Soft Comput (2017) 21:5905–5917
DOI 10.1007/s00500-016-2258-z

FOCUS

Streaming data anomaly detection method based on hyper-grid structure and online ensemble learning

Zhiguo Ding[1,2] · **Minrui Fei**[1] · **Dajun Du**[1] · **Fan Yang**[2]

Published online: 8 July 2016

Abstract This paper proposes a novel online streaming data anomaly detection method. By using the new method, the improved L_1 detection neighbor region optimizes the initial hyper-grid-based anomaly detection method by decreasing the quantity of neighbor detection region, and online ensemble learning adapts to the distribution evolving characteristic of streaming data and overcomes the difficulty of obtaining the optimal hyper-grid structure. To validate the proposed method, the paper uses a real-world dataset and two simulated datasets and finds out that the experimental results are near to the optimal results.

Keywords Hyper-grid structure · Online ensemble learning · Anomaly detection · Streaming data

1 Introduction

With the rapid development of big data applications, data are accumulating in an alarming rate from all walks of life (Subramaniam et al. 2006; Esmaeili and Almadan 2011; Gaber et al. 2005). For example, Wal-Mart supermarket produces about 20 million transaction records each day, Google servers processes about 4 million searching requests each hour and AT&T produces averagely 3 thousands call records each second. For us, most of the data are normal and has little value; however, there exists a very little proportion of data, about from 0.01 to 5 %, which makes a big difference to the decision making. For example, an accidental anomalous transaction record of credit card in financial field could signify fraudulent use of credit cards, an anomalous traffic pattern in computer network security field could indicate a unauthorized access (Gomez et al. 2013), and an anomalous fluctuation of detection signal in industrial control field could stand for a fault component (Limthong et al. 2014; O'Reilly et al. 2014; Salem et al. 2014; Serdio et al. 2014; Gil et al. 2014; Di Martino et al. 2014). These applications all force the anomaly detection method to detect these anomalous observations timely and quickly and to take the efficient measures to warn, track or solve these corresponding problems.

In fact, many anomaly detection methods have been proposed in the literatures based on the statistics, data mining and machining learning methods (O'Reilly et al. 2014; Gupta et al. 2014), which usually builds a profile based on the historical normal dataset. For a new coming observation, it is normal or not is identified by whether it conforms to the normal profile or not. But these methods were usually regarded as a "side effect" and designed for other purpose (classification or clustering) other than anomaly detection (Liu et al. 2012; Yang et al. 2015; Huang et al. 2015). Besides, most of them were proposed for static dataset, which is unfit for the online processing of streaming data. As we all known, the typical characteristics of streaming data, such as continuous generation, massive data, unknown or dynamic changing of data distribution, great un-balance distribution between nor-

Communicated by Y. Jin.

✉ Zhiguo Ding
dzg_jsj@zjnu.cn

Minrui Fei
mrfei@staff.shu.edu.cn

Dajun Du
ddj@shu.edu.cn

Fan Yang
fyang@zjnu.cn

[1] Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronics Engineering and Automation, Shanghai University, Shanghai 200072, China

[2] College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua 321004, Zhejiang, China

mal and anomalous observations, make the static, offline or "multiple-pass" methods un-appropriate for the streaming data situation.

In this paper, a novel online streaming data anomaly detection method is proposed based on improved hyper-grid structure and online ensemble learning. To mitigate the complex searching space of hyper-grid-based anomaly detection method, the L_1 detection neighbor region of initial hyper-grid structure is improved. Besides, considering that the optimal hyper-cube of hyper-grid structure is hardly obtained under the context of streaming data, the online ensemble learning theory is employed, and it has two advantages. The first one is that it can adapt to the distribution evolving characteristic of streaming data, and the other is that obtaining multiple hyper-grid structure based weak individual detectors are more easily than learning an optimal strong detector.

The remainder of this paper is organized as follows: in Sect. 2, some related works about anomaly detection are introduced; in Sect. 3, the online anomaly detection method based on hyper-grid structure and online ensemble learning is presented; in Sect. 4, experiments and result analysis are detailed; at last, conclusions and the look for future work are presented.

2 Related works

Up to now, many researchers have focused on their attentions on the anomaly detection research in the community of statistics, data mining and machine learning. For the static dataset, many anomaly detection methods have been proposed from the different perspective (Lee et al. 2013; Suhailis et al. 2014). For example, two categories of methods are proposed from the perspective of data set characteristics. The one category takes into account the unbalanced characteristics of data distribution and only uses the normal dataset to build the detector model (Segui et al. 2013); the other uses the over-sampling or under-sampling method (He et al. 2008) to balance the dataset and the classification methods to build the detector model (Desir et al. 2013), which regard the anomaly data detection as a binary classification problem. However, it should be noted that the former might lose some useful information, and the latter might decrease the generalization performance. From the perspective of technique adopted, the methods can roughly categorized as distance-based method (Angiulli and Fassetti 2009; Moshtaghi et al. 2011), density-based method (Breunig et al. 2000; Huang et al. 2014), model-based method (Liu et al. 2012) and so on. Distance-based method usually takes the distance, such as Euclidean distance or Mahalanobis distance, as a basic metric to carry out the anomaly detection. The distance is firstly calculated between the object sample and the center of the detection model. If the distance is no more than the pre-defined threshold, it is regarded as normal; otherwise, it is anomalous. Density-based method is an extension of the distance-based method. If the density of region where the object sample is located is less than the pre-defined threshold, it is judged as anomalous; otherwise, it is normal. Usually, density-based methods can obtain the relatively higher detection accuracy than other methods for the local anomaly. Model-based method usually learns a detector model based on the historical dataset, some statistics theory or machine learning methods, such as Gaussian distribution, artificial neural network, support vector machine, were used to build the detection model (Schölkopf 2001). If the object sample conforms to the generation mechanism and fits the obtained model in the process of anomaly detection, it is regarded as normal; otherwise, it is anomalous. From the perspective of whether or not to use the sample label during the procedure of detector building, the methods can be categorized as supervised learning method, un-supervised method and semi-supervised method (Daneshpazhouh and Sami 2014; Yamanishi et al. 2004; Noto et al. 2012; Zhang et al. 2010). In a word, the taxonomy of above mentioned methods has some overlap to some extent and an obvious conclusion can be drawn that machine learning-based method is a dominated method. Recently, from the perspective of essential characteristic of anomaly, i.e., few and different, an isolation-based method is proposed (Liu et al. 2012). Besides, based on the quantity estimation in the detection region, mass-based method is proposed (Ting et al. 2013). Recently, some research results have been obtained based on these two methods (Yu et al. 2009a; Tan et al. 2011).

With the arrival of big data era, an increasing number of data are accumulating rapidly, which makes the anomaly detection methods designed for static dataset improper for the online data processing. Storing massive data in the main memory are impossible and unnecessary, which makes these methods scanning the training dataset many times to learn the detector which is unfit for the one-pass characteristics of online data processing. Recently, researchers at home and abroad have proposed some new methods for online anomaly detection (Quinn and Sugiyama 2014). These methods generally can be categorized as into two directions: one is the modification of the traditional method, i.e., training the initial detector based on all historical dataset and updating it based on the new coming data chunk. However, there exists an obvious disadvantage; that is, the poor adaptability and scalability do not well adapt to the dynamic distribution of online streaming data. Instead, only when the change of data distribution is slow and small can this kind of strategy obtain an acceptable detection performance. The other is to build detector based on incremental learning (He et al. 2011) and online ensemble learning theory (Dietterich 1997; Kolter and Maloof 2007; Breiman 1996, 2001; Bifet et al. 2009a; Ando et al. 2015), which trains multiple individual detector from the different facets of streaming data. Theories and experi-

