Research Issues in Outlier Detection for Data Streams

Shiblee Sadik and Le Gruenwald
The University of Oklahoma, School of Computer Science, Norman OK 73019, USA
{shiblee.sadik, gguenwald}@ou.edu

ABSTRACT

In applications, such as sensor networks and power usage monitoring, data are in the form of streams, each of which is an infinite sequence of data points with explicit or implicit timestamps and has special characteristics, such as transiency, uncertainty, dynamic data distribution, multi-dimensionality, and dynamic relationship. These characteristics introduce new research issues that make outlier detection for stream data more challenging than that for regular (non-stream) data. This paper discusses those research issues for applications where data come from a single stream as well as multiple streams.

1. INTRODUCTION

In this era of information, the data assimilation process has changed significantly. Data generated by applications like sensor network, web-click monitoring, network traffic monitoring, etc. are in the form of data streams. A data stream is a continuous, unbounded sequence of data records accompanied and ordered by implicit or explicit timestamps. Their arrival rate is usually high and their distributions often change over time. Data streams started gaining popularity in early 2000s [1], [2]; since then numerous efforts have been put forward to study them from different perspectives, such as system design [3], [4], query processing [5], [6], [7], [8], resource management [9], optimization [10], [11], scalability [12], storage management [13], data mining [14], [15], [16], etc. However, not much research has been conducted to investigate the outlier detection perspective, which is an important part of the data acquisition process. In this paper, we aim to fill in this gap by identifying research issues that need to be addressed when designing techniques to detect outliers in data streams.

An outlier is a data point which is significantly different from other data points, or does not conform to the expected normal behavior, or conforms well to a defined abnormal behavior [17], [18]. In this definition, the phrases "significantly different," "does not conform to the expected normal behavior," and "conforms well to a defined abnormal behavior" are very subjective and deserve scrutiny; therefore the definition of outlier bears some vagueness. However, outliers are often mentioned as anomalous data points – the ones that do not conform to the expected normal behavior. The data points that are not outliers are often called inliers. Outliers in different domains are different in nature from one another. An outlier in a credit card transaction is not the same as an outlier in meteorological data. Hence, different applications have their own definitions and interpretations of outliers.

Outliers may appear in a dataset for numerous reasons, like malicious activity, instrumental error, setup error, changes of environment, human error, catastrophe, etc. Regardless of the reason, outliers may be interesting and/or important to the user because of their diverse nature compared to normal data points. Some people define outliers as problems, some people define them as interesting items, but in any case, they are unavoidable [17], [19]. Outliers can be classified into three major categories [17], [18] as follows:

Type I Outliers- An isolated individual data point in a dataset is termed as a Type I outlier. By definition they are the simplest type and it is very easy to identify them. Intuitively they are far from other data points in the dataset in terms of attribute values.

Type II Outliers- A data point that is isolated with respect to other data points in the context is called a type II outlier. The context refers to the semantic relationship among the data points. Typically, data in this type of dataset have other contextual attributes (e.g., time, location, etc.). An outlier is a data point that is far from other data points in the same context (i.e. having the same time or the same location) in terms of value. This is a little bit different from a Type I outlier; a Type I outlier is a data point isolated from all the other data points in the dataset. A temperature of 200 °F for any day is most likely an outlier with respect to the entire dataset of temperature (Type I outlier); however a temperature of 100°F is not an outlier with respect to the entire dataset, but is most likely an outlier if we consider the temperature of winter only (Type II outlier). A Type II outlier was first investigated in time series data in the late seventies. Barnett & Lewis [19] defined Type II outliers as the Additive Outliers for time series data. The good thing about additive outliers is that they do not influence the other data points in context; hence it is easy to identify them.

Type III Outliers- A particular group of data points that appear as outliers with respect to the entire dataset is termed type III outliers. No data point in a small subset is an outlier with respect to the other points in the subset, but as a group, they are outliers. For contextual data like time series, the entire dataset forms a sequence; hence a particular subsequence is an outlier with respect to the entire sequence. Barnett & Lewis (1994) called them Innovations Outliers for time series data. The bad thing about innovations outliers is that they influence other data points of the same context and try to hide themselves; therefore it is difficult to identify innovations outliers.

A data stream has one temporal context with each data point; so it might have a type II or type III outlier but never a type I outlier. This is because data streams are considered as infinite series and their processing has to be online; therefore at any particular moment, only a subset of the entire dataset is present; so a data point cannot be an outlier with respect to the entire dataset.

