Distributed Deviation Detection in Sensor Networks

Themistoklis Palpanas Dimitris Papadopoulos Vana Kalogeraki Dimitrios Gunopulos
Department of Computer Science
University of California, Riverside
Riverside, CA 92521
{themis, dimitris, vana, dg}@cs.ucr.edu

Abstract

Sensor networks have recently attracted much attention, because of their potential applications in a number of different settings. The sensors can be deployed in large numbers in wide geographical areas, and can be used to monitor physical phenomena, or to detect certain events.

An interesting problem which has not been adequately addressed so far is that of distributed online deviation detection in streaming data. The identification of deviating values provides an efficient way to focus on the interesting events in the sensor network.

In this work, we propose a technique for online deviation detection in streaming data. We discuss how these techniques can operate efficiently in the distributed environment of a sensor network, and discuss the tradeoffs that arise in this setting. Our techniques process as much of the data as possible in a decentralized fashion, so as to avoid unnecessary communication and computational effort.

1 Introduction

Advances in processor technologies and wireless communications have enabled the deployment of small, low cost and power efficient sensor nodes in both civil and military settings [WLLP01, SS02, OC02, IGE00, IEGH02, SAA03, LBA$^+$02]. The sensors can be deployed in large numbers in wide geographical areas and can be used to monitor, detect and report time-critical events (like earthquakes, chemical spills, or the position and trajectory of moving objects) such that the urgency of the situation is evaluated and distributed efforts are coordinated in a timely manner.

Consider for example a disaster recovery situation where there has been a chemical spill in a given region. Before attempting to control the disaster, we must first find the extent of the current contamination and figure out how the concentration is changing over time so as to estimate the possible reach of the contamination. This requires that the data is collected online from multiple sensors and analyzed dynamically in order to evaluate the accuracy of the threat and respond in real-time.

We propose a technique for distributed deviation, or outlier, detection in real-time streaming data. That is, we are interested in finding those values that deviate significantly from the norm. This problem is especially important in the sensor network setting because it can be used to identify faulty sensors, and to filter spurious reports from different sensors. Even if we are certain of the quality of measurements reported by the sensors, the identification of outliers is still a valuable process. It provides an efficient way to focus on the interesting events in the sensor network.

The context of the sensor networks makes the problem more challenging. First, sensors have limited resource capabilities, including limited battery lifetime, communication bandwidth, CPU capacity and are subject to frequent disconnections. Second, data coming from many different streams have to be examined dynamically and combined to detect deviations. In such an environment, we need to minimize the processing and communication overhead of the sensors. The goal is to process as much of the data as possible in a decentralized fashion, so as to avoid unnecessary communication and computation effort. Identifying deviations is itself a first step towards reducing the communication cost, and prolonging the life span of the sensors. This is true, because there is no need for the sensors to transmit values that follow some general trends already known. What we are really interested in are values that do not follow our models for the data-generating processes.

In this work, we propose the use of kernel density estimators for online deviation detection in streaming data. We describe how these techniques can operate efficiently in the distributed environment of a sensor network, and elaborate on the tradeoffs that arise in this setting. Finally, we discuss techniques for timeliness guarantees on the delivery of the streaming data.

The rest of the paper is organized as follows. Section 2 describes the overall architecture of the system. In Section 3 we elaborate on the problem we are trying to solve. We present the techniques we use for the solution of the problem, and we discuss the properties and functionality of our approach. Section 4 provides some background on the related work, and we conclude in Section 5.

2 Proposed Architecture

We envision a sensor network consisting of a large number of cheap, static sensors. These sensors are limited in terms of their processing and transmission power capabilities, but can be deployed quickly to cover a geographical area.

While the traditional settings assume that the sensor data are gathered in a centralized location and analyzed offline, in our model we assume the existence of additional more powerful and sophisticated nodes that are deployed in small numbers (typically orders of magnitude less than the number of low capacity sensors). We assume that these nodes have large communication range and can communicate with each other using a separate frequency channel so as not to interfere with the sensor communications. These are used to perform more powerful computations, such as the detection of outliers. Figure 1 shows an example of this kind of network. The white circles in the figure denote the low capacity sensors, and the black circles the high capacity ones.

