Tutorial: Mining Massive Data Streams

Michael Hahsler

Lyle School of Engineering
Southern Methodist University

January 23, 2019

Table of Contents

Data Streams

Definition

A data stream is an ordered and potentially infinite sequence of data points:

$$\langle \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \ldots \rangle$$

where $\mathbf{y}_i$ is a tuple (e.g., a vector)

Such streams of constantly arriving data are generated by many types of applications including:

- web click-stream data
- computer network monitoring data
- telecommunication connection data
- readings from sensor nets
- stock quotes

Example: HTTP Server Log

```
208.76.226.148 - - [15/Jan/2012:04:02:42 -0600]
    "GET /MMSA/destroysession.php HTTP/1.0" 302 -
208.76.226.148 - - [15/Jan/2012:04:02:42 -0600]
    "GET /MMSA/index.php HTTP/1.0" 200 11339
129.119.113.115 - - [15/Jan/2012:04:03:43 -0600]
    "GET / HTTP/1.1" 200 1227
208.76.226.148 - - [15/Jan/2012:04:03:48 -0600]
    "GET /PIIH/2011/hurricanes/AL122011/11090118AL1211_PIIH.txt
    HTTP/1.0" 304 -
```

Data stream mining algorithms

- Clustering
- Classification
- Frequent Pattern Mining
- Change Detection
- Database Operations: indexing streams for trend and aggregation queries
- Mining multiple streams

See Aggarwal (2007) and Gama (2010) for current surveys.

Table of Contents

Properties of data streams

- Unbounded size of stream
  - Transient (stream might not be realized on disk)
  - Single pass over the data
  - Only summaries can be stored
  - Real-time processing (in main memory)
- Data streams are not static
  - Incremental updates
  - Concept drift
  - Forgetting old data
- Temporal order may be important

Why can we not use the standard algorithms?

- Why can we not use a regular relational DB and SQL?
- Why not a k-nearest neighbors classifiers?
- Why not k-means/hierarchical clustering?
- Why not Apriori to find frequent itemsets?

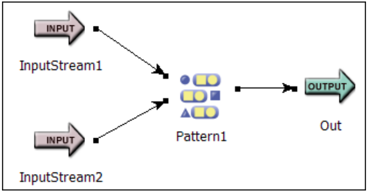
Relational DB vs. Data Streams

Relational DBMS	DSMS (Stream)
persistent relations	transient streams
only current state is important	history matters
not real-time	real-time
low update rate	stream!
one time queries	continuous queries

Source: Babcock *et al.* (2002)

DSMS typically offer SQL-like languages with stream extensions to create continuous queries.

Example: Pattern matching in StreamSQL

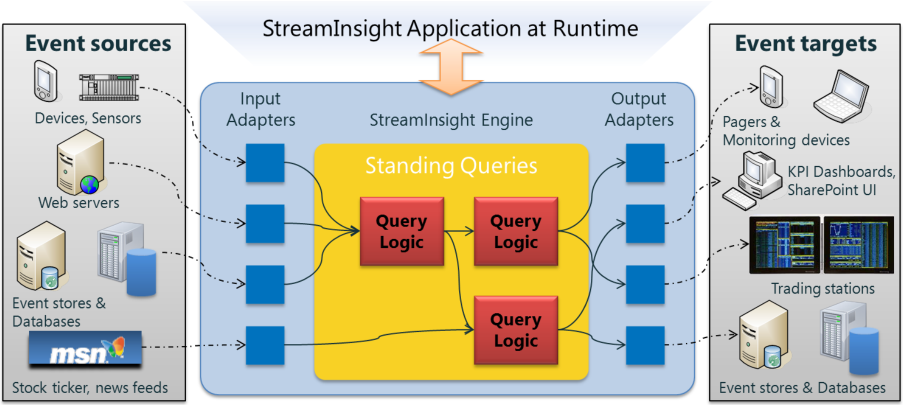



```
CREATE INPUT STREAM InputStream1 (stock string, value double);
CREATE INPUT STREAM InputStream2 (stock string, value double);
CREATE OUTPUT STREAM Out;

SELECT InputStream1.stock AS stock,
       InputStream1.value AS value1,
       InputStream2.value AS value2
    FROM PATTERN (InputStream1 THEN InputStream2) WITHIN 20 TIME
    WHERE (InputStream2.value > InputStream1.value)
        AND (InputStream1.stock = InputStream2.stock)
    INTO   Out;
```

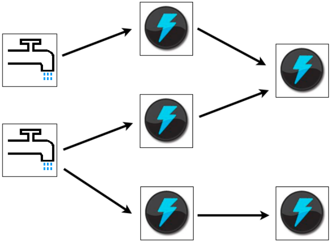
Source: StreamBase, http://www.streambase.com/

Example: Microsoft StreamInsight



Source: Introducing Microsoft StreamInsight, 2009

Example: Apache Storm



Source: Apache Storm (`https://storm.apache.org/`)

- **Topology:** A graph of spouts and bolts that are connected with stream groupings.
- **Spouts:** Read tuples from an external source and emit them into the topology.
- **Bolts:** Do simple stream transformations. Complex stream transformations often requires multiple bolts.