Outlier detection refers to the problem of identifying the outliers in a dataset. Since the definition of outliers is vague and application-dependent, a formal method for outlier detection is less developed. By definition, an outlier detection technique takes a dataset as input and outputs outliers. Despite of the vagueness of outlier definition, several approaches are popular for outlier detection based on the state of the input data. The first approach is called the supervised approach where the outlier detection technique assumes the availability of labeled data [20]. A supervised technique collects knowledge from labeled data and applies the collected knowledge to unlabeled data for outlier detection. The second approach is called the semi-supervised approach which requires only inliers or outliers to be labeled. Both of these approaches are less popular due to the lack of labeled datasets. The third approach is called the unsupervised approach. An unsupervised technique does not require any type of labeling; hence it is very popular for outlier detection. However, unsupervised techniques often suffer from higher false alarms [17].

Outlier detection for data streams is a new area of research compared to the long history of outlier detection in statistical data [19]. Outlier detection for data streams is fundamentally different from that for regular data. In case of regular data, any technique can learn the pattern from the dataset and then compares every data point to the pattern to detect outliers (the store-and-process paradigm). However the store-and-process paradigm is not applicable for data streams because the entire dataset is never available due to their unbounded nature; therefore learning any trend without store-and-process is challenging and requires new research. In addition, due to the characteristics specific to data streams, such as high arrival rate, real time, change of data distribution (or concept drift), outlier detection for this type of data has a different set of research issues as opposed to that for regular data, which we discuss in Section 2.

To address the growing need of outlier detection for data streams, some techniques have been proposed that can be grouped into few categories. The first category includes a set of techniques that use a sliding window to capture a recent subset of the data points and apply a traditional outlier detection technique over the sliding window [21], [22], [23], [24], [25]. The second category includes auto-regression based techniques where a data point is identified as an outlier if it is far from the auto-regression based prediction of the data point [26], [27], [28]. The cluster based approach comprises the third category where a data point is identified as an outlier if it cannot be clustered into any existing cluster [29], [30], [31]. The fourth category consists of data mining and statistics based techniques where a data density function is used to approximate the distribution of the data points, and a data point is identified as an outlier if it belongs to a low density region [19], [32], [33], [34]. Some techniques exist that do not belong to any of the aforementioned categories [23], [35], [36], [37]. However, little research has been done to formally identify all the research issues that need to be addressed when designing an outlier detection technique for data streams. This paper aims to fill this gap.

The rest of the paper is organized as follows. Section 2 presents the research issues of outlier detection for single data streams; Section 3 discusses additional research issues that arise when detecting outliers for multiple data streams; and Section 4 concludes the paper.

2. SINGLE DATA STREAMS

2.1 Definition

A Data Stream Ψ is an infinite set of data points, $\Psi = \{D_t | 0 \leq t\}$ where a data point D_t is a set of attribute values with an explicit or implicit timestamp. Formally a data point is $D_t = (\boldsymbol{V}, \tau_t)$ where $\boldsymbol{V}$ is a p-tuple, each value of which corresponds to one attribute, and τ_t is the timestamp for the t-th data point. A data point D_t is an outlier if it is considerably different from other data points $\{D_i | i \neq t\}$.

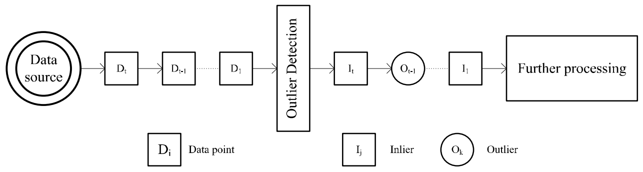

Fig. 1. A Single data stream application with outlier detection

Examples of applications of single data stream include power usage monitoring, meteorological attribute monitoring, stock market monitoring. Fig. 1 shows a single data stream application where a sequence of data points is produced from a single source (double circle in Fig. 1) and the outlier detection component receives one data point at a time and marks the data points as outliers if they are deviated from other data points or inliers otherwise. Those outliers/inliers are continued for further processing.

2.2 Research Issues

Data streams characteristics introduce new issues for outlier detection techniques. In this section, we discuss these issues with respect to those characteristics.

2.2.1 Transient

Data points are transient in a data stream [38], [10]. A particular data point is important for a specific amount of time, after which it is discarded or archived [38], [5], [39], [40]. Therefore it is important to keep the data point moving [41]; otherwise it may lose its importance. However popular outlier detection techniques rely on the store-and-process paradigm [42], [43], [44], where the entire dataset is stored in the first phase to construct an outlier detection model, and each data point is compared to the model or other data points in the second phase to detect outliers among them. These approaches hold the data points for a long period of time and do not detect outliers as they arrive; so for streaming data, these two phase algorithms are inappropriate. A new outlier detection scheme has to be developed that processes data points online.

