1) = \frac{\sum_{k=1}^{n} \text{Inf}(g_{1k}^q, v_k)}{k}, \]

\[ \text{AverInf}(p_2) = \frac{\sum_{k=1}^{n} \text{Inf}(g_{2k}^q, v_k)}{k}. \]

So the similarity weight of data stream patterns can be defined:

\[ \text{Weight}(p_1, p_2) = \frac{\min \left\{ \text{AverInf}(p_1), \text{AverInf}(p_2) \right\}}{\max \left\{ \text{AverInf}(p_1), \text{AverInf}(p_2) \right\}} \]  

(4)

It’s clear that \( 0 \leq \text{Weight}(p_1, p_2) \leq 1 \). The larger the weight, the more similar the pattern is.

(3) Pattern offset: The spatial offset of the two patterns \( p_1 \) and \( p_2 \) can be measured with the Euclidean space distance of the centroids of sub-patterns \( p_1' \) and \( p_2' \). It is expressed as follows:

\[ \text{Dist}(p_1, p_2) = \sqrt{\sum_{k=1}^{n} (v_1^m - v_2^m)^2} \]  

(5)

Where \( v_1^m \) is the m-th coordinate of the centroid of sub-pattern \( p_1' \), \( v_2^m = \sum_{k=1}^{n} \text{Inf}(g_{2k}^q, v_k) \). The definition of \( v_2^m \) is similar to this.

Among the definitions, the pattern overlap ratio \( \text{Overlap}(p_1, p_2) \) can be considered as the spatial similarity of the patterns at different times. The pattern similarity weight and centroid Euclidean distance can be considered as changes in the influence factors of the patterns at different times. \( \text{Overlap}(p_1, p_2) \) and \( \text{Weight}(p_1, p_2) \) have a degree of proportional relationship with the pattern similarity. \( \text{Dist}(p_1, p_2) \) has a degree of inverse relationship with the pattern similarity. Therefore, the measure of pattern similarity is defined as:

\[ \text{Similarity}(p_1, p_2) = \frac{\text{Overlap}(p_1, p_2) \times \text{Weight}(p_1, p_2)}{\text{Dist}(p_1, p_2) + 1} \]  

(6)

3. Generating and Tracking Algorithm of Data Stream Patterns

![Figure 1](image-url)  

Figure 1. Procedure of the data stream pattern tracking algorithm

A real-time data stream dynamic pattern tracking algorithm is proposed to lay the foundation for subsequent pattern anomaly detection. The algorithm process is displayed in Figure 1. After the new data of the data stream is received, they are divided into some data grids and stored in a tree data structure. After that, a pattern clustering algorithm is executed over the grids and some patterns will be generated.
These data stream patterns are tracked to see what happened.

3.1. Grid Generation and Clustering
In the sliding window, when new data flows in, same amount of historical data is deprecated. So the newer data is always retained in the window. The influence factor of every point is calculated according to formula (1). The influence factor of old points will be gradually reduced due to the attenuation function, so that the new data retains a higher weight. The data points are also mapped to the grid of the hypercube \( G \). The influence factor of each grid can be calculated according to formula (2). The feature vector of each grid can be determined according to Definition 2.