Traditional algorithms vs. DS algorithms

	Traditional	**Stream**
passes	multiple	single
processing time	unlimited	restricted
memory	disk	main memory
results	typically accurate	approximate
distributed	typically not	often

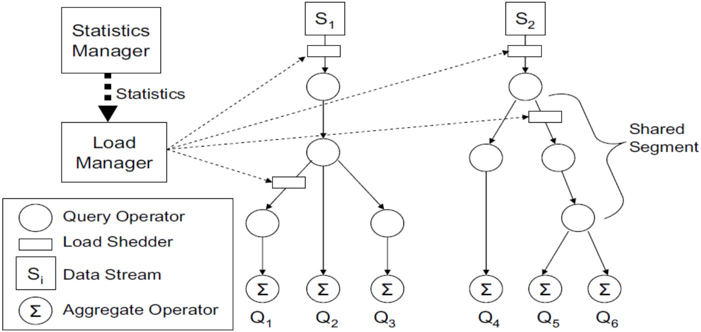
Source: Joao Gama, Data Stream Mining Tutorial, ECML/PKDD, 2007

Table of Contents

Load Shedding

Many data streams have bursts → discard some fraction of the unprocessed data.

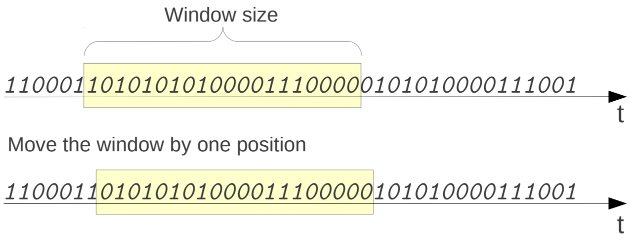
Objective: Minimizing inaccuracy in query answers, subject to the constraint that throughput must match or exceed the data input rate (placement and sampling rate).



Source: Babcock *et al.* (2003)

Table of Contents

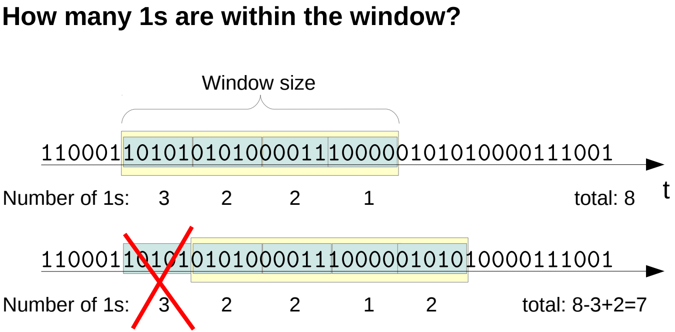
Time window



- Keep the most recent data points.
- Reconstruct a regular model from the window when it changes.
- Typically updated as a sliding window. Sometimes landmark or titled windows.

This is typically expensive!

Time window



- Use buckets
- Models need to be additive (works for count, mean, variance, etc.)
- Can also be used to detect change

Sampling

- Reduce the amount of data to process and store.
- Updating an unbiased sample is tricky since new data is arriving constantly!

What is the problem with the following approach to create a sample of size k:

1. Insert first k elements into sample
2. Add each new element to the sample with a fixed probability p.
3. If a new element was inserted then delete the oldest element in the sample.

Reservoir Sampling

Random Sampling with a Reservoir (Vitter, 1985)

Create a sample of size k:

1. Insert first k elements into sample
2. Then insert ith element with probability $p_i = k/i$.
3. If a new element was inserted then delete an instance at random.

Sketches

A sketch is a small data structure which can be easily updated and helps with estimating frequency moments of a data stream (typically with an error guarantee).

Sketches exist to approximate:

- Count unique values in a stream
- Identify heavy hitters (most frequent items)
- Finding quantiles
- Finding the difference between streams

Sketches: Count distinct values

Method to approximate the number of distinct values M:

- Maintain a Hash Sketch $BITMAP$ which is an array of L bits, where $L = O(log(M))$, initialized to 0.
- Assume a hash function $h(x)$ that maps incoming values x, uniformly across $[0, 2^L - 1]$.
- Let $lsb(h(x))$ denote the position of the least-significant 1 bit in the binary representation of $h(x)$.
- A value x is mapped to $lsb(h(x))$. For each incoming value x, set $BITMAP[lsb(h(x))] = 1$.

Example

$x = 5 \rightarrow h(x) = 101100 \rightarrow lsb(h(x)) = 3$
$BITMAP$: 0 0 0 0 0 0 0 0 0 0 1 0 0

Sketches: Count distinct values

Example

$BITMAP$: 0 0 0 0 1 0 1 1 0 1 1 1 1 1

Left most 0-bit is at position $R = 6$.

Flajolet and Martin proved that $E[R] = log(\phi M)$ with $\phi = .77351$

Estimate of $M = 2^R/\phi$.