Research Issue 1- *An outlier detection technique cannot hold the entire dataset indefinitely and compare each data point D_t to the other data points to detect the outlier-ness of D_t; rather the outlier-ness of D_t should be decided immediately once D_t arrives.*

2.2.2 *Notion of Time*

Unlike regular data, stream data include a notion of time. Each data point has a timestamp associated with it. The association can be explicit (where time is a data attribute) or implicit (when the exact time is not important, but the order of data items is important) [38]. The timestamp gives the temporal context for each data point; thus each data point needs to be processed based on its own temporal context. Outlier detection is no exception; by definition, a data point is an outlier if it has a significantly different value compared to other data points; but if we take temporal context into consideration, a data point must be compared to the other data points with the same temporal context (Type II outlier). Typical outlier detection techniques do not consider the temporal context of the data points [44], [42], rather they compare a data point to the entire dataset; this approach is inherently flawed for data streams since the outlier-ness of a data point can only be detected by comparing it with the data points seen so far. A temperature of 100°F may not look like an outlier if we consider the temperature of an entire year, but it would certainly look like an outlier if we consider the temperature of winter days only. In order to detect outliers meaningfully, an appropriate temporal context has to be selected first (if not given by the user) and then, every data point has to be processed based on its temporal context. Moreover an out-of-order data point should be processed based on its temporal context [41] as well.

Research Issue 2- *A data point has to be compared with the other data points with same temporal context (occurred within the time period which is semantically related to the timestamp of the data point).*

2.2.3 *Notion of Infinity*

Data streams are seen as infinite sequences of data points as they keep coming from a data source indefinitely. The most significant implication of the notion of infinity is that at any particular time, the entire dataset is not available, i.e., a random access to the entire dataset is not possible for outlier detection [38]. Many outlier detection techniques store the entire dataset first and find the outliers later [19]. An outlier detection technique for data streams cannot store all the data points seen so far because the number of data points is infinite; rather it should store only the summary of the data points seen so far using finite memory/resource and detect outliers based on the summary. For example, for outlier detection techniques that determine whether a data point D_t is an outlier based on D_t's neighbor data points, in order to compute the neighbors, a data density function should be used instead of relying on the availability of the entire dataset and using the pairwise distances of all the data points in the dataset. On top of this, the data density function has to be computed incrementally. Thus an outlier detection model has to be incremental and cannot assume the availability of the entire dataset.

Research Issue 3- *In order to detect the outlier-ness of a data point D_t, D_t should be compared with the summary of the other data points, instead of directly comparing D_t to the other data points. In addition, the summary should be computed incrementally.*

2.2.4 *Arrival Rate*

Data points are continuously coming from a data source. The arrival rate might be fixed or variable but every application must finish processing before the next data point arrives [43], i.e., if the outlier detection is a binary classification task, then the classification has to be done before the next data point comes. If the outlier detection technique fails to process a data point before the next one arrives, the result is flooding. Typical outlier detection techniques compare each data point to all other data points and detect outliers. If the dataset size is too large, these approaches would require a vast amount of time and may not be able to keep up with the arrival rate. A reasonable accuracy can be achieved if the data point is compared to a much smaller subset and the size of the subset should be decided based on the available processing time. A similar idea is also applicable to outlier detection model construction [36]. Hence the outlier detection time is bounded by the arrival rate of data streams.

Research Issue 4- *The set of data points or the summary of the data points, to which the current data point is compared to detect outlier-ness, should be adjusted based on the available processing time.*

In some data stream applications like sensor networks, the data arrival rate is not fixed [36], but varies over time. In this kind of applications, the available processing time between every two consecutive data points is not the same. If a long period of time is available, the accuracy of outlier detection could be excelled utilizing the time before the next data point arrives; if a short period of time is available, the data point needs to be processed before the next data point arrives which may compromise outlier detection accuracy [36]. Hence the processing has to be adaptive.

Research Issue 5- *In case of dynamic arrival rate, the set of data points or summary of the data points, to which the current data point is compared to detect outlier-ness,* *should be adjusted dynamically based on the available processing time*.