ment have proved that online ensemble learning can deal with the data dynamic characteristics and concept drift phenomenon (Minku and Yao 2012) of streaming data to some extent, and nowadays, two ensemble models are used widely, i.e., the horizontal ensemble and vertical ensemble. The former uses the latest multiple continuous data chunks to train multiple single detectors. It has the most advantage that it can master the time-dependence relationship of streaming data and can process the noise data items, and the most obvious disadvantage that some individual detector may be too old to adapt to the current data distribution. The latter only uses the latest one chunk to train multiple single detectors by using different learning algorithm, but this strategy requires that the current data chunk should be absolutely accurate; otherwise, the trained detector will have a poor performance. In general, due to the dynamic characteristics of streaming data, the online ensemble learning method becomes a hot direction of online anomaly detection, online bagging, online boosting methods (Oza 2005; Qi et al. 2011; Fern and Givan 2003) providing the solid theory foundation (Bifet et al. 2009b; Chang and Cho 2010) for these methods. There are many researchers proposed some online methods to deal with the classification problem of streaming data (Yamanishi et al. 2004; Palshikar 2005; Yu et al. 2009b; Zhou et al. 2013; Sagha et al. 2013; Ding et al. 2015; Ding and Fei 2013); some of them can be used to detect the anomalous data in streaming data. For example, the experimental results of He et al. (2011) demonstrated that proposed method could master the data distribution in real time and have a good detection performance. Tan et al. (2011) proposed a half-space tree algorithm for streaming data based on the isolation principle and ensemble learning, which dedicated to the online streaming data processing. The most obvious advantage of this algorithm is that the real dataset is not used during the process of building the initial detector and can build the detector fast. But its disadvantage is obvious, for it updates the detector continuously whenever this process is required or not, which degrades the real-time nature. In this paper, a hyper-grid-based ensemble online anomaly detection method is proposed based on the hyper-grid structure and online ensemble learning theory. In the next section, it will be represented in details.

3 Hyper-grid structure-based ensemble anomaly detection method

3.1 Hyper-grid structure-based anomaly detection method

Hyper-grid-based anomaly detection method was initially designed for the offline anomaly detection (Xie et al. 2012), which of course was inappropriate for the online data processing. Here, the hyper-grid structure is firstly introduced.

For the sample x_i ($x_i \in X$,$x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,q})$, $i = \{1 \ldots m\}$), x stands for a q-dimensional vector and X denotes dataset, the hyper-grid structure G, built by X, is represented as follows:

$$G = \left\{ C_{u1,\ldots uq} \mid C_{u1,\ldots uq} \in G \right\}$$

$C_{u1,\ldots uq}$ denotes a hyper-cube, and its side length and diagonal length are h and d, respectively. Here, $h = d/(2\sqrt{q})$ and the hyper-grid structure G are composed of continuous hyper-cubes. For $x \in X$, it can be mapped into a hyper-cube in the hyper-grid structure.

After a hyper-grid structure is built based on training dataset X, the new coming observation, whether it is anomalous or not, is decided by the region where the observation is mapped into. If the mapping position of new observation is located in a sparse region, which means no or few training data are mapped into that hyper-cube, the new observation is detected as anomaly; otherwise, it is normal.

Unfortunately, only using the exact mapped region usually does not accurately judge whether the observation is normal or not, especially these observations are mapped onto the boundary of hyper-cube. Therefore, for the hyper-cube $C_{u1,\ldots uq}$, its 1st neighbor detection region is further introduced and denoted by layer-1 (L_1), which is defined as follows.

$$L_1(C_{u1,\ldots uq}) = \{C_{v1,\ldots vq} \mid v = u \pm 1,\ C_{v1,\ldots vq} \neq C_{u1,\ldots uq}\} \tag{1}$$

To clearly demonstrate the above presented definition, a hyper-grid structure built by a 2-dimension ($2D$) dataset is used as an example, and the layer-1 (L_1) neighbor detection region of mapped hyper-grid cube is depicted in Fig. 1

The black spot in Fig. 1 denotes the mapped observation point located in the different position of corresponding hyper-cube. Figure 1b denotes an observation mapped into one hyper-cube, and the number of its L_1 neighbor detection regions is 8. When an observation is mapped onto the boundary of hyper-cube, the number of L_1 neighbor detection region is relatively small, such as depicted by Fig. 1c, d. Here, under the context of hyper-grid structure, an anomaly is redefined as follows

Definition 1 For an observation x, if its mapping region is sparse, it may be an anomalous sample; otherwise, it is normal (Xie et al. 2012; Knorr et al. 2000).

The above definition is qualitative and is hard to be applied into practical anomaly detecting. Therefore, the threshold k is introduced, and three heuristic rules are concluded to facilitate the anomaly detecting.

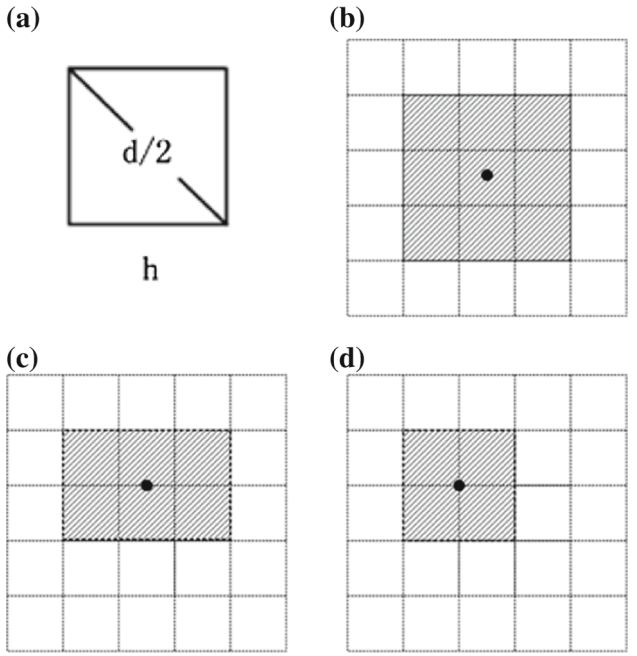

Fig. 1 Layer-1 (L_1) neighbor detection region of hyper-grid structure for a $2D$ observation (*black dot*). **a** One grid of $2D$ hyper-grid structure, **b** L_1 neighbor detection region of one observation mapped in the hyper-grid, **c** L_1 neighbor detection region of one observation mapped on the edge of hyper-grid, **d** L_1 neighbor detection region of one observation mapped on the crossing point of hyper-grid

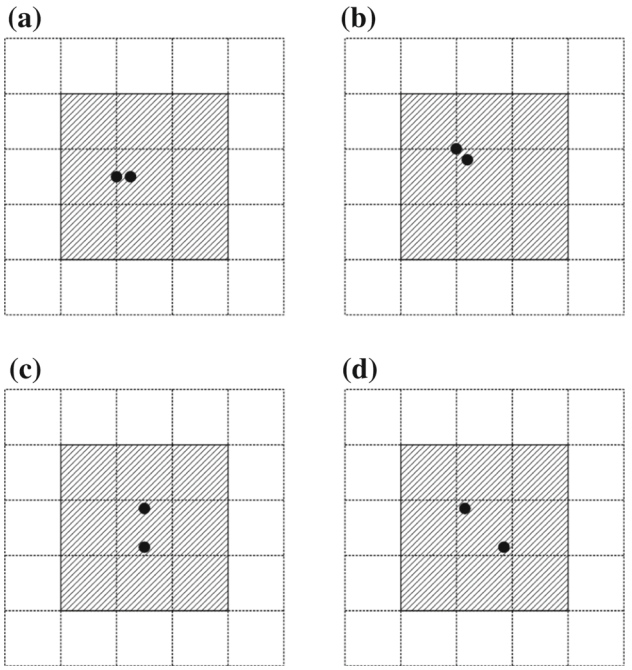

Fig. 2 Layer-1 (L_1) neighbor detection region for two $2D$ observations (*black dots*). **a** L_1 neighbor detection region of two observations (one is close to left edge, the other is mapped on the edge), **b** L_1 neighbor detection region of two observations (one is close to upper-left edge, the other is mapped on the crossing point), **c** L_1 neighbor detection region of two observations (one is close to upper edge, the other is close to lower edge), **d** $L1$ neighbor detection region of two observations (one is close to upper-left edge, the other is close to lower-right edge)

Rule 1 If more than k observations are mapped into the hyper-cube $C_{u1,\ldots uq}$, the new observation mapped into $C_{u1,\ldots uq}$ usually is normal.

