Proceedings of National Conference on Communication and Informatics-2016, Organized by Department of Information Technology, Sri Venkateswara College of Engineering, Sriperumbudur

ISSN:2320-0790

An International Journal of Advanced Computer Technology

A Framework for Real-time Streaming Analytics using Machine Learning Approach

Ms.D.Jayanthi
Assistant Professor
Department of Information Technology
Sri Venkateswara College of Engineering,
Sriperumbudur

Dr.G.Sumathi,
Professor
Department of Information Technology
Sri Venkateswara College of Engineering,
Sriperumbudur

ABSTRACT: The continuous stream of data generated by sensors, machines, vehicles, mobile phones, social media networks, and other real-time sources are compelling organizations to imagine what they could do with this data if they could gain insight into it. A real-time streaming platform must meet the needs of data scientists, developers and data center operations teams without requiring extensive custom code or brittle integration of many third party components. As more and more data is generated and collected, data analysis requires scalable, flexible, and high performing tools to provide insights in a timely fashion. However, organizations are facing a growing big data ecosystem where new tools emerge and "die" very quickly. Therefore, it can be very difficult to keep pace and choose the right tools. This paper focuses on the challenges that stream processing solution should address. Also, this paper analyzes the traditional analytic tools to bridge the gap between data being generated and data that can be analyzed effectively.

Keywords: Streaming analytics, Machine Learning, Streaming platforms.

1. INTRODUCTION

As we become a more digital society, the amount of data being created and collected is growing and accelerating significantly. Analysis of this ever-growing data becomes a challenge with traditional analytical tools. We require innovation to bridge the gap between data being generated and data that can be analyzed effectively. Big data tools and technologies offer opportunities and challenges in being able to analyze data efficiently to better understand customer preferences, gain a competitive advantage in the marketplace, and grow your business. Data management architectures have evolved from the traditional data warehousing model to more complex architectures that address more requirements, such as real-time and batch processing; structured and unstructured data; high-velocity transactions; and so on.

Analyzing large data sets requires significant compute capacity that can vary in size based on the amount of input data and the type of analysis. This characteristic of big data workloads is ideally suited to the pay-as-you-go cloud computing model, where applications can easily scale up and down based on demand. As requirements change, you can easily resize your environment (horizontally or vertically) on AWS to meet your needs, without having to wait for additional hardware or being required to over invest to provision enough capacity.

1.1. Real-time Streaming

The swelling collection of sensors worldwide (plus the extended "Internet of Things") produces large volumes of streaming data that can be leveraged for business advantage. For example, robots have been in use for years in manufacturing, but now they have additional sensors so they can perform quality assurance, not just assembly. For decades, mechanical gauges have been common in many industries such as chemicals and utilities. Now gauges are replaced by digital sensors and "smart meters" to provide real-time monitoring and analysis. GPS and RFID signals now emanate from mobile

devices and assets, ranging from smart phones to trucks to shipping pallets, so all these can be tracked in real time and controlled precisely.

1.2 Streaming analytics

The growing consensus is that analytics is the most direct path to business value drawn from new forms of big data, which includes streaming data. Existing analytic techniques based on mining, statistics, predictive algorithms, queries, scoring, clustering, and so on apply well to machine data once it's captured and stored. Luckily, newer vendor tools are reengineering these and creating new analytic methods so they can operate on data that streams continuously as well as on other data in storage. As devices and systems connected to the Internet of Things generate streaming data on a global scale, firms need effective strategies for identifying and leveraging this data in new applications. That may require the integration and analysis of data from internal legacy sources, current customer-generated data, and external sources from partners and data service providers. The data management and analytics tools of the last decade may be able to handle the volume of data, but they simply can't keep up with the speed at which it must be processed. A variety of vendors are racing to meet this challenges with a new generation of data management and analytics tools. From query optimization solutions to integration techniques to new platforms for device interoperability, there is no shortage of innovation in this space today.

