数理解析研究所講究録 1362 巻 2004 年 124-132

Data Stream Mining: selected tools & algorithms

日本アイビーエム株式会社　東京基礎研究所　小林　メイ

Mei Kobayashi, *mei@jp.ibm.com*

IBM Research, Tokyo Research Laboratory

1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242-8502 Japan

Abstract: Data stream mining, i.e., the on-line processing and analysis of rapidly and continuously arriving large volumes of data, has emerged as an exciting new area of research. Application domains include financial transactions, sensor network measurements, and telecommunication networks. A new generation of mining algorithms are needed for real-time analysis and query response for these applications since most conventional data mining algorithms can only be applied to static data sets that may be updated periodically in large chunks, but not to continuous streams of data. We examine how some classical tools from statistical and signal analysis have been modified and enhanced to solve real world problems involving data streams.

1. Introduction

A *data stream* is a sequence of points that can be read only once, in a fixed order and rate that is normally determined by a system, not by the recipient user. The problem of processing tends of thousands of time series data streams appears in numerous application domains, from financial transactions to sensor network measurements, telecommunication networks, manufacturing systems, and scientific data monitoring systems [1, 2, 9]. Analysis of information in data streams poses some different challenges than those found in classical data mining. Some features that characterize algorithms for stream mining are: fast, real-time processing; means for coping with massive volumes of (possibly infinite) data; a single pass for examination and analysis of stream elements; a system-determined (not user-determined) order of stream elements; and limited availability of memory for analysis and synopsis storage.

Research involving data streams has been conducted from several different perspectives. Some examples are: stream data management systems and their system architecture, stream query models, computer languages for facilitating fast query response, and fast (approximate) algorithms. At least four broad areas of research have emerged from this last perspective –

algorithms. They are: clustering and summarization; classification of stream data; computation and analysis of statistical parameters associated with data streams; and wavelet-based synopsis generation and analysis. In the past few years the volume of published work on stream mining has increased so rapidly that it would be impossible to conduct a comprehensive survey of even a few of the major topics in the field. This paper examines some selected works that have made significant progress towards solving real world problems from the last two perspectives associated with data streams mentioned above, namely statistical parameter and wavelet-based synopses computation and analysis.

The remainder of this paper is organized as follows. In the next section we review some classical tools from statistical analysis and signal analysis. In the following section we review selected examples in which these tools have been modified and enhanced to solve real world problems involving data streams.

2. Models and Tools for Stream Analysis

Algorithms for stream mining rely on several tools from statistical analysis. They include: modeling of the stream as an ordered or unordered sequence; modeling of the stream as *individual items* without pre-processing or as pre-processed *aggregates*; physical- and time-based windowing of streams to facilitate analysis; and approximation and monitoring of classical statistical parameters. Some statistical parameters of interest in typical applications are: averages, standard deviations, best fit slopes, cross-correlation and auto-correlation coefficients, and the sensitivity of values in a stream to values in another stream. In this section we review some statistical tools that are particularly useful in data stream analysis techniques that have been developed to date.

Histograms

Histograms are one of the more frequently used tools in the statistical analysis of data. The main idea in histogram analysis is to:

- partition a given data distribution according to some rule;
- place the partitioned data into separate entitites known as *buckets*; and then
- compute the frequencies, means, standard deviations or other characteristics of data for each bucket.

For one-dimensional problems, histograms are usually constructed to be *equi-width* or *equi-depth* with respect to a user-specified attribute. Sometimes more sophisticated schemes are used, for example, minimizing the standard deviation for each bucket for a given maximum number of buckets.

Multi-dimensional histogram construction is a much more difficult problem (and an entirely different beast) than its one-dimensional counterpart [19]. Algorithms for its construction and dynamic update are needed for query optimization in stream mining systems. This paper does not review this research area, but interested readers should consult the doctoral thesis by Poosala [20] and citations thereof [1].

Point vs. Window Models - for time-series analysis

Time series data stream analysis falls into two broad categories: *point monitoring* and *aggregate monitoring*. In point monitoring, only the most recent data point is important. In a typical application scenario, whenever a data point (representing, e.g., temperature, pressure, stock price) falls above or below certain thresholds, an alarm or notification is sent to a monitor.