Rule 2 If more than k observations are mapped into the hyper-cube $C_{u1,\ldots uq}$ and its layer-1 (L_1) region, i.e., $C_{u1,\ldots uq} \cup L_1(C_{u1,\ldots uq})$, the new observation mapped into $C_{u1,\ldots uq}$ usually is normal.

Rule 3 If less than k observations are mapped into the hyper-cube $C_{u1,\ldots uq}$ and its layer-1 (L_1) region, i.e., $C_{u1,\ldots uq} \cup L_1(C_{u1,\ldots uq})$, the new observation mapped into $C_{u1,\ldots uq}$ usually is anomalous.

Difference from the Knorr et al. (2000), where the initial hyper-grid-based anomaly detection method was proposed to employ the $L_1(C_{u1,\ldots uq})$ and $L_2(C_{u1,\ldots uq})$. The proposed method employs only L_1 neighbor detection region, with the consideration that L_2 neighbor searching space is considerable and introduces the ensemble learning (Sect. 3.3) to build multiple hyper-grid structure rather than an optimized one.

3.2 The improved L_1 neighbor region of hyper-cube

Figure 1 illustrates an obvious fact that the number of neighbor detection region is closely related to the mapped position of observations. In fact, it is hard to judge the exact detection region for any given sample according the aforementioned definition, especially for these samples, which are mapped onto the boundary or near to the boundary, just as is described in Fig. 2.

From Fig. 2a, b, an obvious fact is that even though two observations are very close to each other, the neighbor detection region is significantly different. Besides, even though two observations are mapped into the same hyper-cube, such as Fig. 2c, d, the defined detection region may not represent the true condition because it is hard to obtain the accurate grid structure. Consequently, the traditional definition will lead to some poor detection performances. To solve this difficulty and further reduce the searching space, the improved L_1 neighbor detection region is re-defined.

For sample x, its neighbor space usually refer to the adjacent space from the perspective of geometry. Therefore, for hyper-grid structure, the mapped position of sample is taken into account to judge its L_1 neighbor detection region. Compared to the traditional method, this improvement can decrease the number of searching adjacent hyper-cube to some extent. The improved L_1 neighbor detection region $\bar{J}(x)$ can be defined as follows:

$$\overline{J}(x) = \left\{ C_{v1,\ldots vq} \mid v_i = u_i,\ u_i + e_i,\ \ i = 1, \ldots q \right\} \tag{2}$$

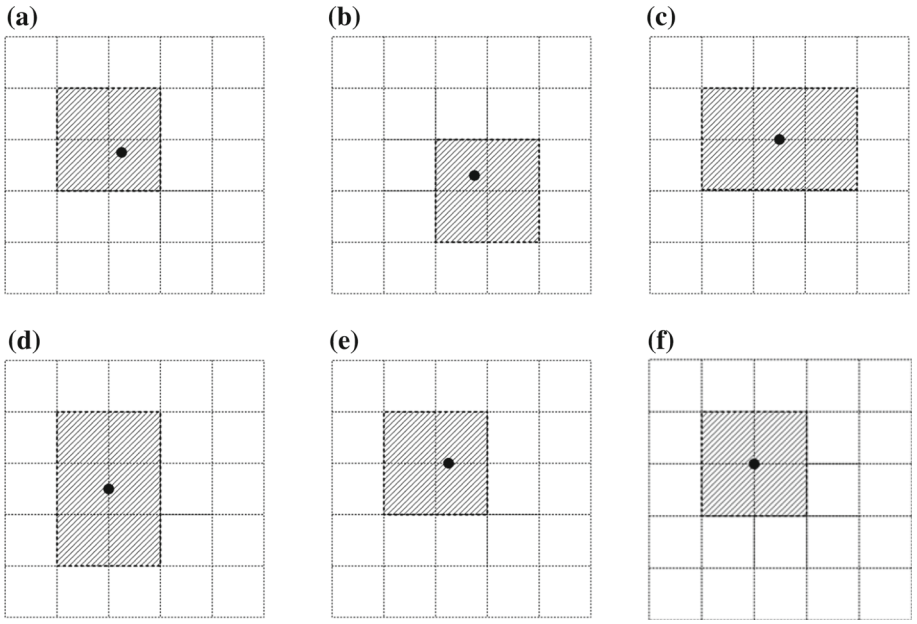

Fig. 3 Proposed layer-1 (L_1) neighbor detection region of hyper-grid structure for a $2D$ observation (*black dot*). **a** Improved L_1 neighbor detection region(observation is close to upper-left edge), **b** improved L_1 neighbor detection region (observation is close to lower-right edge), **c** improved L_1 neighbor detection region (observation is located on the edge), **d** improved L_1 neighbor detection region (observation is located on the edge), **e** improved L_1 neighbor detection region (observation is located on the edge), **f** improved $L1$ neighbor detection region (observation is located on the crossing point)

$$e_i = \begin{cases} +1 & u_i^* - u_i > 0.5 \\ -1 & u_i^* - u_i < 0.5 \\ \{-1,\ +1\} & u_i^* - u_i = 0.5 \end{cases} \tag{3}$$

$$u_i^* = \frac{x_i}{h}, \quad i = 1, \ldots q \tag{4}$$

$$u_i = \lfloor u_i^* \rfloor, \quad i = 1, \ldots q \tag{5}$$

Based on the above improved definition of L_1 neighbor detection region, Fig. 3 partly presents the proposed layer-1 (L_1) neighbor detection region of the hyper-grid structure for a 2D observation.

From Figs. 1 and 3, for observations which are not mapped to the boundary observations which are not mapped in the center, the number of neighbor hyper-cube decreases from 9 to 4 in the $2D$ hyper-grid structure. In fact, most observations can decrease the number of searching region. Because the n-dimension hyper-grid structure and the mapped position of observation are not on the boundary or in the center of hyper-grid cube, the number of neighbor detection region decreases from 2^n to n, which can significantly save the computation cost; and because these observations are mapped on the boundary, the number of searching neighbor detection region is decreased to different degree.

3.3 Hyper-grid-based online anomaly detection algorithm

The initial hyper-grid structure-based anomaly detection method is designed and applied to static dataset, but it could not be directly used for anomaly detection of streaming data. Because the boundary of hyper-grid structure built by a static dataset is fixed, a sample is doubtlessly judged as an anomaly when its mapping position exceeds the hyper-grid boundary. What is more, the concept drift may be occurred under the streaming data environment, which makes the previous hyper-grid structure unfit for the current data distribution for the evolution characteristic of streaming data. That is to say, a sample may be a normal observation even though it exceeds the hyper-grid boundary. Consequently, the hyper-grid structure should be updated corresponding to the distribution change in streaming data. Figure 4 demonstrates the flow chart of our proposed streaming data anomaly detection method, which consists of three critical stages, i.e., detector building, anomaly detecting and detector updating.

The specific procedures are described as follows.

(1) Building of anomaly detection based on hyper-grid structure

The procedure of learning the hyper-grid-based anomaly detector is an activity of mapping the observations into the hyper-cube in hyper-grid space. This process is simple, and only a vector of $q+1$ dimension, i.e., $(u_1, u_2, \ldots, u_q, cnt)$, is required for each observation, the last item cnt is used for storing the number of mapping observations into one hyper-cube $C_{u1,\ldots uq}$, and the front q items of vector, i.e., $(u_1, u_2, \ldots, u_q)$ is used to storing the position index where the observation is mapped to. The number of mapped observations to $C_{u1,\ldots uq}$ can be defined as $S(C_{u1,\ldots uq}) = cnt$. The initial value of cnt is 0. In fact, if the side length of hyper-cube is 1, then hyper-grid-based anomaly detection method can be categorized as a density-based method to some extent. Consequently, the procedure of training the hyper-grid-based anomaly detector is just the process of mapping the observations to one hyper-cube and counting its number. The pseudo-code of training the hyper-grid-based anomaly detector can be described by Algorithm 1.