2.2.5 Concept Drift

The data distribution in a data stream changes over time [14]. This might happen because of changes in environments, changes of trends, etc. This phenomenon is known as *concept-drift* [14]. Many outlier detection techniques use data distribution to identify abnormal behavior [19]. Since data distribution for data streams changes over time, outliers detected for one data distribution might not be the same for another data distribution. For example, the distribution of traffic in a traffic monitoring system during the mornings may be entirely different from the distribution during the evenings; therefore, any assumption about data distribution may lead to incorrect results. Many statistical and machine learning based techniques assume fixed data distribution for outlier detection [19], [33], [34]; they use a training data set to construct the outlier detection model and later detect outliers based on the model. The problem with this approach is that the training data set represents a fixed data distribution which may produce meaningful results for some time, but if a concept drift occurs, the same training data set (or the same outlier detection model) may no longer produce meaningful results.

Research Issue 6- *An outlier detection technique for data streams should not assume any kind of fixed data distribution*.

2.2.6 Uncertainty

Data points in a data stream are further characterized by their uncertainty. Data sources such as sensors in a sensor network are exposed to an open environment. They are vulnerable to external events. The unreliability of the data points in a streaming environment is one of the key challenges for working with data streams [44]. Here the general term *uncertain* is used to describe any element that cannot be relied upon with complete confidence; however it has many facets like Uncertainty (the fact is uncertain, i.e., the attribute value cannot be measured with sufficient confidence), Imprecision (the information is not as specific as it should be), Vagueness (including elements that are inherently vague), Inconsistency (more than one mutually exclusive assertion), and Ambiguity (lack of complete semantics) [45]. On the contrary, existing outlier detection techniques assume data points' values to be correct; therefore, the dissimilarity between two data points can be easily measured by distance (Euclidian or Manhattan) or cosine similarity. However, distance or cosine similarity fails to measure the similarity/dissimilarity between two data points if they are uncertain; and thus outlier detection schemes that use such measure of similarity would inherently fail to detect outlier for data streams [41], [44].

Research Issue 7- *An outlier detection technique for data streams* should use *the similarity metrics that can measure the similarity between two uncertain data points*.

Moreover the uncertainty may arise because some data points could be entirely missing or out-of-date in a data stream which is referred to as imperfection by Stonebraker et al. [41]. Data points may arrive late or even entirely fail to arrive in a data streaming environment. An outlier detection technique must process the existing data regardless of the fate of the failed data points. Consider an example where a senor produces one data point every hour and an outlier detection technique that requires previous three hours of data to decide the outlier-ness of the current data point. Now if the previous two hours of data failed to arrive before the current data point, the fate of the current data point must be decided based on whatever data to which the technique has access (i.e. the current data point and the data point that arrived three hours ago since other two data points in the middle are missing). The problem can be worse if a data point arrives out-of-date. In that case the out-of-date data point must be processed based on its own temporal context. To the best of our knowledge, no existing outlier detection technique processes out-of-date data based on their temporal context. Comparing a data point with other data points having different temporal contexts to identify outliers would produce erroneous results. For example, if the two missing data points arrive some time later, the outlier detection technique needs to process them based on the three hours of data points that are supposed to arrive before them.

Research Issue 8- *The outlier-ness of an out-of-order data point should be decided by comparing it with the data points that have the same temporal context as that of the out-of-order data point*.

2.2.7 Multi-dimensionality

Although multi-dimensionality is not a data stream specific issue, it is worth discussing because of its impacts on outlier detection. Measuring the similarity of a data point to other data points in the dataset is a crucial part of outlier detection because an unusual data point must have very few data points that are similar to it in the dataset. Many outlier detection techniques use Euclidian or Manhattan distance to measure the similarity between data points [21], [35]; but Euclidian distance becomes qualitatively meaningless to represent such similarity and causes instability of nearest neighbor for a high number of dimensions [46], [47]. This is because the distance between two similar data points and the distance between two non-similar data points are approximately equal for a high number of dimensions, which in turn makes distance based outlier detection algorithms less effective. Furthermore, many algorithms use data density function, but the multi-dimensional data density space grows exponentially with the number of dimensions; hence data density function cannot be computed easily [48]. Arguably, outlier detection in a multi-dimensional data stream can be seen as outlier detection in a set of single dimensional data streams; but this approach is fundamentally flawed because it handles all dimensions independently and fails to address the correlation among dimensions.

Research Issue 9- *An outlier detection technique for data streams should use the similarity metric that can measure the similarity among the data points with a large number of dimensions*.

A successful outlier detection technique for single data streams should address the aforementioned research issues. In the next section, we investigate additional research issues that are specifically related to multiple data streams.