3.2. Grid Storage
To track the change of the data stream pattern in the sliding time window, it is required to store the grid of each moment. Referring to the dimensional tree structure proposed in [12], a new tree structure is proposed based on the feature vector to support the subsequent pattern generating and tracking. Since each grid can be expressed as a q-dimensional vector \( V = (v_1, v_2, \cdots, v_q) \), it can be regarded as the coordinates of the grid in the hypercube \( G \). The dimensional tree structure is now used to store all existing grids. That is, each layer corresponds to a dimension, the i-th layer corresponds to the i-th dimension, and the q+1 layer stores the feature vector of the grid (see Definition 2). Therefore, the dimension tree has q+1 layers. When a data point arrives, it is mapped to the vector \( V = (v_1, v_2, \cdots, v_q) \), whose position in the tree can be searched in dimensional order until the leaf node \( L \) is found at the q+1 level. If the position is in the middle i-th layer but the i-th node can’t be found, a node \( v_i \) should be inserted into this layer, and all subsequent nodes should be inserted until the leaf node \( L \) is inserted in the q+1th layer. The feature vector is stored in the leaf node, namely the four-tuple \( (V, F(g, t), class, t) \), where the class is unknown now and is set to 0. After the leaf node \( L \) is found or inserted, the influence factor of each leaf node on the entire tree is updated according to the data point and the grid attenuation model. The influence factor of the non-leaf node is the sum of the influence factor of the children nodes.

3.3. Generating Data Stream Pattern
A clustering algorithm can be applied to aggregate adjacent grids into clusters, and each cluster constitutes a data stream pattern. FCM clustering algorithm is used for unsupervised grid clustering here. The main idea is as follows. Given a data set \( X = (x_1, x_2, \cdots, x_n) \), it’s divided into \( c \) fuzzy classes. The cluster center \( c_i \) of each fuzzy class is computed. The membership degree of each sample \( j \) belonging to a certain fuzzy class \( i \) is expressed as \( \mu_{ij} \). The cluster center and membership matrix can be solved with continuous iterating. The values in the main membership matrix are random numbers between [0,1], which may satisfy some conditions to minimize the objective function. An target function and its constraints can be defined as:

\[
J(U,V) = \sum_{i=1}^{c} \sum_{j=1}^{n} \mu_{ij}^m \|x_j - c_i\|^2
\]

(7)

\[
\sum_{i=1}^{c} \mu_{ij} = 1, (j = 1, 2, \cdots, n)
\]

(8)

In the above expression, \( U = [\mu_{ij}] \) is the membership matrix \( \mu_{ij} \in [0,1] \). \( C = [c_i] \) is the cluster center matrix, and \( c_i \) is the cluster center of the i-th category. \( m \) is the weighted index. \( \|x_j - c_i\|^2 \) represents the distance norm between the sample \( x_j \) and the cluster center \( c_i \). The membership \( \mu_{ij} \) and the cluster center \( c_i \) matrix can be solved with the Lagrangian function. First, formula (7) and formula (8) are combined to obtain formula (9), and then the variables \( \mu_{ij} \) and \( c_i \) are respectively derived to obtain the formulas (10) and (11).
\[
J(U,V) = \sum_{i=1}^{c} \sum_{j=1}^{n} \mu_{ij} \|x_j - c_i\|^2 + \lambda_1 \left( \sum_{i=1}^{c} \mu_{i4} - 1 \right) + \cdots + \lambda_n \left( \sum_{i=1}^{c} \mu_{in} - 1 \right)
\]

\[
\mu_{ij} = \left( \sum_{k=1}^{c} \left( \frac{\|x_j - c_k\|}{\|x_j - c_i\|} \right)^{\frac{2}{m-1}} \right)^{-1}
\]

\[
c_i = \frac{\sum_{j=1}^{n} (\mu_{ij} x_j)}{\sum_{j=1}^{n} (\mu_{ij}^n)}
\]

The fundamental procedure of clustering algorithm is as follows:
1) Specify the number of cluster centers \( c \) and the fuzzy weighting index \( m \). Set the iteration stop threshold \( \varepsilon \) and the maximum iteration times \( b_{\text{max}} \). Set the initial value \( b=0 \). Initialize the membership matrix \( U^0 \);
2) Calculate the fuzzy cluster center matrix \( C^b \) according to formula (8);
3) Update the fuzzy cluster membership degree \( U^{b+1} \) according to formula (7);
4) Select the optimal matrix norm. Compare \( U^b \) and \( U^{b+1} \). Stop iteration when \( \|U^{b+1} - U^b\| \leq \varepsilon \) and output \( U \) and \( V \). Otherwise continue the procedure, set \( b=b+1 \), go to step 2).