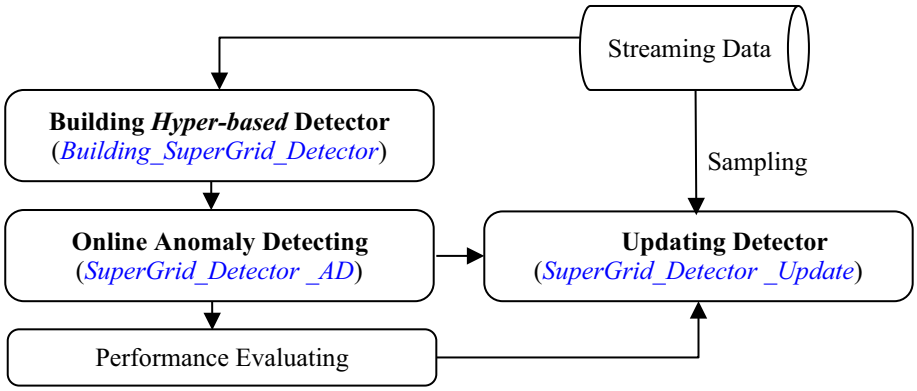

Fig. 4 A flow chart of the streaming data anomaly detection method based on hyper-grid structure

Algorithm 1 *Building_SuperGrid_Detector(h, X,)*

Input: $X \leftarrow \{x_1, x_2, ..., x_m\}$ ($x_i \leftarrow \{x_{i,1}, x_{i,2}, ..., x_{i,q} \mid i = 1, ...m\}$)

Output: $S(C_{u1,...uq}) \leftarrow cnt$

1: $S(C_{u1,...uq}) \leftarrow 0$;

2: For (i ← 1,…m)

3: For(j ← 1,…q) $u_j \leftarrow \left\lfloor \frac{x_{i,j}}{h} \right\rfloor$;

4: $cnt \leftarrow$ ***Find***($S(C_{u1,...uq})$, u);

5: *If (cnt* equal to *0)* $S(C_{u1,...uq}) \leftarrow 1$;

6: else $S(C_{u1,...uq}) \leftarrow S(C_{u1,...uq}) + 1$;

The *Find*() function in algorithm 1 is used to obtain the number of mapped observations in hyper-cube $C_{u1,...uq}$, if the complete hyper-grid space is stored, the random searching strategy is adopted to the sequential storing structure. Unfortunately, the hyper-grid space usually is a sparse space for the greatly unbalanced distribution between normal observations and anomalous observations. In real application, some compressed methods usually are introduced to store such sparse space for saving the memory resource. Consequently, a specially designed searching method is implemented by function *Find*() to accelerate the searching speed.

(2) Anomaly detection based on hyper-grid-based method

After the detector is built, for the new continuous coming observations, the online anomaly detection activity starts up. Firstly, the new observation is mapping to the one hyper-cube of hyper-grid structure; secondly, whether it is anomalous or not is decided by the total *cnt*, which is the sum of number of mapped points corresponding to hyper-cube and its L_1 neighbor detection region. The pseudo-code of anomaly detection method is described by Algorithm 2.

Algorithm 2 *SuperGrid_Detector _AD* ($h, x_i, k, S(C_{u1,...uq})$)

Input: the new coming observation x_i

Output: the label or score to denotes x_i normal or anomalous

1: For (j ← 1,…q) $u_j \leftarrow \left\lfloor \frac{x_{i,j}}{h} \right\rfloor$;

2: $cnt \leftarrow Find$ ($S(C_{u1,...uq})$, u);

3: *If (cnt>=k) return* x_i is normal

4: For j ← 1,…q

5: If $u_{i,j} - \lfloor u_{i,j} \rfloor > 0.5h$ then $e_i \leftarrow 1$;

else if $u_{i,j} - \lfloor u_{i,j} \rfloor < 0.5h$ then $e_i \leftarrow -1$;

else $e_i \leftarrow \pm 1$;

6: $cnt \leftarrow cnt + L_1(C_{u1,...uq})$;

7: *If (cnt>=k) return* x_i is normal;

8: else *return* x_i is anomalous;

(3) Updating strategy of detector

The updating of hyper-grid structure is a re-learning procedure. In order to reduce the computation costs and alleviate the fluctuation of detection performance to some extent, a delay updating strategy is employed (Xie et al. 2012), i.e., a probability p and a buffer is specified based on some prior knowledge. For the new coming observation, p decides whether the new coming observation could be stored in a buffer or not. When the buffer is full, the procedure of updating detector will be activated. Usually, the magnitude of the p value is determined by some prior knowledge of domain experts. For example, if the dynamic characteristic of streaming data is relatively small, a relatively small value may be specified for decreasing the computational cost; otherwise, a relatively big value is specified for reflecting the change in distribution quickly. The pseudo-code of the updating detector method is described by Algorithm 3.

Algorithm 3 *SuperGrid_Detector _Update* (x_i)

Input: the new coming observations x_i

Output: The updated anomaly detector

1: For each x_i , storing the x_i in buffer based on the probability p;

2: *if* Buffer is *FULL*

3: *Building_SuperGrid_Detector*(h, X', $S(C_{u1,\ldots uq})$)

Obviously, the two key parameters, i.e., h and k, have important effects on the performance of proposed method. Due to the known data distribution in static dataset, parameters estimation method can be used to have the relatively accurate values (Xie et al. 2012; Knorr et al. 2000), and in our paper, the initial parameters value can be obtained from the historical dataset. In order to improve the robustness of above mentioned method, online ensemble learning is introduced. Next, the detail is described.

3.4 Hyper-grid-based ensemble anomaly detection algorithm

Our proposed method consists of three key procedures, i.e., training the ensemble detector based on the hyper-grid structure, online anomaly detecting and online ensemble detector updating.

(1) Building of ensemble anomaly detector

In fact, it is hard for the side length h of hyper-cube to obtain the optimal value for X, especially in streaming data. Therefore, ensemble learning is employed to build multiple weak detectors to combine a strong detector based on different hyper-grid structure, and a rough estimation range of h rather than the exact h value can be obtained based on the following description.

Based on the result presented in Xie et al. (2012) and Knorr et al. (2000), if $x \in R^q$ and $d = 2\sqrt{q}h$, the mean square error (MSE) of detection region reaches the minimum. Then the expected h can be estimated by the following formula.

$$h^* = \left(\frac{Z_1 6q}{Z_2 n \sum_{i=1}^{q} R(f_i)} \right)^{\frac{1}{q+2}} \tag{6}$$

where $R(f) = \int_{R^q} f(x)^2 \mathrm{d}x$, $f(x)$ is the probability density function (PDF), which can be estimated by the $\overline{f(x)} = \frac{S(C_{u1,\ldots uq})}{nh^q}$. $S(C_{u1,\ldots uq})$ is the number of mapped data in the hyper-cube $C_{u1,\ldots,uq}$ and it actually subject to a binomial distribution. h^* denotes the expected value of h; the approximate h can be obtained from above formula in the batch learning where the data distribution are known. Unfortunately, the infinite streaming data imply that the probability density function is hard to be obtained, and that an estimated method needs to be adopted. Suppose that in the current data chunk, the streaming data are subject to the Gaussian distribution, i.e., $N(\mu, \sum)$, where $\mu = (0, \ldots, 0)$ and $\sum = \mathrm{diag}(1, \ldots, 1)$. If the current distribution is unknown, it can be learned by the online or incremental learning method (He et al. 2011). Based on above assumption, $Z_1 \in \left[\left(\frac{3}{2}\right)^q h^{2q}, 2^q h^{2q} \right]$ and $Z_2 = (2h)^{2q}$ (Xie et al. 2012), $\frac{Z_1}{Z_2} \in \left[\left(\frac{3}{8}\right)^q, \left(\frac{1}{2}\right)^q \right]$ and $R(f_i) = \frac{1}{2^{q+1}\pi^{\frac{q}{2}}}$ can be induced. Consequently, the rough boundary of h can be calculated by the formula (7).

$$h^* \in \left[\left(\left(\frac{3}{8}\right)^q \frac{6q}{n \sum_{i=1}^{q} R(f_i)} \right)^{\frac{1}{q+2}}, \left(\left(\frac{1}{2}\right)^q \frac{6q}{n \sum_{i=1}^{q} R(f_i)} \right)^{\frac{1}{q+2}} \right] \tag{7}$$

The pseudo-code of the building hyper-grid-based ensemble anomaly detector is described by the Algorithm 4.

