Computers & Industrial Engineering 60 (2011) 195–205

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Increasing availability of industrial systems through data stream mining

Ahmad Alzghoul [1], Magnus Löfstrand *

Division of Computer Aided Design, Luleå University of Technology, Room E218P, SE-97187 Luleå, Sweden

ARTICLE INFO

Article history:
Received 5 August 2010
Received in revised form 13 October 2010
Accepted 14 October 2010
Available online 20 October 2010

Keywords:
Availability
Data stream mining
Data stream management system
Industrial systems
Grid-based classifier

ABSTRACT

Improving industrial product reliability, maintainability and thus availability is a challenging task for many industrial companies. In industry, there is a growing need to process data in real time, since the generated data volume exceeds the available storage capacity. This paper consists of a review of data stream mining and data stream management systems aimed at improving product availability. Further, a newly developed and validated grid-based classifier method is presented and compared to one-class support vector machine (OCSVM) and a polygon-based classifier.

The results showed that, using 10% of the total data set to train the algorithm, all three methods achieved good (>95% correct) overall classification accuracy. In addition, all three methods can be applied on both offline and online data.

The speed of the resultant function from the OCSVM method was, not surprisingly, higher than the other two methods, but in industrial applications the OCSVMs' comparatively long time needed for training is a possible challenge. The main advantage of the grid-based classification method is that it allows for calculation of the probability (%) that a data point belongs to a specific class, and the method can be easily modified to be incremental.

The high classification accuracy can be utilized to detect the failures at an early stage, thereby increasing the reliability and thus the availability of the product (since availability is a function of maintainability and reliability). In addition, the consequences of equipment failures in terms of time and cost can be mitigated.

1. Introduction

Industrial companies seek to manufacture products of high quality. High quality can be achieved by increasing the reliability, the maintainability and thus the availability of a product (Bleed, 1986). Understanding and improving the maintenance phase of the lifecycle of a product is therefore crucial for achieving the goal of high reliability, maintainability and availability. Usually, maintenance tasks are aimed at minimizing failures of industrial plant, machinery and equipment and the consequences of such failures (Alsyouf & Alzghoul, 2009). In this paper we define failures as having faults, interruptions, or stops in a system. Failures are represented in a data by having a deviation in one or more variable. The most common way to detect, predict and avoid failures is to collect and analyze the information produced during the time of operation and maintenance. However, recent advances in technology (hardware and software) have enabled us to collect data from different sources at high generating rates, i.e., streams of data. As a consequence, the development of automated analysis techniques and data stream management systems (DSMS) is necessary Fogelman-Soulié, 2008. Therefore, this paper aims to:

Investigate how to increase the reliability and maintainability and thus the availability of industrial systems by studying data stream mining and data stream management systems.

We will examine how the data stream mining algorithms can be utilized to monitor the status of the machine. The research method consists of a review of data stream mining techniques, a review of data stream mining applications in the field of operation and maintenance, and a real case study. In addition, we will propose a new method, i.e., a grid-based classification method. The research presented in this paper has been carried out in collaboration with the Swedish company Hägglunds Drives AB (http://www.hagglunds.com/), a manufacturer of low-speed, high-torque hydraulic drive systems. Fig. 1 shows an example of a hydraulic motor system from Hägglunds Drives AB.

To monitor Hägglunds Drive AB hydraulic motors a number of sensors were installed to collect information on motor status during operation. The information concerned parameters such as temperature, speed and pressure.

A database management system (DBMS) is used to manage and control data stored in a database (Fogelman-Soulié, 2008). However, as the volume of the collected data has become so large (Exabyte) and the generating rates have increased, the DBMS is

* Corresponding author. Tel.: +46 (0) 920 493874.
E-mail address: Magnus.Lofstrand@ltu.se (M. Löfstrand).
[1] Tel.: +46 (0) 920–492453.

0360-8352/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2010.10.008

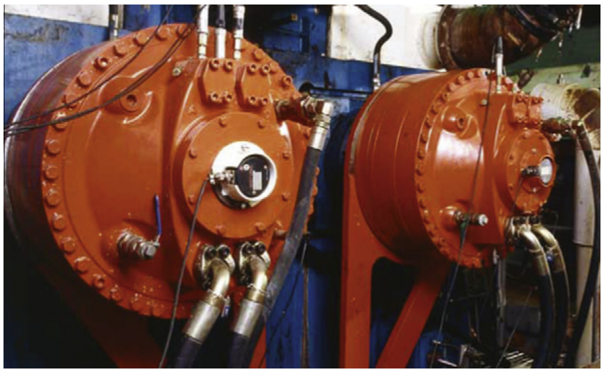
Fig. 1. Hägglunds Drive AB hydraulic motor system.

unable to handle the data. Therefore, a data stream management system may be used to manage the data stream. In addition, DSMS can be used together with data stream mining to analyze the data.

Data mining is a useful tool for analyzing data stored in a database. Data mining is part of the Knowledge Discovery in Databases (KDD) process; its role is to extract patterns from data. Nevertheless, due to their low-speed, the data mining algorithms are not able to handle the data on the fly. Thus, in recent years, new algorithms have been created and some of the data mining algorithms have been modified to handle the data stream. These new and modified data stream mining algorithms can be classified in four categories: Clustering, Classification, Frequency counting, and Time series analysis (Gaber, Zaslavsky, & Krishnaswamy, 2005).

Data mining has been applied in different fields and its applications have attracted many companies, industries and governments. Manufacturing engineering is one field in which data mining has been applied successfully. Manufacturing systems, decision support systems, shop floor control and layout, fault detection, quality improvement, maintenance and customer relationship management are examples of data mining applications in manufacturing (Harding, Shahbaz, & Kusiak, 2006). However, further studies are needed to investigate the application of data stream mining in monitoring with the aim of improving product availability.

A new method (grid-based classifier) and selected algorithms (one-class support vector machines (OCSVM) and polygon-based classifier), which were applied on the reviewed applications (Kargupta, 2004; Matthews & Srivastava, 2008), respectively, have been tested on the data collected from Hägglunds Drives AB hydraulic motor system is reported in this paper. The result, presented in Section 5, Table 1, shows that the algorithms achieved good performance (up to 98%) in detecting the failures. The grid-based method developed in the research leading up to this publication is explained further in Section 4.

This paper is organized as follows: Section 2 presents the research method of our work. Section 3 discusses the data stream issues and includes the review of data stream mining algorithms and their applications. Section 4 presents our proposed classification method. Section 5 presents and discusses the results of the applied algorithms. Finally, Section 6 presents the main conclusions and future work.

2. Research method

Through literature review we studied the issues of data stream mining and data stream management systems. We reviewed the data stream mining algorithms and their applications in the field of operation and maintenance. The literature review has been conducted utilizing several databases, focusing mainly on IEEE along with technical reports and literature published on the World Wide Web. The search included the following keywords: data stream mining, data stream mining algorithm, data stream management system, reliability, maintainability, availability, condition monitoring and machine health monitoring. The review is presented in Section 3.

After the literature review we studied the algorithms which were used in the reviewed applications and selected the algorithms which are suitable for our application. Thereafter, selected algorithms and the proposed approach have been tested on the data collected from Hägglunds Drives AB hydraulic motor system.

The architecture of our fault detection system is illustrated in Fig. 2 below. The first step is to train the algorithms offline using the training data to produce the fault detection functions. Once we finished the training stage, we brought the functions online. The DSMS is responsible for controlling and managing the generated data stream, we used the Super Computer Stream Query processor (SCSQ) Zeitler & Risch, 2006 as a DSMS. If the function output indicates a failure occurrence, then the alarm is activated. The algorithms will be retrained offline using the new incoming data.