In addition to the usual connections among the low capacity sensors, there is another type of connections involving the high capacity sensors. Taking advantage of the limited number of the more powerful sensors, we can assign entire groups of the low capacity sensors to them. An obvious way of making this assignment would be based on spatial proximity. Hence, different parts of the area covered by the deployed sensors are assigned to different groups, and consequently, to different high capacity sensors. Alternatively, the low capacity sensors may be organized into groups in response to some query, so as to provide the best possible answers. In either case, the structure of the groups is not statically assigned, but is rather dynamic in nature.

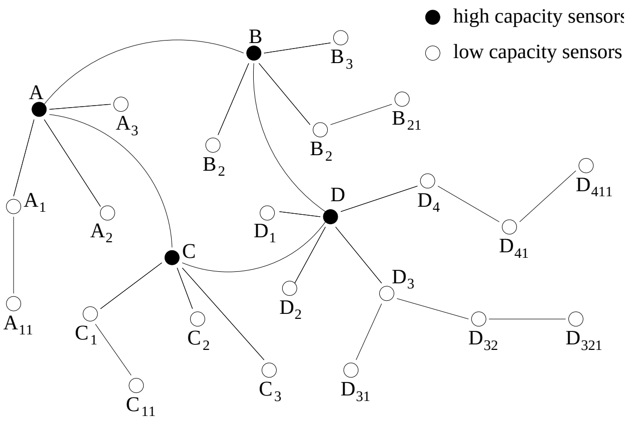

Figure 1: **Example of a sensor network with two different types of sensors.**

Moreover, we may assume that the high capacity sensors are organized in groups as well. These sensors report to operation centers, which can be considered as an additional level in the communication structure.

3 Defining Abnormal Behaviour

An outlier is "an observation that appears to deviate markedly from other members of the sample in which it occurs" [BL94]. This fact may raise suspicions that the specific observation was generated by a different mechanism than the rest of the data. This mechanism may be an erroneous procedure of data measurement and collection, or an inherent variability in the domain of the data under inspection. Nevertheless, in both cases such observations are interesting, and the analyst would like to know about them.

In our case, the goal is to be able to report outlying values coming from unknown generative processes (i.e., the data distribution is not known). As the data values come in, we build a model of these values. In essence, we try to approximate the distribution of the data. Then, we term outliers the values that deviate significantly from that model.

3.1 Outliers in Sensor Networks

We examine the problem of identifying outliers in a sensor network in two stages. First, we discuss the technique we use to model the data distribution. Second, we focus on the problem of managing and combining these models in the network of sensors.

We subsequently discuss the issue of timeliness guarantees on the delivery of the streaming data. Finally, we identify and analyze some tradeoffs that arise in our framework.

3.1.1 Estimation Model

The model we use to estimate the distribution of the values generated by the sensors is based on *kernel density estimators*[1] [Sco92]. For simplicity, we only discuss the one dimensional case. Yet, this approach is easily extended to higher dimensionalities. Assume that we have a static relation, T, that stores the values, t, whose distribution we want to approximate. The recorded values must fall in the interval $[0, 1]$. This requirement is not restrictive, since we can map the operational range of the sensors to the interval $[0, 1]$. Let S be a random sample of T, and $k(x)$ a function, such that $\int_{[0,1]} k(x)dx = 1$, for all tuples in S. We call $k(x)$ the *kernel function*. Then, we can approximate the underlying distribution $f(x)$, according to which the values in T were generated, using the following function

$$f(x) = \frac{1}{n} \sum_{t_i \in S} k(x - t_i).$$

The choice of the kernel function is not significant for the results of the approximation [Cre93]. Hence, we choose the Epanechnikov kernel that is easy to integrate:

$$k(x) = \begin{cases} \frac{3}{4}\frac{1}{B}\left(1 - (\frac{x}{B})^2\right) & , |\frac{x}{B}| < 1 \\ 0 & \text{, otherwise} \end{cases}$$

where B is the bandwidth of the kernel function. In order to set B, we use Scott's rule [Sco92]: $B = \sqrt{5}s|S|^{-\frac{1}{5}}$, where s is the standard deviation of the values in T.