In aggregate monitoring, parameters for analyzing the time series, such as the minimum, maximum, average, or standard deviation, are computed using data points selected according to a rule. One type of selection rule is based on windowing. Windows may be based on geographical information (e.g., location of sensors), physical parameters (e.g., temperature, pressure, or color) as well as temporal data. Some of the more common schemes proposed for time series analysis, assuming no pre-processing of the data, place the data into temporal windows based on endpoints. Some examples of commonly used windows are:

- *fixed windows* – for identifying data points which lie between 2 fixed endpoints; e.g., computing the average value during a fixed time interval $[t_i, t_{i+1}]$.
- *sliding windows* – for identifying data points which lie between 2 sliding endpoints (forward or backward); e.g., computing the highest value in the past 24 hours.
- *landmark windows* – for identifying data points which lie between 1 fixed and 1 sliding window; e.g., computing the average price since Jan 1, 2004.
- *damped windows* – for lending more recent data points higher weight.

[1] e.g., using the *citations* options in *CiteSeer* or *ResearchIndex* search engine: *http://citeseer.nj.nec.com/cs*

Correlation Measures

Understanding correlations between two data streams or a data stream with its own past is important in numerous applications, such as scientific data, stock trading and complex mechanical systems. In classical statistical theory, means for quantitatively evaluating the magnitudes of correlations have been precisely defined [5]. Some important examples that have been extended for use in data stream mining systems are given below. The *synchronized correlation* of two streams $X = \{x_i\}$ and $Y = \{y_i\}$; is defined as

$$\text{correlation } (X,Y) \;=\; \frac{\frac{1}{w}\sum_{i=1}^{w} x_i y_i \;-\; \bar{x}\,\bar{y}}{\sqrt{\sum_{i=1}^{w}(x_i-\bar{x})^2\,(y_i-\bar{y})^2}},$$

where $\bar{x}$ and $\bar{y}$ denote the averages of streams $\{x_i\}$, and $\{y_i\}$, respectively. The *lagged correlation* of two streams $X = \{x_i\}$ and $Y = \{y_{i+k}\}$; with constant $k \in \mathcal{N}$, $k < w$, is

$$\text{lagged correlation } (X,Y) \;=\; \frac{\frac{1}{w}\sum_{i=1}^{w} x_i y_{i+k} \;-\; \bar{x}\,\bar{y}}{\sqrt{\sum_{i=1}^{w}(x_i-\bar{x})^2\,(y_{i+k}-\bar{y})^2}},$$

where $\bar{x}$ and $\bar{y}$ denote the averages of streams $\{x_i\}$, and $\{y_{i+k}\}$, respectively. The lagged correlation of a data stream with itself, i.e., $X = \{x_i\}$ and $Y = \{x_{i+k}\}$ is known as *self-correlation* or *auto-correlation.*

Wavelet Analysis

"Wavelets are families of functions $h_{a,b}$:

$$h_{a,b} = |a|^{-1/2} h\left(\frac{x-b}{a}\right) \;;\quad a,b \in \mathcal{R}\,,\; a \neq 0$$

generated from a single function h *by dilations and translations"* [3]. One of the applications of the theory is to construct a basis set $\{h_{a,b}\}$ for efficient and accurate approximation of functions, operators, and signals [18]. A second class of applications involves the use of wavelet transforms for analysis of non-stationary signals, i.e., time-frequency analysis [21, 22]. For wavelets with a mother function h, the *continuous wavelet transform* for a function $f \in L^2(\mathcal{R})$ is

$$\langle h_{a,b}, f\rangle = |a|^{-1/2} \int dx \cdot h\left(\frac{x-b}{a}\right) \cdot f(x)$$

for $a, b \in \mathcal{R}$, $a \neq 0$, and the *discrete wavelet transform* is

$$\langle h_{m,n}, f\rangle = |a_0|^{-m/2} \int dx \cdot h(a_0^{-m}x - nb_0) \cdot f(x)$$

for $a_0 > 1$, $b_0 \neq 0$ [4].

Data stream mining systems to date that use wavelet technologies have been based only on *Haar wavelets*. The one-dimensional Haar wavelet decomposition amounts to taking the averages and differences of two neighboring signals and scaling (i.e., multiplying) by a factor of $1/\sqrt{2}$. Signal processing scientists describe the Haar wavelet transform process as convolving the signal (or stream) with the *low pass filter* $\{1/\sqrt{2}, 1/\sqrt{2}\}$ and the *high pass filter* $\{-1/\sqrt{2}, 1/\sqrt{2}\}$, then down-sampling by two [23]. The wavelet decomposition is an attractive tool for data stream analysis, because its extension to the multidimensional case is straightforward. Simply apply the one-dimensional decomposition separately to each of the coordinates, one at a time.

3. Selected Real-World Data Stream Mining Problems

In this section we examine how some real world problems with data streams have been tackled using the modeling and analysis described above. These problems involve data streams from securities trading, approximate query processing systems, and telecommunications call detail records.