### 3.4. Tracking Data Stream Pattern

An algorithm is proposed to discover and track patterns in real-time data streams and pattern changes (movement, split, appearance and disappearance). Based on the patterns generating in preceding steps, a method of grid pattern matching and comparing analysis is created to track the patterns. The pattern matching method makes full use of the feature of the grid dimensions. Suppose that \( S_1(p_{11}, p_{12}, \cdots, p_{1n}) \) and \( S_2(p_{21}, p_{22}, \cdots, p_{2m}) \) are two pattern sets at moment \( t_1 \) and \( t_2 \) respectively. The array \( \text{Fade[]} \) is used to classify the pattern \( p_{ij} (i=1,2,\cdots,n) \) whether it disappears at \( t_2 \). The array \( \text{Sim[]} \) stores the similarity of a pattern \( p_{ij} (i=1,2,\cdots,m) \) in the set \( S_j \).

The following pseudo code shows the main process of pattern tracking. The qualitative and quantitative definitions in the previous section is utilized here. The current pattern feature tree and the snapshot of the pattern feature tree at the previous moment is compared and analyzed. The morphological changes such as generating, retreating, mutating, dividing, and merging actions are recorded.

#### Algorithm 1: Pattern Tracking

Start: \( i=1, \text{Fade}[0..n]=0 \)

while \( i < n \) do

\( k = 0, j = 1, \text{Sim}[0..n]=0 \)

while \( j < m \) do:

create the intersection and the union of \( p_{ki} \) and \( p_{kj} \) to calculate the pattern overlap ratio

if overlap > 0 then

\( \text{Fade}[j]=1 \)

else

calculate the similarity between \( p_{ki} \) and \( p_{kj} \), set \( \text{Sim}[j]=s \)

\( j++ \)

end if

end while

find \( \text{smax} = \max(\text{Sim}[0..n]) \), where \( \text{smax} \) is an index

if \( \text{smax} = 0 \) then

\( p_{kj} \) is a new pattern

else

record \( p_{ki} \) evolve to \( p_{kj} \)

end if
end if
find the subsequent pattern set of \( p_i \)
if the set is empty then
record pattern \( i \) disappear
end if
\( i++ \)
end while

In this process, the pattern disappearing can be located by examining the changes in the overlap ratio of the patterns. The pattern dividing can be located by examining the changes in the similarity of the patterns. The instructions is as follows:

1. The overlap ratio of patterns \( p_i \) and \( p_j \) can be calculated by applying formula (3). If the overlap ratio is greater than 0, it is implied that the shape of the pattern \( p_i \) at the previous moment has changed (larger or smaller, divided or merged). But a qualitative change doesn’t happen, that is, it has not completely disappeared or moved elsewhere. If \( p_i \) does not disappear at moment \( t_2 \), but only partially changes, \( \text{Fade}[j] \) will be set to 1. If the overlap ratio is equal to 0, it is implied that the two patterns have no intersection, so the original value of \( \text{Fade}[j] \) will be maintained the same. In the pattern tracking process, \( p_i \) needs to be compare to each pattern in the pattern set \( S_1 \), and calculate the items in the \( \text{Fade} \) array. As long as one item of the \( \text{Fade} \) array element obtained at \( t_2 \) is 1, the dynamic change of the pattern between moments is recorded. If all the elements of the \( \text{Fade} \) array are 0, it is implied that all the patterns in the pattern set \( S_1 \) have no relation with \( p_i \). So far, it is definitive that pattern \( p_i \) disappears at \( t_2 \).