Algorithm 4 *Building_Ensemble_SuperGrid_Detector*(h^*,M,X)

Input: $X \leftarrow \{x_1, x_2, \ldots, x_m\}$ ($x_i \leftarrow \{x_{i,1}, x_{i,2}, \ldots, x_{i,q} \mid i \leftarrow 1, \ldots m\}$)

M: ensemble size

h^*: side length of hyper cube;

Output: Hyper-grid based ensemble detector

1: Generating M different value randomly based on the h^* and forming the set $H \leftarrow \{h_1, h_2, \ldots, h_m\}$

2: For ($i \leftarrow 1, \ldots M$)

3: *Building_SuperGrid_Detector*(h_i, X)

(2) Ensemble anomaly detecting

After the ensemble detector is trained, the procedure of anomaly detection is similar to the description in Sect. 3.3. Here, it needs to be noted that a critical parameters k is still unknown for the streaming dataset. From the perspective of statistics, the value of k can be designated by some prior knowledge in real applications, i.e., if the $\int_{J(y)} f(x)\mathrm{d}x$ is less than a very small probability, for example, 0.01, then y can be regard as anomaly. Therefore, for a dataset consisting of m observations, $k = 0.01 * m$. But for the streaming data, such an estimation method has poor robustness. In fact, after the h is specified firstly, then k can be learned based on the training dataset.

Suppose that $f(x)$ is continuous in the sample space, i.e., $x \in X$, $x_i \in [\min_i, \max_i]$, $i \leftarrow 1, \ldots, q$. Let $|J(y)|$ denotes the number of observation in detection region $J(y)$, if s observations is selected randomly from the X, the estimated value of k, denoted as $\bar{k}$, can be calculated by the formula (9).

$$\overline{k} = \frac{1}{s}\sum_{i=1}^{s} |J(y_i)| \tag{8}$$

$|J(y)|$ can be obtained from the hyper-grid structure built by the X. Unfortunately, the $f(x)$ usually is not continuous in real hyper-grid structure. Therefore, the continuous rate of hyper-grid space can be calculated by the formula (9).

$$r = \frac{|\mathrm{NP}|}{\prod_{i=1}^{q} \left\lceil \frac{\max_i - \min_i}{h^*} \right\rceil} \tag{9}$$

|NP| denotes the number of non-empty hyper-cubes in the hyper-grid structure built by dataset X, and $\prod_{i=1}^{q} \left\lceil \frac{\max_i - \min_i}{h^*} \right\rceil$ denotes the total number of hyper-cube. Then $\overline{k}$ can be estimated by the formula (10).

$$\overline{k} = \frac{r}{s}\sum_{i=1}^{s} |J(x)| \tag{10}$$

After each single detector obtains its respective result $y_i(x)$, a result combination strategy of multiple individual detector can be used to achieve the final result $y_{\mathrm{fin}}(x)$. The commonly used method in the literature is the majority vote (for classification problem) and weighted average (for regression problem). In our paper, the final ensemble detection result can be calculated by (11), where w_i denotes weight coefficient, i.e., $w_i = 1$ means the simple average, otherwise weighted average. What is more, for the sake of simplicity, the simple average strategy is employed to combine the final result.

$$y_{\mathrm{fin}}(x) = \frac{1}{M}\sum_{i=1}^{M} y_i(x) * w_i \tag{11}$$

When the data buffer is full, the ensemble detector updating is triggered. To each single detector of ensemble detector, the updating method is the same to the description of the Algorithm 3 in Sect. 3.3.

4 Experiment and result analysis

In this section, the dataset, performance evaluation metrics, experiment results and analysis will be described, respectively. Experiments were conducted on a personal PC with Intel® Core™ 2 Duo CPU, P7450@2.13 GHz and 4 GB memory. The operating system is Windows 7 professional. The data processing was partly on the MATLAB 2010, and the algorithms mentioned in Sect. 3 were implemented with Microsoft Visual C++ platform. Part of experiments were conducted on the Waikato environment for knowledge analysis software (Weka 2005).

4.1 Dataset

In order to validate proposed method, a real-world dataset, Statlog (*Shuttle*) (UCI Machine Learning Repository 2007), from the UCI machine learning repository was employed; besides, two artificial generated datasets, *SimData1*, *Simdata2* were created. Because the hyper-grid-based method is an un-supervised method, the label attribute was selected only to evaluate proposed algorithm, and further, the input order of dataset was regard as the order of streaming data for simulating the online environment.

The *Shuttle* is a widely used dataset from UCI repository, which contains 9 attributes and has little or no distribution change. The classes of this dataset are labeled from 1 to 7, and about 80 % samples of dataset belong to class 1 (i.e., normal class). To validate our method, this paper selected the label 2, 3, 5, 6, 7 as the anomaly class, the total number of which amount to 49,097.

Artificial dataset *SimData1* and *Simdata2* have the form of $X(x, y)$, in which x denotes the multiple dimension data sample and y denotes the label. Here, $y \in \{0, 1\}$, 0 denotes the normal sample and 1 anomalous sample. The dimension of *SimData1* and *SimData2* are 4 and 5, respectively. Each attribute value is continuous, and each dataset has 50,000 samples. *SimData1* has 500 anomalies and the expectation anomaly rate is 1 %, and *SimData2* has 2000 anomalies and the expectation anomaly rate is 4 %. Besides, for each artificial generated dataset, the normal data follow the Normal (Gaussian) distribution, i.e., $p(x) \sim N(\mu, \Sigma)$(μ and Σ denotes average and covariance matrix), which can be generated by the function $normrnd(\mu, \Sigma, m, n)$ using the MATLAB 2010; the anomalous dataset follows the uniform distribution. In order to simulate the streaming data which data distribution maybe changed, some strategy is designed during the process of dataset generation.

Suppose μ_i is an average of the ith dimension and its value is changed dynamically. According to $\mu_i r_i(1 + \tau)$, $\tau(\tau \in [0, 1])$ denotes the evolution degree and $r_i(r_i \in \{-1, 1\})$ denotes the evolution direction which is activated with a 5 % probability; further, the dataset evolution follows the next three assumptions.

1. Only the average of dataset was changed gradually;
2. Only the variance of dataset was changed gradually;
3. Both the average and the variance were changed gradually.

The above three assumptions were activated with a specified probability during the process of generating dataset. The summary of real-world dataset and simulated dataset can be seen in Table 1.

Table 1 Summary of three datasets

Dataset	Sample number	Attribute number	Anomaly number	Anomaly label	Anomaly rate (%)
Shuttle	49,097	9	3511	2, 3, 5, 6, 7	7.15
SimData1	50,000	4	500	1	1
SimData2	50,000	5	2000	1	4

4.2 Performance evaluation metrics

For the test observation, the trained detector defines its label as normal or anomalous. Without losing the generality, "1" and "−1" are used to denote the anomalous and normal observation, respectively. The true label for the observation x is denoted as $y \in \{-1, +1\}$, the detection result obtained from detector is denoted as $\widehat{y}$, and if $\widehat{y} = y$, the detector made a correct decision; otherwise, it made a mistake.

For anomaly detection, the results of each observation can be classified into four categories:

1. True positive, TP, i.e., $\widehat{y} = y = +1$;
2. False positive, FP, i.e., $\widehat{y} = +1,\ y = -1$;
3. True negative, TN, i.e., $\widehat{y} = y = -1$;
4. False negative, FN, i.e., $\widehat{y} = -1,\ y = +1$.