Given a value t_o and the density distribution function $f(x)$, we can estimate the number of values that are in the neighborhood of t_o. This allows us to identify distance based outliers [KN98]. In particular, we have to check the number of values, $N(t_o, r)$, in T that fall in a sphere of radius r around t_o [KGKB03]. If this number is less than an application-specific threshold p then t_o is an outlier. The equation for the computation of $N(t_o, r)$ is as follows

$$N(t_o, r) = \int_{\text{sphere of radius } r \text{ around } t_o} f(x)dx.$$

We should add that the kernel density estimation technique can also be employed in the presence of multivariate distributions, and has been shown to approximate unknown data distributions efficiently and effectively [GKTD00].

In the sensor network environment we require that each sensor maintains a model for the distribution of values it generates. Since we are not interested in the entire history of the values produced by the sensors, it suffices to consider the values in a sliding window of size N. Then, T holds only the values in the sliding window, that is, the N most recent values.

Note that in this case we cannot directly apply the above analysis, since T is continuously changing as a function of time. Therefore, we adapt our technique as follows. The set S is once again a random sample of T, but it is now computed as a sample of a sliding window. The other quantity we need for the estimation of the data distribution is the standard deviation s of the values in the sliding window (used for the computation of the bandwidth, B). Both the above operations can be efficiently supported in a data streaming environment with sliding windows [BDM02, BDMO03].

3.1.2 Distributed Deviation Detection

In the previous section we described the model we use to estimate the data distributions, and explained how we can use this model to identify outliers among the values reported by a sensor. We will now discuss the issues that arise when we consider the setting of a network with a large number of sensors.

Assume we have a sensor network similar to the one shown in Figure 1. The presence of low and high capacity sensors offers a multiresolution view to the deviation detection problem. When we are at the lowest level (i.e., low capacity sensors), we identify local outliers. As we move up (i.e., high capacity sensors), we attain a more

[1] Other model estimation techniques can be used in our framework as well.

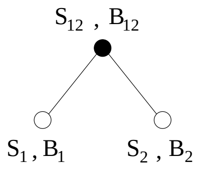

Figure 2: **Example of model composition.**

global perspective, and the outliers we report are with respect to all the sensors of a particular area.

In order for this setting to work, we have in place a mechanism for model composition. This allows us to take the data distribution models of two different sensors in the network and come up with a single model that describes the behaviour of the data of both sensors. We then assign this combined model to another sensor, e.g., a sensor of high capacity (see Figure 2).

In the case of kernel density estimators, there are two quantities that we have to take care of. Namely, the sample set, S, and the bandwidth of the kernel function, B (refer to Section 3.1.1). We can combine the sample sets just by taking their union. This is possible because both samples are uniform. We may then reduce the size of the resulting set by resampling, if necessary. In order to combine the bandwidths of two kernel functions, we only need to combine the two standard deviations upon which the bandwidths depend. This is accomplished using the same techniques as the ones for computing the standard deviation in a sliding window of streaming data.

The above process gives to the high capacity sensors a coarse view of the sensor network, where the details specific to different parts of the deployment area have been masked away. If we wish to recover these details, we have to query the low capacity sensors directly.

3.1.3 Real-Time Delivery of Sensor Data

One important issue in distributed online deviation detection, is how to deliver the streaming data to remote sensors in a timely manner. This is important because if the high capacity sensors fail to collect and analyze current data generated by the sensors, this may lead to inconsistencies in identifying an outlier and determining the criticality of a situation. In such circumstances, we need distributed real-time scheduling algorithms to provide timeliness guarantees.

Streaming data may need to be delivered in sequence, or in parallel, on one or more sensors (e.g., a report delivered from one sensor to another until it reaches a high capacity sensor). These are delivered through the generation of *distributed events*. With each distributed event we associate the following timing parameters:

- *Deadline:* The relative time after the initiation of the event, within which the event must be delivered to the destination.