2. The similarity of any two patterns can be determined by formula (6) and stored in the array \( \text{Sim} \). There are merging and recombining between data stream patterns. If the occurrence of this is wanted, each pattern in the pattern set \( S_1 \) need to be compared with each pattern in the subsequent pattern set \( S_2 \) one by one. Hence the variation relationship can be established, while each time the relationship between one source pattern and several subsequent patterns is recorded in the array \( \text{Sim} \). Comparing the array \( \text{Sim} \) of multiple source patterns, pattern dividing can also be determined.

From the above analysis, it is clear that the amount in two pattern sets and the structure of the pattern feature tree will have a greater impact on the efficiency of the algorithm. From the most pessimistic point, if \( q \) is the spatial dimension of the space \( S \), and \( \beta \) is the amount of segments in each dimension of the hypercube, the time complexity of a pattern matching is \( O(\beta^{q+1}) \) and the time complexity of all pattern matching is \( O(mn\beta^{q+1}) \). So the amount of pattern sets is an important variable. If the data stream has a many dimension, the parameter \( \beta \) can be tuned to a small value to reduce the time complexity effectively. Hence the processing time spent on discovering and tracking can be reduced. It means that the algorithm is more efficient and suitable for more applications.

4. Experiment Analysis
In order to verify the effectiveness of the algorithm, some experiments are conducted on the real data collected by the silk A line in the energy management and control system of Ningbo Cigarette Factory. The sensor data generated for 10 consecutive days in a certain process section is collected. There are 5 main attributes, including time, fan current, fan operating frequency, negative pressure value, pressure difference, and negative pressure deviation. The experiment programs are built by Pandas and Matplotlib, using the method proposed in Section 3. The data stream generating and tracking algorithm proposed in this paper are executed to discover the potential rules in the real-time data stream, to track the evolution of the dynamic patterns, and to report abnormal changes in the patterns. Because the data involves 4 dimensions, the pattern tracking of all attributes cannot be directly displayed on the graph. Here, three attributes of negative pressure value, pressure difference and current are chosen for analysis and display. The algorithm parameters are set as: the attenuation factor constant \( \lambda = 0.99 \), and the data
space dimension interval division number $\beta=100$. Applying the method in this article, the grids are created and the data stream pattern are stored and generated. The following are the patterns at two moments:

**Figure 2.** Data stream pattern at two moments

**Figure 3.** Changes in the pattern similarity measurement

It shows at moment $t_1$ that the pressure difference value fluctuates around 670Pa, the fluctuation range is not large, the negative pressure is distributed in the interval of [-31,-30.5] (unit of mbar), and the current distribution is wide, but most of them are in [39, 41] (unit of A). At moment $t_2$, most of the pressure difference values are still around 670Pa, but some values jump to the interval [700,860], the negative pressure distribution is more dispersed, and the current distribution is more concentrated. Based on these two modules, the pattern features values at different moments can be calculated:

**Table 1.** Pattern feature values at different moments

|      | Overlap ratio (%) | Similarity (%) |
|------|------------------|---------------|
| $t_1$ | 93.5             | 94.4          |
| $t_2$ | 72.7             | 79.1          |

It shows that the patterns at the moment $t_1$ is relatively stable, because the pattern overlap ratio and the similarity are both high. While at moment $t_2$, the patterns has changed greatly, which is worthy of attention. And if the two measurement values in a period of time are continuously calculated, two curves can be obtained:

The trend of these two curves indirectly reflects the trend of each feature of the data stream. It shows that at moment $t=2\sim6$, the pattern overlap ratio and the pattern similarity change by a large margin, and the overlap ratio is higher than the similarity. It indicates that the shape of the grids of the patterns have a greater change than the changes of locations, implying possible abnormal changes in energy supply. After a period of time, the two measurement values tended to be consistent and remained above 90%, indicating that the energy system parameters remained stable. Some of these changes can be detected by comparing feature value thresholds, but the magnitude and form of changes cannot be described only by feature thresholds. The method in this paper can detect the changes of the data stream patterns over time in the hypercube space by calculating the pattern overlap ratio and pattern similarity. It can provide more information related to the physical state change.