Example

$M = 2^6/\phi = 82.7$ distinct values.

Source: Flajolet and Martin (1985). Adapted from Joao Gama, Data Stream Mining Tutorial, ECML/PKDD, 2007

Wavelets

Idea: Concentrate on the important features of the data.

- Wavelet transforms (e.g., Discrete Cosine and Fourier transforms) split the data up into components (e.g., basic trend and local variations)
- Retain only the most important components.
- For data stream summarization fast to compute Wavelets are used (e.g., Haar Wavelet)

Interactive Example:

`http://www.tomgibara.com/computer-vision/haar-wavelet`

Others

- Histograms
- Micro-clusters (see Clustering)
- Decision trees (see Classification)

Table of Contents

Clustering Data Streams

Conventional clustering algorithms need several passes over the complete data set!

Main ideas:

- Strategies
  1. **Time Window:** Split stream into time windows and cluster each window independently. Then combine the clusterings (STREAM).
  2. **Micro-clusters:** A small set of statistics which can be iteratively updated (mean, variance, etc.). (CluStream, DenStream)
  3. **Density based:** Map each data point into a predefined grid. (D-Stream, MR-Stream)
- **Reclustering:** Use conventional clustering (e.g., k-means, DBSCAN) off-line to combine micro-clusters/grids.
- **Exponential decay** to decrease the influence of older data on the micro-clusters. This deals with concept drift.

See Silva *et al.* (2013) for a current survey.

A very simple algorithm

1. Start with an empty set of micro-clusters
2. For each new data point x
    1. Find for x the closest micro-cluster c
    2. If x is closer to c then a set threshold δ then
        1. add update x to absorb c

       otherwise
        1. create a new micro-cluster for c.

Some research questions

- Temporal structure?
- No off-line reclustering?
- How do we compare different algorithms?

Table of Contents

Classification

Several classification methods suitable for data streams have been developed. Examples are

- *Very Fast Decision Trees (VFDT)* (Domingos and Hulten, 2000) using Hoeffding trees
- *Online Information Network (OLIN)* (Last, 2002) using time-windows
- *On-demand Classification* (Aggarwal *et al.*, 2004a) based on data-stream clustering

For a detailed discussion of these and other methods we refer the reader to the survey by Gaber *et al.* (2007).

Decision Trees

Problem: How do we decide on splits if new data is constantly arriving?

- **Solution 1:** Use a time window.
- **Solution 2 (Very Fast Decision Trees):** Uses the current best attribute to make a split once the number of examples satisfies the *Hoeffding* bound. This gives a guarantee on how different the tree will be from a tree built on all the data. (Domingos and Hulten, 2000)

Problems with decision trees: Need to be rebuilt to adapt to concept drift.

Classification by Clustering

Idea: Cluster the data stream (CluStream; (Aggarwal *et al.*, 2003)) into groups and assign a label to each cluster. Find for a new data point the closest cluster and use its label. (Aggarwal *et al.*, 2004b)

Advantages:

- DS clustering is fast
- Takes care of concept drift
- Micro-clusters allow for an arbitrary decision boundary. Essentially is k-nearest neighbor with $k = 1$ and micro-clusters instead of data points)

Possible problems:

- What if micro-clusters contain points of several classes?
- How do we label a new developing micro-cluster?

Table of Contents

Conclusion

- Data streams are everywhere.
- Often a good approximation is all that is needed.
- Some streaming algorithms produce results of similar quality as traditional algorithm at a fraction of the computational cost $\rightarrow$ apply them to large non-streaming data.
- Data stream extensions for DBs are/will become available (e.g., MS SQL Server's StreamInsight).
- Distributed stream mining (cluster, grid, cloud) will become important.

References I

Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the International Conference on Very Large Data Bases (VLDB '03)*, pages 81–92, 2003.

Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. On demand classification of data streams. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 503–508, New York, NY, USA, 2004. ACM.

Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. On demand classification of data streams. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–508. ACM, 2004.

Charu Aggarwal, editor. *Data Streams – Models and Algorithms*. Springer-Verlag, 2007.

Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, pages 1–16, New York, NY, USA, 2002. ACM.

Brian Babcock, Mayur Datar, and Rajeev Motwani. Load shedding techniques for data stream systems. In *Proceedings of the 2003 Workshop on Management and Processing of Data Streams (MPDS*, 2003.

Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 71–80, New York, NY, USA, 2000. ACM.

Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, September 1985.

Mohamed Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. A survey of classification methods in data streams. In Charu Aggarwal, editor, *Data Streams – Models and Algorithms*. Springer-Verlag, 2007.

João Gama. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, Boca Raton, FL, 1st edition, 2010.

Mark Last. Online classification of nonstationary data streams. *Intelligent Data Analysis*, 6:129–147, April 2002.

Jonathan A. Silva, Elaine R. Faria, Rodrigo C. Barros, Eduardo R. Hruschka, Andre Carvalho, and Joao Gama. Data stream clustering: A survey. *ACM Computer Surveys*, 46(1):13:1–13:31, July 2013.

Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, March 1985.