1.2.1 Today's real-world use cases for streaming analytics

1. Monitor and maintain the availability, performance, and capacity of interconnected infrastructures, such as utility grids, computer networks, and manufacturing facilities.
2. Understand customer behavior across multiple channels, so you can improve the customer experience as it's happening
3. Device Telemetry
4. Infrastructure monitoring
5. Identify compliance and security breaches, then halt and/or correct them immediately
6. Spot and stop fraudulent activity, even as fraud is being perpetrated
7. Inventory management
8. Web analytics/Content management

Evaluate sales performance in real time and achieve sales quotas through instant incentives such as discounts, bundles, free shipping, and easy payment terms

Such compelling use cases typically result from a "perfect storm" of emerging data types, software technologies, and fast-paced business methods

2. REAL TIME STREAMING PLATFORMS

Real-time analytics can keep you up-to-date on what's happening right now, such as how many people are currently reading your new blog post and whether someone just liked your latest Facebook status. For most use cases, real time is a nice-to-have feature that won't provide any crucial insights. However, sometimes real time is a must.

2.1. Hadoop

Hadoop was first out of the gate, and enjoyed (and still does enjoy) widespread adoption in industry. So why would you still use Hadoop, given all of the other options out there today? Despite the fact that Hadoop processes often complex Big Data, and has a slew of tools that follow it around like an entourage, Hadoop (and its underlying MapReduce) is actually quite simple. If your data can be processed in batch, and split into smaller processing jobs, spread across a cluster, and their efforts recombined, all in a logical manner, Hadoop will probably work just fine for you. A number of tools in the Hadoop ecosystem are useful far beyond supporting the original MapReduce algorithm that Hadoop started as. Of particular note, and of a foreshadowing nature, is YARN, the resource management layer for the Apache Hadoop ecosystem. It can be used by systems beyond Hadoop, including Apache Spark.

2.2 Apache Spark

Spark is an open-source data-processing framework that is really hot at the moment. Because Spark runs in-memory on clusters, and it isn't tied to Hadoop's MapReduce two-stage paradigm, it has lightning-fast performance. Spark can run as a standalone or on top of Hadoop YARN, where it can read data directly from HDFS. In addition to its in-memory processing, graph processing, and machine learning, Spark can also handle streaming. Companies like Yahoo, Intel, Baidu, Trend Micro, and Groupon are already using it. Spark is the heir apparent to the Big Data processing kingdom. Spark and Hadoop are often contrasted as an "either/or" choice, but that isn't really the case. The Hadoop ecosystem can accommodate the Spark processing engine in place of Map Reduce, leading to all sorts of different environment make-ups that may include a mix of tools and technologies from both ecosystems. As one specific example of this interplay, Big Data powerhouse Cloudera is now replacing MapReduce with Spark as the default processing engine in all of its Hadoop implementations moving forward. As

another example, Spark does not include its own distributed storage layer, and as such it may take advantage of Hadoop's distributed filesystem (HDFS), among other technologies unrelated to Hadoop (such as Mesos).Spark differs from Hadoop and the MapReduce paradigm in that it works in-memory, speeding up processing times. Spark also circumvents the imposed linear dataflow of Hadoop's default MapReduce engine, allowing for a more flexible pipeline construction. When would you choose Spark? If you don't want to be shackled by the MapReduce paradigm and don't already have a Hadoop environment to work with, or if in-memory processing will have a noticeable effect on processing times, this would be a good reason to look at Spark's processing engine. Also, if you are interested in tightly-integrated machine learning, MLib, Spark's machine learning library, exploits its architecture for distributed modeling. Again, keep in mind that Hadoop and Spark are not mutually exclusive.

2.3 Apache Storm

Storm is a distributed real-time computation system that claims to do for streaming what Hadoop did for batch processing. It can be used for real-time analytics, machine learning, continuous computation, and more. The cool thing is that it was designed to be used with any programming language. It runs on top of Hadoop YARN and can be used with Flume to store data on HDFS. Storm is already used by the likes of WebMD, Yelp, and Spotify. Apache Storm is a distributed real-time computation system, whose applications are designed as directed acyclic graphs. Storm is designed for easily processing unbounded streams, and can be used with any programming language. It has been benchmarked at processing over one million tuples per second per node, is highly scalable, and provides processing job guarantees. Apache Storm can be used for real-time analytics, distributed machine learning, and numerous other cases, especially those of high data velocity. Storm can run on YARN and integrate into Hadoop ecosystems, providing existing implementations a solution for real-time stream processing.