Analysis of Stock Market Data Streams using the DFT

Detection of correlations among multiple data streams appears in many applications. For example, more than 20,000 sensors send one tick per second as part of the monitoring system for NASA's Space Shuttle system [12]. A more difficult problem involving 50,000 streams with up to 100,000 ticks per second is the analysis of securities trading in the New York Stock Exchange (NYSE). Zhu and Shasha [24, 25] developed a means for fast approximation of cross-correlation and auto-correlation coefficients.

Zhu and Shasha's algorithm works as follows. The data are placed into very long sliding windows, each of which consists of an integer number of much smaller, *basic* windows. Sliding windows can shift by one, two or any integer number of basic windows. Summaries or *digests* for each of these smaller windows are computed and stored. The digests facilitate fast updating of information in the sliding windows.

The central idea in the algorithm is to take advantage of some valuable relationships and approximation properties of the discrete fourier transform (DFT) applied to time-series data:

- the DFT preserves Euclidean distances between two sequences;

- the DFT possesses the symmetry property; and
- the first few DFT coefficients usually approximate well the curve of the time-series of data from real-world problems in the field.

Bceause of the third property, the DFT is suited for data stream applications since good approximations can be quickly determined even when only the first few DFT coefficients for each basic window are computed. The inner products of these coefficients can then be used to compute cross-correlations and auto-correlations of the two time series.

Typically there are several trades per second for each of the approximately 300 stocks on the NYSE. If a stock experiences no trading activity within a time frame, then its most recent value is used. Sliding windows vary between 0.5 hours to 2 hours, and basic windows are 0.5 minutes to several minutes, depending on the number of stocks to be monitored and the computational resources of the user. The target is to identify only those pairs which show positive, high-value *synchronized* and *lagged* correlations that exceed a user-specified threshold. More recently, Zhu and Shasha have extended their work to detection of abnormal patterns in data stream aggregates, known as *elastic bursts* [25].

Wavelets for Highest B-Term Approximation

Wavelet-based techniques have been applied in a variety of data mining applications [14], and recently, they are being used in data stream mining. Some attractive properties associated with the wavelet transform are:

- For a signal with wavelet coefficients $\{w_1, w_2, \ldots, w_{N-1}\}$, sorted in decreasing order so that $|w_{l1}| \geq |w_{l2}| \geq |w_{l3}| \geq \cdots$, the *highest B-term approximation* $\sum_{k=1}^{B} w_{lk} xi_{lk}$ (where the xi_l are the wavelet basis vectors) is also the *best* B-term approximation, i.e., it minimizes the sum squared error for any given B.
- The energy of a signal, which is equivalent to the square of its L_2-norm, is preserved under the wavelet transform.
- In empirical observations of many types of natural signals, the highest B-term approximation of many naturally occurring signals is very accurate for very small values of B (i.e., has relatively small error).

- Futher efficiencies in computation and storage space can be achieved by approximating the highest B-term representation with a similar error.
- wavelet transform techniques can be extended straightforwardly to higher dimensions, without scalability problems, unlike some traditional signal processing techniques.

Matias, Vitter, and Wang studied multiresolution and wavelet transform-based analysis methods for histogram construction for on-line selectivity estimation [16]. They subsequently extended the work to consider tracking of the most significant wavelet transform coefficients over time [17]. The work also extends that by Lee, Kim, and Chung [13] on tracking the most significant DFT coefficients over time. Although results from experiments by Matias et al. using synthetic data appear to be promising, no provable error bounds are given.

Garofalakis and Gibbons [6] proposed a method for generating *probabilistic wavelet synopses*, i.e., synopses that are generated using wavelet transform coefficients and probabilistic thresholding. Their method can generate synopses fast enough for data stream analysis and provides error guarantees insuring highly accurate answers for approximate query processing systems. Work on dynamic maintenance of probabilistic wavelet synopses is currently on going.

Gilbert, Kotodis, Muthikrishnani, and Strauss [7] developed a method to generate small space memory requirement (or "*small space*), one-pass summaries of *call detail records* from AT&T to "enable answering point, range, and aged aggregate queries from one or more data streams". In addition to new results, the paper presents an excellent, short review of related works on one-pass algorithms, small space representations of massive amounts of data, dynamic maintenance of synopses and the difficulties associated with maintaining summaries of the largest k items and with identifying new trends in an evolving stream. For the particular application considered by the authors, sampling techniques cannot be used to reduce the scale of the problem size, so the use of wavelet transforms is critical to achieving the high processing speeds needed for data stream analysis. Gilbert et al. recognized that in addition to the desirable properties associated with the highest B-term approximation of the wavelet transform noted by Matias, Vitter, and Wang, further efficiencies in computation and storage space can be achieved by *approximating* the highest B-term representation without significant increase in the error. They developed and applied an approximation technique that performs well in call detail record query processing.