Above the commonly used evaluation metrics of anomaly detection are presented for assessing proposed method performance.

1. True positive rate (TPR)

$$\text{TPR} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}} \tag{12}$$

# denotes the number of set; i.e., #TP denotes the number of true positive observations of test dataset. This metric is renamed as *sensitivity*, *recall*, which represents the percentage of anomalies that are correctly detected, i.e., the proportion between the number of correctly detected anomalies and the total number of anomalies.

2. Positive predictive rate (PPR)

$$\text{PPR} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FP}} \tag{13}$$

It is renamed as *precision*, which means the proportion that the number of anomalous observations that are correctly detected and the number of anomalies identified by the anomaly detector.

3. F-measure

$$\begin{aligned}\text{F-measure} &= \frac{\left(1+\beta^2\right)\text{precision} * \text{recall}}{\beta^2 * \text{precision} + \text{recall}} \\ &= \frac{\left(1+\beta^2\right) * \#\text{TP}}{(1+\beta^2) * \#\text{TP} + \beta^2 * \#\text{FN} + \#\text{FP}}\end{aligned} \tag{14}$$

This metric is renamed as F-score, which is an adjustable average of precision and recall and its value is near to the relative small value between precision and recall. Consequently, the relative big value of F-measure means that the precision and recall both have higher value. β is a coefficient to adjust the relative importance of precision recall, and usually, β is set as 0.5, 1 or 2. In our paper, $\beta = 1$.

4.3 Experimental results and analysis

The purpose of the experiment is to validate the performance of proposed algorithm. Firstly, the dataset should be regularization pre-processing before the experiment; Secondly, one-fifth dataset is used to build the hyper-grid-based detector and the rest is used as a test dataset. For each dataset, h needs to be specified in advance and its expectation value can be obtained with the method proposed in Sect. 3.4. However, it is usually difficult and nearly impossible to obtain an exact value of h in real situation, a rough boundary of h is specified relatively easy based on the historical dataset, i.e., $[h_l^*,\ h_u^*]$, h_l^* and h_u^* denote the expectation lower boundary and upper boundary, respectively. With the evolution of streaming data, a more roughly boundary is specified according to the prior knowledge of domain expert. It can be evaluated by the following formula.

$$\begin{aligned} h'_u &= h_u^* + (h_u^* - h_l^*) * \lambda_1 \\ h'_l &= h_l^* - (h_u^* - h_l^*) * \lambda_2 \end{aligned} \tag{15}$$

λ_1 and λ_2 denote the relax coefficient, and its value range is [0, 1]. The h value of the three datasets can be seen in Table 2. The second column ($h*$) shows the expectation value of h which is acquired based on the formulas described in Sect. 3.4; the third column (h') shows the experimental value of h which is estimated based on the corresponding expectation value. In our paper, the value of h' is gradually changing

Table 2 The experimental value of h

Dataset	h^* (expectation value)	h' (experimental value)	k (experimental value)
Shuttle	[0.5927, 0.7499]	[0.54, 0.78]	{7, 8, 9}
SimData1	[0.3014, 0.3651]	[0.21, 0.45]	{3, 4, 5}
SimData2	[0.3726, 0.4576]	[0.26, 0.50]	{4, 5, 6}

Table 3 The performance of shuttle dataset

h	k	TPR	PPR	F-measure
0.54	7	0.7400	0.7185	0.7291
0.55	9	0.7571	0.8053	0.7805
0.56	8	0.7762	0.7235	0.7489
0.57	7	0.7652	0.8992	0.8268
0.58	8	0.7905	0.8945	0.8393
0.59	9	0.8032	0.8621	0.8316
0.60	8	0.7851	0.8502	0.8164
0.61	8	0.8345	0.8576	0.8459
0.62	7	0.8476	0.8339	0.8407
0.63	7	0.8590	0.8491	0.8540
0.64	8	0.8686	0.8559	0.8622
0.65	8	0.8771	0.8338	0.8549
0.66	7	0.8562	0.8645	0.8603
0.67	9	0.8679	0.8710	0.8694
0.68	8	0.8357	0.8741	0.8545
0.69	7	0.7995	0.8546	0.8261
0.70	9	0.7446	0.8319	0.7858
0.71	8	0.7143	0.8275	0.7667
0.72	8	0.7286	0.7837	0.7551
0.73	9	0.6986	0.7410	0.7192
0.74	8	0.6610	0.7102	0.6847
0.75	8	0.6762	0.7941	0.7304
0.76	7	0.6762	0.7526	0.7124
0.77	9	0.6667	0.7843	0.7207
0.78	7	0.6576	0.7452	0.6987

Table 4 The performance of SimData1 dataset

h	k	TPR	PPR	F-measure
0.21	3	0.8647	0.8798	0.8722
0.22	3	0.8469	0.8980	0.8717
0.23	3	0.8871	0.8850	0.8860
0.24	3	0.8700	0.8945	0.8821
0.25	4	0.8750	0.8834	0.8792
0.26	3	0.8845	0.8902	0.8873
0.27	4	0.8930	0.8456	0.8687
0.28	4	0.8845	0.8845	0.8845
0.29	4	0.8950	0.8875	0.8912
0.30	5	0.9047	0.8980	0.9013
0.31	4	0.9104	0.8813	0.8956
0.32	5	0.9022	0.8922	0.8972
0.33	5	0.8959	0.9005	0.8982
0.34	5	0.9094	0.8900	0.8996
0.35	5	0.9082	0.8770	0.8923
0.36	5	0.8980	0.8545	0.8757
0.37	4	0.8961	0.8420	0.8682
0.38	5	0.8730	0.8463	0.8594
0.39	5	0.8745	0.8448	0.8594
0.40	4	0.8654	0.8243	0.8444
0.41	4	0.8460	0.8340	0.8400
0.42	5	0.8732	0.8447	0.8587
0.43	4	0.8554	0.8045	0.8292
0.44	4	0.8274	0.7849	0.8056
0.45	5	0.8361	0.7897	0.8122

from lower boundary to upper boundary with a step length 0.01, 25 hyper-grid spaces are built for each dataset. Then, as is shown in Table 2, the corresponding k value of three datasets can be acquired based on the h' value.

1. The algorithm performance under different parameter setting

For each dataset, all 75 independent experiments were carried out smoothly based on different h and k value. Each experiment repeated itself 10 times, and we presented the average for different performance evaluation metric. In view of the limit space, for each h, only the best experiment results about k are presented in Tables 3, 4 and 5.

From the experimental results presented in Tables 3, 4 and 5, it can be seen that the appropriate h and k values affect the algorithm performance. In real applications, based on some prior knowledge and for the sake of simplicity, a relative fixed k value can be usually specified, while the hyper-grid detectors are being built. In our paper, the hyper-grid-based ensemble detector was combined by different individual hyper-grid-based detectors with different h value.

The value of three performance evaluation metrics on three dataset, i.e., TPR, PPR and F-measure, are not relatively high. Nevertheless, our proposed method proves useful, for the next section will detail the comparative experiments between hyper-grid-based between the hyper-grid-based anomaly detector and some existed methods.