2.4 Apache Samza

Samza is a distributed stream-processing framework that is based on Apache Kafka and YARN. It provides a simple callback-based API that's similar to MapReduce, and it includes snapshot management and fault tolerance in a durable and scalable way. Samza manages snapshotting and restoration of a stream processor's state. When the processor is restarted, Samza restores its state to a consistent snapshot. Samza is built to handle large amounts of state (many gigabytes per partition) Whenever a machine in the cluster fails, Samza works with YARN to transparently migrate your tasks to another machine. Samza uses Kafka to guarantee that messages are processed in the order they were written to a partition, and that no messages are ever lost. Samza is partitioned and distributed at every level. Kafka provides ordered, partitioned, replayable, fault-tolerant streams. YARN provides a distributed environment for Samza containers to run in.

2.5 Flume

Flume is a distributed, reliable, and available system for efficiently collecting, aggregating, and moving large amounts of event data. The topology of Flume is made of multiple agents, each of which runs in a separate Java Virtual Machine (JVM). An agent consists of three pluggable components, named source, sink and channel. Source collects incoming data as events, sink writes events
out, and channels connect the source and sink. The channel on each agent functions as the data buffer that stores the events in case of the downstream failure or shutdown. Flume has file-based and memory based channels. The original purpose of Flume is used to log data aggregation. But, since data sources are customizable, Flume can be used to transport massive quantities of event data, such as network traffic data, social-media-generated data, and email messages.

2.5 Scribe

Scribe is the system for aggregating streaming log data. It is designed to scale to a very large number of nodes and be robust to network and node failures. Scribe running on each node is configured to aggregate messages, and to send messages to a central Scribe server (or multiple Scribe servers in a large cluster). If the central Scribe server is not available, Scribe will write the messages to the local disk and sends them when the central server recovers. The central Scribe server(s) write the messages to the files as the final destination, typically on a network file system or a distributed file system, or send them to another layer of Scribe servers.

2.6 S4

S4 is a general-purpose, near real-time, distributed, decentralized, scalable, event-driven, modular platform for processing continuous unbounded streaming data. S4 has a decentralized and symmetric architecture which all the nodes in a cluster are identical, different to the classic master-nodes architecture. S4 employs ZooKeeper as the communication layer to coordinate the nodes within the cluster. But, this is transparent to users since S4

can automatically handle communication, scheduling and distribution across all the nodes. The design of S4 is derived from the combination of MapReduce and Actors model The computation on S4 is performed by the so-called Processing Elements (PEs). S4 transmits messages between the PEs in the form of data events. The state of each PE is inaccessible to the other PEs, and the event emission and consumption are the only mode of interaction between PEs.

An S4 cluster contains many PEs to process events. Since data is streamed between PEs, no on-disk checkpoint is required. Thus, only partial fault tolerance is supported in S4. For example, if a processing node fails, its processes are automatically moved to a standby server, but the states of these processes are lost, and cannot be recovered.

2.7 HStreaming

HStreaming is an analytics platform built on top of Hadoop and MapReduce. The architecture of HStreaming consists of two components: data acquisition and data analytics. The data acquisition component is able to collect data in near real-time, and has ETL capabilities, while the analytics component allows to analyze unstructured and structured data on HDFS in a real-time fashion. HStreaming also provides the connectors for connecting both SQL and NoSQL data stores, which make it possible to ana lyze the data from different databases. HStreaming provides both enterprise and community editions.

2.8 All-RiTE.

All-RiTE is an ETL middleware system, which enables efficient processing live data warehouse (DW) data. The live DW data, such as accumulating facts and early arriving facts, will eventually be updated to the data warehouse, but also queried in an online fashion. It is typically not efficient to deal with live data using traditional technologies, such as using SQL INSERTs followed by the possible updates and deletions, when the data has been loaded into the DW. All-RiTE solves this problem by making use of an intermediate data store, called catalyst. The catalyst locates between data producers and the DW, and accumulates live data in the in-memory store. Therefore, if there necessitates to update or delete the rows in the catalyst, the updates and deletions are done on-the-fly when the data is queried or materialized to the DW. All-RiTE exploits a number of user-defined flush policies to decide when to move data from source systems towards the DW. Through the flush policies, the data could be queried in a near realtime or right-time fashion (right-time means only the data satisfying the specified time accuracy is read). All-RiTE can run in stand-alone mode or be integrated with other ETL tools to process live DWdata

2.9 Impala.

Impala is an open source real-time analytics system developed by Cloudera. This system is inspired by Google's Dremel which is a scalable, interactive ad-hoc query system for data analytics. Impala provides an SQL-like engine to execute queries against the data in HDFS and HBase, which is somewhat similar to Hive. In an Impala cluster, Impala daemons run on all the nodes, which cache some of the data read from HDFS, and process the in-memory data. Therefore, the results can be returned very quickly. Impala uses the metastore of Hive to save the metadata, and supports the subset of Hive SQL, ODBC driver and friendly user-interfacing (through Hue Beeswax). Therefore, to Hive users, Impala provides a familiar and unified platform for both batch-oriented and real-time queries. The first beta version supports text files and SequenceFiles, and additional formats including Avro, RCFile, LZO text files, and new Parquet columnar format. However, Impala does not provide fault-tolerance compared to Hive due to its use of the in-memory based operations.