4. Conclusion

We surveyed some selected works that solved real world problems that can be effectively modeled as data stream mining systems. Since the data streams evolve massive volumes of temporally changing data, traditional techniques from data mining cannot necessarily be applied straightforwardly. Many more difficult problems involving sensor systems and incomplete data pose even greater challenges in this emerging area of research. Faster and more accurate methods for approximating and analyzing data streams are being sought.

References

[1] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, Models and issues in data stream systems, *ACM Principles of Database Systems (PODS) tutorial*, 2002.

[2] Y. Chen, G. Dong, J. Han, J. Pei, B.W. Wah, and J. Wang, Online analytical processing stream data: Is it feasible?, *Proceedings of Data Mining and Knowledge Discovery 2002: www.cs.uiuc.edu/~hanj*

[3] I. Daubechies, Orthonornal bases of compactly supported wavelets, *Communications on Pure and Applied Mathematics*, Vol. XLI, pp. 909–996, 1998.

[4] I. Daubechies, The wavelet transform, time-frequency localization and signal analysis, *IEEE Transactions on Information Theory*, Vol. 36, pp. 961–1005, Sept. 1990.

[5] *Encyclopedia of Statistical Sciences*, Vol. 2, John Wiley and Sons, New York, NY, pp. 193–204, 1982.

[6] M. Garofalakis and P. Gibbons, Wavelet synopses with error guarantees, *Proceedings of ACM SIG Modeling of Data (SIGMOD)*, pp. 476–487, 2002.

[7] A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss, Wavelets on streams: one-pass summaries for approximate aggregate queries, *The VLDB Journal*, pp. 79–88, 2001.

[8] J. Gehrke, F. Korn, and D. Srivastava, On computing correlated aggregates over continuous data streams, *Proceedings of ACM SIGMOD*, 2001.

[9] L. Golab and M. Ozsu, Data Stream management issues, *University of Waterloo, Canada, Department of Computer Science, Technical Report,* No. CS-2003-08, April 2003.

[10] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, Clustering data streams, *Proceedings of ACM Foundations of Computer Systems (FOCS)*, 2000.

[11] G. Hulten, L. Spencer, and P. Domingos, Mining time-changing data streams, *Proceedings of ACM SIG Knowledge Discovery and Data Mining (KDD)*, 2001.

[12] E. Keogh and P. Smyth, A probabilistic approach to fast pattern matching in time series databases, *Proceedings of ACM SIGKDD*, 1997.

[13] J. Lee, D. Kim, and C. Chung, Multi-dimensional selectivity estimation using compressed histogram information, *Proceedings of ACM SIGMOD*, 1999.

[14] T. Li, Q. Li, S. Zhu, and M. Ogihara, A survey on wavelet applications in data mining, *ACM SIGKDD Explorations*, Vol. 4, Issue 2, pp. 49–68, Dec. 2002.

[15] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, New York, NY, 1997.

[16] Y. Matias, J. Vitter, and M. Wang, Wavelet-based histograms for selectivity estimation, *Proceedings of ACM SIGMOD*, 1998.

[17] Y. Matias, J. Vitter, and M. Wang, Dynamic maintenance of wavelet-based histograms, *Proceedings of VLDB*, 2000.

[18] Y. Meyer, *Ondelettes et Algorithmes Concurrents*, *Ondelettes et Operateurs*, and *Opérateurs de Calderon-Zygmund*, Hermann, Paris, 1990.

[19] S. Muthukrishnan, V. Poosala, and T. Suel, Partitioning two dimensional arrays: algorithms, complexity and applications, *Proceedings of the International Conference on Database Theory*, 1998.

[20] V. Poosala, Histogram-based estimation techniques in database systems, *Ph.D. Thesis*, Computer Science, University of Wisconsin, Madison, 1997.

[21] S. Qian and D. Chen, *Joint Time-Frequency Analysis*, Prentice Hall, Englewood Cliffs, New Jersey, 1996.

[22] O. Rioul and M. Vetterli, Wavelets and signal processing, *IEEE Signal Processing Magazine*, Vol. 10, pp. 14–38, 1991.

[23] M. Vetterli and C. Herley, Wavelets and filter banks, *IEEE Transactions on Signal Processing*, Vol. 40, No. 9, pp. 2207–2232, 1992.

[24] Y. Zhu and D. Shasha, StatStream: statistical monitoring of thousands of data streams in real time, *Proceedings of VLDB*, 2002.

[25] Y. Zhu and D. Shasha, Fast approaches to simple problems in financial time series streams, *Proceedings of the Workshop on Management and Processing of Data Streams*, June 8, 2003.