Table 5 The performance of SimData2 dataset

h	*k*	TPR	PPR	F-measure
0.26	4	0.8825	0.8800	0.8812
0.27	4	0.8500	0.9714	0.9067
0.28	4	0.8752	0.9140	0.8942
0.29	4	0.8700	0.8851	0.8775
0.30	5	0.8861	0.8506	0.8680
0.31	5	0.9014	0.8896	0.8955
0.32	4	0.9021	0.8762	0.8890
0.33	5	0.8843	0.8247	0.8535
0.34	5	0.9179	0.8998	0.9088
0.35	4	0.8967	0.9185	0.9075
0.36	5	0.8500	0.8488	0.8494
0.37	4	0.8778	0.8947	0.8862
0.38	4	0.8875	0.9050	0.8962
0.39	5	0.8429	0.9043	0.8725
0.40	5	0.8667	0.8609	0.8638
0.41	5	0.8542	0.8435	0.8488
0.42	6	0.8430	0.8830	0.8625
0.43	5	0.8605	0.8660	0.8632
0.44	6	0.8660	0.8598	0.8629
0.45	4	0.8100	0.8708	0.8393
0.46	6	0.8367	0.8654	0.8508
0.47	5	0.8275	0.8300	0.8287
0.48	5	0.7813	0.7996	0.7903
0.49	4	0.8207	0.8006	0.8105
0.50	6	0.7747	0.7920	0.7833

2. Performance comparison of different algorithm

To compare the proposed method with the existed methods, Weka (Waikato environment for knowledge analysis) software, an open and intelligent experiment platform in data mining and machine learning community, is introduced. It is worth noticing that the Weka experimental platform only serves as a reference to evaluate proposed method. Here, Weka's two library functions, i.e., classifier based on support vector machine and ensemble learning (BSVME), classifier based on multiple perception machine and ensemble learning (BMLPE) are employed. Two experiments are conducted using all dataset, that is to say, the data distribution is known. If the experimental result of the proposed method is close to the results obtained from Weka, it can prove that our proposed method is effective to some extent. The experiments on Weka follow such strategies

1. Each dataset was divided into training sub-dataset and test sub-dataset according to the proportion of 2:1.
2. Bagging ensemble strategy and simple voting result combination method were employed.
3. Algorithms of one-class support vector machine and multiple layer perception were used to train the individual detector.
4. Ensemble size was set as 40, and the experimental result was the average of 50 times independent experiments.

Table 6 Performance of different anomaly detection algorithm

Dataset	Algorithm	TPR	PPR	F-measure
Shuttle	HypGridE	0.8886	0.9086	0.8985
	AHIForest	0.8802	0.9075	0.8936
	HSTree	0.8900	0.8932	0.8916
	BSVME	0.9028	0.9136	0.9082
	BMLPE	0.9085	0.9064	0.9074
SimData1	HypGridE	0.9004	0.8775	0.8888
	AHIForest	0.9048	0.8765	0.8904
	HSTree	0.8920	0.8826	0.8873
	BSVME	0.9387	0.9079	0.9230
	BMLPE	0.9038	0.9226	0.9131
SimData2	HypGridE	0.9292	0.8821	0.9050
	AHIForest	0.9184	0.8904	0.9042
	HSTree	0.9150	0.8845	0.8995
	BSVME	0.9343	0.9410	0.9376
	BMLPE	0.9428	0.9684	0.9554

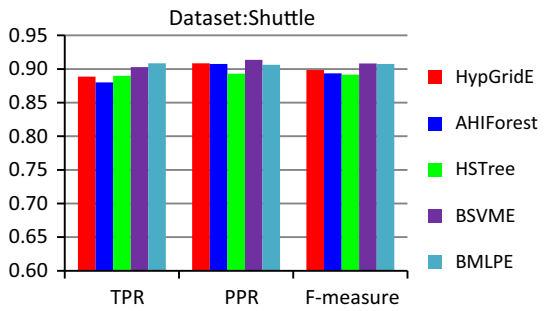

Fig. 5 Shuttle dataset experimental results of different evaluation metrics based on different algorithm

Besides, the *AHIForest* (Ding et al. 2015) and *HSTree* (Tan et al. 2011) algorithms are employed to evaluate proposed *HypGridE* algorithm. The windows size, ensemble size and statistic histogram size are set as 256, 40 and 10, respectively. The experimental results can be seen in Table 6.

From the results presented in Table 6, it can be seen that the performance of *HypGridE* has the similar experimental results to those of the *AHIForest* and HSTree algorithm, even though they are all inferior to the *BSVME* and *BMLPE*. Further, from Figs. 5, 6 and 7, the results of three performance evaluation metrics of different algorithms have slight difference. Here, it is worth noticing that the experiments of

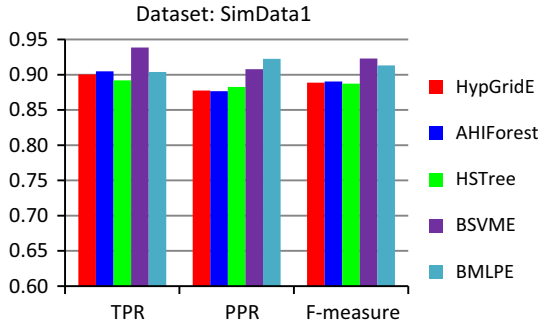

Fig. 6 SimData1 dataset experimental results of different evaluation metrics based on different algorithm

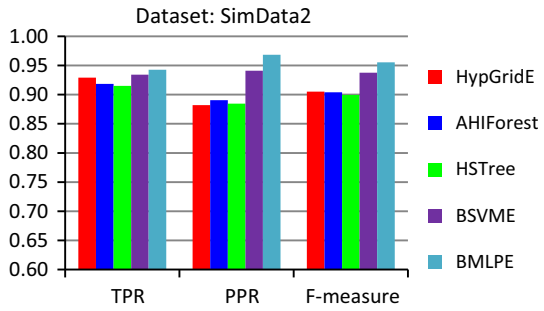

Fig. 7 SimData2 dataset experimental results of different evaluation metrics based on different algorithm

BSVME and *BMLPE* were conducted on the static dataset, while the experiments of *HypGridE*, *AHIForest* and *HSTree* were conducted on the dynamic dataset, which to some extent turns out that our proposed method is effective. Besides, compared with *BSVME* and *BMLPE, HypGridE* has an exclusive advantage that by mapping the observations to the hyper-grid space saves not only the complicated computation but also the computation cost, making itself appropriate for online streaming data processing.

5 Conclusion and future work

After exploring the drawbacks of existed hyper-based anomaly detection algorithms, an improved version with the re-definition of L_1 neighbor searching space was firstly introduced at first. Then, considering it is hard to obtain the optimized parameters related to the algorithm performance and difficult for the single detector to fit the streaming data environment, this paper employs online ensemble learning technique and proposes hyper-grid-based ensemble anomaly detection method. The greatest advantage of the proposed method is not to compute the distance or density directly and different from existed distance-based or density-based methods with low computation complexity; the method fits the online streaming data processing well, about which was demonstrated by the experiments conducted on the real and simulated data set.

Because the space complexity of hyper-grid-based anomaly detection method is $O(\mathbb{N}^q)$, when dealing with the high dimension data set, our proposed method demand significantly high memory requirement, but this may be feasible in the real applications. Hence, our future work will be centered on how to use sparse theory and some compression algorithms to tackle the problem.

Acknowledgements This study was funded by the National High Technology Research and Development Program of China (Grant No. 2011AA040103-7), the National Key Scientific Instrument and Equipment Development Project (Grant No. 2012YQ15008703), The Open Project of Top Key Discipline of Computer Software and Theory in Zhejiang Provincial (Grant No. ZC323014100), the Zhejiang Provincial Natural Science Foundation of China (Grant No. LY13F020015), National Science Foundation of China (Grant No. 61104089), Science and Technology Commission of Shanghai Municipality (Grant No. 11JC1404000), Shanghai Rising-Star Program (Grant No. 13QA1401600).