3. MULTIPLE DATA STREAMS

3.1 Definition

Multiple data streams consist of a set of data streams where each stream produces an infinite sequence of data points, each of which is accompanied with an explicit or implicit time stamp. Outlier detection for a single data stream compares a data point with respect to the history data points in order to detect whether the data point is an outlier. In case of multiple data streams, a data point can be detected as an outlier either by comparing it to the history data points from the same stream or comparing it to the data points from the other streams. The opportunity of having multiple data streams to compare facilitates better accuracy and richer semantics across the data streams.

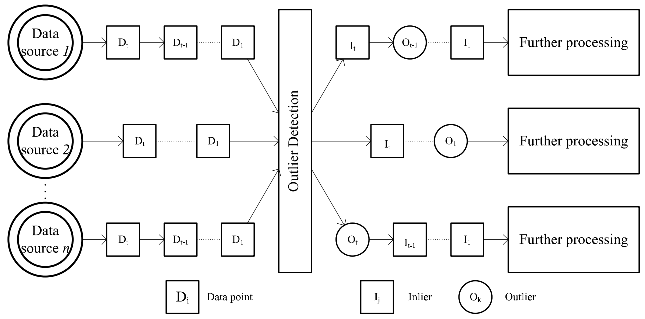

Fig 2. Multiple data streams application with outlier detection

There are many applications that involve multiple data streams. One of such applications is network traffic monitoring where network traffic is monitored to keep track of network usage, detect potential threats, etc. Traffic status at different network devices is continuously monitored; and a data stream is produced from each device that gives birth to a multiple data streams application [49]. Consider a network of two routers, four switches, one server and a couple of hosts. A network traffic monitoring application monitors the entire network and collects different attributes from different network devices, such as number of service requests, average service time, average throughput, etc. from the server, number of incoming packets, number of outgoing packets, etc. from switches, number of successful logins, number of failure logins, number of root accesses, etc. from hosts, and numbers of routing packets and failed packets, type of requests, etc. from routers. Each network device reports one data point encapsulating all necessary attributes and sends it to the monitoring application, which then collects data points from all devices and analyzes the data points for further knowledge. Other multiple data stream applications include large-scale scientific observation, web log analysis, and security surveillance system. A good discussion about these applications can be found in [50].

Figure 2 shows a multiple data stream application that includes outlier detection. Each heterogeneous data source produces a sequence of data points. The outlier detection component receives all data points from different sources and marks a data point as an outlier comparing it to the other data points from the same stream as well as comparing it to the other data points from different streams. The outlier detection component releases the data point once it decides the outlier-ness of the data point asynchronously. Since the outlier detection based on data points from the same stream was discussed earlier in Section 2, in this section we focus our discussion only on outlier detection based on data points from other data streams.

3.2 Research Issues

The research issues for single data streams that we identified earlier in Section 2.2 are also applied to multiple data streams. Therefore in this section we discuss only the additional research issues that exist due to multiple streams specifically.

3.2.1 Cross-correlation

Multiple data streams produce multiple data points with explicit or implicit timestamps. Some outlier detection techniques assume that data points from multiple data streams should be close to one another [23], [35], and a data point is an outlier if it is far from other data points from other streams at any point in time. This definition is too restrictive because in many applications, such as Chlorine monitoring and temperature in the same building, data points from different streams are cross-correlated although their values could be far from one another [51]. A data point is considered to be an outlier if it violates the expected cross-correlation (nonconformist to other values). The temperatures from different cities from different states could be very different from one another but they could be related. In order to detect outliers, the outlier detection technique should find the cross-correlation among the data points from different data streams and compare them to the data points based on their explored/expected cross-correlations.

Research Issue 10- *The outlier detection technique should be capable of detecting outliers that are non-conformist to the other data points with respect to their relationship with other data points*.

3.2.2 Asynchronous Data Points

Data sources in a multiple data streams application may be independent of one another; and thus they may generate data points with different arrival rates. These data points are asynchronous [50]. In order to detect whether a data point is an outlier or not, the outlier detection technique has to choose an appropriate temporal context not only for the same stream but also for other streams. The data point with a predefined temporal context can be compared with other data points with the same temporal context in other streams for outlier detection. Since data points do not arrive synchronously, it is not only difficult to choose an appropriate temporal context from all data streams, but also, at a particular timestamp, some data points from some data streams may be present while those from other data streams may not, and thus might be considered as missing data. To our best knowledge no existing outlier detection technique is designed to tackle this kind of missing data.