**Figure 4.** Relationship between algorithm performance and $\beta$ value

**Figure 5.** Relationship between pattern similarity and window length

The performance of the method in this paper and the accuracy of mining are highly dependent on the window length, the number of dimensional interval division $\beta$, the number of features. Figure 4 shows
the relationship between the execution time of the algorithm and the value of $\beta$. If $\beta$ increases a few, the number of grids increases exponentially, the computational burden will increase accordingly, which will cause a greater burden on memory and CPU. Figure 5 shows the relationship between the similarity of the data stream pattern and the length of the window. If the length of the window increases (indicated by the number of points in the window), the difference in the number of data points in the hypercube will be more obvious, and the data stream pattern will change more significantly. But if the window is too long, minor changes in the data pattern will also be recognized, resulting in some subtle alarms. Therefore, the data stream on different sections should be tuned to choose the optimal parameters.

5. Summary
The requirements on data stream processing of energy control system is introduced. An evolution analysis method based on grid division and data stream pattern is proposed. Through dividing the data space, multi-dimensional grids are created and stored. Based on these grids, the procedure of grid clustering, data stream patterns generating and pattern tracking is applied according to measurement values such as pattern overlap and similarity. The method provides support for subsequent applications such as pattern tracking and anomaly detecting based on data stream patterns. Experiments based on real data sets show that the algorithm can effectively create dynamic patterns in data streams, and can track the evolution activity of data stream patterns. For future directions, some more measurements on Similarity between Data Stream Patterns can be discovered so that the features of a pattern can be more clearly distinguished.

Acknowledgement
This research was financially supported by the Key Research & Development Program of Zhejiang Province (Project Code: 2021C01110).

References
[1] C. Zhang. Research and application of energy saving technology for power and energy system, Xihua University, 2017.
[2] A. Bifet, Adaptive stream mining: pattern learning and mining from evolving data stream, Proceedings of the 2010 Conference on Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams, Amsterdam, Netherlands, 2010: 1-21.
[3] Y. Xue, Y. Lai, Integration of macro energy thinking and big data thinking, part two applications and explorations, Automation of Electric Power Systems, 2016, 40(8):1-13.
[4] A. Ragab, M. El-Koujok, B. Poulin, ect., Fault diagnosis in industrial chemical processes using interpretable patterns based on Logical Analysis of Data, Expert Systems With Applications, 2018, 95:368–383.
[5] M. Han, J. Ding, Survey of frequent pattern mining over data streams, Journal of Computer Applications, 2019,39(3): 719-727.
[6] C. Park, Outlier and anomaly pattern detection on data streams, Journal of Supercomputing, 2019, 75(9):6118-6128.
[7] R. Domingues, M. Filippone, P. Michiardi, etc., A comparative evaluation of outlier detection algorithms: Experiments and analyses Pattern Recognition, 2018, 74:406-421.
[8] X. Liu, J. Guan, P. Hu, Mining frequent closed itemsets from a landmark window over online data streams, Computers and Mathematics with Applications, 2009, 57(6):927-936.
[9] Y. Baek, U. Yun, H. Kim, etc., Erasable pattern mining based on tree structures with damped window over data streams, Engineering Applications of Artificial Intelligence, 2020, 94:103735.
[10] S.K. Tanbeer, C.F. Ahmmed, B.S. Jeong, et al., Sliding window-based frequent pattern mining over data streams, Information Seciences, 2009, 179(22):3843-3865.
[11] X.L Zhang, W. Zeng, Real-time data stream clustering based on damped window and pruning dimension tree, Journal of Computer Applications, 2009,29(5): 1331-1334+1341.
[12] K. Aditya, H. Kolla, W. Kegelmeyer, et al., Anomaly detection in scientific data using joint statistical moments, Journal of Computational Physics, 2019, 387: 522–538.