To test this fault detection system we stored the collected data on a PC and then used a DSMS to stream the data to another PC, where the algorithms (fault detection function) are implemented. We used SCSQ data stream management system to stream, control, and manage the data. SCSQ was developed at Uppsala DataBase Laboratory (UDBL) http://user.it.uu.se/~udbl/. The SCSQ is currently considered the world's fastest data stream management system (DSMS), since it has achieved the best score thus far for the Linear Road Benchmark (http://user.it.uu.se/~udbl/SCSQ.html). The fault detection function was implemented using Active Mediator Object System (Amos II) query language (AmosQL) Risch, Josifovski, & Katchaounov, 2003. Offline algorithm training and Figs. 3 and 5–7 were done and produced using Matlab. The next subsections discuss the data set and the algorithms which will be used for fault detection.

2.1. The data set

Data were collected at three different motor speeds from motors in the Hägglunds Drives AB laboratrory. The collected data did not have any failure sample. The data contain 11,153 data points which correspond to a 22 variables; i.e. 11,153 $\times$ 22. The data was divided into two groups, the first of which was used to train the algorithms, i.e., training data, and represented approximatly10% of the data (1118 data points). Usually one uses 40% or more of the data points

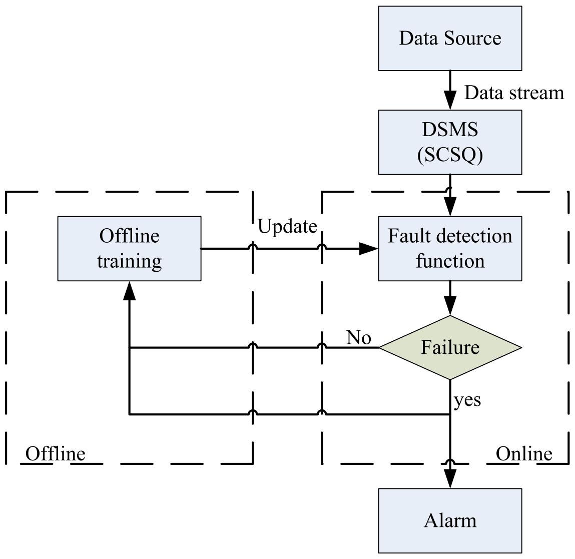

Fig. 2. The architecture of the fault detection system.

for training and the rest for testing. However, if we are able to get good result using fewer data points for training that will give two advantages: (1) We do not need large storage capacity to save all the data, i.e. a summary of the data is enough. (2) The algorithm training time will be less. Having tried different percentages, the authors came to the conclusion that in this particular case using 10% data points for training yielded acceptable result in terms of classification accuracy and training time. The second group was used for the testing purposes, i.e., testing data, and represented around 90% of the data (10,035 data points). In addition, we created some artificial abnormal data to examine the algorithms, since no failures were present initially in the collected data. The abnormal data simulate two failures which are represented by 68 data points (68×22) for the three different motor speeds. Note that other faulty cases exist, based on the number of variables and possible machine errors. However, the other errors can not be simulated since the available data does not support it, i.e. we did not have access to enough data to simulate other possible faulty cases.

The distribution of and comparison between the abnormal and normal cases of the data are presented in Fig. 3 below. The two variables which are reported in Fig. 3 have been modified with artificial abnormal data based on interviews with experienced engineers and researchers working in close collaboration with Hägglunds Drives AB. (The artificial data represents possible errors which may occur in the hardware.)

2.2. The applied algorithms for fault detection

Our problem is a one-class classification, since the data refer only to the normal behavior; i.e., the machine works without any problem. The created artificial failures were used only for examining the algorithms, not for training. Therefore, the two one-class classification methods, which were used by (polygon-based method) Kargupta (2004) and (OCSVM) Matthews and Srivastava (2008), and the proposed grid-based classifier, which is described in Section 4, were selected to be tested on our data.

3. Data stream issues

Data stream involves two important issues: data stream management and data stream mining. While data stream management concerns either the building of aggregated historical data from the input stream or alarming on some conditions, the data stream mining concerns application of the data mining algorithm to either the whole or part of the data stream (Fogelman-Soulié, 2008). In the following subsections both terms are discussed in more detail.

3.1. Data stream management systems

As identified in the popular magazine *The Economist*, (Data, 2010) the quantity of generated data and the available storage are continuously increasing. However, the rate of growth of the generated data is faster than that of the available storage capacity. This overload has compelled researchers to find systems and tools that can handle data on the fly without the need to store it. Data stream management system has been found to be an effective means of handling continuously generated data.

Data stream management system can be defined as an extension of a database management system which has the ability to deal with a data stream. A DSMS has the same structure as a DBMS (i.e., table in a relational database), but a data stream has no associated disk storage; it arrives continuously, its arrival rate may vary from time to time, and the missed data may be lost. In order to handle the stream data a special query language, which is compatible with the structure of the DSMS, has to be defined. These queries are applied continuously and permanently on the data stream, producing a new stream or updating a permanent table (Fogelman-Soulié, 2008).

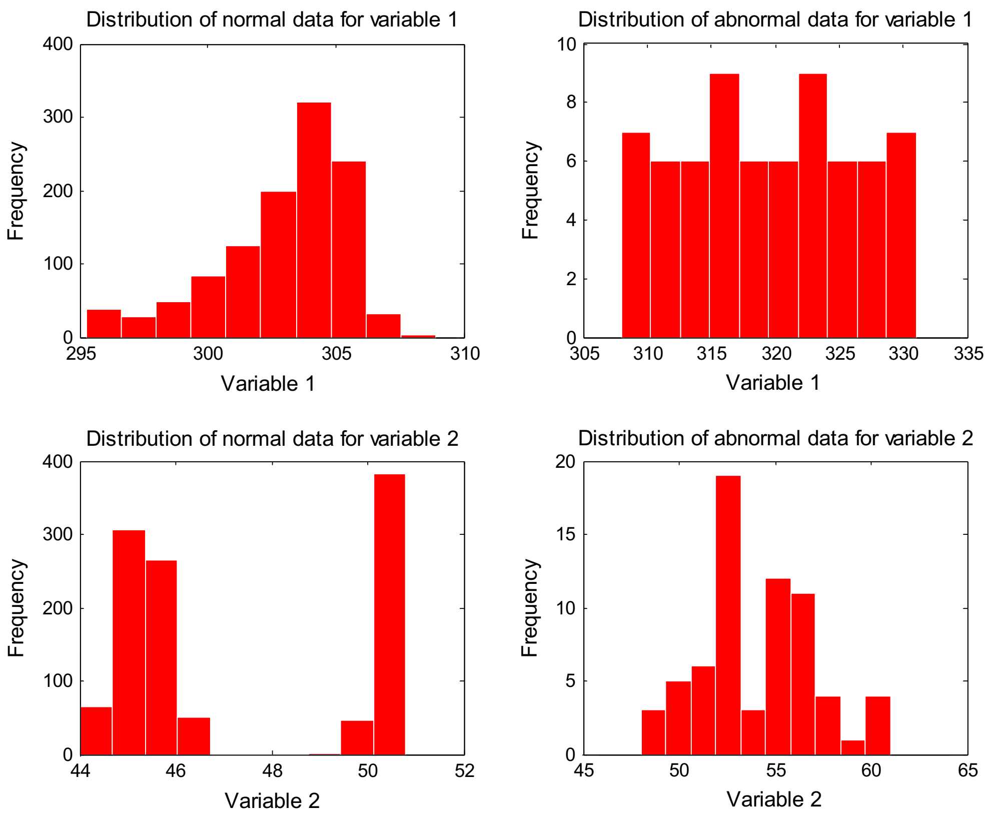

Fig. 3. The distribution between abnormal and normal cases.

A DSMS faces a number of challenges which must be considered during its design. It should have an optimized execution plan that enables it to deal with the varying arrival rates of the data stream, works with limited memory, and allows memory sharing between the different queries. SCSQ, STREAM (Arasu & et al., 2004), TelegraphCQ (Chandrasekaran & et al., 2003), and Aurora–Medusa–Borealis (Abadi & et al., 2003) are examples of general-purpose DSMSs, that can be used for more than one purpose or application. On the other hand, Gigascope (Cranor & et al., 2003) and Hancock (Cortes & et al., 2000) are examples of specialized, application-based, DSMSs. They were designed for network monitoring and analysis of telecommunications (Fogelman-Soulié, 2008).