Research Issue 11- *The outlier-ness of a data point should be decided as it arrives, minimizing the effect of missing data due to asynchronous processing*.

Furthermore, the asynchronous behavior of multiple data streams hinders synchronous processing. Some outlier detection schemes for data streams assume all data points from multiple data streams to arrive together [35], [52] and then process them together to detect an outlier. However, in practical situations, it is difficult to achieve synchronization for different data sources [53]. Moreover the processing of a data point cannot be delayed and wait for other data points from other data streams to arrive. Therefore the data points from multiple data streams may need to be processed and outliers may need to be detected asynchronously. However, in that case it would be extremely difficult to exploit the cross-correlations among the data points from multiple streams. If an outlier detection technique ignores the cross-correlations completely, it will fail to exploit the advantage of having multiple data streams, and might produce less accurate results.

Research Issue 12- *The outlier detection technique should have the capability of learning cross-correlation among streams and detecting outliers based on the learned cross-correlation asynchronously*.

3.2.3 Dynamic Relationship

The cross-correlations among the data points from multiple data streams may vary over time and this dynamic relationship is due to two phenomena: (1) asynchronous behavior and (2) concept drift. The data point from multiple data streams have temporal correlations which may vary with the varying time differences among the data points. Imagine two temperature sensors are mounted in a close proximity to detect temperatures. One sensor produces one data point every 3 hours and another sensor produces one data point every 5 hours. The time difference between the most recent data points from the two sensors may vary from 0 to 3 hours. In that case the temporal correlations among them also vary over time. However typical outlier detection relies on comparing a data point to its cross-correlated data points; if the relationship changes over time, the cross-correlation among the data points from multiple streams has to be monitored continuously.

Concept drift is the second driving factor for dynamic relationship. If concept drifts occur independently in multiple data streams, the correlations among the data points from multiple streams vary as well. Thus the relationships among the data points become dynamic.

Research Issue 13- *An outlier detection technique for multiple streams should continuously monitor their cross-correlation and compare a data point with other cross-correlated data points only to decide its outlier-ness*.

3.2.4 Heterogeneous Schema

In a multiple data streams application, different data streams might have different schemas [50]. Comparing multiple data points with different schemas is a complicated problem and the definition of outlier is even vaguer in that case. By definition, a data point is an outlier if it is a non-conformist compared to other data points, that is, if it has a value considerably different from those of other data points. However, if we consider a heterogeneous set of data points where all data points have different attributes, it becomes intrinsically difficult to identify which data point is a non-conformist. Imagine two data streams with one producing temperature and another producing humidity of any location. The direct comparison of temperature to humidity does not make sense and, hence, neither of them can be detected as an outlier on the basis of the other. Nonetheless, it is perceivable that temperature and humidity might have a correlation between them and the observed values of temperature and humidity might violate the predefined or previously traced/predefined correlation and, therefore, one of them is an outlier. Although intuitively this kind of heterogeneous comparison may produce meaningful outliers, it requires a new definition of outliers unlike what we have seen before.

Research Issue 14- *An outlier detection technique for multiple data streams should be able to compare data points with the same or different schemas in order to detect outliers*.

4. CONCLUSIONS

Data streams have received a great deal of attention in the last decade and several attempts have been made to solve the outlier detection problem for data streams. However there exists no work that presents a comprehensive set of research issues for detecting outliers in single stream and multiple stream applications. In this paper, we have identified and discussed those issues in depth.

5. REFERENCES

[1] D. J. Abadi, D. P. Carney, U. Cetintemel, M. F. Cherniack, C. Convey, C. Erwin, E. Galvez, M. Hatoun, A. S. Maskey, A. Rasin, A. Singer, M. R. Stonebraker, N. Tatbul, Y. Xing, R. Yan

and S. B. Zdonik, "Aurora: A Data Stream Management System," SIGMOD, 2003.

[2] R. Motwani, J. Widom, A. Arasu, B. Babcock, S. Babu, M. Datar, G. Manku, C. Olston, J. Rosenstein and R. Varma, "Query Processing, Resource Management, and Approximation ina Data Stream Management System," Stanford InfoLab, 2002.

[3] D. J. Abadi, Y. Ahmad, M. Balazinska, U. ?etintemel, M. Cherniack, J.-H. Hwang, W. Lindner, A. Maskey, A. Rasin, E. Ryvkina, N. Tatbul, Y. Xing and S. B. Zdonik, "The Design of the Borealis Stream Processing Engine," Innovative Data Systems Research , 2005.