- *Importance:* a parameter that represents the relative priority of the events. A low importance event may need to be delayed to meet the deadline of a high importance event.

The degree to which end-to-end timeliness is achieved in a sensor network is a combination of two factors; the relative importance of the event (e.g., high urgency or low urgency) and its deadline. The deadline essentially represents a measure of priority for the event; the higher priority goes to the event with the smallest deadline.

3.1.4 Tradeoffs

The framework we describe provides a flexible means for outlier detection in sensor networks. Yet, it also raises some questions that need to be explored in more detail.

The kernel density estimation model that we propose is capable of adjusting itself to the input data distribution, as this distribution changes over time. Both the sample and the standard deviation of the data values, needed for the estimation of the underlying distribution, are being constantly adjusted, following the way the values in the sliding window change. However, we also need a self-correcting mechanism, able of fine-tuning other parameters of the model, e.g., the size of the sample, in case the distribution of the data changes drastically.

We also need to be able to quantify the accuracy of the combined models. That is, we need to know how much information we lose just by combining two different models into one. This process will provide quality guarantees for the generated results.

Finally, once merged, the accuracy of the combined models will depend greatly on the frequency of updates of their parameters from the underlying models. This is a question of how much and how often we should propagate information from the low capacity sensors to the high capacity ones, in order to keep the models at all levels synchronized with the changes in the data distributions.

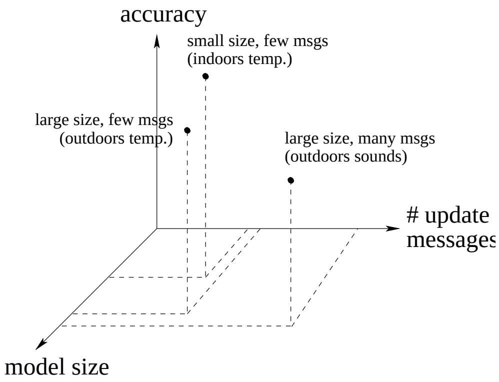

Figure 3: **The tradeoffs space.**

The common denominator to all the above issues is the introduction of the right set of statistics. These statistics can give an indication of the accuracy of the models we employ for approximating the data distributions, and subsequently of the way that the parameters of the models should be altered. They can also help to control the tradeoff between the desired accuracy of results and the performance penalty. This tradeoff is expressed in terms of memory space and number of messages exchanged in the network, and quality of results. The goal is to satisfy the application accuracy requirements while maintaining the resource consumptions to a minimum.

In general, we expect to achieve higher accuracy in the results when we increase the size of the models and the frequency with which we update these models. Nevertheless, in several cases, the requirements are dictated by the applications. Assume, for example, that we are interested in identifying outliers in a sensor network that monitors temperature inside a building. Then, small sized models, updated with modest frequency, are adequate to provide qualitative answers. If we perform the same task in a desert, where there are big temperature variations (e.g., day vs. night and shadow vs. sunlight), we need more sophisticated models to achieve the same degree of accuracy. Finally, if instead of temperature we monitor sounds, then we may need to update our models more frequently, in order to adapt to a fast changing environment. The above tradeoffs are graphically depicted in Figure 3.

4 Related Work

There has been much work in the areas of sensor networks, outliers, and streaming data, but no work that lies in the intersection of these areas.

Madden and Franklin [MF02] present a framework for the efficient execution of queries in a sensor network. The problem of evaluating aggregate operators in a sensor network is addressed by Madden et al. [MFH02]. The authors present a taxonomy of various aggregate operators, and propose techniques for efficient, distributed execution of these operators in the network. A subsequent study [HHMS03] presents applications of the above framework, namely, topographic mapping, wavelet-based compression, and vehicle tracking. Yao and Gehrke [YG03] investigate the problem of query processing in sensor networks as well. They present schemes for in-network aggregation similar to the previous studies, and crash recovery techniques. Bulut et al. [BSV03] describe a scalable, distributed indexing architecture for streaming data.