3.2. Data stream mining

Data stream mining can be defined as the extraction of patterns from continuous and rapidly arriving streams of data (Gaber, Zaslavsky et al., 2005; Karacal, 2006). In this case, the data cannot be stored and must be manipulated immediately. Therefore, the data mining algorithm has to be sufficiently fast to handle the fast rate of arriving data.

Data mining algorithms can be applied either on the whole or on a part (window) of the stream. Thus, the algorithms differ according to the type of the window. According to (Fogelman-Soulié, 2008) there are three types of windows: (1) Whole stream: the algorithm has to be incremental, e.g., artificial neural network and incremental decision tree. (2) Sliding window: the algorithm has to be incremental and also it must have the ability to forget the past, e.g., incremental principal component analysis. (3) Any past portion of the stream: the algorithm has to be incremental and able to keep a summary of the past in a limited memory, e.g., Clustream algorithm.

In the next subsections we will review the data stream mining algorithms and their applications in the field of maintenance and operation.

3.2.1. Data stream mining algorithms review

In this section, data stream mining systems and techniques are reviewed. We classify the existing algorithms into four categories: Clustering, Classification, Frequency counting and Time series analysis. This work is based on a continuation of the review done in 2005 by Gaber, Zaslavsky et al. (2005). The literature review is conducted utilizing several databases, with a main focus on IEEE, along with technical reports and literature published on the World Wide Web. The next subsections summarize these algorithms according to their categories: clustering, classification, frequency counting and time series analysis.

3.2.1.1. Clustering. Clustering is unsupervised learning, where we categorize a given data into subsets so that each subset represents a cluster which has distinctive properties. Clustering algorithms have been applied in different engineering applications such as manufacturing system design, quality assurance and production processes (Pham & Afify, 2006). The summary of the data stream clustering techniques review can be found in Appendix A, Table A1.

3.2.1.2. Classification. Classification is supervised learning, where we group the data into classes based on labeled training data set. Fault detection, fault diagnostics and decision making are examples of engineering applications in which the classification techniques can be used (Harding et al., 2006). The review of the data stream classification techniques is summarized in Appendix A, Table A2.

3.2.1.3. Frequency counting. Frequency counting aims to find the most interesting relations between variables. Product design improvement and shop floor control and layout are examples of frequency counting applications in the manufacturing field (Harding et al., 2006). Table A3 in Appendix A summarizes the frequency counting techniques found in this review.

3.2.1.4. Time Series Analysis. Time series is a sequence of data points taken sequentially in time and the adjacent data points are dependent. The analysis of these dependences concerns time series analysis and its techniques (Box, Jenkins, & Reinsel, 1976). Controller tuning is an example of a time series analysis application in the engineering field (Smith, 1992). A review of these techniques is presented in Appendix A, Table A3.

3.2.2. Applications of data stream mining in the field of operation and maintenance

Data stream mining algorithms have been applied in different fields such as financial transactions, sensor network measurements, telecommunication networks, manufacturing systems and scientific data monitoring systems (Kobayashi, 2004). The maintenance phase of the lifecycle processes of a product is concerned with how the system is operated, used and maintained. Usually, maintenance tasks are aimed at minimizing the failure of industrial plant, machinery and equipment and the consequences of such failure (Alsyouf & Alzghoul, 2009). In this section the applications of data streaming mining in the field of operation and maintenance are reviewed. The following list summarizes the applications found in this review:

- Machine monitoring and reliability analysis (Karacal, 2006; Karacal, 2007).
- Online failure prediction (Youree & et al., 2008).
- Mobile and distributed data stream mining for vehicle-health monitoring (Kargupta, 2004; Kargupta, 2007).
- Anomaly detection from solid rocket motor data (Matthews & Srivastava, 2008).
- Fault detection in industrial measurement (Angelov & et al., 2006).
- Tool condition monitoring (Karacal, Cho, & Yu, 2009).

The description and the methods used in these applications are reported in Sections 3.2.2.1–3.2.2.6.

3.2.2.1. Machine monitoring and reliability analysis. A framework for the use of multivariate statistical process monitoring technique and data stream mining techniques to detect the change in the sensor data stream was proposed by Karacal (2006, 2007). The vibration, noise, current and temperature in the motors for the CNC machine were suggested as the sensor channels targeted. In addition to these stream data, some of the processing parameters were suggested for cross referencing. Examples of such processing parameters are: feed rate, rpm, cutting tool, geometry and material properties of the work piece. It is suggested that the data be collected at different sampling rates under normal and distinctive processing conditions. The collected data will be then used for referencing and validating simulated data. Also, the authors suggested imitating different breakdown scenarios. After that, the empirical reference streams along with the breakdown scenarios are used as an input for generating simulated test streams. They proposed a tilted time window-based on seconds, minutes and hours. The algorithm will shift between these different time windows depending on the most recent shift detection results. This model will allow use of both the most recent and the historical data, which will make the data stream mining more effective A number of methods for detecting the failure of a machine at early stage were suggested:

- Scaled-up MEWMA (multivariate exponentially weighted moving average).
- The regression adjustment method.
- Principal component method.

Once the shift is detected a clustering method is applied to see which micro-cluster the shift belongs to. To detect patterns on individual sensors, using the dynamic window frame, a pattern recognition algorithm is applied. Nonparametric correlation can be utilized to explore the nonlinear relations between the streams before sensor fusion. The dynamic weight hierarchy was suggested for sensor fusion. The sensor fusion will give the global status of the machine at a given time. After the sensor fusion a classification routine is applied to predict the remaining machine life. Finally, the frequency of correct estimates and the average degree of deviation from the true values will be combined to measure the performance (Karacal, 2007).

3.2.2.2. Online failure prediction. A design of multivariate statistical analysis technique for online failure prediction was developed and tested by Youree et al. (2008). The system was designed for monitoring the health of mission equipment, industrial equipment and facilities. In this process, similarity test and trend determination are the two stages through which the input data passes. Once the sensor data arrive, a feature extraction algorithm is applied, and then the extracted features are compared statistically with the feature patterns which are considered to be representative of the nominal operating condition for equipment. The nominal operating conditions for equipment can be obtained either from equipment specification or from a sensor-based data stream. The trend determination stage will be activated if a specific threshold is exceeded at the similarity test stage. The result of the trend stage is used to determine the status of the machine and the required maintenance actions. For statistical similarity test, which depends on the dimension of the data stream, Youree et al. (2008) have suggested the following tests:

- Kolmogorvo–Smirnov (KS) test, for single dimensional data stream.
- Hotelling's T-square test, for multidimensional data stream.

To perform the trending Youree et al. (2008) have used a combination of several tests to determine the best trend from a group of a given trends. These tests are:

- Residual noise.
- Sum of squares due to Error.
- Degree-of-freedom adjusted R-square.
- Confidence bounds for the coefficients.
- Slope of the fitted curve.

Finally, they used a Bayes risk criterion to select the state which results in minimum expected cost.

A multivariate similarity-based modeling (SBM) technique for machine health monitoring (detecting incipient faults in an H-60 intermediate gearbox (IGB)) using vibration data was discussed by Wegerich (2004)., who used this technique due to its scalability, flexibility and usability in condition monitoring, and it has been successfully applied in different condition monitoring applications. He used two steps to monitor the gearbox status: First step, feature extraction from the input raw data. Second step, apply the SBM to the extracted features to produce estimates and residuals. The input data contains the vibration component of each individual tooth in the gear. The residual signatures, which are generated in the second step, together with a classifier or logical rules, were used to diagnose the faults. Wegerich assessed his approach on two seeded fault test cases; the results showed that SBM is able to detect the faults early and to monitor the machine health of the gearbox. In addition, the residual signatures simplified the fault diagnostics.

3.2.2.3. Mobile and distributed data stream mining for vehicle-health monitoring. Mobile and distributed data stream mining systems were used in Kargupta (2004) for real time vehicle-health monitoring. In this work they developed the VEhicle DAta Stream mining (VEDAS) system, which is used to monitor the vehicle data streams and driver characterization. VEDAS used an on-board PDA-based distributed data stream mining system connected with remote modules through wireless network to monitor vehicle fleets.