[4] L. Gurgen, C. Roncancio, C. Labbé, A. Bottaro and V. Olive, "SStreaMWare: A Service Oriented Middleware for Heterogeneous Sensor Data Management," Pervasive services, 2008.

[5] A. Arasu, S. Babu and J. Widom, "An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations," dbpubs.stanford.edu:8090/pub/2002-57, Palo Alto, 2002.

[6] A. Dobra, M. Garofalakis, J. Gehrke and R. Rastogi, "Sketch-Based Multi-query Processing over Data Streams," Advances in Database Technology, Lecture Notes in Computer Science, p. 551 – 568, 2004.

[7] J.-H. Hwang, S. Cha, U. Cetintemel and S. Zdonik, "Borealis-R: A Replication-Transparent Stream Processing System for Wide-Area Monitoring Applications," SIGMOD, 2008.

[8] J. Kr?mer and B. Seeger, "Semantics and Implementation of Continuous Sliding Window Queries Over Data Streams," ACM Transactions on Database Systems, 2009.

[9] M. Cammert, J. Kramer, B. Seeger and S. Vaupel, "A Cost-Based Approach to Adaptive Resource Management in Data Stream Systems," In IEEE Transactions on Knowledge and Data Engineering, pp. 230-245, 2008.

[10] D. Carney, U. Setintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul and S. Zdonik, "Monitoring Streams: A New Class of Data Management Applications," VLDB, 2002.

[11] C. Heinz, J. Kramer, T. Riemenschneider and B. Seeger, "Toward Simulation-Based Optimization in Data Stream Management Systems," ICDE, 2008.

[12] B. Gedik, H. Andrade, K.-L. Wu, P. S. Yu and M. Doo, "SPADE: The System S Declarative Stream Processing Engine," SIGMOD, 2008.

[13] I. Botan, G. Alonso, P. M. Fischer, D. Kossmann and N. Tatbul, "Flexible and Scalable Storage Management for Data-intensive Stream Processing," EDBT, 2009.

[14] N. Jiang and L. Gruenwald, "Research Issues in Data Stream Association Rule Mining," ACM SIGMOD Record, pp. 14 - 19, 2006.

[15] B. Mozafari, H. Thakkar and C. Zaniolo, "Verifying and Mining Frequent Patterns from Large Windows over Data Streams," ICDE, 2008.

[16] Y. Kim and U. Kim, "WSFI-Mine: Mining Frequent Patterns in Data Streams," Advances in Neural Networks, Lecture Notes in Computer Science, pp. 845-852, 2009.

[17] V. Chandola, A. Banerjee and V. Kumar, "Anomaly Detection: A Survey," ACM Computing Surveys, vol. 41, pp. 1-58, July 2009.

[18] V. Hodge and J. Austin, "A Survey of Outlier Detection Methodologies," Artificial Intelligence Review, vol. 22, pp. 85-126, October 2004.

[19] V. Barnett and T. Lewis, Outliers in Statistical Data, New York: John Wiley & Sons, Inc.,, 1994.

[20] B. Z. J. L. Naoki Abe, "Outlier Detection by Active Learning," SIGKDD, 2006.

[21] F. Angiulli and F. Fassetti, "Distance-based outlier queries in data streams: the novel task and algorithms," Data Mining and Knowledge Discovery, vol. 20, no. 2, pp. 290-324, 2010.

[22] S. Basu and M. Meckesheimer, "Automatic Outlier Detection for Time Series: An Application to Sensor Data," Knowledge and Information System, pp. 137-154, 2006.

[23] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki and D. Gunopulos, "Online Outlier Detection in Sensor Data Using Non-parametric Models," VLDB, 2006.

[24] L. Kuncheva, "Change Detection in Streaming Multivariate Data Using Likelihood Detectors," IEEE Transactions on Knowledge and Data Engineering, vol. 99, p. PrePrints, 2011.

[25] X. Lu, T. Yang, Z. Liao, M. Elahi, W. Liu and H. Wang, "Incremental outlier detection in data streams using local correlation integral," SAC, 2009.

[26] D.-I. Curiac, O. Banias, F. Dragan, C. Volosencu and O. Dranga, "Malicious Node Detection in Wireless Sensor Networks Using an Autoregression Technique," ICNS, 2007.

[27] V. Puttagunta and K. Kalpakis, "Adaptive Methods for Activity Monitoring of Streaming Data," ICMLA, 2002.