2.10 Amazon Kinesis

Kinesis is Amazon's service for real-time processing of streaming data on the cloud. It's deeply integrated with other Amazon services via connectors, such as S3, Redshift, and DynamoDB, for a complete Big Data architecture. Kinesis also includes Kinesis Client Library (KCL) that allows you to build applications and use stream data for dashboards, alerts, or even dynamic pricing.

2.11 Enterprise Solutions

The big firms don't just sit and twiddle their thumbs while the Big Data keeps growing. IBM InfoSphere Streams, Microsoft StreamInsight, and Informatica Vibe Data Stream are just a few of the commercial enterprise-grade solutions that are available for real-time processing. Chances are that if your company already uses, for example, Microsoft-based products, you'll stay with your vendor and add on StreamInsight to your roaster. Otherwise, you'll probably go for one of the open-source solutions.

3. SEVEN ESSENTIAL ELEMENTS IN REAL TIME STREAMING ANALYTICS PLATFORM

1. Open source

software created by a limited number of developers or a software package created by thousands of developers around the globe? Open source is, of

course, the latter and allows for countless developers and users to create innovative new features and enhancements to the code.

2. Future-proof

The technological advances related to real-time streaming analytics are moving and changing as rapidly as data itself. The ideal real-time streaming analytics platform would have an architecture with a common abstraction layer below the user interface that allows the selection of one or more streaming engines and allows you to change engines as your goals, requirements and strategies change. This abstraction layer would also streamline upgrades and embed new technologies into the existing system, based on relevance and credibility.

3. Low latency

Latency refers to the time required for a system to respond to input. In the case of streaming data analytics - this is typically measured in how many milli-seconds or seconds it takes to ingest, process, analyze and respond to an incoming event or data-point. This depends on the architecture of the underlying stream processing engine. Some popular micro-batch based architectures like Spark Streaming may not be able to process a response in anything less than 500 milli-seconds where as discrete event processing architectures like Apache Storm could provide responses in less than 10 milli-seconds. This will vary and depend on data size of each event and the complexity of the calculations performed.

4. Data Integration

Hadoop MapReduce, Storm and Spark for massively parallel processing; Ni-fi, Kafka, Flume, Active MQ along with traditional messaging and queuing software for real time data movement; Mesos and YARN for resource management. We encourage you to find a platform that provides an easily usable UI driven abstraction over the stack of complex technologies used in Big Data platforms.The platform should make the experience easy for developers, analysts and business users and have them be much more productive and able to focus on business needs rather than infrastructure issues.

5. Pre-built operators

The ideal RTSA platform will include a wide range of dataprocessing operators including but not limited to the following:

In addition to such a rich array of pre-built operators; it would be good to watch for developer life-cycle enablement features like automatic porting from development, to test to production; ability to track and maintain multiple application versions and easily upload and manage custom code and extensions for tailored business logic.

6. Elastic scaling

Ideally, you want a platform that the operations team can easily scale out or scale down by visually changing the resource allocation to various tasks and the newly configured resources seamlessly join the cluster and are able to handle changes in load or traffic without any overall interruption to the real-time streaming data processing.