VEDAS software has the following components: (1) interfacing with the on-board data bus, (2) on-board data stream management and mining, (3) central control station module and (4) a privacy management module. In order to monitor the vehicle-health, the clusters which represent good conditions for the desired performance of the vehicle were generated. Then, these clusters were used to determine the healthy operation regimes which are subsequently used to monitor the new data. Once the new data arrive the on-board data stream mining module checks whether the data fall into the safe operating regimes or not. If so, that indicates that the vehicle is in good condition; otherwise, the vehicle is not in the normal mode. The implementation of this process was done by using the incremental principal component analysis (which uses the matrix perturbation theory to detect large changes) for data projection, the incremental k-means clustering algorithm for separating the data into k safe regimes, and the Delaunay Triangulation based polygonization approach to calculate the polygon which represents the clusters (Kargupta, 2004). Test reports of applying this approach on the collected data from Hägglunds Drives are given in Section 5.

MineFleet and fast resource constrained monitoring of correlation matrices were used by Kargupta (2007) for on-board vehicle data stream monitoring. The work has the same general structure as in Kargupta (2004) but with some modifications to the applied algorithms. They used a probabilistic test to quickly detect the change in the correlation matrix and to determine the portions of the matrix which contain the significant coefficients.

3.2.2.4. Anomaly detection from solid rocket motor data. A number of data-driven anomaly detection methods were applied on continuous data stream from a solid rocket motor in Matthews and Srivastava (2008). Orca, one-class support vector machines (OCSVM) and Inductive Monitoring System (IMS) were the three methods used for the anomaly detection, and their performances were compared. Three ground firing tests of a 10-in. solid rocket motor TD-31 were used to obtain the data. A total of 25 sensors, which corresponds to the three main factors: pressure, temperature and strain, were used in this work. While a 2000 Hz sampling rate was used for the pressure sensors, a sampling rate of 250 Hz was used for the strain and the temperature. Since the indication of a failure can be observed within the first second after ignition, the implemented algorithm attempted to detect the O-ring fault within the first second. Due to the differences in the sampling rates between the different factors, the algorithms were tested on each factor separately. The results showed good performance for the suggested methods in detecting faults by either directly detecting the O-ring failure or indirectly, i.e., a drop in pressure indicates a growing in a hole. The algorithms achieved the best performance in early fault detection when strain sensors were used as inputs for the algorithms. The best time result was achieved by the OCSVM method. Test reports of applying this algorithm on the collected data from Hägglunds Drives are given in Section 5. However, all three methods (Orca, IMS, OCSVM) were able to be used in detecting failures before the critical failure point.

3.2.2.5. Fault detection in industrial measurement. A model-based approach to detect the faults in the industrial application is proposed in Angelov et al. (2006). The proposed approach, which was built using data-driven and hybrid approaches, was tested and verified on both simulated and real data sets from car engine test benches. The result showed that the method was capable of detecting the fault effectively in industrial measurement and it also outperformed the conventional correlation- and regression-based models (Angelov et al., 2006).

3.2.2.6. Tool condition monitoring. A sensor stream mining system for tool condition monitoring was developed by Karacal et al. (2009). They used chemical data from the gases released during the cutting process to monitor the cutting tool condition. They proposed a system of two stages through which the stream data will pass for the monitoring purpose: (1) Dimension reduction stage and (2) classifier stage. They suggested using PCA to compress the data from high dimension to a small dimension space. In the second stage they use a binary support vector machine as a classifier. Since the SVM is computationally expensive, they suggested training the classifier, using collected data, before the system is brought online.

3.2.3. Applications review result

The review showed that there are some applications, where the data stream mining was applied successfully. In addition, several frameworks were suggested for the monitoring purpose. However, since the application of data stream mining in manufacturing is still at the early stage, most of the reviewed data stream mining algorithms were not applied in the field of operation and maintenance. Moreover, there is a need to compare the performance of the different algorithms in the different applications.

As discussed in Section 2, the problem is a one-class classification, since the data which we have represent only the normal behavior. Therefore, the two one-class classification methods, which were used by (polygon-based method) Kargupta (2004) and (OCSVM) Matthews and Srivastava (2008), have been selected to be tested on our data.

In Section 4, below, we will describe the proposed grid-based classification method, and in Section 5 we compare our method to the polygon-based and OCSVM methods.

4. The Grid-based classification method

According to Saini and Dua (2010) even though there are some techniques which are closely related to grid-based classifier such as (Wei, Huang, & Tian, 2007), there is no approach which can be purely classified as a grid-based classifier. Saini and Dua (20100 suggested a grid-based scalable classifier which can be used for large and high dimensional datasets. The training phase of their algorithm consists of three steps: grid generation, class mapping and class boundary detection.

Our grid-based classification uses a grid to partition the data space into small elements. The elements can have different shapes such as squares, rectangles and triangles. Each element keeps information regarding the training data points, i.e., the data which we normally use to train an algorithm. In this case, information refers to how many training data points belong to every class. A new data point is classified according to its corresponding element information, not depending on the whole body of data, which makes the classification process faster. The flow chart for the grid-based methods is illustrated in Fig. 4 below.

Our approach consists of two stages: (1) data mapping and (2) data classification. In the first stage the training data are mapped from high dimensional into two-dimensions (2D). The data space is then partitioned by a uniform grid, as in Fig. 5. After that, we count and save the number of the data points, which belong to every class, for each element in the grid. For elements which do not have any of the classes, we set the number for these classes in these elements at zero. For example, in Fig. 5, if we consider element number five, then we set the count for class1 to 1 (has one point which corresponds to class1) and 9 for class2 (has nine points correspond to class2) while for element number seven we set the count for class1 to 0 (has no points which corresponds to class1) and 0 for class2 (has no points correspond to class2). In the data classification stage, we classify a new data point depending on the probability of a class in the element which the new data point, after mapping in 2D, corresponds to. For example, if a new data point maps in the area of element 1 then the probability that the data point belongs to class1 is 20% (3/15, number of data points which belong to class1 in element 1 divided by the total number of data points in element 1) and to class2 80% (12/15). Therefore, the new data point will be classified as class2. Note that one could have the output as a list of all the available classes and their probabilities.

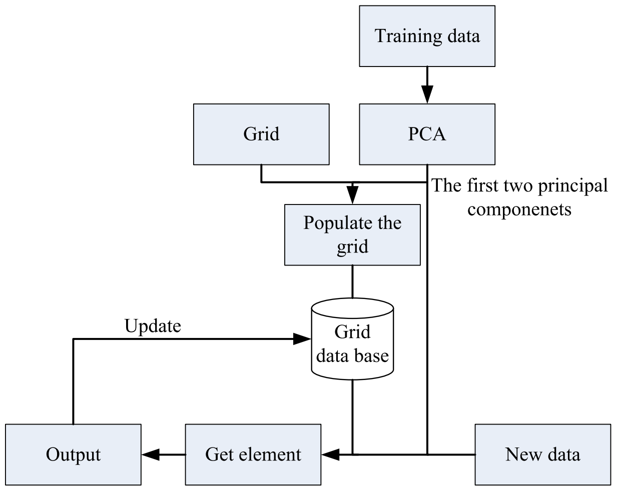

Fig. 4. Grid-based classification method architecture.

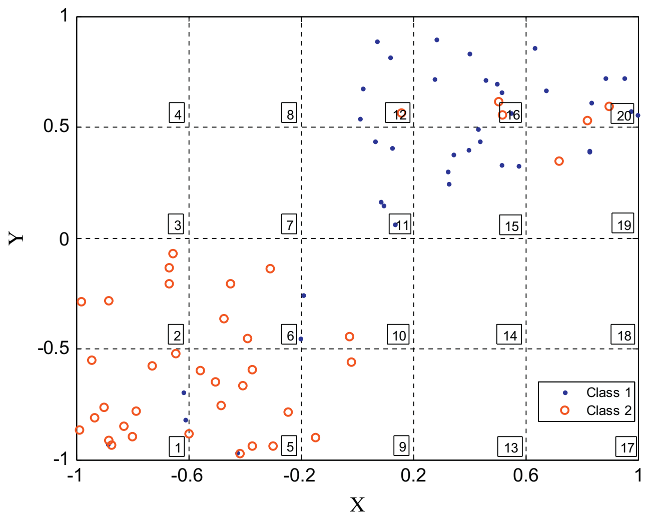

Fig. 5. A grid with 20 elements (squares), the number in the corner of every square (element) corresponds to the element number.