There is extensive literature in the statistics community regarding outlier detection [BL94]. Several algorithms have been proposed for the problem of finding deviations in large datasets [KN98] [RRS00] [BKNS00] [AAR96] [SAM98] [PK01]. However, none of the above approaches is directly applicable to a streaming data environment.

Several studies have proposed new algorithms for solving traditional database problems in the data streaming context. A decision tree classification algorithm for streaming data was presented by Hulten et al. [HSD01]. Two additional algorithms for streaming data, one for computing correlated aggregates [GKS01] and another for answering aggregate queries approximately [GKMS01], have been described in the literature. Two recent studies [CDIM02] [DM02] present solutions to specific problems in data stream management, with approximation guarantees. The first one finds the number of distinct items in a single data stream, and the number of unequal item counts in a combination of two streams. The second study estimates the rarity of data items, and the similarity of two data streams. A related problem, is the one of identifying correlations in streaming data [GGK03]. This work proposes techniques based on singular value decomposition for capturing correlations between multiple streams. Finally, Palpanas et al. [PVK+04] describe algorithms for online approximation of streaming data, using user-defined time-decaying functions to specify the accuracy of the approximation.

5 Conclusions

During the recent years, sensor networks have become increasingly popular. Recent efforts have been devoted in this area, demonstrating their functionality and wide applicability.

In this paper, we address the problem of deviation detection in the environment of sensor networks, which has not been studied in the literature. We describe a technique for online identification of outliers in a stream of data, such as the one produced by a sensor. We then discuss how to extend this technique to an entire network of sensors, taking into consideration the distributed processing of events, as well as the need for response time guarantees in the delivery of the sensor data.

References

[AAR96] Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. A Linear Method for Deviation Detection in Large Databases. In *International Conference on Knowledge Discovery and Data Mining*, pages 164–169, Portland, OR, USA, August 1996.

[BDM02] Brian Babcock, Mayur Datar, and Rajeev Motwani. Sampling From a Moving Window Over Streaming Data. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 633–634, San Francisco, CA, USA, January 2002.

[BDMO03] Brian Babcock, Mayur Datar, Rajeev Motwani, and Liadan O'Callaghan. Maintaining Variance And k-medians Over Data Stream Windows. In *ACM PODS International Conference*, pages 234–243, San Diego, CA, USA, June 2003.

[BKNS00] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. In *ACM SIGMOD International Conference*, pages 21–32, Dallas, TX, USA, May 2000.

[BL94] V. Barnet and T. Lewis. *Outliers in Statistical Data*. John Wiley and Sons, Inc., 1994.

[BSV03] Ahmet Bulut, Ambuj K. Singh, and Roman Vitenberg. An Adaptive and Scalable Middleware for Distributed Indexing of Data Streams. In *International Workshop on Databases, Information Systems and Peer-to-Peer Computing*, Berlin, Germany, September 2003.

[CDIM02] Graham Cormode, Mayur Datar, Piotr Indyk, and S. Muthukrishnan. Comparing Data Streams Using Hamming Norms (How to Zero In). In *VLDB International Conference*, Hong Kong, China, August 2002.

[Cre93] N. A. C. Cressie. *Statistics for Spatial Data*. Wiley & Sons, 1993.

[DM02] Mayur Datar and S. Muthukrishnan. Estimating Rarity and Similarity over Data Stream Windows. In *Annual European Symposium (ESA)*, pages 323–334, Rome, Italy, September 2002.

[GGK03] Sudipto Guha, Dimitrios Gunopulos, and Nick Koudas. Correlating Synchronous and Asynchronous Data Streams. In *International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 2003.

[GKMS01] Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, and Martin Strauss. Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries. In *VLDB International Conference*, pages 79–88, Rome, Italy, sep 2001.

[GKS01] Johannes Gehrke, Flip Korn, and Divesh Srivastava. On Computing Correlated Aggregates Over Continual Data Streams. In *ACM SIGMOD International Conference*, pages 13–24, Santa Barbara, CA, USA, may 2001.