7. Data Visualization

Attractive and easy-to-understand pictorial illustration of data with graphs charts and dashboards are a big factor in increasing adoption of any platform among the user community. A strong RTSA platform will provide data visualization tools that 7 allow you to:

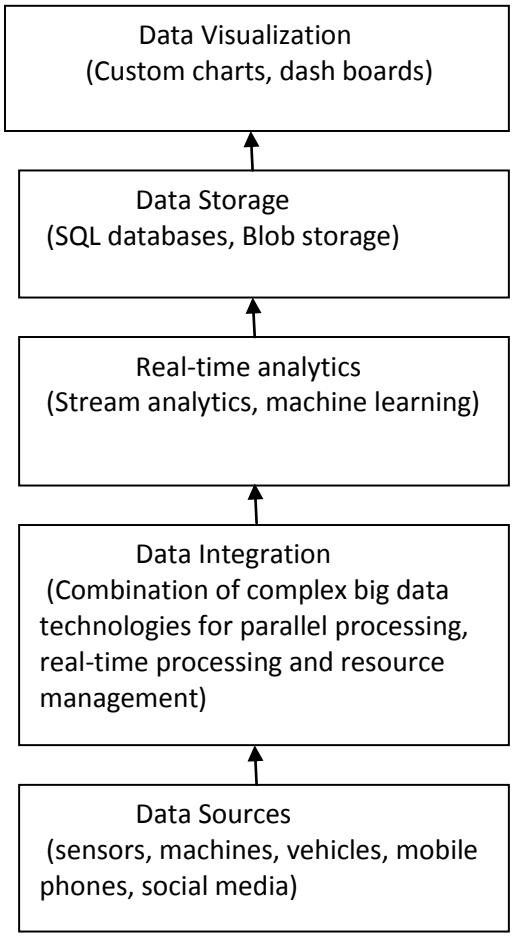
i) Visualize live streaming data with add-ons like thresholds, alerts, range markers, etc.

ii)Dynamically create custom charts or dashboards

iii) See trending and comparisons

iv) Easily build custom UI extensions

4. FRAMEWORK FOR REAL TIME STREAMING ANALYTICS



4.1 Data sources

Data may be generated from systems already in the cloud, or from devices local to a person or piece of equipment. Additionally, the events from these systems may be delivered on time as they occur, or batched with timestamps to be delivered when connectivity it available.

4.2 Data Integration

The Data-Integration layer provides a set of capabilities for ingesting data of varying formats and velocity from either external sources or existing cloud storage. Both Event Hubs and Blob Storage are foundational storage components of Azure that often support additional solution scenarios outside of real-time analytics. As such, they create a natural data-integration point for additional workloads such as archival storage, batch analytics, or supporting self-service business intelligence.

4.3 Real-time streaming analytics

There are two technologies that support the real-time analytics layer of this event processing reference architecture. Stream Analytics acts as the linchpin of this reference architecture by providing the engine capable of temporal analytics over moving data streams. The second technology in the real-time analytics layer is Machine Learning, which is a predictive analytics service capable of consuming either a single record made up of multiple columns (sometimes referred to in the context of machine learning as “features”) via request / response API, or consuming a file for asynchronous batch scoring. Given the real-time scope of this document, the role of Machine Learning will be based on the request / response method only.

4.4 Data Storage

Typically, solutions that require interactive query response rates across aggregate / filtered data will use SQL Database as the destination data store, whereas solutions that capture all events for large scale analytics such as training machine learning models should leverage Blob Storage as the store. HDInsight compute can then be bound to this storage on demand for processing the data.

4.5 Data Visualization

The Presentation / Consumption layer supports two primary endpoints. Whereas a more traditional batch analytics solution comprised of ETL, Data Warehouses and Cubes will generally support end users only, real-time analytics solutions will commonly support both end users and/or applications that consume specific events and initiate additional automated action. Visualizations representing data from the Azure SQL Database, in conjunction with other data sources outside of the scope of the real-time analytics solution can be centralized onto a dashboard to support a single view across all information, regardless of the underlying source. The following diagram demonstrates the connectivity method for enabling Power BI to connect with SQL Database.

5. MACHINE LEARNING WITH STREAMING

Predictive analysis defines functions that can perform analytic algorithms. Incremental machine learning algorithms learn and update a model on the fly, so that predictions are based on a dynamic model. Traditional supervised learning algorithms train data models based on historical, static data. In the traditional scenario, training and retraining are infrequent events that require a large pre-existing data set to be maintained. Once training is complete, a learned model is stored in a table. When new data arrives, the scoring function makes predictions based on this stored model. As patterns change, the model needs to be retrained with more historical, labelled, data to ensure the accuracy of the algorithm. In contrast, supervised learning in streaming can continuously learn as new data arrives and is labelled, thus allowing accurate scoring in real time, which adapts to changing situations.