The grid-based classification can be modified to be incremental by continuously updating the element information whenever a new data point is received. So, when we confirm the class to which a new data point belongs, we increase the number of that class in the corresponding element. In case of a new class appearing, we add this class to all elements and count and save the number of data points which belong to this class in each element in the grid.

Grid-based classification can be also modified to deal with one-class problem by counting the number of training data points in each element in the grid (in this case, we count for the class which we have). Then we classify a new data point by:

(1) Calculating the probability that a new data point belongs to the class which we have. The probability is computed by dividing the number of data points in the corresponding element by the maximum number of data points which an element, in the grid, has. For example, if we assume that Class1 in Fig. 5 is the only class that we have, then, element number 15 has the maximum number of data points which equal to 8. So, if we get a new data point mapped in element 11, then the probability that the new data point belongs to the class in hand is 62.5% (5/8) and the probability to have it in element number 5 is 12.5% (1/8).

(2) Comparing the number of the data points in the corresponding element to a pre- specified threshold. If it is more than the threshold, then it belongs to the class which we have. For example, if we specify the threshold to be 2, then all the new data points which will be mapped in elements 1, 11, 12, 15, 16, and 20 will be classified as Class1.

The limitation of this method is that it is based on the first two principal components. To evaluate the limitation, a test is needed to check if the first two components are sufficient to cover the data set information. Scree plots may for example be used to find how many components should be retained. In the next section we test our algorithm and compare it to the polygon-based and OCSVM methods.

5. Results

In this section we discuss the results from testing all three algorithms: polygon-based, OCSVM and grid-based classification methods. For the OCSVM we used the Radial Basis Function (RBF) as a kernel. As we discussed in Section 2, we train the OCSVM algorithm offline and then brought the resulted function online. Fig. 6 shows the resultant polygons which represent the safe areas using the polygon-based method (blue dots represent the normal behavior, while the red circles represent the failures). These polygons were constructed by mapping the training data, which represent the normal behavior, into two-dimensions, then find the clusters, and finally construct the polygon for every cluster. The resultant three clusters are due to the three different speeds at which the hydraulic motor operates.

For the grid-based method we used a triangle grid, as in Fig. 7, and we used the threshold which achieves the highest classification accuracy. By trying different numbers we selected the value 7 as a threshold. Thereafter, the three trained algorithms were tested on the streamed data. The result of the test is presented in Table 1.

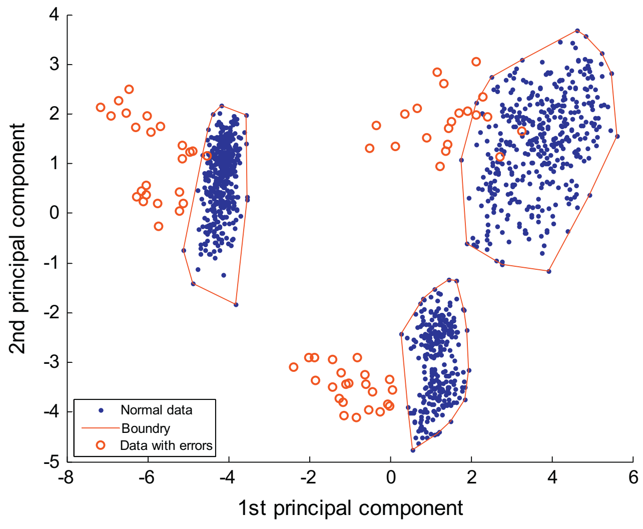

Fig. 6. Polygons represent safe areas.

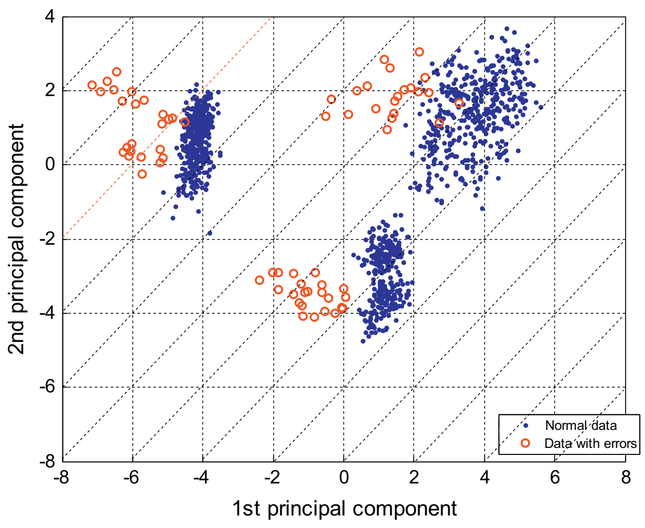

Fig. 7. Grid-based method, the total number of normal and abnormal data are 1118 and 68 respectively.

Table 1
classification accuracy for polygon-based, OCSVM and grid-based algorithms and their speed.

Method	Classification accuracy (normal data) (%)	Classification accuracy (abnormal data) (%)	Classification accuracy (overall) (%)	Processing time for one data point (s)
Polygons-based	96.02	92.54	95.99	0.02
OCSVM	98.40	98.53	98.40	0.00026
Grid-based	96.82	83.82	96.73	0.035

Table 1 shows that the classification accuracy is achieved by the OCSVM method. It achieved the highest classification accuracy for both normal and abnormal data with 98.4% and 98.53% correctness, respectively, and thus for overall (normal + abnormal) with 98.4%. In addition, it has the highest speed by spending only 0.00026 s to process one data point; i.e., it can handle around 3846 data points per second. On the other hand, while the polygon-based method outperforms the grid-based method in classifying abnormal data and algorithm speed (processing time per data point), the grid-based method outperforms the polygon-based method in classifying normal data and also in overall classification accuracy.

The OCSVM algorithm outperforms the polygon-based and grid-based methods because it maps the data into a higher dimension and then find the decision boundary (a hyper-plane), while the other two methods map the data from high dimension to a lower dimension, which could cause information lost.

The grid-based method is a bit slower than the polygons-based method, since the grid-based method has to find the element to which a data point corresponds before the classification process starts, while for the polygon-based method the search, itself, is the classification process. For instance, the polygon-based method required only finding whether a data point is in three areas (which can be considered as three elements) or not, while the grid-based method needs to search more than three elements before the classification process begins. On the other hand, even the training process for the OCSVM is computationally expensive; the resultant function can be used to classify new data very quickly, since the function depends only on the support vectors.

The weakness of the tests as such is that there is only one data set tested. Thus, the results lack some generality. To address this issue, the authors used three parts of the data set which correspond to three different motor speeds and took their average. One limit of the test is that using only one case (as has been done here) always includes the potential danger that it produces clearly poorer or better results than that it would be on average, by using several data sets to represent "generality" more reliably.

6. Conclusions

High product availability is of great interest in industrial companies. Monitoring and analyzing the data, from different kinds of sensors used during the product lifecycle, can improve the availability by detecting, predicting, and avoiding the faults at an early stage. There is a need to handle the data in real time, since the generated data volume exceeds the available storage capacity. Data stream management systems and data stream mining algorithms are effective tools for searching and analyzing a high-volume data stream.

We have reviewed the data stream mining algorithms and monitoring applications. We found that the application of data stream mining for monitoring purposes is still in the early stages, and most of the data stream mining algorithms have not been applied in industrial applications. However, some research describes the application of data stream mining in the field of operation and maintenance such as: machine monitoring and reliability analysis, online failure prediction, mobile and distributed data stream mining for vehicle-health monitoring, anomaly detection from solid rocket motor data, fault detection in industrial measurement, and tool condition monitoring.