[GKTD00] Dimitrios Gunopulos, George Kollios, Vassilis J. Tsotras, and Carlotta Domeniconi. Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. In *ACM SIGMOD International Conference*, pages 463–474, Dallas, TX, USA, May 2000.

[HHMS03] Joseph M. Hellerstein, Wei Hong, Samuel Madden, and Kyle Stanek. Beyond Average: Toward Sophisticated Sensing with Queries. In *International Workshop on Information Processing in Sensor Networks*, Palo Alto, CA, USA, April 2003.

[HSD01] Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining Time-Changing Data Streams. In *International Conference on Knowledge Discovery and Data Mining*, pages 71–80, San Fransisco, CA, USA, aug 2001.

[IEGH02] Chalermek Intanagonwiwat, Deborah Estrin, Ramesh Govindan, and John Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *Proceedings of ICDCS*, 2002.

[IGE00] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of ACM MOBICOM*, 2000.

[KGKB03] George Kollios, Dimitrios Gunopulos, Nick Koudas, and Stefan Berchtold. Efficient Biased Sampling for Approximate Clustering and Outlier Detection in Large Datasets. 15(5):1170–1187, 2003.

[KN98] Edwin M. Knorr and Raymond T. Ng. Algorithms for Mining Distance-Based Outliers in Large Datasets. In *VLDB International Conference*, pages 392–403, New York, NY, USA, August 1998.

[LBA+02] C. Lu, B. Blum, T. Abdelzaher, J. Stankovic, and T. He. Rap: A real-time communication architecture for large-scale wireless sensor networks. In *Proceedings of the Real-Time Technology and Applications Symposium*, San Jose, CA, September 2002.

[MF02] Samuel Madden and Michael J. Franklin. Fjording the Stream: An Architecture for Queries Over Streaming Sensor Data. In *International Conference on Data Engineering*, San Jose, CA, USA, February 2002.

[MFH02] Samuel Madden, Michael J. Franklin, and Joseph M. Hellerstein. TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *Symposium on Operating Systems Design and Implementation*, Boston, MA, USA, December 2002.

[OC02] C. Okino and M. Corr. Statistically accurate sensor networking. *IEEE WCNC*, 2002.

[PK01] Themistoklis Palpanas and Nick Koudas. Entropy Based Approximate Querying and Exploration of Datacubes. In *International Conference on Scientific and Statistical Database Management*, pages 81–90, Fairfax, VA, USA, July 2001.

[PVK+04] Themistoklis Palpanas, Michail Vlachos, Eamonn Keogh, Dimitrios Gunopulos, and Wagner Truppel. Online Amnesic Approximation of Streaming Time Series. In *International Conference on Data Engineering*, Boston, MA, USA, March 2004.

[RRS00] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient Algorithms for Mining Ouliers from Large Data Sets. In *ACM SIGMOD International Conference*, pages 427–438, Dallas, TX, USA, May 2000.

[SAA03] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz. Esrt: Event to sink reliable transport protocol in wireless sensor networks. *ACM MOBIHOC*, 2003.

[SAM98] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. Discovery-driven Exploration of OLAP Data Cubes. In *International Conference on Extending Database Technology*, pages 168–182, Valencia, Spain, March 1998.

[Sco92] D. Scott. *Multivariate Density Estimation: Theory, Practice and Visualization*. Wiley & Sons, 1992.

[SS02] A. Savvides and M. B. Srivastava. A distributed computation platform for wireless embedded sensing. In *Proceedings of ICCD 2002*, Freiburg, Germany, September 2002.

[WLLP01] B. Warneke, M. Last, B. Liebowitz, and K. Pister. Smart dust: Communicating with a cubic-millimeter computer. *IEEE Computer Magazine*, pages 44–51, January 2001.

[YG03] Yong Yao and Johannes Gehrke. Query Processing for Sensor Networks. In *Conference on Innovative Data Systems Research*, Asilomar, CA, USA, January 2003.