Traditional unsupervised learning algorithms analyze a large dataset to detect hidden patterns in data, without any labels being provided to the algorithm. When new data needs to be analyzed, the entire dataset must be re-examined to determine patterns. Conversely, unsupervised learning in streaming is able to detect novel patterns in streaming data in real time without any re-analysis of previously examined data.

"Fast Data" via stream processing is the solution to embed patterns - which were obtained from analyzing historical data - into future transactions in real-time. It deals with how patterns and statistical models of R, Spark MLlib and other technologies can be integrated into real-time processing using open source frameworks (such as Apache Storm, Spark or Flink) or products (such as IBM InfoSphere Streams or TIBCO StreamBase).

6. **Comparison of Real-time processing Systems**

Name of system	Architecture	Catalog	Integration/ Query	In-memory	Recovery	License
Hadoop Online	Master/Slaves	No	Integration	Yes	No	Apache License 2.0
Strom	Peer	No	Integration	No	Manual	Eclipse public License
Flume	Peer	No	Integration	Yes/backed by files	Manual	Apache License 2.0
Spark Streaming	Master/Slaves	No	Both	Yes	Parallel	Apache License 2.0
Kafka	Publish/subscribe	No	Integration	Yes/backed by files	Yes	Apache License 2.0
Scribe	Master/Slaves	No	Integration	Yes/backed by files	Yes	MIT License
S4	Peer	No	Integration	Yes	Manual	Apache License 2.0
HStreaming	Master/Slaves	No	Both	Semi-in memory	Yes	community
All-RiTE	Peer	No	Both	In-memory	Yes	GPL 2.0
Impala	Peer	Yes	Analytical Query	Yes	Yes	Apache License 2.0
Amazon Kinesis	Master/Slave	Yes	Both	In-memory	Yes	Amazon Software License

7. CONCLUSION

With a broad set of managed services to collect, process, and analyze big data the AWS platform makes it easier to build, deploy and scale big data applications, allowing you to focus on business problems instead of updating and managing these tools. There are many solutions to address big data analytic requirements. Most big data architecture solutions use multiple tools to build a complete solution: this can help meet the stringent business requirements in the most cost-optimized, performant, and resilient way possible. The result is a flexible, big data architecture that is able to scale along with your business on the global infrastructure.

8. REFERENCES

1. 1.André Leon Sampaio Gradvohl, Hermes Senger, Luciana Arantes, Pierre Sens," Comparing Distributed Online Stream Processing Systems Considering Fault Tolerance Issues,Journal of Emerging Technologies in Web Intelligence, Vol 6, No 2 (2014), 174-179, May 2014,doi:10.4304/jetwi.6.2.174-179.

2. Rahnama A.H.A, "Distributed real-time sentiment analysis for big data social streams", IEEE International Conference on Control, Decision and Information Technologies (CoDIT), (Nov 2014) page(s):789-794,doi:10.1109/CoDIT.2014.6996998"

3. Gianmarco De Francisci Morale, "SAMOA: A Platform for Mining Big Data Streams", 22nd International Conference on WWW 2013 Companion, May 13–17, 2013, Rio de Janeiro, Brazil. ACM 978-1-4503-2038-2/13/05.

4. Mohit Maske*, Dr. Prakash Prasad, "A Real Time Processing and Streaming of Wireless Network Data using Storm ", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE),vol 5,Issue 1, Jan 2015, ISSN:2277 128X,page(s):506-510

5. Gianmarco De Francisci Morales,Albert Bifet,SAMOA: "Scalable Advanced Massive Online Analysis", Journal of Machine Learning Research 16 (2015) page(s):149-153

6. Bifet, A. ,De Francisci Morales, G.,Big "Data Stream Learning with SAMOA", 2014 IEEE International Conference on Data Mining Workshop (ICDMW),page(s):1199 – 1202, 978-1-4799-4275-6

7. Dilpreet Singh, Chandan K Reddy,"A survey on platforms for big data analytics", Journal of Big Data 2014,page(s): 2:8 doi:10.1186/s40537-014-0008-6

8. Georg Krempl,Indre Žliobaite," Open challenges for data stream mining research",ACM SIGKDD Explorations - Special issue on big data archive,Volume 16 Issue 1, June 2014,Pages 1-10
9. Min Chen · Shiwen Mao · Yunhao Li," Big Data: A Survey", © Springer Science+Business Media New York 2014, Mobile Netw Appl (2014) 19:171–209 DOI 10.1007/s11036-013-0489-0