A new method (Grid-based classifier) was proposed and validated. We compared the proposed method to the OCSVM and the polygon-based classifier by testing them on the data collected from Hägglunds Drives AB hydraulic motor system data. We found that the OCSVM outperformed the grid-based and the polygon-based methods in both classification accuracy and execution time. However, all three methods achieved good (>95% correct) overall classification accuracy. In addition, all the methods can be applied on both offline and online data.

The main advantages of the grid-based classification method are that it allows for calculation of the probability (%) that a data point belongs to a specific class, and the method can be easily modified to be incremental or used for one-class classification problems. Further work is needed to determine how to optimize the design of the grid, i.e., element shape and element dimensions.

The results achieved by the OCSVM method are promising. Its high classification accuracy can be utilized to detect failures at an early stage, and thus increase the maintainability and, ultimately, the availability of the product. However, the OCSVM method needs to be further validated by being applied on data containing actual measured failures. Also, since the normal behavior of a machine can change over time (concept drift), the algorithm needs to be retrained periodically (or retrained according to a data analysis method) or modified to be incremental.

It is necessary to study the cost when an algorithm output is not correct, i.e., issues an alarm when there is no failure or there is a failure with no alarm. Future work also includes studying the frequency of the different failures, their costs and the time needed to fix them. In addition, further optimization of the algorithms is needed to see if it is possible to speed up the implementation of the algorithms.

Acknowledgements

This work was funded by the SSPI (SSF) http://www.ltu.se/tfm/cad/home/d3919/d748/d18397/1.42699?l=en and the iStreams (VINNOVA) http://user.it.uu.se/~udbl/iStreams.html projects. The authors wish to thank the collaborating industrial partners for their participation and kind help. Bengt Liljedahl and Anders Westerlund deserve special mention. In addition, the authors wish to thank our research partners at Uppsala DataBase Laboratory (http://user.it.uu.se/~udbl/) and Professor Tore Risch for their help and support.

Appendix A

See Tables A1–A4.

Table A1
Clustering techniques summary.

Clustering technique	Brief description	References
K-median technique	Uses single pass over data	Guha and et al. (2000)
K-median combined with exponential histogram (EH)	Improve the above technique Guha et al. (2000) (they addressed the problem of merging clusters)	Babcock and et al. (2003)
Improved k-median algorithm	Overcome the problem of increasing approximation factors in Guha et al. (2000).	Charikar, O'Callaghan, and Panigrahy (2003)
VFKM	Very fast k-means (scaled up machine learning algorithm applied on k-means)	Domingos and Hulten (2001)
Incremental k-means algorithm	An improved version of k-means used for clustering binary data stream	Ordonez (2003)
Stream and localsearch algorithms	Used to cluster high quality data stream clustering.	O'Callaghan (2002)
CluStream algorithm	A frame work for clustering data stream. Used incremental algorithm and keep a summary of the past.	Aggarwet al. (2003)
HPStream	Used for high dimensional data stream clustering. Outperforms the CluStream.	Aggarwet al. (2004)
Technique based on K-motif	Produce meaningful results in subsequence clustering.	Keogh and Lin (2005)
LWC	Lightweight Clustering. It based on Algorithm Output Granularity (AOG).	Gaber, Krishnaswamy, and Zaslavsky (2005)
DenStream	A density-based data streams clustering algorithm over damped windows.	Cao (2006)
AcluStream	Based on the density and space	Wegerich (2004)
D-Stream	Density and grid-based algorithm, adopts two-phased clustering framework.	Chen and Tu (2007)
SOStream	Clusters high dimensional data Streams (the data space is partitioned into grids)	Wang and et al. (2008)
EStream	Evolution-based technique, it uses a distance function to test the similarity between two objects	Udommanetanakit, Rakthanmanon, and Waiyamai (2007)
SWClustering	It combines the exponential histogram with the temporal cluster features, and uses the EHCF data structure.	Zhou and et al. (2008)
SUBFCM	Subtractive Fuzzy cluster means (an energy efficient algorithm)	Sabit, Al-Anbuky, and Gholam-Hosseini (2009)
Weighted fuzzy algorithm	An extension to Fuzzy c-means (FCM)	Wan (2008)
MovStream	Based on the measurement of the movements	Tang (2008)
DD-Stream	A grid and density-based clustering algorithm	Jia (2008)
HDenStream	A density-based algorithm for clustering data stream with heterogeneous features	Jinxian (2009)
SPDStream	Subspace partition clustering based on CDTree lattice structure	Zhang (2009)
SDStream	Density-based algorithm which adopts the CluStream clustering framework	Ren (2009)

Table A2
Classification techniques summary.

Classification technique	Brief description	References
GEMM and FOCUS	For data mining model maintenance and change detection (under block evolution).	Ganti, Gehrke, and Ramakrishnan (2002)
VFDT	Very fast decision tree. decision tree based on hoeffding trees.	Domingos and Hulten (2000)
CVFDT	Concept drifting very fast decision tree learner (extension to VFDT)	Hulten, Spencer, and Domingos (2001)
OLIN	Online information network	Last (2002)
AWSOM	Arbitrary window stream modeling method. Used to find interesting pattern discovery	Papadimitriou, Brockwell, and Faloutsos (2003)
On-Demand classification	Classify data using the clustering result	Aggarwal and et al. (2004)
Decision tree classification on spatial data streams	Fast building algorithm based on Peano count tree (P-tree) structure	Ding and Perrizo (2002)
LWClass	Lightweight classification. It based on algorithm output granularity (AOG)	Gaber, Krishnaswamy et al. (2005)
LELC	PU learning by extracting Likely positive and negative micro-clusters	Li (2009)
Dynamic incremental SVM learning	The most appropriate classifier is selected according to the statistical performance	Li (2007)
EVFDT	Efficient-VFDT, based on uneven interval numerical pruning (UINP) and naïve Bayes	Li (2008)
Clustering-training	A semisupervised learning algorithm for unlabeled data stream mining	Shuang (2006)
HashCVFDT	On top of CVFDT, based on extended Hash table and list	Ouyang (2008)
VFDTb	On top of VFDT, based on binary search trees	Wang (2007)
Active learning method	Mine time-changing data streams, can be used for unlabeled data	Shucheng (2008)
Basic incremental IFN multi-model IFN Pure multi-model IFN advanced incremental IFN	Algorithms based on Info-Fuzzy Network (IFN) which is an advanced decision tree learning methodology	Cohen (2008)

Table A3
Frequency counting techniques summary.

Frequency counting technique	Description	References
Frequent itemsets mining algorithm	Uses the tilted windows to calculate the frequent patterns for the most recent transactions.	Giannella and et al. (2003)
Approximate frequency counts in data stream	Uses all the previous historical data	Manku and Motwani (2002)
Counting frequent items using group testing	Used with the turnstile data stream	Cormode and Muthukrishnan (2005)
LWF	Lightweight frequency counting. It based on algorithm output granularity (AOG)	Gaber, Krishnaswamy et al. (2005)
MISM	Mining Frequent Item sets over data Streams by Matrix	Jin and Agrawal (2005)
DSM-MFI	Based on the lossy counting algorithm	Li, Lee, and Shan (2005)
FpMax	An algorithm based on FP-tree, uses MFI-tree to store maximal frequent itemsets.	Liu (2006)
THUI-Mine	Mining temporal high utility itemsets from data streams	Chu (2008)
Exclusive Incremental	An efficient technique for incrementally mining contrast patterns	Bailey (2010)
CP-tree	Work same as FP-growth algorithm but with only one scan instead of two	Tanbeer (2009)
estAcc	The count is monitored by a lexicographic tree resided in main memory. There is no candidate generation process.	Chang (2006)
IDSM-MFI	Uses a synopsis data structure to store the items of transactions embedded data streams so far	Mao (2009)
MRFI-SW	Mining recent frequent itemsets by sliding window (each items is denoted a bit-sequence representations)	Ren (2008)
MFS-HT	Mining frequent itemsets based on D-Hashed table	Chunhua (2009)
MFI-TD	Mine maximal frequent itemsets based on time decay model	Huang (2009)
HCFI	Based on hash-based closed itemsets monolayer tree(HCI-Mtree) for mining closed frequent Itemsets	Ren (2008)
UF-streaming SUF-growth	Tree-based algorithms for mining uncertain data stream	Leung (2009)
STAGGER	Incremental algorithm for mining patterns using expanding sliding window	Elfeky (2007)
CPS-tree	Compact pattern stream tree, based on FP-growth mining technique	Tanbeer (2009)

Table A4
Time series analysis techniques summary.