[28] M. Shuai, K. Xie, G. Chen, X. Ma and G. Song, "A Kalman Filter Based Approach for Outlier Detection in Sensor Networks," CSSE, 2008.

[29] S. Ando and E. Suzuki, "Detection of Unique Temporal Segments by Information Theoretic Meta-clustering," SIGKDD, 2009.

[30] K. Sequeira and M. Zaki, "ADMIT: Anomaly-based Data Mining for Intrusions," SIGKDD, 2002.

[31] G. Sheikholeslami, S. Chatterjee and A. Zhang, "WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases," VLDB, 1998.

[32] S. Sadik and L. Gruenwald, "DBOD-DS: Distance Based Outlier Detection for Data Streams," DEXA, 2010.

[33] E. Eskin, "Anomaly Detection over Noisy Data using Learned Probability Distributions," ICML, 2000.

[34] D. Agarwal, "An Empirical Bayes Approach to Detect Anomalies in Dynamic Multidimensional Arrays," ICDM, 2005.

[35] K. Ishida and H. Kitagawa, "Detecting Current Outliers: Continuous Outlier Detection over Time-Series Data Streams," in Lecture Notes in Computer Science, Berlin , Springer Berlin / Heidelberg, 2008, pp. 255-268.

[36] I. Assent, P. Kranen, C. Baldauf and T. Seidl, "AnyOut: Anytime Outlier Detection on Streaming Data," in Database Systems for Advanced Applications, vol. 7238, S. Lee, Z. Peng, X. Zhou, Y. Moon, R. Unland and J. Yoo, Eds., Springer Berlin / Heidelberg, 2012, pp. 228-242.

[37] S. Sadik and L. Gruenwald, "Online outlier detection for data streams," IDEAS, 2011.

[38] B. Babcock, S. Babu, M. Datar, R. Motwani and J. Widom, "Models and Issues in Data Stream Systems," PODS, 2002.

[39] N. A. Chaudhry, "Introduction to Stream Data Management," in Stream Data Management, Advances in Database Systems, Springer, 2006, p. 1 – 13.

[40] W. Lindner and J. Meier, "Towards a Secure Data Stream Management System," Lecture Notes in Computer Science, pp. 114-128, 2006.

[41] M. Stonebraker, U. ?etintemel and S. Zdonik, "The 8 Requirements of Real-time Stream Processing," ACM SIGMOD Record, pp. 42 - 47, 2005.

[42] H. Chok and L. Gruenwald, "An online Spatio-Temporal Association Rule Mining Framework for Analyzing and Estimating Sensor Data," IDEAS, 2009.

[43] B. Liu, Classifying Data Streams Using a Concept Drifting Indicator, Norman: University of Oklahoma Thesis Library, 2006.

[44] N. Tatbul, "Streaming Data Integration: Challenges and Opportunities," IEEE ICDE International Workshop on New Trends in Information Integration, 2010.

[45] A. Motro, "Management of Uncertainty in Database Systems," in Modern Database Systems: The Object Model, Interoperability, and Beyond, New York, ACM Press/Addison-Wesley Publishing Co., 1995, pp. 457 - 476 .

[46] K. Beyer, J. Goldstein, R. Ramakrishnan and U. Shaft, "When Is "Nearest Neighbor" Meaningful?," ICDT, 1999.

[47] C. Aggarwal, A. Hinneburg and D. Keim, "On the Surprising Behavior of Distance Metrics in High Dimensional Spaces," ICDT, 2001.

[48] D. W. Scott, Multivariate Density Estimation: Theory, Practice, and Visualization, New York: John Wiley & Sons Inc., 1992.

[49] S. Stolfo, W. Fan, W. Lee, A. Prodromidis and P. Chan, "Cost-based modeling for fraud and intrusion detection: results from the JAM project," in DARPA Information Survivability Conference and Exposition, 2000.

[50] W. Wu and L. Gruenwald, "Research issues in mining multiple data streams," International Workshop on Novel Data, 2010.

[51] S. Papadimitriou, J. Sun and C. Faloutsos, "Streaming pattern discovery in multiple time-series," VLDB, 2005.

[52] C. Franke and M. Gertz, "ORDEN: Outlier Region Detection and Exploration in Sensor Networks," SIGMOD, 2009.

[53] G. Barrenetxea, F. Ingelrest, G. Schaefer and M. Vetterli, "The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments," SenSys, 2008.

[54] L. Golab and T. M. Ozsu, "Issues in Data Stream Management," ACM SIGMOD Record, pp. 5 - 14, 2003.