Compliance with ethical standards

Conflict of interest The four authors declare that they have no conflict of interests.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Ando S, Thanomphongphan T, Seki Y, Suzuki E (2015) Ensemble anomaly detection from multi-resolution trajectory features. Data Min Knowl Discov 29:39–83

Angiulli F, Fassetti F (2009) Dolphin: an efficient algorithm for mining distance-based outliers in very large datasets. ACM Trans Knowl Discov Data (TKDD) 3:1–57

Bifet A, Holmes G, Pfahringer B, Gavald R (2009a) Improving adaptive bagging methods for evolving data streams, advances in machine learning. Springer, Berlin, pp 23–37

Bifet A, Holmes G, Pfahringer B, Kirkby R, Gavald R (2009b) New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 139–148

Breiman L (1996) Bagging predictors. Mach Learn 24:123–140

Breiman L (2001) Random forests. Mach Learn 45:5–32

Breunig MM, Kriegel H-P, Ng RT, Sander J (2000) LOF: identifying density-based local outliers. ACM Sigmod Rec 29(2):93–104

Chang WC, Cho CW (2010) Online boosting for vehicle detection. IEEE Trans Syst Man Cybern Part B Cybern 40:892–902

Di Martino F, Sessa S, Barillari UES, Barillari MR (2014) Spatio-temporal hotspots and application on a disease analysis case via GIS. Soft Comput 18:2377–2384

Ding Z-G, Du D-J, Fei M-R (2015) An online anomaly detection method for stream data using isolation principle and statistic histogram. Int J Model Simul Sci Comput (IJMSSC) 6:1–22

Ding Z, Fei M (2013) An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. In: 3rd IFAC conference on intelligent control and automation science, ICONS 2013. IFAC Secretariat, Chengdu, pp 12–17

Daneshpazhouh A, Sami A (2014) Entropy-based outlier detection using semi-supervised approach with few positive examples. Pattern Recognit Lett 49:77–84

Desir C, Bernard S, Petitjean C, Heutte L (2013) One class random forests. Pattern Recognit 46:3490–3506

Dietterich TG (1997) Machine-learning research—four current directions. AI Mag 18:97–136

Esmaeili M, Almadan A (2011) Stream data mining and anomaly detection. Int J Comput Appl 34:38–41

Fern A, Givan R (2003) Online ensemble learning: an empirical study. Mach Learn 53:71–109

Gaber MM, Zaslavsky A, Krishnaswamy S (2005) Mining data streams: a review. ACM Sigmod Rec 34:18–26

Gil P, Santos A, Cardoso A (2014) Dealing with outliers in wireless sensor networks: an oil refinery application. IEEE Trans Control Syst Technol 23:1589–1596

Gomez J, Gil C, Banos R, Marquez AL, Montoya FG, Montoya MG (2013) A Pareto-based multi-objective evolutionary algorithm for automatic rule generation in network intrusion detection systems. Soft Comput 17:255–263

Gupta M, Gao J, Aggarwal CC, Han JW (2014) Outlier detection for temporal data: a survey. IEEE Trans Knowl Data Eng 26:2250–2267

He H, Bai Y, Garcia EA, Li S (2008) ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: IEEE international joint conference on neural networks, IEEE World congress on computational intelligence. IEEE, pp 1322–1328

He H, Chen S, Li K, Xu X (2011) Incremental learning from stream data. IEEE Trans Neural Netw Learn Syst 22:1901–1914

Huang C-W, Lin K-P, Wu M-C, Hung K-C, Liu G-S, Jen C-H (2015) Intuitionistic fuzzy c-means clustering algorithm with neighborhood attraction in segmenting medical image. Soft Comput 19:459–470

Huang H, Yoo S, Qin H, Yu DT (2014) Physics-based anomaly detection defined on manifold space. ACM Trans Knowl Discov Data 9:1–39

Knorr EM, Ng RT, Tucakov V (2000) Distance-based outliers: algorithms and applications. VLDB J 8:237–253

Kolter JZ, Maloof MA (2007) Dynamic weighted majority: a new ensemble method for tracking concept drift. J Mach Learn Res 8:2755–2790

Lee YJ, Yeh YR, Wang YCF (2013) Anomaly detection via online oversampling principal component analysis. IEEE Trans Knowl Data Eng 25:1460–1470

Limthong K, Fukuda K, Ji YS, Yamada S (2014) Unsupervised learning model for real-time anomaly detection in computer networks. IEICE Trans Inf Syst E 97D:2084–2094

Liu FT, Ting KM, Zhou ZH (2012) Isolation-based anomaly detection. ACM Trans Knowl Discov Data 6:1–39

Minku LL, Yao X (2012) DDD: a new ensemble approach for dealing with concept drift. IEEE Trans Knowl Data Eng 24:619–633

Moshtaghi M, Havens TC, Bezdek JC, Park L, Leckie C, Rajasegarar S, Keller JM, Palaniswami M (2011) Clustering ellipses for anomaly detection. Pattern Recognit 44:55–69

Noto K, Brodley C, Slonim D (2012) FRaC: a feature-modeling approach for semi-supervised and unsupervised anomaly detection. Data Min Knowl Discov 25:109–133

Oza NC (2005) Online bagging and boosting. In: 2005 IEEE international conference on systems, man and cybernetics. IEEE, pp 2340–2345

O'Reilly C, Gluhak A, Imran MA, Rajasegarar S (2014) Anomaly detection in wireless sensor networks in a non-stationary environment. IEEE Commun Surv Tutor 16:1413–1432

Palshikar GK (2005) Distance-based outliers in sequences. In: Chakraborty G (ed) Distributed computing and internet technology, proceedings. Springer, Berlin, pp 547–552

Qi ZQ, Xu YT, Wang LS, Song Y (2011) Online multiple instance boosting for object detection. Neurocomputing 74:1769–1775

Quinn JA, Sugiyama M (2014) A least-squares approach to anomaly detection in static and sequential data. Pattern Recognit Lett 40:36–40

Sagha H, Bayati H, Mill JDR, Chavarriaga R (2013) On-line anomaly detection and resilience in classifier ensembles. Pattern Recognit Lett 34:1916–1927

Salem O, Liu YN, Mehaoua A, Boutaba R (2014) Online anomaly detection in wireless body area networks for reliable healthcare monitoring. IEEE J Biomed Health Inform 18:1541–1551

Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC (2001) Estimating the support of a high-dimensional distribution. Neural Comput 13:1443–1471

Segui S, Igual L, Vitria J (2013) Bagged one-class classifiers in the presence of outliers. Int J Pattern Recognit Artif Intell 27:1–21

Serdio F, Lughofer E, Pichler K, Buchegger T, Pichler M, Efendic H (2014) Fault detection in multi-sensor networks based on multivariate time-series models and orthogonal transformations. Inf Fusion 20:272–291

Subramaniam S, Palpanas T, Papadopoulos D, Kalogeraki V, Gunopulos D (2006) Online outlier detection in sensor data using non-parametric models. In: Proceedings of the 32nd international conference on very large data bases, VLDB Endowment, pp 187–198

Suhailis A, Kadir A, Abu Bakar A, Hamdan AR (2014) Frequent positive and negative (FPN) itemset approach for outlier detection. Intell Data Anal 18:1049–1065

Tan SC, Ting KM, Liu TF (2011) Fast anomaly detection for streaming data. In: Proceedings of the twenty-second international joint conference on artificial intelligence. AAAI Press, pp 1511–1516

Ting K, Zhou G-T, Liu F, Tan S (2013) Mass estimation. Mach Learn 90:127–160

UCI Machine Learning Repository (2007) http://archive.ics.uci.edu/ml/datasets.html

Weka (2005) http://www.cs.waikato.ac.nz/ml/weka/

Xie M, Hu J, Han S, Chen H (2012) Scalable hyper-grid k-NN-based online anomaly detection in wireless sensor networks. IEEE Trans Parallel Distrib Syst 24:1661–1670

Yamanishi K, Takeuchi JI, Williams G, Milne P (2004) On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. Data Min Knowl Discov 8(3):275–300

Yang X, Han L, Li Y, He L (2015) A bilateral-truncated-loss based robust support vector machine for classification problems. Soft Comput 19:2871–2882

Yu X, Tang LA, Han J (2009a) Filtering and refinement: a two-stage approach for efficient and effective anomaly detection. In: ICDM'09. Ninth IEEE international conference data mining. IEEE, pp 617–626

Yu Y, Guo SQ, Lan S, Ban T (2009b) Anomaly intrusion detection for evolving data stream based on semi-supervised learning. Adv Neuro-Inf Process 5506:571–578

Zhang Y, Meratnia N, Havinga P (2010) Outlier detection techniques for wireless sensor networks: a survey. IEEE Commun Surv Tutor 12:159–170

Zhou XZ, Li SP, Ye Z (2013) A novel system anomaly prediction system based on belief Markov model and ensemble classification. Math Probl Eng 2013:831–842