Time series analysis technique	Description	References
Approximate solutions with probabilistic error bounding	Used to solve the problems of relaxed periods and average trend	Indyk, Koudas, and Muthukrishnan (2000)
Mining astronomical time series streams technique	Uses two phases: one for clustering and the other for association rule discovery.	Perlman and Java (2002)
StatStream	Uses discrete fourier transform	Zhu and Shasha (2002)
Technique based on the symbolic representation of time series data stream	Includes two transformation: from time series data to piecewise aggregate approximation (PAA) then from PAA to discrete string symbols	Lin (2003)
Regression cubes for data stream	Uses multidimensional regression analysis	Chen and et al. (2002)
Generic event detection approach	For batch and online incremental processing of time series data.	Guralnik and Srivastava (1999)
SPRING	Streaming method for subsequence matching in data streams	Sakurai (2007)
ASM	Accurate Subsequence Matching, based on SPRING algorithm	Niennattrakul, Wanichsan, and Ratanamahatana (2009)
ARES	Adaptive Radius-based Search, efficient similarity join over multiple stream time series	Lian and Chen (2009)
ODAC	Online divisive agglomerative clustering. a clustering system for streaming time series	Rodrigues (2008)
FSM	Fast subsequence matching algorithm, three times faster than SPRING algorithm	Zou (2008)

References

Abadi, D. et al. (2003). Aurora: A new model and architecture for data stream management. *The VLDB Journal, 12*(2), 120–139.

Aggarwal, C. et al. (2003). *A framework for clustering evolving data streams*. VLDB Endowment.

Aggarwal, C. et al. (2004). *A framework for projected clustering of high dimensional data streams*. VLDB Endowment.

Aggarwal, C. et al. (2004). *On demand classification of data streams*. ACM.

Alsyouf, I., Alzghoul, A. (2009). Soft computing applications in wind power systems: a review and analysis. In *European offshore wind conference and exhibition*. Stockholm, Sweden.

Angelov, P. et al. (2006). An approach to model-based fault detection in industrial measurement systems with application to engine test benches. *Measurement Science and Technology, 17*(7), 1809–1818.

Arasu, A., et al. 2004. Stream: The stanford data stream management system. In Garofalakis, Gehrke, and Rastogi (Eds.), *A book on data stream management*.

Babcock, B., et al. (2003). *Maintaining variance and k-medians over data stream windows*.

Bailey, J. (2010). Efficient incremental mining of contrast patterns in changing data. *Information Processing Letters, 110*(3), 88–92.

Bleed, P. (1986). The optimal design of hunting weapons: Maintainability or reliability. *American Antiquity, 51*(4), 737–747.

Box, G., Jenkins, G., & Reinsel, G. (1976). *Time series analysis: Forecasting and control*. Holden-day San Francisco.

Cao, F. (2006). Density-based clustering over an evolving data stream with noise. In *Proceedings of the sixth SIAM international conference on data mining* (p. 328–339).

Chandrasekaran, S. et al. (2003). *TelegraphCQ: Continuous dataflow processing*. ACM.

Chang, J. H. (2006). Finding frequent itemsets over online data streams. *Information and Software Technology, 48*(7), 606–618.

Charikar, M., O'Callaghan, L., Panigrahy, R. (2003). *Better streaming algorithms for clustering problems*.

Chen, Y. et al. (2002). *Multi-dimensional regression analysis of time-series data streams*. VLDB EndowmentVLDB Endowment.

Chen, Y., & Tu, L. (2007). *Density-based clustering for real-time stream data*. ACM.

Chu, C. J. (2008). An efficient algorithm for mining temporal high utility itemsets from data streams. *The Journal of systems and software, 81*(7), 1105–1117.

Chunhua, J. (2009). Mining approximate frequency itemsets over data streams based on D-Hash table. In *10th ACIS conference on software engineering, artificial intelligence, networking and parallel/distributed computing, SNPD* (pp. 249–254).

Cohen, L. (2008). Info-fuzzy algorithms for mining dynamic data streams. *Applied Soft Computing, 8*(4), 1283–1294.

Cormode, G., & Muthukrishnan, S. (2005). What's hot and what's not: tracking most frequent items dynamically. *ACM Transactions on Database Systems (TODS), 30*(1), 249–278.

Cortes, C. et al. (2000). *Hancock: A language for extracting signatures from data streams*. ACM.

Cranor, C. et al. (2003). *Gigascope: A stream database for network applications*. ACM.

Data, 2010. Data everywhere (a special report on managing information). In *The Economist*. February 27th, 2010.

Ding, Q., & Perrizo, W. (2002). *Decision tree classification of spatial data streams using Peano Count Trees*. ACM.

Domingos, P., Hulten, G. (2001). *A general method for scaling up machine learning algorithms and its application to clustering*.

Domingos, P., & Hulten, G. (2000). *Mining high-speed data streams*. ACM.

Elfeky, M. G. (2007). STAGGER: Periodicity mining of data streams using expanding sliding windows. In *Proceedings of IEEE international conference on data mining, ICDM* (pp. 188–199).

Fogelman-Soulié, F. (2008). *Data stream management and mining*. Mining Massive Data Sets for Security: Advances in Data Mining, Search, Social Networks and Text Mining, and Their Applications to Security. p. 89.

Gaber, M., Krishnaswamy, S., and Zaslavsky, A. (2005). On-board mining of data streams in sensor networks. In: *Advanced methods for knowledge discovery from complex data* (p. 307–335).

Gaber, M., Zaslavsky, A., & Krishnaswamy, S. (2005). Mining data streams: A review. *ACM Sigmod Record, 34*(2), 26.

Ganti, V., Gehrke, J., & Ramakrishnan, R. (2002). Mining data streams under block evolution. *ACM SIGKDD Explorations Newsletter, 3*(2), 10.

Giannella, C. et al. (2003). Mining frequent patterns in data streams at multiple time granularities. *Next Generation Data Mining, 212*, 191–212.

Guha, S. et al. (2000). *Clustering Data Streams*.

Guralnik, V., & Srivastava, J. (1999). *Event detection from time series data*. ACM.

Harding, J., Shahbaz, M., & Kusiak, A. (2006). Data mining in manufacturing: A review. *Journal of Manufacturing Science and Engineering, 128*, 969.

http://user.it.uu.se/~udbl/ Cited 04.08.10.

http://user.it.uu.se/~udbl/iStreams.html Cited 01.08.10.

http://user.it.uu.se/~udbl/SCSQ.html Cited 09.06.10.

http://www.hagglunds.com/ Cited 03.08.10.

http://www.ltu.se/tfm/cad/home/d3919/d748/d18397/1.42699?l=en Cited 0.08.10.

Huang, G. Y. (2009). An efficient algorithm based on time decay model for mining maximal frequent itemsets. In *Proceedings of the 2009 international conference on machine learning and cybernetics*.

Hulten, G., Spencer, L., & Domingos, P. (2001). *Mining time-changing data streams*. ACM.

Indyk, P., Koudas, N., & Muthukrishnan, S. (2000). *Identifying representative trends in massive time series data sets using sketches*. Morgan Kaufmann Publishers Inc.

Jia, C. (2008). A grid and density-based clustering algorithm for processing data stream. In *Proceedings of 2nd international conference on genetic and evolutionary computing, WGEC*.

Jin, R., and Agrawal, G. (2005). An algorithm for in-core frequent itemset mining on streaming data.

Jinxian, L. (2009). A density-based clustering over evolving heterogeneous data stream. In *2nd ISECS international colloquium on computing, communication, control, and management, CCCM*. 2009.

Karacal, S. C. (2006). Data stream mining for machine reliability. In *IIE Annual conference and exhibition*.

Karacal, S. C. (2007). Mining machine data streams using statistical process monitoring techniques. In *IIE annual conference and expo*.

Karacal, C., Cho, S., & Yu, W. (2009). Sensor stream mining for tool condition monitoring. *Computers and Industrial Engineering*, 1429–1433.

Kargupta, H. (2004). VEDAS: A mobile and distributed data stream mining system for real-time vehicle monitoring. *SIAM Proceedings Series, 1*, 300–311.

Kargupta, H. (2007). On-board vehicle data stream monitoring using mine-fleet and fast resource constrained monitoring of correlation matrices. *New Generation Computing, 25*(1), 5–32.

Keogh, E., & Lin, J. (2005). Clustering of time-series subsequences is meaningless: Implications for previous and future research. *Knowledge and Information Systems, 8*(2), 154–177.

Kobayashi, M. (2004). Data stream mining: Selected tools & algorithms (numerical analysis and new information technology). *RIMS Kokyuroku, 1362*, 124–132.

Last, M. (2002). Online classification of nonstationary data streams. *Intelligent Data Analysis, 6*(2), 129–147.

Leung, C. K. S. (2009). Mining of frequent itemsets from streams of uncertain data. In *Proceedings of international conference on data engineering*.

Li, Z. W. (2007). Dynamic incremental SVM learning algorithm for mining data streams. In *Proceedings of the 1st international symposium on data, privacy, and E-commerce, ISDPE*.

Li, F. (2008). An improved algorithm of decision trees for streaming data based on VFDT. In *International symposium on information science and engineering, ISISE* (pp. 597–600).

Li, X. L. (2009). Positive unlabeled learning for data stream classification. In *Society for industrial and applied mathematics – 9th SIAM international conference on data minin, Proceedings in applied mathematics* (pp. 256–267).

Li, H., Lee, S., and Shan, M. (2005). Online mining (recently) maximal frequent itemsets over data streams. In *15th IEEE Int'l workshop on research issues on data engineering (RIDE–SDMA)*.

Lian, X., & Chen, L. (2009). Efficient similarity join over multiple stream time series. *IEEE Transactions on Knowledge and Data Engineering, 21*(11), 1544–1558.

Lin, J. (2003). A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on research issues in data mining and knowledge discovery, DMKD '03*.

Liu, X. (2006). Mining frequent closed patterns from a sliding window over data streams. *Journal of Computer Research and Development, 43*(10), 1738–1743.

Manku, G., & Motwani, R. (2002). *Approximate frequency counts over data streams*. VLDB Endowment.

Mao, Y. (2009). A mining maximal frequent itemsets over the entire history of data streams. In *First international workshop on database technology and applications, DBTA*.

Matthews, B., and Srivastava, A. N. 2008. Comparative analysis of data-driven anomaly detection methods. In *JANNAF conference on propulsion systems. Proceedings of the joint army navy NASA air force conference on propulsion*, Orlando, FL.

Niennattrakul, V., Wanichsan, D., and Ratanamahatana, C. (2009). *Accurate subsequence matching on data stream under time warping distance*.

O'Callaghan, L., et al. (2002). Streaming-data algorithms for high-quality clustering. In *Proceedings of IEEE international conference on data engineering*.

Ordonez, C. (2003). Clustering binary data streams with K-means. In *Proceedings of the 8th ACM SIGMOD workshop on research issues in data mining and knowledge discovery* (p. 12–19).

Ouyang, Z. (2008). An efficient decision tree classification method based on extended hash table for data streams mining. In *Proceedings of 5th international conference on fuzzy systems and knowledge discovery, FSKD 2008*.

Papadimitriou, S., Brockwell, A., & Faloutsos, C. (2003). *Adaptive, hands-off stream mining*. VLDB Endowment.

Perlman, E., and Java, A. (2002). *Predictive mining of time series data in astronomy*. Arxiv preprint astro-ph/0212413.

Pham, D., & Afify, A. (2006). *Engineering applications of clustering techniques*. Elsevier Science Ltd.

Ren, J. D. (2008). Online data stream mining of recent frequent itemsets based on sliding window model. In *Proceedings of the 7th international conference on machine learning and cybernetics, ICMLC* (pp. 293–298).

Ren, J. (2008). Mining closed frequent itemsets in sliding window over data streams. In *Third international conference on innovative computing information and control, ICICIC'08*.

Ren, J. (2009). Density-based data streams clustering over sliding windows. In *Sixth international conference on fuzzy systems and knowledge discovery, FSKD*. 2009.

Risch, T., Josifovski, V., & Katchaounov, T. (2003). Functional data integration in a distributed mediator system. *Functional Approach to Computing with Data*.

Rodrigues, P. P. (2008). Hierarchical clustering of time-series data streams. *IEEE Transactions on Knowledge and Data Engineering, 20*(5), 615–627.

Sabit, H., Al-Anbuky, A., & Gholam-Hosseini, H. (2009). *Distributed WSN data stream mining based on fuzzy clustering*. IEEE Computer Society.

Saini, S., & Dua, S. (2010). A grid-based scalable classifier for high dimensional datasets. *Information Systems, Technology and Management, 1*, 404–415.

Sakurai, Y. (2007). Stream monitoring under the time warping distance. In *Proceedings of international conference on data engineering*.

Shuang, W. (2006). *Clustering–training for data stream mining*.

Shucheng, H. (2008). An active learning method for mining Time-Changing data streams. In *Proceedings of 2nd international symposium on intelligent information technology application, IITA*.

Smith, K. (1992). Methodology for applying time series analysis techniques. In *International conference of the instrument SOC of America (ISA/92 Canada)*.

Tanbeer, S. K. (2009). Efficient single-pass frequent pattern mining using a prefix-tree. *Information Sciences, 179*(5), 559–583.

Tanbeer, S. K. (2009). Sliding window-based frequent pattern mining over data streams. *Information sciences, 179*(22), 3843–3865.

Tang, L. (2008). MovStream: An efficient algorithm for monitoring clusters evolving in data streams. In *Granular computing in IEEE international conference on 2008* (pp. 582–587).

Udommanetanakit, K., Rakthanmanon, T., and Waiyamai, K. (2007). E-stream: Evolution-based technique for stream clustering. *Advanced Data Mining and Applications*, 605–615.

Wan, R. (2008). A weighted fuzzy clustering algorithm for data stream. In *Proceedings of ISECS international colloquium on computing, communication, control, and management, CCCM* (pp. 360–364).

Wang, T. (2007). An efficient classification system based on binary search trees for data streams mining. In *2nd International conference on systems, ICONS*.

Wang, S., et al. (2008). Subspace clustering of high dimensional data streams. In *Proceedings of the 7th international conference on computer and information science*.

Wegerich, S. W. (2004). Similarity based modeling of time synchronous averaged vibration signals for machinery health monitoring. *IEEE Aerospace Conference Proceedings, 6*, 3654–3662.

Wei, X., Huang, H., Tian, S. (2007). A grid-based clustering algorithm for network anomaly detection. isdpe, 104–106.

Youree, R., et al. (2008). *A multivariate statistical analysis technique for on-line fault prediction*.

Zeitler, E., & Risch, T. (2006). Processing high-volume stream queries on a supercomputer. In *Proceedings of the 22nd international conference on data engineering workshops*. IEEE Computer Society.

Zhang, Z. (2009). A fast subspace partition clustering algorithm for high dimensional data streams. In *Proceedings of IEEE international conference on intelligent computing and intelligent systems, ICIS*.

Zhou, A. et al. (2008). Tracking clusters in evolving data streams over sliding windows. *Knowledge and Information Systems, 15*(2), 181–214.

Zhu, Y., & Shasha, D. (2002). *Statstream: Statistical monitoring of thousands of data streams in real time*. VLDB Endowment.

Zou, P. (2008). Fast similarity matching on data stream with noise. in *IEEE 24th international conference on data engineering workshop ICDEW 2008*.